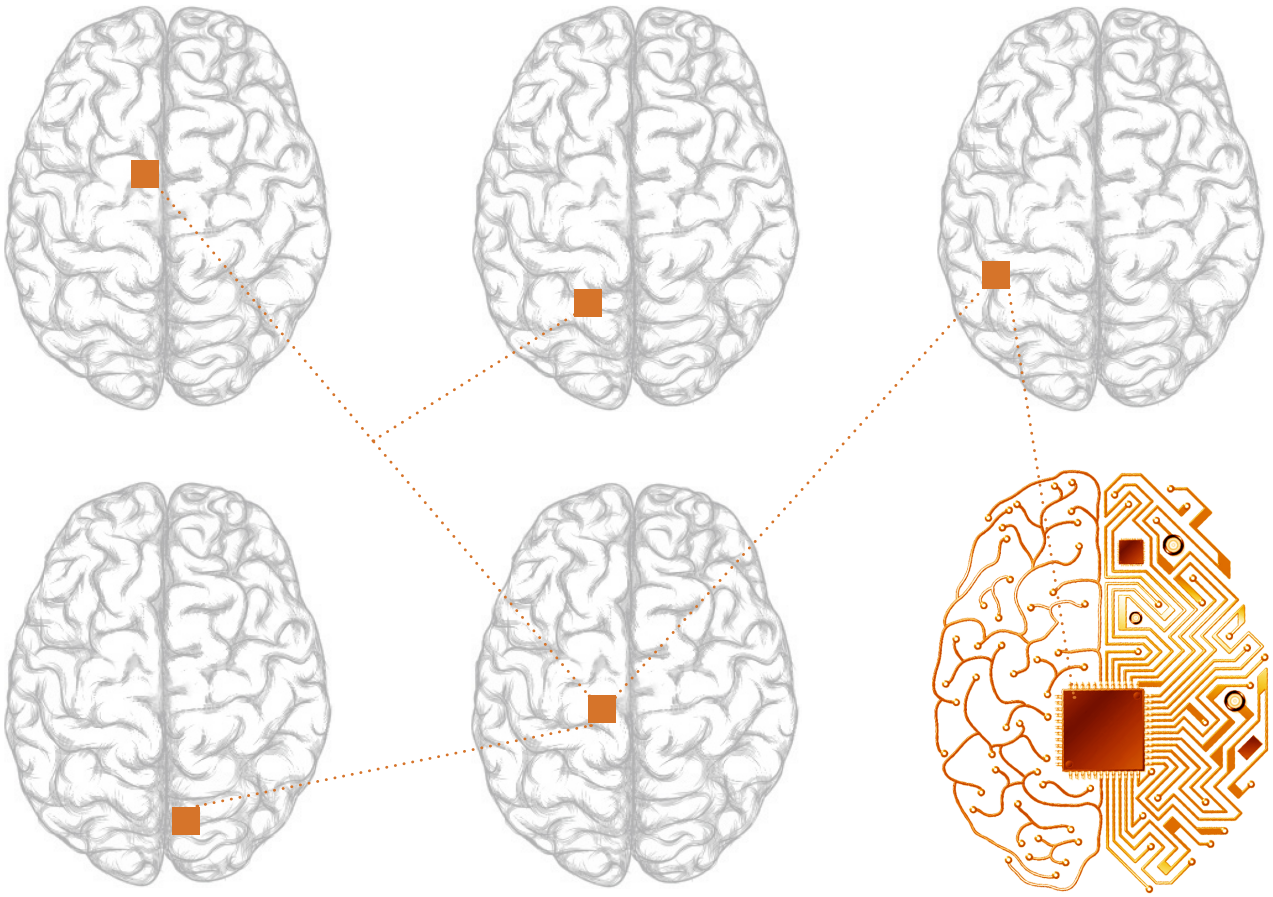


PFIA 2015

Plate-forme Intelligence Artificielle

Rennes

Actes JFSMA

Président CP : Laurent Vercouter

Inria
INVENTEURS DU MONDE NUMÉRIQUE

AfIA
Association française
pour l'Intelligence Artificielle

23es Journées Francophones sur les Systèmes Multi-Agents

Impact des environnements socio-techniques, physiques ou simulés, dans les modèles et développements de systèmes multi-agents

Présentation de la conférence

La plate-forme accueille la vingt-troisième édition des Journées Francophones sur les Systèmes Multi-Agents (JFSMA). Les JFSMA sont le rendez-vous annuel de la communauté des chercheurs francophones travaillant dans le domaine des SMA. Placées sous le signe de l'échange et de l'interdisciplinarité, ces journées sont ouvertes vers d'autres disciplines (intelligence artificielle, vie artificielle, sciences humaines, systèmes distribués ou génie logiciel) et vers les entreprises et les organismes de recherche privés. Cette conférence constitue un lieu privilégié d'échanges scientifiques et technologiques. Les précédentes journées se sont tenues à Toulouse (1993), Grenoble (1994), Chambéry (1995), Port-Camargue (1996), Nice (1997), Nancy (1998), L'Île de la Réunion (1999), Saint-Étienne (2000), Montréal (2001), Lille (2002), Hammamet (2003), Paris (2004), Calais (2005) et Annecy (2006), Carcassonne (2007), Brest (2008), Lyon (2009), Mahdia (2010), Valenciennes (2011), Honfleur (2012), Lille (2013), Loriol-sur-Drôme (2014).

Domaine

Le domaine des Systèmes Multi-Agents a atteint depuis quelques années un stade de maturité qui en fait une approche appropriée au développement d'applications informatiques ouvertes, adaptatives et évolutives. Puisant souvent son inspiration dans la pluridisciplinarité, la modélisation à base d'agents logiciels mis en interaction au sein d'un environnement partagé facilite l'appréhension de systèmes complexes grâce à des outils, modèles ou plate-formes, développés par notre communauté de chercheurs depuis plus de 20 ans, adressant, parfois de manière combinée, les niveaux microscopique et macroscopique.

La prise en compte des dimensions locale et globale d'un système multi-agent, ainsi que de leur influence réciproque, permet de considérer au plus près l'hétérogénéité des acteurs, services et matériels d'une application. C'est un des facteurs du succès de l'approche multi-agent lorsqu'il est nécessaire d'intégrer dès la conception du système des spécificités propres à la présence d'humains, de dispositifs matériels aux ressources limitées ou à des dynamiques d'environnement.

Thématique

Il est de tradition que chaque édition des JFSMA mette en avant une thématique privilégiée. Cette année, l'accent sera mis sur l'impact des environnements socio-techniques, physiques ou simulés, dans les modèles et développements de systèmes multi-agents. L'objectif est d'étudier de quelles manières la conception d'un système multi-agent prend en compte les contraintes inhérentes à l'immersion d'un système multi-agent dans un environnement physique (où les agents sont amenés à interagir avec des dispositifs matériels et/ou des utilisateurs humains), virtuel (en interactions avec d'autres ressources numériques), ou mixtes (dans une approche type web des objets). Selon la nature des environnements considérés, il s'agira de s'intéresser à la manière dont les travaux de recherche gèrent des aspects tels que la perception d'une dynamique propre à l'environnement, la prise d'initiative et l'élicitation des préférences des utilisateurs, la sûreté de fonctionnement et la protection de données sensibles...Il sera également intéressant d'aborder cet impact dans le cadre de simulations multi-agents, en étudiant de quelles manières une simulation peut reproduire, et avec quelles limites, les caractéristiques et contraintes réelles d'environnements socio-techniques simulés.

Les soumissions sur cette thématique privilégiée ont été encouragées. Des contributions sur d'autres travaux originaux en systèmes multi-agents seront aussi considérés, notamment ceux visant à répondre aux 4 grands champs applicatifs des systèmes multi-agents :

- le développement de systèmes informatiques décentralisés où l'approche SMA permet l'intégration flexible et la coopération de logiciels et de services autonome ;
- la résolution collective de problème pour laquelle il s'agit de résoudre de manière distribuée un problème qui se pose globalement à la collectivité d'agents ;
- la simulation de phénomènes complexes où la modélisation multi-agents apporte un cadre conceptuel permettant la représentation et la simulation de systèmes faisant intervenir différentes entités en interaction ;
- le développement de systèmes médiatisés ou utilisateurs humains et agents artificiels interagissent directement ou indirectement, dans le cadre d'activités collectives de type éducatif, culturel ou social.

Conférencier Invité : Jeremy Pitt (Imperial College London)

Biographie

Jeremy Pitt est responsable adjoint du groupe de recherche sur les systèmes intelligents et les réseaux de l'Imperial College de Londres. J. Pitt s'intéresse notamment à l'informatique affective, aux normes et aux sociétés dans les systèmes multi-agents et aux réseaux ad hoc. Impliqué dans de nombreux projets européens de développement des systèmes d'agents intelligents et multi-agents, J. Pitt a également été impliqué dans l'initiative de normalisation de la conception d'agents logiciels (FIPA).

Fair and Sustainable Resource Allocation in Self-Organising Multi-Agent Systems

Many open computing systems, for example grid and cloud computing, and ad hoc networks, such as sensor or vehicular networks, face a similar problem : how to collectivise resources, and distribute them fairly, in the absence of a centralized component. In this talk, we apply the methodology of sociologically-inspired computing, in which the study of human (social) models are formalised as the basis of engineering solutions to technical problems. In this case, we present formal models of Ostrom's design principles for self-governing institutions and Rescher's theory of distributive justice, for defining executable specifications of electronic institutions which support fair and sustainable resource allocation. This work has opened up a programme of research called computational justice : capturing some notions of "correctness" in the outcomes of algorithmic decision-making. However there are various different "aspects" of justice, and some of these – natural, distributive, retributive, procedural and interactional are discussed in the context of supporting self-governance for self-organising multi-agent systems.

Comité de Programme

Présidente du comité de programme

Laurent Vercouter, LITIS, INSA de Rouen

Comité consultatif

— Pierre Chevaillier (Lab-STICC, ENIB, France)
— Rémy Courdier (LIM, Université de la Réunion, France)
— Zahia Guessoum (LIP6, CReSTIC-Reims, France)
— Salima Hassas (LIRIS, Université de Lyon, France)
— Michel Occello (LCIS, Valence, France)

Membres du comité de programme

— Emmanuel Adam (LAMIH, UVHC, Valenciennes, France)
— Frédéric Armetta (LIRIS, Université de Lyon, France)
— Flavien Balbo (MINES Saint-Etienne, France)
— Jean-Paul Barthes (Heudiasyc, UTC, France)
— Grégory Bonnet (GREYC, Université de Caen, France)
— Olivier Boissier (MINES Saint-Etienne, France)
— Valérie Camps (IRIT, Université Paul Sabatier, France)
— Vincent Chevrier (LORIA, Université de Lorraine France)
— Caroline Chopinaud (MASA Group, France)
— Vincent Corruble (LIP6, UPMC, Paris, France)
— Yves Demazeau (LIG, CNRS, France)
— Alexis Drogoul (UMMISCO, IRD Can-Tho, Vietnam)
— Cécile Duchêne (COGIT, IGN, France)
— Amal El Fallah Seghrouchni (LIP6, UPMC, Paris, France)
— Stéphane Galland (SET, UTBM, France)
— Catherine Garbay (LIG, CNRS, France)
— Marie-Pierre Gleizes (IRIT, Université Paul Sabatier, France)
— Emmanuelle Grislin (LAMIH, UVHC, Valenciennes, France)
— Guillaume Hutzler (IBISC, Université d'Evry, France)
— Jean-Paul Jamont (LCIS, Université de Grenoble, France)
— René Mandiau (LAMIH, UVHC, Valenciennes, France)
— Philippe Mathieu (LIFL, Université Lille 1, France)
— Bruno Mermet (GREYC, Université du Havre, France)
— Fabien Michel (LIRMM, Université Montpellier II, France)

— Frédéric Migeon (IRIT, Université Paul Sabatier, France)
— Maxime Morge (LIFL, Université Lille 1, France)
— Bernard Moulin (Université Laval, Canada)
— Jean-Pierre Muller (UPR Green, CIRAD Montpellier, France)
— Denis Payet (LIM, Université de la Réunion, France)
— Sylvie Pesty (LIG, IMAG, Grenoble, France)
— Gauthier Picard (MINES Saint-Etienne, France)
— Sébastien Picault (LIFL, Universite Lille 1, France)
— Suzanne Pinson (LAMSADE, Université Paris-Dauphine, France)
— Joël Quinqueton (LIRMM, Université Montpellier II, France)
— Lilia Rejeb (TIM, FMM Monastir, ISSAT Sousse, Tunisie)
— Nicolas Sabouret (LIMSI, Université Paris-Sud, France)
— Julien Saunier (LITIS, INSA de Rouen, France)
— Olivier Simonin (CITI, INSA Lyon, France)
— Mahdi Zargayouna (IFSTTAR, France)

Coordinateur de l'organisation des journées

Gauthier Picard (MINES Saint-Etienne, France)

Autres relecteurs

— Christine Bourjot (LORIA, Université de Lorraine, France)
— Emmanuel Hermellin (LIRMM, Université Montpellier II, France)
— Elsy Kaddoum (IRIT, Université de Toulouse, France)
— Julien Vaubourg (LORIA, Inria, France)

Table des matières

Présentations longues

Présentations courtes

Démonstrations

Posters de jeunes chercheur(e)s

Conception d'une place de marché pour la vente et la distribution d'énergie dans les *smart grids*

Jesús Cerquides[†] cerquide@iiia.csic.es
Gauthier Picard[⋆] gauthier.picard@emse.fr
Juan A. Rodríguez-Aguilar[†] jar@iiia.csic.es

[†]IIIA-CSIC, Campus UAB, 08193 Cerdanyola, Catalonia, Spain

[⋆]Laboratoire Hubert Curien UMR CNRS 5516 + Institut Henri Fayol, MINES Saint-Etienne, France

Résumé

Cet article introduit un nouveau mécanisme de marché qui permet aux prosommateurs d'échanger de l'électricité tout en satisfaisant les contraintes physiques du réseau. La règle d'allocation de notre marché est mise en œuvre au moyen de RadPro*, un algorithme efficace de programmation dynamique qui évalue en temps polynomial combien d'énergie chaque prosommateur échange ainsi que comment l'énergie doit être distribuée au travers du réseau. Nos résultats empiriques montrent que* RadPro *surclasse de manière significative CPLEX et Gurobi en temps lors du calcul de l'allocation optimale dans des réseaux acycliques. De plus, la gestion par envoi de messages de* RadPro *offre la possibilité d'exécuter notre marché d'une manière décentralisée (pair-à-pair).*

Mots-clés : *smart grid, marché de l'énergie, prosommateurs,* RadPro

Abstract

This paper introduces a novel market that allows prosumers to trade electricity while satisfying the constraints of the grid. Our market's allocation rule is implemented by means of the so-called RadPro*, an efficient dynamic programming algorithm that assesses in polynomial time how much energy each prosumer trades as well as how energy must be distributed throughout the grid. Our empirical results show that* RadPro *significantly outperforms both CPLEX and Gurobi in solving time when computing the optimal allocation over acyclic networks. Furthermore, the message-passing nature of* RadPro *offers the possibility of running our market in a decentralized (peer-to-peer) manner.*

Keywords: *smart grid, energy market, prosumers,* RadPro

1 Introduction

Notre modèle centralisé de production et de transmission de l'énergie est une source d'énormes gâchis [6]. Comme les stations de production sont généralement éloignées des centres de demande, la plupart de la chaleur produite est non utilisée, mais ventilée dans des cheminées ou rejetée dans des rivières. Des pertes additionnelles surviennent lors du transport de l'électricité le long des câbles des systèmes de transmission et de distribution [6, 23]. Favoriser une génération décentralisée de l'énergie devrait réduire les inefficacités de génération et de distribution et faciliter des contributions accrues des énergies renouvelables [23]. Ce nouveau modèle est destiné à être pris en charge par les *smart grids* (ou *SG*).

Une SG est un réseau d'électricité qui peut intégrer les actions de tous les utilisateurs connectés —générateurs, consommateurs— afin de livrer efficacement (en terme d'économie, de sécurité) de l'électricité. Dans une SG, le consommateur peut être un individu ou un foyer, mais aussi une communauté ou une PME. Dans sa forme la plus générale, une SG est peuplée de *prosommateurs* capables à la fois de production et de consommation d'énergie. Ainsi, les SG jouent un rôle central dans l'intégration de tous ces usagers au moyen de l'adoption d'un système qui satisfait un certain nombre d'objectifs sociétaux. Parmi ces objectifs, il y a celui de la fixation des prix du marché de l'électricité en tenant compte des contraintes structurelles de la SG. Un défi majeur de la recherche au cœur de plusieurs feuilles de route pour les SG [3, 4] est donc la conception de marchés pour prosommateurs qui prennent en compte des contraintes du réseau de distribution, ce qui permettra aux prosommateurs de commercer directement sur la SG [8]. D'après [17], les opérations de marché impliqueront un grand nombre de prosommateurs hétérogènes, distribués à travers le réseau (plus près du point d'utilisation de l'électricité), et un commerce d'énergie négociée en quantité plus faible qu'aujourd'hui. La distribution de l'électricité emploie l'un des trois types de topologies de réseaux : radial, en anneau, et interconnecté [5, 7, 21]. Les réseaux radiaux sont acycliques, mais comme on l'observe dans [11], bien que les anneaux et les réseaux interconnectés contiennent des cycles, ils

sont configurés comme des réseaux acycliques au moyen de commutateurs [7, 21].

La vision des SG a stimulé de nombreuses recherches sur la conception de marchés et d'agents commerciaux. L'état de l'art a surtout considéré l'emploi de différents types de ventes aux enchères. Ainsi, l'échange d'énergie basé sur le marché est généralement traité dans la littérature par un ensemble de prosommateurs qui participent à une double vente aux enchères où l'énergie est négociée sur une base journalière [8, 9, 10, 12, 14, 19]. Les exceptions à cette approche commune envisagent des ventes aux enchères multi-unitaires adaptées [22] et des enchères inversées combinatoires simultanées [15] pour faire correspondre la demande à l'offre. Dans la limite de nos connaissances, aucun des mécanismes de marché utilisés dans la littérature à ce jour ne tient compte des contraintes physique de la SG. Ainsi, la compensation du marché se produit sans prendre en compte, par exemple, le fait que la transmission d'énergie est réalisée le long des réseaux de distribution aux capacités limitées [21]. Par conséquent, le commerce et la distribution sont considérés comme des activités découplées.

L'objectif de cet article est donc de concevoir un nouveau marché qui permet aux prosommateurs de faire le commerce de l'électricité dans une SG tout en satisfaisant les contraintes de distribution de la SG. Plus précisément, (i) nous formalisons le problème d'allocation d'énergie (EAP) comme le problème de la décision de la quantité d'énergie échangée entre chaque prosommateur de telle sorte que le bénéfice global soit maximisé tout en respectant les contraintes physiques et les préférences des utilisateurs. Résoudre l'EAP revient à compenser (ou équilibrer) notre marché de prosommation. (ii) Nous proposons un nouvel algorithme de programmation dynamique, par envoi de messages, pour résoudre des instances acycliques d'EAP, appelé Radial Energy Network Algorithm for Prosumer Market (RadPro).

Cet article est structurée comme suit. La section 2 présente l'algorithme Acyclic-Solving sur lequel est basé RadPro. La section 3 définit formellement la règle d'allocation que nous proposons pour équilibrer les marchés d'énergie pour prosommateurs. Ensuite, la section 4 introduit RadPro. La section 5 analyse empiriquement RadPro et la section 6 conclut et trace les perspectives pour nos travaux à venir.

2 Fondements algorithmiques

Cette section introduit la notion de problèmes d'optimisation sous contraintes puis présente Acyclic-Solving, l'algorithme de programmation dynamique à la base de RadPro.

Soit $X = \langle x_1, \ldots, x_n \rangle$ une séquence de variables, avec chaque variable x_j prenant ses valeurs dans un ensemble fini $\mathcal{D}_j$, son *domaine*. Le domaine joint $\mathcal{D}_X$ est le produit cartésien des domaines de toutes les variables. Nous notons $\mathbf{x}_j$ un état possible de x_j, c'est-à-dire $\mathbf{x}_j \in \mathcal{D}_j$. Etant donné un ensemble de variables $Y \subseteq X$, un tuple $\mathbf{X}_Y$ affecte un état possible à chaque variable de Y, c'est-à-dire $\mathbf{X}_Y \in \mathcal{D}_Y$. Une fonction d'utilité f de portée Y est une fonction $f : \mathcal{D}_Y \to \mathbb{R}$. Nous notons $s(f)$ la portée de f. La somme de deux fonctions d'utilité f et f' de portée Y et Z respectivement est une nouvelle fonction d'utilité $h = f + f'$ de portée $Y \cup Z$, de telle sorte que $h(\mathbf{T}) = f(\mathbf{T}^{\downarrow Y}) + f'(\mathbf{T}^{\downarrow Z})$ où $\mathbf{T}^{\downarrow Y}$ est la projection du tuple $\mathbf{T}$ à la portée Y. La projection d'une fonction d'utilité f de portée Y à la portée Z, est $f^{\downarrow Z}(\mathbf{Z}) = \max\{f(\mathbf{Y}) \mid \mathbf{Y} \in \mathcal{D}_Y, \mathbf{Y}^{\downarrow Z} = \mathbf{Z}\}$.

Formellement, un COP (*Constraint Optimisation Problem*) est un problème d'optimisation dont l'entrée est un tuple $\langle X, \mathcal{D}, F \rangle$, où F est un ensemble de fonctions d'utilité et dont l'objectif est de trouver le tuple $\mathbf{X}^*$ qui maximise la somme $g = \sum_{f \in F} f$ des fonctions d'utilité, c'est-à-dire de trouver $\mathbf{X}^* = \text{argmax}_{\mathbf{X}}\, g(\mathbf{X})$. Etant donné un COP, le *réseau de contraintes* est défini comme le graphe ayant un nœud pour chaque contrainte et un arc entre deux nœud si leur portée partage au moins une variable. Lorsque le réseau de contraintes est acyclique, l'algorithme Acyclic-Solving peut résoudre le COP efficacement [2, Chapitre 9, page 248], si toutefois la portée des fonctions d'utilité est limitée. Acyclic-Solving sélectionne un nœud comme étant la racine. Chaque nœud j (non racine) se voit attribué un parent p_j et une fonction d'utilité f_j (représentant sa propre vue du problème). La portée d'un nœud j est la portée de f_j. Le séparateur s_j entre j et p_j est l'intersection de leurs portées (l'ensemble des variables partagées par ces deux agents).

Acyclic-Solving, présenté dans l'algorithme 1, s'exécute en deux phases : (1) les coûts sont envoyés des feuilles jusqu'à la racine ; (2) les affectations optimales sont décidées et communiquée de manière descendante. En fait, il détermine la solution optimale en utilisant le principe suivant. Premièrement, chaque feuille commence par envoyer sa propre utilité à son parent. Lorsque qu'un nœud a reçu les messages de tous ses enfants, il les combine avec sa propre utilité pour produire une utilité agrégée de son propre sous-arbre, puis l'envoie à son parent. Une fois que la racine a reçu les messages de tous ses enfants, il évalue l'utilité agrégée du problème entier, puis décide de la meilleure affectation (tuple

Algorithme 1 : ACYCLIC-SOLVING (adapté de [2])

```
// chaque nœud j de l'arbre exécute
1 pour chaque enfant k faire recevoir un message μ_{k→j}
2 si j n'est pas racine alors
3     évaluer le message μ_{j→p_j} = (f_j + Σ_k μ_{k→j})^{↓s_j}
4     envoyer le message μ_{j→p_j} à son parent p_j
5     recevoir le message X*_{s_j} de son parent p_j
6 évaluer la meilleure affectation
  X*_j = argmax_{X_j}(f_j + Σ_k μ_{k→j})(X*_{s_j}, X_j)
7 pour chaque enfant k faire envoyer un X*^{↓s_k} à k
```

de coût maximal) pour ses propres variables. Enfin, il envoie cette affectation à ses enfants, qui évaluent à leur tour la meilleure affectation et l'envoient dans leur sous-arbre. Après l'exécution de l'algorithme, chaque agent connaît les valeurs optimales pour les variables de sa portée.

Comme ACYCLIC-SOLVING peut être exprimé comme un algorithme par envoi de messages, il peut aisément être appliqué à des COP distribués [1], où il peut être vu comme un cas particulier de DPOP [16] ou d'Action-GDL [18]. La complexité calculatoire de chaque agent dans ACYCLIC-SOLVING est exponentielle selon le nombre de variables dans sa portée —ce qui est fondamentalement dû au calcul des messages de la ligne 3. Habituellement, une technique de *bookkeeping* est utilisée lors de ce processus pour rendre l'évaluation de la meilleure affectation de la ligne 6 peu coûteuse. La complexité en temps de chaque agent j est exponentielle selon le nombre de variables partagées avec son parent (taille du séparateur s_j). Comme le nombres de messages est linéaire, la complexité globale est alors exponentielle.

3 Problème d'allocation d'énergie

Cette section fournit un modèle mathématique simple pour le marché énergétique dans le réseau de prosommateurs, et la règle d'allocation proposée pour ce marché. Nous illustrons le modèle des prosommateurs et le modèle du réseau énergétique que nous considérons à l'aide d'un exemple de scénario d'échange d'énergie. Ensuite, nous présentons la règle d'affectation pour ce marché comme la solution d'un problème d'optimisation : le problème d'allocation d'énergie (EAP, *Energy Allocation Problem*).

3.1 Scénario d'échange d'énergie

La figure 1 montre un exemple d'un scénario d'échange d'énergie impliquant quatre prosommateurs (représentés par des cercles). Chaque arc connectant deux prosommateurs signifie qu'ils sont physiquement connectés. De plus, chaque lien est étiqueté avec sa capacité à transporter de l'énergie. Par exemple, le prosommateur 1 est connecté au prosommateur 2, et leur lien peut transporter jusqu'à 2 unités d'énergie (e.g. $2kW$). Chaque prosommateur peut offrir d'acheter, de vendre ou bien de transmettre de l'énergie. L'offre de chaque prosommateur est représentée par une table à côté de chaque prosommateur, où chaque entrée est une paire (unités,prix). Par convention, une offre de vente est exprimée par un nombre négatif d'unités et un prix négatif, alors qu'une offre d'achat est traduite par un nombre positif d'unité et de prix. Par exemple, le prosommateur 4 offre (entre autre) : d'acheter 2 unités d'énergie et de payer 1.75c€, de vendre 3 unités pour 11c€, et de transmettre de l'énergie gratuitement (0 unités pour 0c€). Dans la figure 1, nous pouvons observer que le prosommateur 1 ne fait que vendre de l'énergie, et le prosommateur 2 ne fait qu'en vendre, alors que les prosommateurs 3 et 4 peuvent soit vendre soit acheter.

Remarquons que les entrées d'une table représentent des offres mutuellement exclusives. Ainsi, par exemple, l'offre du prosommateur 4 indique qu'il peut soit acheter 2 unités, soit vendre 3 unités, mais pas les deux.

3.2 Définition du problème

Le problème auquel les prosommateurs font face dans la figure 1 est de décider combien d'énergie échanger et avec qui de telle sorte que le bénéfice global soit maximisé et que les contraintes de capacité du réseau soient respectées. Ceci signifie que : (i) chaque prosommateur doit choisir une seule offre dans sa table (quelle quantité) ; et (ii) chaque paire de prosommateurs connectés par un lien doivent se mettre d'accord sur la quantité à transférer et la direction de ce transfert (avec qui). Dans ce qui suit, nous exprimons ce problème comme un problème d'optimisation.

D'après la figure 1, le réseau connectant un ensemble de prosommateurs P peut être modélisé comme un graphe (P, E), où les nœuds représentent les prosommateurs et les arcs de E représentent les paires de prosommateurs. Un arc $\{i, j\} \in E$ signifie que le prosommateurs i et j sont physiquement connectés pour échanger de l'énergie. Lorsque $i < j$ on dit que i est un voisin entrant (de l'ensemble $in(j)$) de j et que j est un voisin sortant (de l'ensemble $out(j)$) de i.

Chaque prosommateur j exprime ses offres d'achat et de vente au moyen d'une *fonction d'offre* $o_j : \mathbb{Z} \to \mathbb{R} \cup \{-\infty\}$. Par exemple, $o_j(3) = 2$ indique que le prosommateur j souhaite acheter 3

1. où chaque agent est affecté à une seule contrainte et le réseau de contraintes est acyclique.

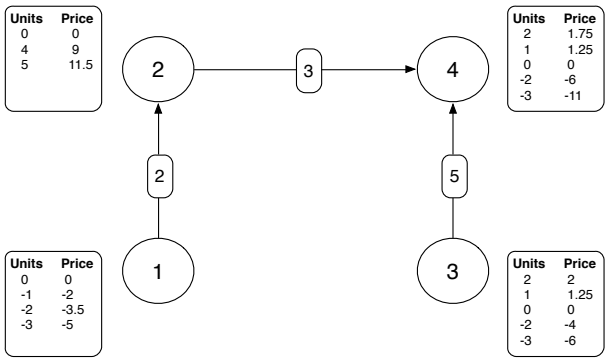


FIGURE 1 – Un scénario d'échange d'énergie : quatres prosommateurs (cercles) publient des offres (tables) dans un réseau aux capacités limitées (arcs).

unités à 2c€, alors que $o_j(-4) = -2$ signifie qu'il souhaite vendre 4 unités pour 2c€. De plus, nous permettons à un prosommateur d'exprimer qu'il souhaite laisser l'énergie transiter par son nœud en vendant autant d'unités qu'il achète. Nous supposons que $o_j(0) \neq -\infty$ pour tout $j \in P$. Remarquons que les fonctions d'offres capturent les contraintes des prosommateurs. Pour communiquer ses offres, chaque prosommateur envoie une table comme celle de la figure 1 rendant explicite ses états possibles et leurs valeurs. Si une offre pour k unités n'apparaît pas dans la table, ceci signifie qu'un tel état est non admissible pour le prosommateur et qu'ainsi sa valeur $o_j(k)$ est $-\infty$. Formellement, une fonction d'offre est une *valuation*.

Définition 1. *Une* valuation α *est une fonction* $\alpha : \mathbb{Z} \to \mathbb{R} \cup \{-\infty\}$. *Le domaine de* α *est décomposé en deux parties : le* domaine à valeurs finies $FVD(\alpha)$ *est le sous-ensemble de* $\mathbb{Z}$ *dans lequel* α *prend ses valeurs finies. La taille* n_α *d'une valuation est le nombre d'éléments de* $FVD(\alpha)$. *Nous définissons la valuation vide* $\diamond$ *comme étant celle qui associe* 0 *à* 0 *et* $-\infty$ *à tout autre élément.*

Au-delà des offres, nous considérons également que le réseau d'énergie est physiquement contraint par la capacité des connexions entre prosommateurs. Nous noterons c_{ij} la capacité du lien $\{i, j\}$, à savoir le nombre maximum d'unités échangeables entre les prosommateurs i et j. Une allocation (ou affectation) spécifie le nombre d'unités d'énergie que chaque prosommateur échange avec ses voisins. Nous formaliserons une allocation par un ensemble de variables $Y = \{y_{ij} \mid i \in P, j \in out(i)\}$, où $\mathbf{y}_{ij}$ représente le nombre d'unité que i vend à j et qui est bornée par la capacité limite c_{ij}. Le domaine de la variable y_{ij} est ainsi $D_{ij} = [-c_{ij} .. c_{ij}]$. Par conséquent, si $\mathbf{y}_{ij}$ prend la valeur k strictement positive, ceci signifie que i vend k unités d'énergie à j. Si-non, si $\mathbf{y}_{ij}$ prend une valeur strictement négative $-k$, on dit que i achète k unités à j. Et donc, $\mathbf{y}_{ij}$ représente un échange du point de vue de i.

Maintenant, nous souhaitons évaluer la valeur d'une affectation donnée. Avant cela, nous définissons la valeur locale d'une affectation pour un unique prosommateur. Nous devons évaluer la quantité d'énergie qu'un prosommateur acquiert et vend suivant une affectation $\mathbf{Y}$. Le prosommateur j considérera uniquement sa vue locale de l'affectation, représentée par $\mathbf{Y}_j = \mathbf{y}_{.j} \cup \mathbf{y}_{j.}$. Nous pouvons évaluer l'*équilibre net d'énergie* pour le prosommateur j comme

$$net(\mathbf{Y}_j) = \sum_{i \in in(j)} \mathbf{y}_{ij} - \sum_{k \in out(j)} \mathbf{y}_{jk}, \qquad (1)$$

où les $\mathbf{y}_{ij}$ et $\mathbf{y}_{jk}$ sont agrégées avec différents signes car j prend le rôle de vendeur dans $\mathbf{y}_{ij}$ et d'acheteur dans $\mathbf{y}_{jk}$. Ainsi, la valeur locale v_j de l'affectation $\mathbf{Y}$ pour j peut être évaluée comme la valeur de son équilibre d'énergie net au moyen de sa fonction d'offre

$$v_j(\mathbf{Y}_j) = o_j(net(\mathbf{Y}_j)). \qquad (2)$$

Par conséquent, la valeur d'une affectation $\mathbf{Y}$ peut être obtenue en ajoutant les valeurs locales des affectations des prosommateurs :

$$Value(\mathbf{Y}) = \sum_{i \in P} v_j(\mathbf{Y}_j). \qquad (3)$$

Maintenant, nous sommes prêts à définir le problème allocation d'énergie comme celui qui consiste à trouver la valeur maximale qui satisfait les capacités du réseau.

Problème 1. *Etant donnés un ensemble de prosommateurs* P, *leurs offres* $\{o_j | j \in P\}$, *et un graphe* E *où chaque arc est étiqueté avec sa capacité* c_{ij}, *le problème d'allocation d'énergie (EAP) revient à trouver une affectation* $\mathbf{Y}$ *qui maximise* $Value(\mathbf{Y})$. *Lorsque le graphe* E *est acyclique, nous disons que l'EAP est acyclique.*

Examinons l'exemple de la figure 1. Lors sa résolution, nous obtenons l'affectation de variables présentée dans la figure 2. La solution indique que le prosommateur 1 transfert 2 unités au prosommateur 2 ($y_{12} = 2$), le prosommateur 2 reçoit également 3 unités du prosommateur 4 ($y_{24} = -3$), et le prosommateur 3 transfert 3 unités au prosommateur 4. Dans chaque table d'offres, nous soulignons l'offre choisie par chaque prosommateur. Pour chaque prosommateur, le premier nombre souligné correspond à l'équilibre d'énergie net (équation 1), alors que le prix souligné correspond à la valeur locale de l'affectation (équation 2). Ainsi, l'affectation qui maximise l'équation 3 a la valeur 2.

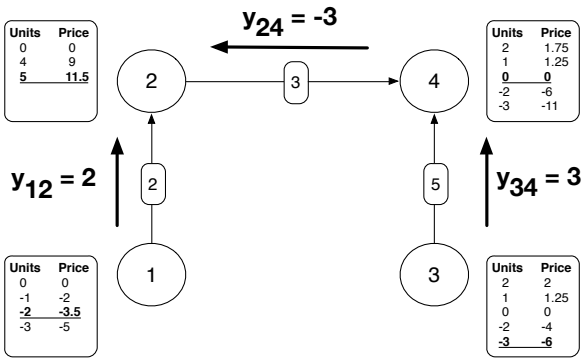


FIGURE 2 – Solution à l'EAP du scénario d'échange d'énergie.

Notons que le prosommateur 2 obtient 5 unités en agrégeant les unités reçues des prosommateurs 1 et 4. Cependant le prosommateur 4 ne vend rien au prosommateur 2. Le rôle du prosommateur 4 est de *relayer* au prosommateur 2 l'énergie transférée du prosommateur 3, qui est celui qui vend l'énergie. En général, notre modèle considère que chaque prosommateur (i) agrège l'énergie reçue de ses voisins lors d'achats ; ou bien (ii) partage et distribue de l'énergie à ses voisins lors de ventes ; ou bien (iii) relaie de l'énergie de telle sorte que les autres prosommateurs puissent satisfaire leur demande.

4 Envoi de messages pour EAP acyclique

Cette section présente RADPRO, un nouvel algorithme pour EAP acyclique basé sur des envois de messages, et qui peut ainsi être facilement implanté de manière distribuée. Comme expliqué dans la section 2, l'algorithme ACYCLIC-SOLVING repose sur des envois de messages pour résoudre des COP acycliques. Le but de cette section est double. Premièrement, la section 4.1 détaille comment employer ACYCLIC-SOLVING directement pour résoudre EAP. Deuxièmement, comme l'évaluation des messages dans ACYCLIC-SOLVING est de complexité exponentielle, la section 4.2 détaille la théorie mathématique qui sous-tend l'évaluation des messages d'ACYCLIC-SOLVING en complexité polynomiale. En raison de la limitation d'espace, nous formulons les lemmes et théorèmes, et laissons les preuves dans un rapport technique [1]. Enfin, la section 4.3 introduit RADPRO et compare les complexités en nombre de messages et en temps avec celles d'ACYCLIC-SOLVING.

4.1 Solution directe

Notons que la valeur d'une allocation, d'après l'équation 3 est la somme des fonctions utilité, une par prosommateur. Ainsi, on peut directement faire correspondre l'EAP à un COP. Lorsque l'EAP est acyclique, le réseau de contrainte l'est également et on peut ainsi directement appliquer ACYCLIC-SOLVING (voir section 2). Notons maintenant que le séparateur s_j entre chaque nœud j et son parent p_j correspond soit avec l'unique variable y_{ip_j} si p_j est un voisin sortant de j ou avec y_{p_jj} si p_j est un voisin entrant de j. Dans tous les cas, la taille du message $\mu_{j\to p_j}$ est le nombre d'états possibles pour le lien entre j et p_j, et le message peut être interprété comme la communication de l'utilité de chacun de ces états au réseau composé des prosommateurs dans le sous-arbre de racine j. Ainsi, la quantité d'information communiquée nécessaire à l'exécution de ACYCLIC-SOLVING est très faible.

L'expression du message de j à son parent peut être obtenue en particularisant l'équation générale d'ACYCLIC-SOLVING (équation **??**). Lorsque le parent p_j est un voisin sortant, j envoie le message suivant :

$$\mu_{j\to p_j}(\mathbf{y}_{jp_j}) = \max_{\mathbf{Y}_{j-p_j}} v_j(\mathbf{y}_{jp_j}, \mathbf{Y}_{j-p_j}) + \sum_{k\in out(j)\setminus\{p_j\}} \mu_{j\to k}(\mathbf{y}_{jk}) + \sum_{i\in in(j)} \mu_{i\to j}(\mathbf{y}_{ij}) \quad (4)$$

et lorsque le parent p_j est un voisin entrant, j envoie le message suivant :

$$\mu_{j\to p_j}(\mathbf{y}_{p_jj}) = \max_{\mathbf{Y}_{j-p_j}} v_j(\mathbf{y}_{p_jj}, \mathbf{Y}_{j-p_j}) + \sum_{k\in out(j)} \mu_{j\to k}(\mathbf{y}_{jk}) + \sum_{i\in in(j)\setminus\{p_j\}} \mu_{i\to j}(\mathbf{y}_{ij}) \quad (5)$$

où $\mathbf{Y}_{j-p_j}$ représente une affectation de valeurs à chacune des variables de Y_j exceptée y_{jp_j} (ou y_{p_jj}). En accord avec la section 2, l'évaluation des messages prend un temps exponentiel selon le nombre de variables dans la portée de la contrainte, qui dans ce cas correspond au nombre de voisins du prosommateur. Le coût de calcul pour un prosommateur j pour évaluer le message destiné à son parent est en $\mathcal{O}((2C_j+1)^{N_j})$, où C_j est la capacité du lien le plus puissant entre j et un agent voisin, et N_j est le nombre de voisins de j. Comme expliqué en section 2, il est possible d'utiliser du *bookkeeping* durant l'évaluation des messages pour réduire la complexité du calcul de la meilleure affectation (équation **??**) en temps constant. Ainsi ACYCLIC-SOLVING sera efficace dès lors que le degré des nœuds du réseau est faible.

Ceci peut se produire dans des scénarios ruraux [13], cependant dans des zones urbaines [20] il a été analysé que le degré des nœuds peut suivre une distribution géométrique avec certains atteignant des degrés supérieurs à 35. Ceci entrave l'usage direct d'Acyclic-Solving dans de tels scénarios. Des approches par programmation dynamique similaires pour la réduction des émissions de CO_2 dans les réseaux électriques ont été qualifiée de fonctionnelle avec des degrés maximums égaux à 4 [11].

4.2 Calcul efficace des messages

L'évaluation des messages par les équations 4 et 5 prend un temps exponentiel selon le nombre de voisins du prosommateur. Ainsi, pour des réseaux denses, cela peut fortement limiter l'applicabilité de l'algorithme. Pour palier ce problème, nous introduisons une algèbre des valuations dont le prosommateur peut prendre avantage lors de l'évaluation de ses messages.

Objets et opérations des *valuations*. Notre objectif est de construire une algèbre nous permettant d'accélérer l'évaluation des messages. L'objet mathématique principal de l'algèbre, appelé *valuation*, a déjà été présenté dans la définition 1. A la fois les offres o_j et les messages $\mu_{j\to p_j}$ peuvent directement être représentés par des valuations. Ces dernières peuvent à leur tour être implantées au moyen de tables de hachage. Dans la suite, nous introduisons les opérations nécessaires pour l'évaluation des messages. Tout d'abord considérons le cas où un prosommateur j avec un seul voisin entrant p_j. Le message à envoyer d'après l'équation 5 sera tout simplement une copie de l'offre o_j. Cependant, comme les état possibles de $y_{p_j j}$ sont limités par la capacité du lien, nous avons besoin d'une opération permettant de supprimer de la valuation o_j les valeurs hors de ces limites. Cette opération est appelée *restriction*.

Définition 2. *Etant donnés une valuation α, et un sous-ensemble $D \subseteq \mathbb{Z}$ on définit $\alpha[D]$, la* restriction de α à D *comme*

$$\alpha[D](k) = \begin{cases} \alpha(k) & k \in D \\ -\infty & \textit{sinon.} \end{cases}$$

Ainsi, le message d'un prosommateur j à son unique voisin entrant p_j peut être écrit comme $\mu_{j\to p_j} = o_j[D_{p_j j}]$.

Considérons maintenant le cas d'un prosommateur j ayant un unique voisin sortant p_j. Ici, l'équation d'équilibre de flux pour j montre que $net(\mathbf{y}_{jp_j}) = -\mathbf{y}_{jp_j}$. Ainsi, le message à envoyer, d'après l'équation 4 est la version symétrique d'o_j, restreinte à la capacité du lien. Nous appelons *complément* l'opération permettant d'obtenir la version symétrique d'une valuation.

Définition 3. *Etant donnée une valuation α, on définit $\overline{\alpha}$, le* complément de α *comme*

$$\overline{\alpha}(k) = \alpha(-k)$$

Ainsi, le message de j à son unique voisin entrant p_j peut s'écrire comme $\mu_{j\to p_j} = \overline{o_j}[D_{p_j j}]$.

Enfin, prenons le cas d'un prosommateur j ayant son parent p_j comme voisin entrant et un voisin sortant k. Ici l'équation 5 est réduite à

$$\mu_{j\to p_j}(\mathbf{y}_{p_j j}) = \max_{\mathbf{y}_{jk}} \left(v_j(\mathbf{y}_{p_j j}, \mathbf{y}_{jk}) + \mu_{j\to k}(\mathbf{y}_{jk}) \right) = \max_{\mathbf{y}_{jk}} \left(o_j(\mathbf{y}_{p_j j} - \mathbf{y}_{jk}) + \mu_{j\to k}(\mathbf{y}_{jk}) \right) \quad (6)$$

Notons que pour certaines des affectations $\mathbf{y}_{p_j j}, \mathbf{y}_{jk}$, il peut arriver que o_j soit non défini pour l'équilibre énergétique $\mathbf{y}_{p_j j} - \mathbf{y}_{jk}$. On peut éviter cette situation en introduisant une nouvelle opération sur les valuations, l'*agrégation*.

Définition 4. *Etant données deux valuations α et β, on définit $\alpha \cdot \beta$, l'*agrégation de α et β *comme*

$$(\alpha \cdot \beta)(k) = \max_{\substack{i,j \\ k=i+j}} \alpha(i) + \beta(j) \quad (7)$$

Maintenant le message de j, lorsque son parent p_j est un voisin entrant et k est un voisin sortant, peut être écrit comme $\mu_{j\to p_j} = (o_j \cdot \mu_{k\to j})[D_{p_j j}]$.

Structure de l'algèbre des *valuations*. Maintenant que les valuations et leurs opérations ont été définies, nous présentons la structure algébrique que nous avons créée, sous la forme de deux lemmes. Concentrons-nous d'abord sur l'agrégation.

Lemme 1. *L'ensemble de valuations muni de l'opération d'agrégation forme un monoïde commutatif, $\diamond$ étant l'élément neutre. Ceci signifie que pour toutes valuations α, β, et γ nous avons*

$$\alpha \cdot \beta = \beta \cdot \alpha$$
$$(\alpha \cdot \beta) \cdot \gamma = \alpha \cdot (\beta \cdot \gamma)$$
$$\alpha \cdot \diamond = \alpha$$

D'un point de vue mathématique le lemme 1 nous permet d'écrire des expressions telles que $\alpha \cdot \beta \cdot \gamma$ qui n'établit aucun ordre spécifique dans lequel les agrégations doivent être effectuées. On peut ainsi étendre la définition d'agrégation de paires de valuations à des ensembles de valuations. Nous définissons cette agrégation n-aire de valuations dans $A = \{\alpha_1, \ldots, \alpha_n\}$

Algorithme 2 : Agrégation de deux valuations

```
γ ← ◇
pour i ∈ FVD(α) faire
    pour j ∈ FVD(β) faire
        γ(i + j) ← max(γ(i + j), α(i) + β(j))
```

comme $(\prod_{i=1}^{n} \alpha_i)(k) = \max_{\substack{j_1,\ldots,j_n \\ \sum_{i=1}^{n} j_i = k}} \sum_{i=1}^{n} \alpha_i(j_i)$. Remarquons que nous avons $\prod_{i=1}^{n} \alpha_i = \alpha_1 \cdot \ldots \cdot \alpha_n$. Ainsi, l'agrégation n-aire d'un ensemble fini de valuations peut être évalué en utilisant uniquement des agrégations binaires de la définition 4. Du point de vue de la complexité, le lemme 1 nous permet de grouper les opérations d'agrégation de la manière qui nous arrange. Ainsi, il est possible de suivre un ordre séquentiel, ou d'évaluer les messages de manière arborescente dans le cas où cela peut être plus efficace.

Maintenant intéressons-nous à la caractérisation des relations entre les différentes opérations.

Lemme 2. *Le complément définit un automorphisme involutif du monoïde des valuations, c'est-à-dire que pour toutes valuations α et β nous avons*

$\overline{\alpha \cdot \beta} = \overline{\alpha} \cdot \overline{\beta}$
$\overline{\overline{\alpha}} = \alpha$
$\overline{\diamond} = \diamond$
$\bar{\cdot} : \Phi \to \Phi$ *est une association bijective*

De plus, $\overline{\alpha[D]} = (\overline{\alpha})[-D]$ *où* $-D = \{-x | x \in D\}$.

Certaines de ces propriétés vont se montrer très pratiques pour travailler sur des expressions impliquant l'agrégation, le complément et la restriction.

Efficacité de l'agrégation. Intéressons-nous maintenant au coût de calcul de l'agrégation d'un ensemble de valuations. L'algorithme 2 peut être utilisé pour implanter l'agrégation, avec le résultat suivant :

Lemme 3. *L'agrégation de deux valuations α et β peut être effectuée en temps $\mathcal{O}(n_\alpha \times n_\beta)$. De plus, $n_{\alpha\cdot\beta} \leq n_\alpha \times n_\beta$.*

De là, nous pouvons montrer que l'évaluation de l'agrégation de M valuations de taille N peut être effectuée en temps $O(N^M)$.

Notre objectif était d'améliorer les calculs d'agrégation des messages. Cependant, nous pouvons observer que dans le cas le plus général, l'agrégation de M messages prend un temps exponentiel. Prenons maintenant avantage d'une des particularités des messages à agréger pour nous permettre de calculer l'agrégation en temps polynomial. Remarquons que le FVD d'un message $\mu_{i\to j}$ est toujours inclus dans un intervalle réduit $D_{ij} = [-c_{ij}..c_{ij}]$ autour de 0. Nous disons qu'une valuation α est *de capacité restreinte C* si $FVD(\alpha) \subseteq [-C..C]$. Pour des valuations de capacité restreinte nous obtenons le résultat suivant.

Lemme 4. *L'agrégation de deux valuations α et β de capacité restreinte C peut être faite en temps $\mathcal{O}(C^2)$. De plus, $\alpha\cdot\beta$ est une valuation de capacité restreinte $2C$.*

Remarquons que, tandis que pour les valuations en général la taille de la valuation agrégée grandit de manière quadratique, pour les valuations à capacité restreinte, elle grandit de manière linéaire. Ceci devient fondamental pour prouver le résultat suivant, qui nous dit qu'il est possible d'agréger des valuations en temps polynomial dès lors qu'elles sont de capacité restreinte.

Lemme 5. *Etant donné un ensemble A de M valuations à capacité restreinte C on peut évaluer son agrégation $\prod_{\alpha\in A} \alpha$ en temps $\mathcal{O}(M^2C^2)$.*

La preuve utilise la propriété d'associativité pour diviser l'agrégation en deux agrégations plus petite, chacune ayant la moitié des valuations, pour évaluer ces agrégations plus réduites et finalement agréger les résultats.

Usage de l'algèbre pour l'évaluation des messages. Le théorème suivant montre qu'il est possible de fournir une expression des messages d'Acyclic-Solving des équations 4 et 5 en termes d'opérations de l'algèbre des valuations et ceci en temps polynômial.

Théorème 1. *Le message de j à son parent et voisin entrant p_j défini dans l'équation 5 peut être évalué comme*

$$\mu_{j\to p_j} = \left(o_j \cdot \prod_{k\in out(j)} \mu_{j\to k} \cdot \prod_{i\in in(j)\setminus\{p_j\}} \overline{\mu_{i\to j}} \right)[D_{p_j j}] \tag{8}$$

et le message de j à son parent et voisin sortant p_j défini dans l'équation 4 peut être évalué comme

$$\mu_{j\to p_j} = \left(\overline{o_j} \cdot \prod_{k\in out(j)\setminus\{p_j\}} \overline{\mu_{j\to k}} \cdot \prod_{i\in in(j)} \mu_{i\to j} \right)[-D_{jp_j}]. \tag{9}$$

De plus la complexité en temps de l'évaluation de chaque message est en $\mathcal{O}(N_j C_j n_{o_j} + N_j^2 C_j^2)$ où N_j est le nombre de voisins de j et C_j est la plus grande capacité de tous les liens connectés à j.

La preuve repose sur la transformation des expressions des équations 4 et 5 en une agrégation n-aire qui est ensuite transformée en un ensemble d'agrégations binaires manipulées à l'aide du lemme 2. Le résultat de complexité repose sur le lemme 5.

	Acyclic-Solving	RadPro
Messages	$\mathcal{O}(nC_{max})$	$\mathcal{O}(nC_{max})$
Temps	$\mathcal{O}(n(2C_{max}+1)^{N_{max}})$	$\mathcal{O}(nN_{max}^2C_{max}^2)$

Table 1 – Complexités théoriques comparées

4.3 Envoi de message dans RadPro

Le théorème 1 montre que les messages d'Acyclic-Solving peuvent être évalués en temps polynomial pour EAP. Nous appelons RadPro le nouvel algorithme résultant de l'exécution d'Acyclic-Solving comme décrit dans l'algorithme 1 mais en évaluant les messages en utilisant les équations 8 et 9. Notons que nous ne modifions que l'algorithme utilisé pour évaluer les messages. Comme les messages sont échangés de la même manière pour les deux algorithmes, le nombre et la taille des messages restent les mêmes : $2n$ messages envoyés, avec n le nombre de prosommateurs, pour une taille de messages bornée par $2C_{max}+1$, où C_{max} est la plus grande capacité du réseau. Ainsi, le nombre de valeurs échangées par les deux algorithmes est en $\mathcal{O}(nC_{max})$.

Concernant la complexité en temps, comme précisé dans la section 4.1, dans Acyclic-Solving la complexité pour un prosommateur j d'évaluer le message à son parent est en $\mathcal{O}((2C_j+1)^{N_j})$ où C_j est la plus grande capacité des liens connectés à j et N_j est le nombre de voisins de j. Ainsi, la complexité globale est bornée par $\mathcal{O}(n(2C_{max}+1)^{N_{max}})$ où N_{max} est le nombre de voisins du prosommateur le plus connecté et n est le nombre total de prosommateurs. D'autre part, la complexité en temps pour évaluer un message dans RadPro, comme vu dans le théorème 1 est en $\mathcal{O}(N_jC_jn_{o_j}+N_j^2C_j^2)$. Comme le nombre de messages à évaluer est égal au nombre de prosommateurs dans le réseau, la complexité globale est bornée par $\mathcal{O}(n(N_{max}C_{max}n_{max}+N_{max}^2C_{max}^2))$ où n_{max} est la taille de l'offre la plus large. En supposant que dans le cas le plus commun $n_{max} < N_{max}C_{max}$, on peut simplifier l'expression en une complexité finale en $\mathcal{O}(nN_{max}^2C_{max}^2)$.

5 Expérimentation et évaluation

Dans cette section nous évaluons le temps d'exécution de RadPro et le comparons avec celui d'Acyclic-Solving et ceux de solveurs MIP (*Mixed Integer Programming*) classiques, après avoir transformé EAP en programme linéaire, suivant le modèle de [1].

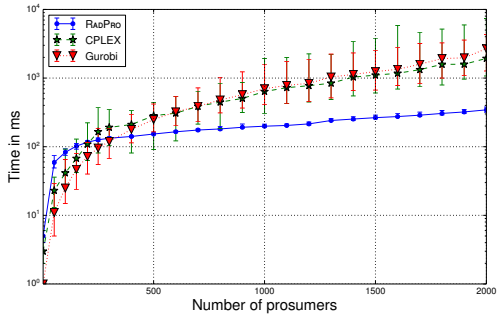
Comparaison avec des solveurs MIP. Ici nous comparons les performances de RadPro à celles de solveurs commerciaux (IBM CPLEX et Gurobi). Pour ce faire, nous générons des instances d'EAP acycliques dont la structure simule la topologie de réseaux radiaux. Conformément à [20], une bonne approximation de la distribution empirique du degré des nœuds dans les SG réelles est obtenue grâce à une distribution géométrique. Ainsi, nous générons des arbres dont les degrés des nœuds suivent une distribution géométrique, avec $p = 0.5$, pour simuler la structure de larges réseaux de distribution. Chaque nœud de l'arbre représente un prosommateur, qui est déterminé de manière aléatoire comme étant soit un producteur (10% des prosommateurs) soit un consommateur (90% restants). Pour chaque consommateur j, l'offre o_j est :

$$o_j(t) = \begin{cases} t \cdot price_j & t \in [min_j..max_j] \\ 0 & t = 0 \\ -\infty & sinon \end{cases}$$

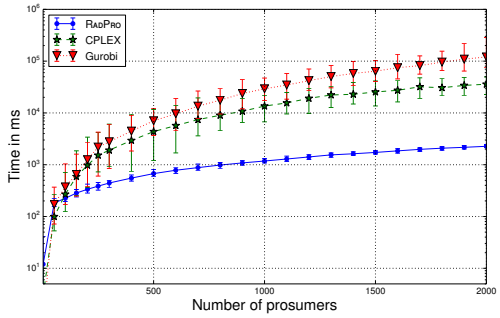
où min_j et max_j sont les nombres minimum et maximum d'unités permises pour le prosommateur j, et $price_j$ est le prix par unité d'énergie pour ce prosommateur. Le nombre maximum d'unités max_j est déterminé suivant une distribution normale $\mathcal{N}(\kappa, \frac{\kappa}{2})$ où κ est un paramètre de capacité. Le nombre minimum d'unités min_j est ensuite déterminé suivant une distribution uniforme $\mathcal{U}(1, max_j)$. Le prix $price_j$ est déterminé suivant une loi normale $\mathcal{N}(1, 0.5)$. Les producteurs sont générés suivant la même procédure, à l'exception de l'offre qui est définie dans l'intervalle $[-max_j..-min_j]$. La capacité d'un lien entre deux nœuds est fixée au nombre maximum d'unités produites ou requises par chacun des prosommateurs. Les expérimentations ont été conduites sur un unique Intel Core i7 2.66GHz, avec 8GB RAM. RadPro a été développé en Java.

Nous analysons deux capacités différentes $\kappa = 10$ et $\kappa = 100$. Le nombre d'agents (prosommateurs) n varie par pas de 100 jusqu'à 2000. Pour chaque κ et n, nous avons généré 100 instances différentes. La figure 3a montre la valeur médiane et l'écart interquartile 5% – 95% du temps d'exécution pour $\kappa = 10$, et la figure 3b pour $\kappa = 100$. Dans les deux cas, CPLEX et Gurobi sont plus rapides quand le nombre de prosommateurs est faible. Cependant, RadPro passe mieux à l'échelle et pour $n = 2000$ il est plus rapide de plus d'un ordre de grandeur que CPLEX, qui lui surpasse Gurobi. Dans le scénario le plus extrême ($n = 2000$, $\kappa = 100$), RadPro affiche un temps médian inférieur à 2.3 secondes alors que CPLEX a besoin de plus de 35.8 secondes et Gurobi plus de 2 minutes.

Efficacité du calcul des messages. Pour souligner le bénéfice de l'évaluation efficace des messages avec une taille de voisinage grandissante, nous exécutons RadPro sur des instances



(a) $\kappa = 10$



(b) $\kappa = 100$

FIGURE 3 – Temps d'exécution de RADPRO comparé à CPLEX et Gurobi, pour des réseaux à distribution géométrique.

plus simples basées sur des réseaux en étoile (un nœud central connecté à plusieurs voisins). les offres sont générées comme précédemment, sauf que min_j et max_j sont maintenant fixées à $min_j = 1$ et $max_j = \kappa$. Comme les prosommateurs font des propositions pour chaque valeur de leurs capacités, ceci représente le pire scénario possible. De plus, ceci résulte dans une capacité fixée pour les liens (tous les liens ont la même capacité κ), facilitant ainsi l'analyse des résultats. La figure 4 montre l'accélération – i.e. le rapport $\frac{time(\text{ACYCLIC-SOLVING})}{time(\text{RADPRO})}$. Par exemple, en taille $n = 7$ (un agent central ayant 6 voisins), pour $\kappa = 50$, RADPRO est 5000 fois plus rapide qu'ACYCLIC-SOLVING. Ceci souligne comment le calcul efficace des messages que nous proposons permet à RADPRO de passer à l'échelle polynomialement, alors que le temps d'exécution d'ACYCLIC-SOLVING grandit exponentiellement : à $n = 7$ et $\kappa = 50$, RADPRO prend environ 1 seconde alors qu'ACYCLIC-SOLVING prend environ 83 minutes pour résoudre une instance. Notons que ceci est également équivalent à comparer RADPRO à d'autres approches multiagents par programmation dynamique comme DPOP [16], ActionGDL [18] ou D–DYDOP [11]. La figure 5 illustre comment le temps de RADPRO est affecté par la capacité pour des réseaux plus larges (ces scénarios sont inaccessibles pour ACYCLIC-SOLVING). Ici, nous voyons que RADPRO résout un problème de capacité 100 avec 100 voisins en moins de 1 minute. Ces résultats font de RADPRO un très bon candidat pour s'attaquer à de larges EAP ayant un facteur de branchement élevé —ce qui est le cas dans les configurations urbaines.

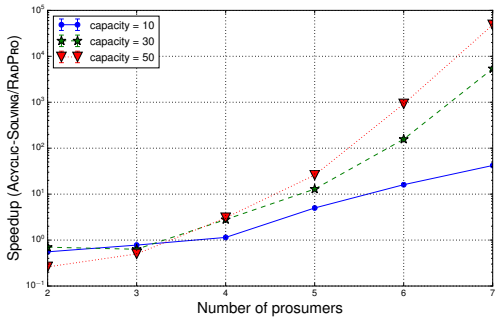


FIGURE 4 – Accélération de RADPRO par rapport à ACYCLIC-SOLVING) dans des réseaux en étoile.

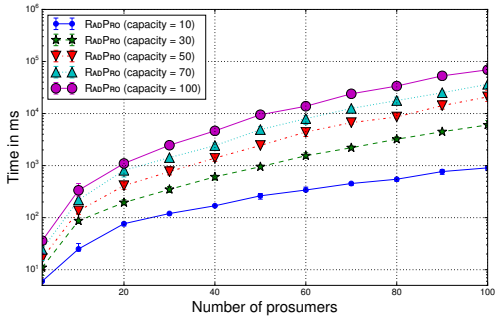


FIGURE 5 – Temps d'exécution de RADPRO dans le larges réseaux en étoile.

6 Conclusions et perspectives

Dans cet article nous avons étudié comment mettre en place un échange d'énergie entre prosommateurs tout en prenant en compte les limites physiques du réseau électrique. Nous avons considéré ce problème comme une problème d'optimisation, l'EAP. Pour des configurations acycliques (ce qui est un hypothèse très réaliste, compte tenu de la structure actuelle des réseaux électriques), nous avons conçu RADPRO, un nouvel algorithme de programmation dynamique pour résoudre de manière optimale et efficace l'EAP acyclique. RADPRO se base sur ACYCLIC-SOLVING car il repose sur un réseaux arborescent, qui est parfaitement adapté aux réseaux radiaux. Cependant, à cause du temps exponentiel du calcul des messages dans ACYCLIC-SOLVING, nous avons apporté à RADPRO une évaluation efficace des messages grâce à une nouvelle *algèbre des valuations*. Grâce à celle-ci, RADPRO peut efficacement calculer les messages en temps polynomial, et ainsi pallier les limitations d'ACYCLIC-SOLVING en réduisant le temps de calcul de plusieurs ordres de grandeurs. De plus, nos résultats expérimentaux montrent que RADPRO surclasse

largement CPLEX et Gurobi en temps de calcul pour trouver l'allocation optimale d'une EAP, avec une taille de marché grandissante. En général, nos expériences démontrent que les complexités en temps et en nombre de messages de RadPro le positionne clairement comme un algorithme optimal et adapté aux larges réseaux.

Voici quelques pistes de travail futur. Premièrement, la nature par envoi de messages de RadPro offre la possibilité de résoudre EAP de manière centralisée ou distribuée. Ceci est en ligne directe avec les défis dans [3], et ouvre la possibilité d'explorer les impacts économiques et environnementaux de la mise en place d'un protocole de négociation pair-à-pair. Deuxièmement, comme les futures infrastructures des SG sont amenées à évoluer et éventuellement contenir des boucles (e.g. par maillage), nous projetons d'explorer comment améliorer RadPro pour gérer efficacement les EAP cycliques. Troisièmement, la vitesse de RadPro ainsi que la convergence d'autres technologies (comme le *smart metering*) rendent la *conscience énergétique* en temps réel possible. Et ainsi, de manière alternative aux enchères centralisées journalières, RadPro permet de considérer des fenêtres de temps de l'ordre de la minute, donnant ainsi la possibilité aux prosommateurs de mettre en place des échanges en temps presque réel.

Références

[1] J. Cerquides, G. Picard, and J. A. Rodríguez-Aguilar. Designing a marketplace for the trading and distribution of energy in the smart grid - extended version with proofs, available at http://bit.ly/18PSEb0, 2015.

[2] R. Dechter. *Constraint processing*. Morgan Kauffman, 2003.

[3] European Technology Platform. SmartGrids SRA 2035. Strategic Research Agenda. Update of the SmartGrids SRA 2007 for the needs by the year 2035, March 2012.

[4] Federation of German Industries (BDI). The Energy Industry on the Way to the Internet Age. BDI publication No.439, 2010.

[5] T. Gonen. *Electric power distribution engineering*. CRC press, 2014.

[6] Greenpeace. Decentralising power : An energy revolution for the 21st century. http://bit.ly/1xf1RCk, 2005.

[7] L. L. Grigsby. *Electric Power Generation, Transmission, and Distribution*. CRC press, 2012.

[8] D. Ilic, P. G. Da Silva, S. Karnouskos, and M. Griesemer. An energy market for trading electricity in smart grid neighbourhoods. In *6th IEEE International Conference on Digital Ecosystems Technologies (DEST), 2012*, pages 1–6. IEEE, 2012.

[9] K. Kok, B. Roossien, P. MacDougall, O. van Pruissen, G. Venekamp, R. Kamphuis, J. Laarakkers, and C. Warmer. Dynamic pricing by scalable energy management systems—field experiences and simulation results using powermatcher. In *Power and Energy Society General Meeting, 2012 IEEE*, pages 1–8. IEEE, 2012.

[10] S. Lamparter, S. Becher, and J.-G. Fischer. An agent-based market platform for smart grids. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems : Industry track*, pages 1689–1696. International Foundation for Autonomous Agents and Multiagent Systems, 2010.

[11] S. J. O. Miller. *Decentralised Coordination of Smart Distribution Networks using Message Passing*. PhD thesis, University of Southhampton, 2014.

[12] J. Mockus. On simulation of the nash equilibrium in the stock exchange contest. *Informatica*, 23(1) :77–104, 2012.

[13] B. Neagu and G. Georgescu. Optimization Possibilities for Radial Electric Energy Distribution Network Routes. *Bul. Inst. Politehnic, Iaşi, LIX (LXIII)*, (Lxiii), 2013.

[14] M. A. Olson, S. J. Rassenti, V. L. Smith, M. L. Rigdon, and M. J. Ziegler. Market design and motivated human trading behavior in electricity markets. In *Proceedings of the 32nd Annual Hawaii International Conference on Systems Sciences, 1999. HICSS-32.*, pages 1–27. IEEE, 1999.

[15] Y. K. Penya and N. R. Jennings. Optimal combinatorial electricity markets. *Web Intelligence and Agent Systems*, 6(2) :123–135, 2008.

[16] A. Petcu and B. Faltings. A scalable method for multiagent constraint optimization. *IJCAI International Joint Conference on Artificial Intelligence*, pages 266–271, 2005.

[17] S. D. Ramchurn, P. Vytelingum, A. Rogers, and N. R. Jennings. Putting the 'smarts' into the smart grid : A grand challenge for artificial intelligence. *Commun. ACM*, 55(4) :86–97, Apr. 2012.

[18] M. Vinyals, J. A. Rodríguez-Aguilar, and J. Cerquides. Constructing a unifying theory of dynamic programming DCOP algorithms via the generalized distributive law. *Autonomous Agents and Multi-Agent Systems*, 3(22) :439–464, May 2011.

[19] P. Vytelingum, S. D. Ramchurn, T. D. Voice, A. Rogers, and N. R. Jennings. Trading agents for the smart electricity grid. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems : volume 1-Volume 1*, pages 897–904. International Foundation for Autonomous Agents and Multiagent Systems, 2010.

[20] Z. Wang, A. Scaglione, and R. J. Thomas. The Node Degree Distribution in Power Grid and Its Topology Robustness under Random and Selective Node Removals. *2010 IEEE International Conference on Communications Workshops*, (1) :1–5, May 2010.

[21] B. M. Weedy, B. J. Cory, N. Jenkins, J. Ekanayake, and G. Strbac. *Electric power systems*. John Wiley & Sons, 2012.

[22] T. K. Wijaya, K. Larson, and K. Aberer. Matching demand with supply in the smart grid using agent-based multiunit auction. *2013 Fifth International Conference on Communication Systems and Networks (COMSNETS)*, pages 1–6, Jan. 2013.

[23] P. Wolfe. The implications of an increasingly decentralised energy system. *Energy policy*, 36(12) :4509–4513, 2008.

Planification de missions multi-satellites par système multi-agent coopératif

J. Bonnet[a,b] jonathan.bonnet@irt-saintexupery.com
M-P. Gleizes[b] Marie-Pierre.Gleizes@irit.fr
E. Kaddoum[b] Elsy.Kaddoum@irit.fr
S. Rainjonneau[a] serge.rainjonneau@irt-saintexupery.com

[a]Institut de Recherche Technologique Saint Exupéry, Toulouse, France

[b]IRIT, Université Paul Sabatier - Toulouse III, France

Résumé

La planification de mission de constellations de satellites est un problème complexe soulevant d'importants défis technologiques pour les systèmes spatiaux de demain. L'augmentation de la taille des constellations, les performances évoluées et l'hétérogénéité des satellites sont tous des critères impliquant une combinatoire très élevée. Les techniques actuelles présentent des limites, car elles planifient un satellite à la fois et non la constellation dans son ensemble.
Dans cet article, nous proposons de résoudre ce problème difficile par les systèmes multi-agents coopératifs. Une amélioration du modèle AMAS4Opt est présentée, permettant de maximiser la coopération entre les agents du système. De par leurs interactions locales, ces agents, permettent d'obtenir une solution de bonne qualité en un temps raisonnable, en assurant un partage équitable des tâches au sein de la constellation. Enfin, une comparaison avec l'algorithme Glouton Chronologique, couramment utilisé dans le domaine spatial, souligne les avantages de l'approche proposée.

Mots-clés : *Planification, Multi-satellites, Systèmes multi-agents coopératifs.*

Abstract

Mission planning of constellation of Earth observation satellites is a complex problem raising significant technological challenges for tomorrow's space systems. The increasing size of constellations, the advanced performances and heterogeneity of satellites are different criteria involving a huge combinatorial search space. The techniques used today have limitations : they are planning one satellite at a time and not the whole constellation.
In this paper, we propose to solve this difficult problem by cooperative multi-agent systems. An improvement of the AMAS4Opt agent model is presented to maximize the cooperation level. In the proposed system, the agents, through their local interactions, allow to reach a good solution in a reasonable time, while ensuring a fair distribution of tasks within the constellation. Finally, a comparison with the chronological greedy algorithm, commonly used in the spatial domain, underlines the advantages of the presented system.

Keywords: *Scheduling, Multi-Satellite, Cooperative Multi-Agent Systems.*

1 Introduction

Une constellation de satellites d'observation de la Terre est un ensemble potentiellement hétérogène de satellites. Elle permet de couvrir une large surface terrestre avec une fréquence de revisite élevée de chaque zone, tout en proposant des images de types différents et en garantissant la robustesse du système. Planifier une mission d'observation pour une telle constellation est une tâche difficile. En effet, beaucoup de paramètres et de contraintes souvent contradictoires sont à prendre en compte : le nombre de satellites et leurs caractéristiques, le volume des demandes des clients et les nombreuses contraintes qui y sont associées (le type de prise de vue, la priorité du client, la qualité demandée, la plage temporelle de validité, etc.) ainsi que les contraintes extérieures (la couverture nuageuse dans le cas de satellites optiques, etc.). La durée de mise en place d'un plan doit être raisonnable, notamment dans le cadre de demandes urgentes, typiquement inférieures à cinq minutes dans un contexte opérationnel, ce qui n'est pas toujours le cas actuellement.

Nous supposons connu l'algorithme qui permet à un satellite de construire son plan de mission

à partir d'un ensemble de prises de vues qui lui est affecté. Dans cet article nous nous intéressons au problème de la répartition des requêtes dans une constellation de satellites. Cette répartition consiste à planifier les demandes des clients sur les différents satellites, tout en respectant les contraintes et en assurant une charge équilibrée pour chaque satellite.

Pour cela, les systèmes multi-agents coopératifs (Gleizes, 2012) et leur capacité à prendre en compte un grand nombre d'entités et de contraintes sont une bonne approche pour résoudre ce type de problème. Dans ce travail nous nous appuierons sur AMAS4Opt *(Adaptive Multi-Agent System For Optimization)* (Kaddoum, 2011), un modèle d'agent générique qui fournit des patrons de conception pour résoudre des problèmes d'optimisation combinatoire à l'aide des systèmes multi-agents coopératifs.

La partie 2 de cet article formalise le problème et présente son contexte. La partie 3 développe le fonctionnement du système multi-agent ATLAS *(Adaptive saTellites pLanning for dynAmic earth obServation)*. Enfin, la partie 4 présente une évaluation du système ATLAS ainsi que sa comparaison à un algorithme Glouton Chronologique.

2 Le problème de planification multi-satellite

Cette partie décrit le problème puis présente un résumé des méthodes de résolution actuellement utilisées.

2.1 La formalisation du problème

En nous appuyant sur les travaux de (Bensana and Verfaillie, 1999) et (Bonnet, 2008), nous allons tout d'abord présenter formellement notre problème. Dans la partie 3, nous nous appuierons sur cette modélisation pour *agentifier* le système.

Constellation de satellites. Une constellation est un ensemble potentiellement hétérogène de satellites, $Sat = \{s_1, s_2, ..., s_n\}$, dans lequel chaque satellite s_i a ses propres caractéristiques :
- une trajectoire légèrement elliptique autour de la terre,
- une gestion d'énergie,
- une capacité mémoire,
- une charge utile, ici des instruments d'observation optique ou radar, et leurs attributs propres,
- un système de contrôle orbital et de gestion d'attitude, qui permet au satellite de contrôler son pointage *(vers la zone à observer)*.

Requêtes. Une requête est une commande client. Nous noterons l'ensemble des requêtes à planifier $R = \{r_1, r_2, ..., r_m\}$, avec r_i une requête définie par un ensemble de données :
- son type,
- une date de soumission,
- un intervalle de temps qui correspond à la période durant laquelle la requête est valable *(quelques heures à quelques jours, voire plusieurs semaines)*,
- une zone géographique,
- une priorité donnée par le client,
- w, le taux de couverture nuageuse toléré, $0 <= w <= 1$,
- un ensemble de mailles, $M^i = \{m^i_1, m^i_2, ..., m^i_p\}$, associé à la requête r_i.

Maille. Chaque requête est découpée en un ensemble de mailles dites *sœurs*, M^i. Une maille est une entité élémentaire que le satellite pourra acquérir en une seule prise de vue. Dans cet article, nous ne parlerons pas de l'algorithme de découpage des requêtes. Chaque maille reprend les contraintes de sa requête, mais est définie par une zone géographique plus petite. Une requête r_i est satisfaite si toutes ses mailles sont acquises. A chaque maille, est associé un ou plusieurs accès de visibilité correspondant à une période durant laquelle la maille est visible par le satellite. C'est dans cet accès de visibilité que l'acquisition sera effectuée. La durée du créneau d'acquisition est très inférieure à la durée de l'accès de visibilité. La figure 1 représente un accès de visibilité *A* sur la maille *M1*, par un satellite. Un créneau de réalisation *C* placé à l'intérieur de l'accès *A*. Le calcul de ces accès de visibilité est une donnée externe au problème.

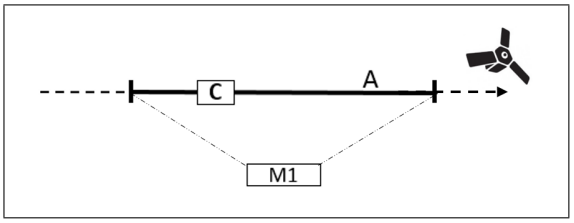


FIG. 1 – Représentation d'un accès de visibilité et d'un créneau de réalisation

Contraintes dures et souples. Deux catégories de contraintes sont à prendre en compte : les contraintes dures (la plage de validité ou le type d'image souhaité par le client par exemple) et souples (le taux de couverture nuageuse). L'ensemble C_h de contraintes dures doit être satisfait, les contraintes souples, C_s peuvent être relâchées.

But. L'objectif du problème est donc de trouver pour chaque satellite un ensemble de mailles, avec pour chaque maille une date d de couverture. Les mailles d'une même requête ne seront pas obligatoirement attribuées au même satellite. Les satellites devront assurer la couverture tout en :
– satisfaisant l'ensemble des contraintes dures C_h,
– maximisant le nombre de contraintes souples C_s satisfaites,
– maximisant le nombre de mailles à satisfaire.

2.2 État de l'art

Différentes publications traitent du problème de la planification de mission d'une constellation, en le comparant à des problèmes d'optimisation tel que le problème du **sac à dos multidimensionnel** (Lemaître et al., 2002) et (Grasset-Bourdel et al., 2011) ou au problème du **voyageur de commerce** (Mancel, 2004). Le sac à dos représentant la capacité mémoire des satellites, et le chemin à trouver entre les villes modélisant l'enchaînement entre les acquisitions. De nombreux algorithmes existent pour résoudre ces problèmes, et les méthodes de planification spatiale s'en inspirent. Nous regroupons ici les approches les plus couramment utilisées dans le domaine, en commençant par les méthodes exactes puis les méthodes approchées.

Les méthodes exactes garantissent de trouver la solution optimale, si elle existe. La **Programmation Dynamique** est utilisée pour décomposer le problème en sous-problèmes plus simples. La difficulté de tels algorithmes vient de la possibilité de découper récursivement le problème initial. Comme le montre (Grasset-Bourdel, 2011), si on applique l'algorithme linéairement, son déroulement va rapidement utiliser la majorité des ressources mémoires et le rendre inapplicable en cas de requêtes stéréoscopiques (requêtes qui demandent des prises de vue d'un même endroit, mais avec un décalage temporel). (Mansour and Dessouky, 2010) émettent la même critique concernant des algorithmes construits sur une recherche de type **Séparation et Évaluation**. En effet, même si toutes les solutions ne sont pas explorées car l'algorithme se sert des propriétés du problème pour guider la recherche, le déroulement d'une telle méthode est extrêmement coûteux (en espace mémoire et en temps de calcul). Ces méthodes ne sont donc pas adaptées pour des problèmes ayant de très grands espaces de recherche. Le temps de parcours et l'espace mémoire nécessaires étant dépendants de la taille de cet espace.

Les méthodes approchées peuvent être appliquées pour pallier ces différents inconvénients. Elles permettent de trouver une solution d'une bonne qualité en un temps raisonnable. La différence entre ces méthodes vient de l'heuristique qui guide la recherche et impacte donc la qualité de la solution finale.

L'algorithme le plus utilisé, car rapide et facile à mettre en oeuvre, est l'algorithme **Glouton Chronologique** (souvent nommée par son terme anglophone *Greedy Algorithm*). Son principe est simple, on planifie chronologiquement les requêtes, et en cas de conflit on applique une heuristique locale. De plus, cet algorithme sert souvent de base de travail, comme par exemple pour (Lemaître et al., 2002). Enfin, différentes méthodes approchées sont adaptées pour être appliquées au problème de planification de mission, comme les méthodes de **Recuit Simulé** (Wu et al., 2014), les **Recherches Tabou** (Bianchessi et al., 2007), ou bien les **Algorithmes Génétiques** tels que ceux présentés par (Globus et al., 2003) et (Mansour and Dessouky, 2010).

Tous ces algorithmes, même s'ils produisent des résultats plutôt satisfaisants présentent des limites. Tout d'abord, comme nous l'avons évoqué, ils dépendent fortement de l'heuristique qui guide la recherche. Il faut donc que le concepteur étudie et comprenne le problème dans sa globalité, et si celui-ci change, il faudra réadapter l'heuristique et l'algorithme. De plus, en cas d'ajout de requêtes pendant le déroulement du processus de planification, il faut reprendre le processus depuis la première opportunité de placement de la requête ajoutée. Enfin, ces algorithmes sont essentiellement conçus pour planifier la mission d'un satellite. Même si le problème reste globalement le même, ils font abstraction des nouvelles contraintes, comme par exemple l'équilibre de la charge de la constellation.

(Bonnet and Tessier, 2009) proposent une approche basée sur les systèmes multi-agents

pour la planification à bord des satellites d'une constellation. Dans cette approche, les satellites sont définis comme étant des agents communicants via des liaisons inter-satellites. L'objectif des satellites est de négocier entre eux la répartition des demandes de prises de vues reçues du sol permettant de maximiser le nombre de demandes satisfaites. Par l'utilisation des liaisons inter-satellites, l'applicabilité de cette approche requiert un certain nombre de conditions (taille de la constellation, distance entre satellites, etc.). Dans le travail présenté, nous proposons l'utilisation des systèmes multi-agents pour **la planification au sol de la mission d'une constellation de satellites**.

3 Le système ATLAS

Le modèle AMAS4Opt (Kaddoum, 2011) est basé sur l'approche par AMAS (*Adaptive Multi-Agent System*) (Gleizes, 2012). Dans cette approche, la coopération locale des agents permet au système de s'auto-organiser pour réaliser la fonction pour laquelle il est conçu. Ainsi, AMAS4Opt fournit les patrons de conception de deux rôles d'agents coopératifs : le rôle « contraint » et le rôle « service ». Les agents ayant le rôle « contraint » portent le problème et doivent être satisfaits, et les agents qui ont le rôle « service » possèdent les capacités et les compétences pour satisfaire les agents ayant le rôle « contraint ». Ce modèle utilise la *criticité* locale des agents ayant le rôle « contraint » comme moteur de la coopération entre agents. Nous avons utilisé ce modèle pour concevoir les agents et l'architecture générale du système et nous en proposons ici une amélioration.

Dans cette partie nous présentons les agents et leur rôle, puis la mesure de la criticité utilisée par les agents ayant le rôle « contraint ». Enfin, nous introduisons la mesure du coût représentant la criticité des agents ayant le rôle « service ».

3.1 Les agents

A l'aide de la modélisation du problème proposée dans la partie 2.1, et en se basant sur AMAS4Opt nous avons identifié neuf entités, parmi lesquelles :

– **trois types d'agents coopératifs** : le type agent requête, le type agent maille et le type agent satellite (leur comportement sera détaillé plus tard dans cet article),
– **trois types d'entités actives** : la couverture nuageuse, l'éphéméride solaire et les stations de télé-déchargement (ces entités n'ont pas de but à satisfaire),
– **trois types d'entités passives** (ressources du système) : la mémoire des satellites, leur batterie et un module de calcul de trajectoire et d'attitude orbitales.

L'utilisation du modèle AMAS4Opt permet de définir le comportement et les interactions des agents : les agents satellites prennent le rôle « service », et les agents requêtes et mailles le rôle « contraint ». Nous présentons ci-dessous les trois agents coopératifs en indiquant leur objectif local, les entités du système avec lesquelles ils seront en interaction au cours de leur processus de résolution et les agents avec lesquels ils seront amenés à négocier (échanges d'information, demandes de services, etc.), dans le but d'atteindre leur objectif.

Le type agent satellite (rôle « service »)
– objectif : acquérir des prises de vues,
– interactions avec :
  - le module de gestion trajectoire,
  - la couverture nuageuse,
  - la batterie,
  - la mémoire,
  - le soleil,
– négociations avec :
  - les agents mailles.

Le type agent requête (rôle « contraint »)
– objectif : avoir toutes ses mailles planifiées,
– négociations avec :
  - les agents mailles.

Le type agent maille (rôle « contraint »)
– objectif : être planifiée,
– négociations avec :
  - les agents satellites,
  - les agents requêtes.

3.2 La planification multi-satellite

Le système ATLAS assure la planification de la constellation grâce à la coopération de ses agents. Cette coopération est assurée via l'échange de messages entre les agents et est guidée par deux indicateurs : la criticité et le coût. C'est cette coopération qui permet de produire une planification respectant les critères suivant : un maximum de requêtes planifiées et un équilibre au sein de la constellation (les satellites doivent avoir tous la même charge de travail). Le fonctionnement du système ATLAS

TAB. 1 – Les agents coopératifs et leurs différents messages

Agent	Message	Signification
Satellite	estimationSlot(Cost c)	Estimation du coût de la planification d'une maille
	confirmSlot(Boolean b)	Confirmation ou non de la planification
Maille	askForASlot(Access a)	Demande de planification
	askConfirmationForASlot(Access a)	Demande de confirmation pour l'accès
	informRequest()	Transmet de nouvelles informations à sa requête
Requête	informMesh()	Transmet de nouvelles informations à une maille

est basé sur la répétition par les agents du cycle « Perception - Décision - Action ». La phase de décision est l'étape centrale. En fonction de ses perceptions et de son état, l'agent choisit l'action à réaliser.

Les agents mailles envoient leurs demandes à tous les agents satellites ayant un accès de visibilité sur eux, et par la suite, privilégient l'agent satellite au coût le plus faible. Les agents satellites vont traiter tous les agents mailles les ayant sollicités en commençant par ceux les plus critiques. Les agents requêtes peuvent influencer la criticité de leurs agents mailles en fonction de l'avancement de la planification de la requête dans son ensemble. Les différents messages émis par les agents sont présentés dans la table 1.

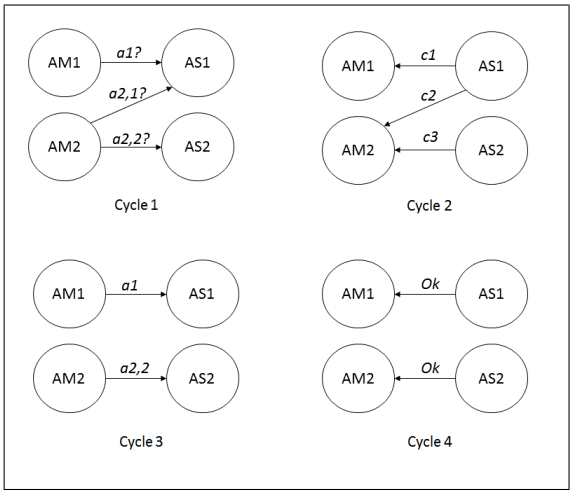


FIG. 2 – Exemple de planification

La figure 2 illustre le problème suivant : deux agents mailles (AM1 et AM2) communiquent avec deux agents satellites (AS1 et AS2). AM1 possède un seul accès de visibilité et AM2 deux. La résolution suit les cycles suivants :

– cycle 1 : AM1 et AM2 émettent pour tous leurs accès (a1, a2.1 et a2.2) des demandes aux agents satellites,
– cycle 2 : AS1 et AS2 classent les demandes en fonction de la criticité des agents mailles, et calculent les coûts (c1, c2 et c3) de planification qu'ils renvoient aux agents mailles,
– cycle 3 : AM1 et AM2 choisissent les agents satellites les moins coûteux et demandent confirmation,
– cycle 4 : AS1 et AS2 valident.

3.3 La criticité et le coût

La criticité. Dans ATLAS, le degré de non satisfaction des agents mailles est représenté par la criticité. (Bouziat et al., 2014) définissent la criticité comme « la distance qui sépare l'état courant [de l'agent], de l'état dans lequel son objectif local est atteint ».

Les règles de comportement local des agents satellites sont donc écrites pour favoriser les agents mailles les plus *critiques*. Cette criticité, qui indique donc le degré de non satisfaction de l'agent maille, est représentée par un ensemble de valeurs ordonnées. Dans la version actuelle du système, cette mesure comprend trois critères.

Le premier critère concerne l'urgence de la planification de la maille. Celle-ci est définie par le client au moment de sa commande. En cas d'égalité de ce critère, les agents satellites utilisent le nombre d'accès de visibilité restants (deuxième critère). Ainsi, un agent maille qui n'est visible qu'une seule fois par les satellites sera prioritaire par rapport aux autres agents mailles visibles plusieurs fois. En effet, si l'agent satellite privilégie un agent maille ayant plusieurs accès de visibilité sur un agent maille en ayant un seul, ce dernier ne pourra pas être planifié. Enfin, les agents mailles font évoluer leur criticité en fonction de leur état et des réponses à leurs messages envoyés aux agents satellites. Le troisième critère concerne donc l'influence des agents requêtes sur les agents mailles les composants.

L'évolution de la criticité d'un agent maille augmente si le nombre de ses créneaux de visibilité

diminuent, ou si ses mailles *sœurs* sont planifiées. Elle baisse lorsque l'agent maille est pris en compte. Finalement, elle est nulle lorsque l'acquisition de l'agent maille est planifiée.

Le coût. La criticité, telle qu'elle est définie dans le modèle AMAS4Opt, ne permet pas une coopération *totale* entre les agents : la coopération est seulement assurée par les agents ayant le rôle « contraint ». En effet, elle ne permet qu'aux agents ayant le rôle « service » de savoir quels agents ayant le rôle « contraint » sont prioritaires. Ainsi, un agent ayant le rôle « contraint » ne peut pas privilégier un agent ayant le rôle « service » sur un autre. Nous avons donc ajouté la notion de coût pour résoudre ce problème, et rendre les agents ayant le rôle « contraint » plus coopératifs.

Dans ATLAS, le coût est un indicateur sur la difficulté pour l'agent satellite à prendre en compte et planifier un agent maille. Il est calculé et retourné à l'agent maille ayant émis un message de demande de couverture. L'agent maille va donc recevoir un coût pour chaque demande émise aux agents satellites. Pour favoriser la coopération, un agent maille privilégie l'agent satellite lui répondant avec le coût le plus faible. Voici des éléments qui favorisent un coût élevé :
– une charge mémoire importante (beaucoup de mailles sont déjà planifiées par le satellite),
– un grand nombre de messages de demande de planification reçus,
– la nécessité d'adapter le plan, en déplaçant des éléments planifiés.

4 Résultats et discussions

Dans cette partie, nous présentons les résultats obtenus par le système multi-agent ATLAS pour planifier une constellation de satellites. Pour le tester nous avons développé un générateur de *scenarii*. Ces différents *scenarii* permettent de présenter l'intérêt du système ATLAS et de le comparer avec l'algorithme Glouton Chronologique couramment utilisé dans le domaine spatial.

4.1 Le générateur de solutions

Lors de l'établissement d'un plan de mission, les demandes des clients sont pré-traitées pour conduire à une liste de mailles correspondantes à acquérir. Cette phase de pré-traitement ne concerne pas ATLAS. Chacune de ces mailles terrestres est visible par un ou plusieurs satellites de la constellation. Chaque maille peut donc être acquise par différents satellites à différents instants, ce qui augmente le nombre de solutions possibles.

Nous avons donc développé et utilisé un générateur pour construire des plans de mission *complets*, c'est-à-dire que tous les créneaux sont occupés par des acquisitions. Ces plans se concentrent sur les données temporelles des requêtes. Pour éviter de fournir une liste dans laquelle chaque requête n'a qu'un seul accès de visibilité, le générateur ajoute à chaque requête un nombre aléatoire d'accès de visibilité vers des satellites eux aussi tirés aléatoirement dans la constellation. Ainsi, nous obtenons une liste de requêtes ayant toutes plusieurs accès de visibilité. Chaque requête peut donc être couverte par N satellites et ceci à différents moments de la mission.

Ce générateur permet de générer un nombre conséquent de *scenarii* représentatifs de la combinatoire du problème réel. Cette génération a été validée par des experts du domaine spatial. Nous pouvons ainsi tester le système ATLAS, prouver sa validité, sa robustesse et son passage à l'échelle.

4.2 Expérimentations sur ATLAS

Nous allons tout d'abord nous intéresser à l'évolution de la criticité globale du système et des messages échangés. Pour cela, nous avons généré un scénario comprenant cinq satellites et cinq cents requêtes, scénario que nous nommerons S1. Les résultats que nous exposons sont obtenus en moyenne après avoir lancé cent fois S1.

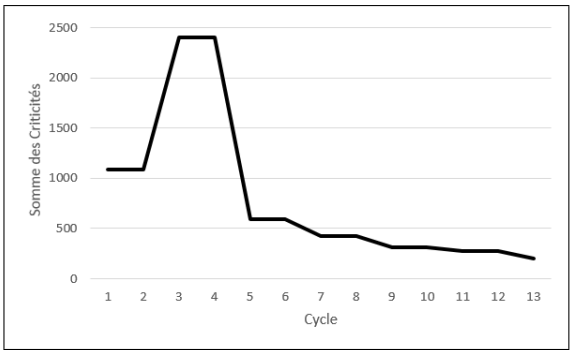


FIG. 3 – Évolution de la criticité globale d'ATLAS

Criticité du système. La figure 3 montre l'évolution de la criticité globale du système tout au long de son déroulement. Pour cela, nous avons donné une valeur numérique aux criticités des agents mailles. Cette valeur augmente

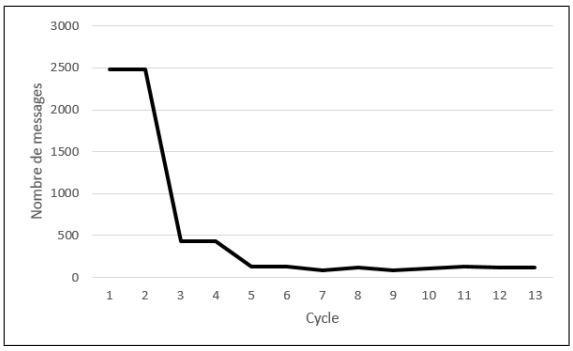


FIG. 4 – Évolution du nombre de messages par cycle

en fonction du nombre d'accès de visibilité restant et du nombre de messages envoyés, elle devient nulle quand la maille est planifiée. Après une croissance lors des premiers cycles, quand les agents mailles émettent des demandes de couverture aux agents satellites, la criticité globale ne fait que décroître avant de se stabiliser. La forte décroissance est due au grand nombre de planifications initiales. Le fait que la criticité globale décroît rapidement pour stabiliser vers une valeur assez faible montre qu'ATLAS converge rapidement vers une solution, où il reste peu d'agents mailles non planifiés.

Communications et cycles de résolution. La figure 4 permet de voir l'évolution du nombre de messages au cours des cycles de décision, et ce pour la résolution de S1. On note deux fortes décroissances entre les cycles 2 - 3 et 4 - 5. Les agents mailles initialisent le système, puis, ayant tous émis des requêtes vers les agents satellites au cycle 1, ce sont ces derniers qui traitent les demandes lors du cycle 2, les agents mailles étant en attente de réponses. Cinq fois moins de messages sont envoyés au cycle 3, en effet, à ce stade ce sont les agents mailles qui traitent les réponses à leurs requêtes et demandent confirmation au satellite qu'elles choisissent de favoriser. On note le même phénomène, un palier puis une forte diminution aux cycles 3, 4 et 5, mais avec moins de messages car davantage d'agents mailles sont déjà dans un état « planifiés », ils ne communiquent donc plus. En convergeant rapidement vers une solution, les agents n'encombrent pas le système avec un trop-plein de communication.

La table 2 permet de visualiser le nombre de messages émis par les agents mailles et le nombre de cycles de résolution nécessaires pour des constellations de plus en plus grandes. Un satellite est ajouté à chaque nouvelle constellation. La colonne S de la table 2 donne le nombre de satellites pour chaque constellation. Les différents *scenarii* produits par le générateur durent 100 unités de temps, ce qui correspond à une moyenne de 86 requêtes pour chaque satellite, la croissance du nombre de requêtes est donc linéaire. Pour simplifier les expérimentations, nous avons décidé que chaque requête correspond à une maille. La proportion d'accès de visibilité pour chaque maille est donnée par un facteur en entrée du générateur, que nous avons augmenté en fonction du nombre de satellites, afin d'accroître la complexité du problème : rajouter des accès de visibilité revient à accroître le nombre de solutions possibles. L'augmentation de la complexité, et donc du nombre de messages émis, n'empêche pas le système de toujours converger vers une solution en un faible nombre de cycles et dans un temps de calcul restreint.

TAB. 2 – Nombre de messages échangés et nombre de cycles de résolutions

S	Mailles	Accès	Messages	Cycles
2	175	546	780	7
3	259	792	1 125	11
4	331	1 177	1 644	13
5	417	1 518	2 126	13
6	518	2 146	2 921	15
7	605	2 478	3 346	17
8	694	3 470	4 526	17
9	767	4 286	5 540	17
10	860	5 738	7 384	20

4.3 Comparaison avec un algorithme Glouton Chronologique : ChronoG

ChronoG. Pour analyser la qualité des solutions d'ATLAS, nous le comparons à la solution standard qui est actuellement utilisée par les références du spatial : l'algorithme Glouton Chronologique, nommé ici **ChronoG**. ChronoG traite la constellation un satellite à la fois. Nous supposerons aussi qu'une maille sera acquise par un seul satellite.

L'algorithme suivant est donc répété pour chaque satellite : pour chaque pas de temps t de la durée de planification parcourue chronologiquement, ChronoG regarde si une maille est planifiée ou si l'espace est libre.

– Si une maille est planifiée, ChronoG passe à $t + 1$, sinon, ChronoG vérifie si une maille peut être placée à cet instant en déterminant si la maille en question possède un accès de visibilité qui englobe la date courante t.

– si plusieurs mailles sont possibles, ChronoG utilise une heuristique, détaillée ci-dessous, pour sélectionner une maille.
– en cas d'égalité, ChronoG choisira la maille ayant la plus petite différence entre la fin de son accès de visibilité et t.

Finalement, ChronoG planifie la maille choisie en réservant la durée nécessaire à son acquisition.

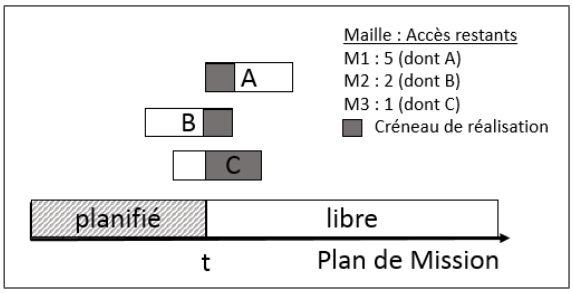


FIG. 5 – Heuristique de choix

La figure 5 permet de comprendre l'heuristique utilisée par ChronoG : les accès A, B et C (appartenant respectivement aux mailles M1, M2 et M3) sont possibles à t. C sera sélectionné car il appartient à la maille la plus critique : M3 n'a plus qu'un seul accès de disponible.

Expérience 1 : Taux de planification. La première expérimentation permet de comparer les systèmes ATLAS et ChronoG, sur des constellations de tailles différentes. Pour mener à bien cette expérimentation, nous avons généré aléatoirement neuf *scenarii*, en rajoutant à chaque fois à la constellation un satellite. De plus, nous augmentons progressivement le degré d'accessibilité des requêtes par les satellites, ainsi, plus la constellation est grande, plus les requêtes sont visibles. Cela permet de complexifier la résolution : en augmentant le nombre d'accès possibles, le nombre de solutions possibles augmente aussi. Nous définissons un indicateur sur la solution : le pourcentage de requêtes planifiées. La solution proposée par le générateur est optimale sur ce critère, car toutes les requêtes sont planifiées : le taux de planification est de 100%. Ainsi il est facile de comparer facilement les deux systèmes, un taux élevé nous indiquant donc que la solution est de bonne qualité.

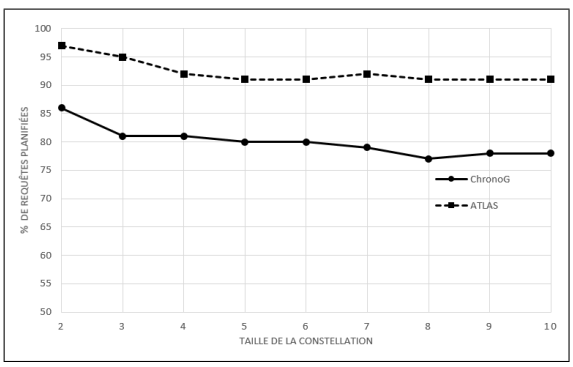


FIG. 6 – Comparaison des résultats entre ATLAS et ChronoG

La figure 6 montre que les résultats obtenus avec le système ATLAS sont meilleurs. Certes, les deux premières exécutions (sur des constellations de deux et trois satellites et une moyenne de deux accès par maille) produisent de bons résultats sur les deux systèmes, mais cela s'explique par le fait que chaque maille possède peu d'accès de visibilité.

Quand la taille de la constellation et le facteur d'accessibilité augmentent, les résultats se stabilisent, ATLAS assurant un taux de planification avoisinant les 90%, alors que ChronoG est en moyenne à 80%. Cette stabilisation est plus visible sur les résultats produits par ATLAS où la convergence se fait dès que la constellation atteint 4 satellites. Enfin, il est important de noter que l'écart type des pourcentages de planification pour ATLAS est de l'ordre de 2%, et ce quelle que soit la taille de la constellation. ATLAS produit donc des solutions homogènes.

TAB. 3 – Comparaison taux de mailles planifiées et temps d'exécution

	% planification		Temps d'exécution	
S	ChronoG	ATLAS	ChronoG	ATLAS
2	86 %	97 %	20 ms	50 ms
3	81 %	95 %	31 ms	31 ms
4	81 %	92 %	32 ms	34 ms
5	80 %	91 %	32 ms	30 ms
6	80 %	91 %	47 ms	28 ms
7	79 %	92 %	50 ms	38 ms
8	77 %	91 %	64 ms	29 ms
9	78 %	91 %	50 ms	36 ms
10	78 %	91 %	50 ms	71 ms

La table 3 détaille les statistiques obtenues lors de l'exécution des neuf *scenarii* : taux de planification et temps moyen d'exécution. Ces résultats montrent qu'ATLAS planifie toujours plus de mailles que ChronoG. Concernant les temps moyens d'exécution, ATLAS est aussi meilleur. La variation de ces durées dépend de la complexité des *scenarii*. Un nombre de conflits important entraînent de nombreux appels à l'heu-

ristique ce qui ralenti ChronoG. A l'inverse, les agents du système ATLAS traitent localement ces conflits, ce qui ne ralentit pas la résolution.

Expérience 2 : Équilibre de charge. Pour illustrer cette seconde expérience, nous avons généré un scénario particulier. Dans ce dernier, deux satellites *agiles* identiques (que nous nommerons A et B) se suivent, cette agilité leur permet d'effectuer des prises de vues parallèles à leur déplacement, mais aussi en avant ou en arrière. De plus, ils possèdent les mêmes accès de visibilité. Ainsi, chaque maille peut être acquise par n'importe quel satellite. Le but ici est de montrer la distribution de la charge au sein de la constellation. Les figures 7 et 8 sont des représentations des tâches que les satellites devront effectuer durant leur passage. L'axe central représente la trace au sol, c'est-à-dire le déplacement vu du sol des deux satellites, avec sur la partie haute (rectangle noir) les tâches du premier satellite et sur la partie basse (damier) celles du second. Les rectangles sont hachurés en fonction du satellite responsable de l'acquisition. Les rectangles vides correspondent quant à eux, aux mailles non affectées.

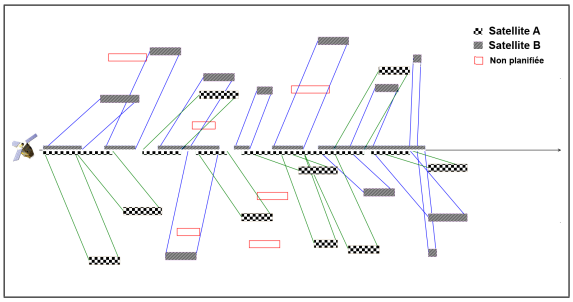


FIG. 7 – Mission planifiée avec ChronoG

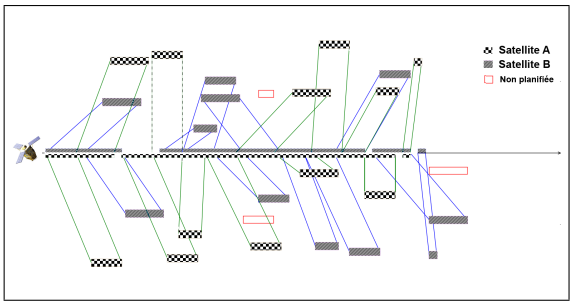


FIG. 8 – Mission planifiée avec ATLAS

Il est intéressant de noter que les résultats obtenus sur ce cas d'étude sont proches de ceux que l'on avait pu obtenir via des *scenarii* aléatoires, ATLAS fournissant toujours de meilleurs taux de planification (88% pour ATLAS contre 76% pour ChronoG). Pour ChronoG (figure 7), le satellite A a seulement neuf tâches de planifiées et B en compte onze. En comparaison, nous voyons sur la figure 8 que la planification avec ATLAS donne douze prises de vues pour A et onze pour B. Enfin, nous constatons aussi que la planification via la méthode gloutonne entraîne davantage de périodes d'inactivités. En effet, ChronoG n'étant pas coopératif mais reposant sur une heuristique, les requêtes attribuées bloquent des enchaînements qui auraient permis d'ordonnancer davantage de tâches.

4.4 Discussion

Les systèmes actuels sont basés sur une heuristique globale et sont dépendants de cette dernière. Changer les entités du problème impose de modifier certains paramètres. Dans notre système multi-agent, ce sont la criticité des agents mailles et le coût de planification émit par les agents satellites qui guident la recherche. Or ces deux critères sont déduits du fonctionnement normal des entités et définis afin de mettre en avant la coopération, on peut donc changer les entités (les caractéristiques des satellites par exemple) sans avoir à les modifier.

Une autre limitation est l'approche *réductionniste* des méthodes utilisées pour planifier les constellations. Actuellement, les satellites sont planifiés individuellement, et ne tirent pas profit des avantages que présentent les constellations. Les agents composant ATLAS coopèrent pour planifier efficacement la constellation dans son ensemble. Le coût et la criticité sont donc là aussi deux puissants outils.

Enfin, l'ouverture et le dynamisme du système sont deux points intéressants. Dans les systèmes de planification actuellement utilisés, l'ajout et la suppression de requêtes durant le calcul du plan est une opération particulièrement difficile. C'est d'autant plus problématique dans une approche chronologique comme ChronoG : les requêtes sont traitées dans leur ordre d'accès. Supprimer ou ajouter une requête implique de recalculer le plan à partir de la date d'accès de la requête, et donc perdre du temps de calcul. Les requêtes urgentes, qui arrivent en cours de planification ne peuvent pas toujours être prise en compte, en fonction du temps restant. Au contraire, ATLAS est lui un système ouvert. L'ajout ou la suppression d'une requête entraîne une perturbation locale. Les agents vont la prendre en compte et s'adapter pour converger vers une nouvelle solution. Nos prochains travaux concerneront cette auto-adaptation.

5 Conclusion et perspectives

Dans cet article, nous avons présenté une approche basée sur les systèmes multi-agents adaptatifs pour planifier efficacement une constellation de satellites d'observation de la Terre : ATLAS. Dans ce système, la coopération des agents est guidée par la criticité des agents mailles, et par le coût indiqué par les agents satellites. Ces deux moteurs de la coopération permettent ainsi d'assurer qu'un nombre de requêtes important soit planifié de façon équilibrée entre les satellites de la constellation. Les différentes expérimentations ont montré qu'ATLAS fournissait des résultats de meilleure qualité que l'algorithme Glouton Chronologique couramment utilisé (ChronoG). De plus, ATLAS fournit des solutions ayant un meilleur équilibrage de la charge de la constellation. Ces premiers résultats sont donc très encourageants et prouvent l'intérêt d'une approche par système multi-agent pour planifier efficacement une constellation de satellites.

Nos futurs travaux vont concerner la prise en compte de la dynamique et l'auto-adaptation du système. Nous envisageons aussi de tester ATLAS sur des cas d'utilisations réels. Ainsi, nous pourrons comparer nos résultats avec des plans de mission produits par le CNES *(Centre National d'Études Spatiales)* ou Airbus D&S-Geo (anciennement Spot Image), deux organismes qui gèrent les constellations des satellites Spot et Pléiades.

Remerciements

Les auteurs souhaitent remercier l'IRT Saint Exupéry pour le financement de cette recherche.

Références

Bensana, E. and Verfaillie, G. (1999). Earth Observation Satellite Management. In *Constraints*, volume 299, pages 293–299.

Bianchessi, N., Cordeau, J. F., Desrosiers, J., Laporte, G., and Raymond, V. (2007). A heuristic for the multi-satellite, multi-orbit and multi-user management of Earth observation satellites. *European Journal of Operational Research*, 177(2) :750–762.

Bonnet, G. (2008). *Coopération au sein d'une constellation de satellites.* PhD thesis.

Bonnet, G. and Tessier, C. (2009). Évaluation d'un système multirobot cas d'une constellation de satellites. *Revue d'Intelligence Artificielle*, 23 :565–593.

Bouziat, T., Combettes, S., Camps, V., and Glize, P. (2014). La criticité comme moteur de la coopération dans les systèmes multi-agents adaptatifs (short paper). In *Journées Francophones sur les Systèmes Multi-Agents, 2014*, pages 149–158.

Gleizes, M.-P. (2012). Self-adaptive Complex Systems (regular paper). In *European Workshop on Multi-Agent Systems, Maastricht, The Netherlands*, volume 7541, pages 114–128.

Globus, A., Crawford, J., Lohn, J., and Pryor, A. (2003). Scheduling earth observing satellites with evolutionary algorithms. In *Conference on Space Mission Challenges for Information Technology*.

Grasset-Bourdel, R. (2011). Planification dynamique et réactive pour des satellites agiles d'observation de la Terre.

Grasset-Bourdel, R., Flipo, A., and Verfaillie, G. (2011). Planning and replanning for a constellation of agile Earth observation satellites.

Kaddoum, E. (2011). *Optimization under Constraints of Distributed Complex Problems using Cooperative Self-Organization*. PhD thesis.

Lemaître, M., Verfaillie, G., Jouhaud, F., Lachiver, J. M., and Bataille, N. (2002). Selecting and scheduling observations of agile satellites. *Aerospace Science and Technology*, 6(5) :367–381.

Mancel, C. (2004). *Modélisation et résolution de problèmes d'optimisation combinatoire issus d'application spatiales*. PhD thesis.

Mansour, M. A. and Dessouky, M. M. (2010). A genetic algorithm approach for solving the daily photograph selection problem of the spot5 satellite. *Computers & Industrial Engineering*, 58(3) :509–520.

Wu, G., Wang, H., Li, H., Pedrycz, W., Qiu, D., Ma, M., and Liu, J. (2014). An adaptive Simulated Annealing-based satellite observation scheduling method combined with a dynamic task clustering strategy. *Computing Research Repository*, abs/1401.6098 :23.

Délégation GPU des perceptions agents : application aux boids de Reynolds

Emmanuel Hermellin[a]
emmanuel.hermellin@lirmm.fr

Fabien Michel[a]
fmichel@lirmm.fr

[a]LIRMM - Université de Montpellier - CNRS
161 rue Ada, 34090 Montpellier, France

Résumé

L'utilisation du GPGPU (General-Purpose Computing on Graphics Processing Units) pour la simulation multiagent permet d'améliorer les performances des modèles et lève une partie des contraintes liées au passage à l'échelle. Cependant, adapter un modèle pour qu'il utilise le GPU est une tâche complexe car le GPGPU repose sur une programmation extrêmement spécifique et contraignante. C'est dans ce contexte que la délégation GPU des perceptions agents a été proposée. Ce principe consiste à réaliser une séparation claire entre les comportements de l'agent (gérés par le CPU) et les dynamiques environnementales (manipulées par le GPU) dans le but de faciliter l'implémentation de SMA sur GPU. Il a été appliqué sur un cas d'étude et a montré de bons résultats en termes de performances et de conception. Dans cet article, nous proposons de tester la généricité de cette approche en appliquant le principe de délégation GPU sur un modèle agent très classique : les boids de Reynolds. Nous montrons que le principe de délégation offre des résultats intéressants au niveau des performances mais aussi d'un point de vue conceptuel.

Mots-clés : *SMA, GPGPU, Flocking, CUDA*

Abstract

General-Purpose Computing on Graphics Processing Units (GPGPU) allows to extend the scalability and performances of Multi-Agent Based Simulations (MABS). However, GPGPU requires the underlying program to be compliant with the specific architecture of GPU devices, which is very constraining. In this context, the GPU Environmental Delegation of Agent Perceptions principle has been proposed to ease the use of GPGPU for MABS. This principle consists in making a clear separation between the agent behaviors, managed by the CPU, and environmental dynamics, handled by the GPU. For now, this principle has shown good results, but only on one single case study. In this paper, we further trial this principle by testing its feasibility and genericness on a classic ABM, namely Reynolds's boids. To this end, we first review existing boids implementations and propose our own benchmark model. The paper then shows that applying GPU delegation not only speeds up boids simulations but also produces an ABM which is easy to understand, thanks to a clear separation of concerns.

Keywords: *GPGPU, MABS, Flocking, CUDA*

1 Introduction

La *délégation GPU des perceptions agents* (que nous nommerons *délégation GPU*) fait partie des approches visant à faciliter l'utilisation du GPGPU (*General-Purpose Computing on Graphics Processing Units*) dans le domaine des simulations multiagents. Réelle révolution technologique, le GPGPU utilise l'architecture massivement parallèle des cartes graphiques pour effectuer du calcul généraliste. Il est ainsi possible d'utiliser des ordinateurs classiques pour accélérer les simulations multiagents [6]. Cependant, de par l'architecture matérielle très spécifique des GPU (*Graphics Processing Unit*), ce type de programmation requiert des connaissances avancées qui limitent son utilisation [9].

La *délégation GPU* est basée sur une approche hybride qui repose sur une utilisation conjointe du CPU (*Central Processing Unit*) et du GPU. En offrant la possibilité de sélectionner ce qui va être adapté puis exécuté par le GPU, les approches hybrides offrent une solution attractive pour contourner les difficultés occasionnées par le GPGPU [4]. La *délégation GPU* consiste à déplacer dans l'environnement certains calculs effectués par les agents au sein de leur propre comportement. Un cas d'étude a été proposé dans [7] et montre de bons résultats en terme de performances, d'accessibilité et de réutilisabilité.

Dans le but de tester la généricité et les avan-

tages que peut apporter ce principe, nous appliquons, dans cet article, la *délégation GPU* sur un modèle classique de SMA : les *boids* de Reynolds [13].

La section 2 présente le modèle de Reynolds et son implémentation dans différentes plates-formes SMA. Nous introduisons notre propre modèle en section 3. La section 4 se focalise sur le principe de *délégation GPU des perceptions agents*. La section 5 décrit l'application de la *délégation GPU* sur notre modèle de *flocking* ainsi que son implémentation. Les résultats sont présentés et discutés en section 6. Enfin, la section 7 conclut l'article et présente des perspectives de recherche qui lui sont associées.

2 Les boids de Reynolds

2.1 Présentation du modèle original

Reynolds a remarqué qu'il n'était pas possible d'utiliser des scripts pour réaliser des animations réalistes de nuées d'oiseaux artificiels (*boids*) [13]. Son idée a donc été la suivante : les *boids* doivent être influencés par les autres pour se déplacer d'une manière cohérente et crédible. La citation issue de [13] résume cette approche : "*Boid behavior is dependant not only on internal state but also on external state*".

Chaque agent va ainsi suivre trois règles comportementales :

- **R.1** *Collision Avoidance* : éviter les collisions entre entités ;
- **R.2** *Flock Centering* : rester le plus proche possible des autres entités ;
- **R.3** *Velocity matching* : adapter sa vitesse à celles des autres entités.

Le modèle de *flocking* de Reynolds est l'une des simulations multiagents la plus représentative et connue. Ainsi, de nombreuses plates-formes spécialisées dans le développement de SMA l'intègrent en proposant leur propre implémentation.

2.2 Les boids dans les plates-formes SMA

Dans cette section, nous comparons plusieurs implémentations du modèle de Reynolds. Parmi tous les travaux trouvés, nous sélectionnons uniquement les modèles pouvant être testés et qui mettent à disposition leur code source : NetLogo, StarLogo, Gama, Mason et FlameGPU. Pour chaque modèle, nous décrirons comment l'implémentation des trois lois de Reynolds a été réalisée.

NetLogo Dans NetLogo [1] [16], tous les agents (*turtle*) se déplacent et essayent de se rapprocher les uns des autres. Si la distance entre eux est trop faible, ils tentent de se dégager pour éviter de rentrer en collision (R.1), sinon ils s'alignent (R.2). Cependant, R.3 n'est pas implémenté ; il n'y a aucune gestion de la vitesse qui reste ainsi constante tout au long de la simulation.

StarLogo Dans StarLogo [2] [12], chaque agent recherche son plus proche voisin. Si la distance entre eux est trop faible, (R.1) il tourne et s'éloigne pour éviter de rentrer en collision. Sinon, il s'approche de lui et s'aligne sur sa direction de déplacement. La recherche de cohésion n'est pas explicitement exprimée (R.2) et comme pour le modèle précédent, la gestion de la vitesse n'est pas présente (R.3).

Gama Dans Gama [3] [3], les agents commencent par rechercher une cible (assimilable à un but) à suivre. Une fois la cible acquise, les agents se déplacent grâce à trois fonctions indépendantes qui implémentent les règles de Reynolds : une fonction pour éviter les collisions (R.1), une fonction de cohésion (R.2) et une fonction permettant d'adapter la vitesse des agents (R.3). Ce modèle diffère de celui présenté par Reynolds car les agents ont besoin d'une cible pour avoir un comportement de *flocking*.

MasOn MasOn [4] [5] utilise un ensemble de vecteurs pour implémenter R.1 et R.2. Ainsi, le mouvement de chaque agent est calculé à partir d'un vecteur global, ce dernier étant composé d'un vecteur d'évitement, d'un vecteur de cohésion (un vecteur dirigé vers le "centre de masse" du groupe d'entités (R.2)), d'un vecteur moment (un vecteur du déplacement précédent), d'un vecteur de cohérence (un vecteur du mouvement global) et d'un vecteur aléatoire. La vitesse est ici aussi constante pendant toute la simulation, R.3 n'étant pas implémentée.

FlameGPU FlameGPU [5] [15] est la seule implémentation GPGPU que nous avons pu tester. Dans ce modèle, R.1, R.2 et R.3 sont implémentées dans trois fonctions indépendantes. La particularité de ce framework est la nécessité d'adopter un formalisme de conception de

1. https ://ccl.northwestern.edu/netlogo/
2. http ://education.mit.edu/starlogo/
3. https ://code.google.com/p/gama-platform/
4. http ://cs.gmu.edu/˜eclab/projects/mason/
5. http ://www.flamegpu.com/

Plate-forme	Implémentation règles de Reynolds			Caractéristiques principales	Performances
	Collision R.1	Cohésion R.2	Vitesse R.3		
NetLogo	X	X		R.3 n'est pas implémentée : la vitesse des agents est fixée pendant toute la simulation	214 ms (CPU / Logo)
StarLogo	X			Implémentation minimaliste (seul l'évitement d'obstacle est implémenté)	*1000 ms (CPU / Logo)
Gama	X	X	X	Comportement de *flocking* seulement lorsque les agents acquièrent une cible	375 ms (CPU / GAML)
MasOn	X	X		Les règles R.1 et R.2 sont réinterprétées en un calcul de vecteur global qui intègre de l'aléatoire, aucune gestion de la vitesse	45 ms (CPU / Java)
Flame GPU	X	X	X	Les trois règles sont respectées et implémentées telles que définies par Reynolds	*82 ms (GPU / C, XML)

TABLE 1 – Le flocking dans les plate-formes SMA

SMA, basé sur les langages XML et C, qui n'est pas intuitif. Le but est de cacher à l'utilisateur toute la partie GPGPU.

Résumé Dans le but d'avoir un aperçu global des différentes implémentations des *boids* de Reynolds au sein des plates-formes SMA, le tableau 1 résume pour chaque modèle les règles de Reynolds implémentées, énonce les caractéristiques principales des modèles et donne des informations de performances.

Performances Nous évaluons pour chaque modèle le temps de calcul moyen en millisecondes pour une itération. Le but de cette évaluation est de donner une idée des possibilités de chaque implémentation. Ainsi, nous utiliserons comme paramètre commun un environnement de 512 par 512 contenant 4000 agents. Notre configuration de test est composée d'un processeur Intel Core i7 (génération Haswell, 3.40GHz) et d'une carte graphique Nvidia Quadro K4000 (768 cœurs).

Il faut noter que pour StarLogo, nous avons observé un temps de calcul d'une seconde dès 400 agents simulés. Les performances étant très en dessous des autres plates-formes, nous n'avons pas poussé les tests plus loin. Enfin, pour FlameGPU, il n'a pas été possible de modifier le nombre d'agents dans la simulation qui est de 2048.

3 Proposition d'un modèle de boids

De l'étude précédente, nous remarquons des disparités entre les différents modèles présentés. En effet, les règles de *flocking* proposées par Reynolds autorisent une grande variété d'interprétations. Ainsi, nous remarquons que la règle pour l'adaptation de la vitesse (R.3) est la moins prise en compte (en comparaison de R.1 et R.2 implémentées dans chaque modèle vu à l'exception de StarLogo). Cependant, lorsque R.3 est implémentée, les comportements collectifs deviennent beaucoup plus convaincants et le mouvement global possède alors une dynamique et une fluidité plus intéressante. De même, dans certains travaux, les comportements d'alignement et de cohésion sont confondus ou fusionnés. Les modèles explicitant la différence entre ces deux comportements offrent cependant des déplacements plus intéressants.

Le modèle que nous proposons prendra en compte les points intéressants observés précédemment. Nous avons en effet remarqué que lorsque les trois règles sont intégrées, la dynamique et le mouvement des agents sont plus intéressants. Cela est d'autant plus vrai quand R.3 est prise en compte. Ainsi, notre modèle intègrera R.1, R.2 et R.3 et suivra le principe de parcimonie dans le but de créer une version minimaliste (avec le moins de paramètres possible) se focalisant sur la vitesse et l'orientation de l'agent [6].

Chaque entité a un comportement global qui consiste à se déplacer dans l'environnement tout en adaptant sa vitesse et sa direction en fonction de ses voisins. Ainsi, la proximité avec les autres agents est testée et selon la distance trouvée, les différentes règles de Reynolds sont activées. Plus précisément, chaque agent vérifie si il n'a pas de voisin. Si aucun agent n'est présent dans son champ de vision, il continue à se déplacer dans la même direction. Sinon, l'agent vérifie sa proximité avec ses voisins. Selon la distance trouvée, l'agent va soit se séparer (R.1) s'ils sont trop proches, s'aligner si le nombre de voisins est inférieur à un seuil de cohésion ou

6. L'orientation est un angle en degré (entre 0 et 360) qui donne la direction de l'agent en fonction du repère fixé dans l'environnement

former un groupe dans le cas ou le nombre de voisins est supérieur au seuil défini (R.2). Ensuite, l'agent adapte sa vitesse (R.3), se déplace et recommence le processus. La figure 1 illustre ce comportement global.

Dans notre modèle, il existe deux types différents de paramètres : 5 constantes pour le modèle et 3 attributs spécifiques aux agents. Les constantes sont les suivantes :

- *fieldOfView* (le champ de vision de l'agent) ;
- *minimalSeparationDistance* (la distance minimum entre deux agents) ;
- *cohesionThreshold* (le nombre minimal de voisins pour déclencher un comportement de cohésion) ;
- *maximumSpeed* (la vitesse maximum) ;
- *maximumRotation* (l'angle maximum de rotation).

Les attributs spécifiques à chaque agent sont les suivants :

- *heading* (son orientation) ;
- *velocity* (sa vitesse) ;
- *nearestNeighborsList* (la liste des voisins présents dans son champ de vision).

Comportement de séparation R.1 Ce comportement consiste en la récupération des deux directions (celle de l'agent et de son plus proche voisin). Si ces deux directions mènent à une collision, les deux agents tournent pour s'éviter (voir l'algorithme 1).

Comportement d'alignement R.2 L'alignement se produit lorsque deux agents se rapprochent l'un de l'autre. Ils vont dans ce cas adapter leur orientation de mouvement pour s'aligner et ainsi se diriger vers la même direction (voir l'algorithme 2).

Comportement de cohésion R.2 Quand plusieurs agents sont proches sans avoir besoin de se séparer, ils ont un comportement de cohésion. Ce dernier consiste à calculer la direction moyenne de tous les agents présents dans le champ de vision. Chaque agent va ensuite adapter son orientation en fonction de la valeur trouvée (voir l'algorithme 3).

Adaptation de la vitesse R.3 Avant de se déplacer, les agents doivent adapter leur vitesse (R.3). Durant toute la simulation, chaque agent modifie sa vitesse en fonction de celle de ses voisins. Si l'agent vient d'exécuter le comportement de séparation (R.1), il accélère pour se dégager plus rapidement. Sinon, l'agent ajuste sa vitesse pour la faire correspondre à celle de ses voisins (dans la limite autorisée par la constante *maximumSpeed*).

Tester notre modèle Nous avons mis en ligne un ensemble de vidéos qui montrent notre modèle en action [7]. Sur cette page sont aussi disponibles les codes sources des différents modèles mentionnés et les ressources nécessaires pour tester la solution.

4 Délégation GPU des perceptions agents

4.1 Simulations multiagents et GPGPU

Notions relatives au GPGPU Pour comprendre le principe de programmation associé au GPGPU, il faut avoir à l'esprit qu'il est fortement lié à l'architecture matérielle massivement multicœur des GPU. Le CPU (*host*) gère la répartition des données et exécute les *kernels* : des fonctions spécialement créées pour s'exécuter sur le GPU (*device*). Le GPU est capable d'exécuter un *kernel* de manière parallèle grâce aux *threads* (les fils d'instructions). Ces *threads* sont regroupés par *blocs* (les paramètres *blockDim.x*, *blockDim.y* définissent la taille de ces blocs), qui sont eux-mêmes rassemblés dans une *grille globale* (la figure 2 donne un exemple de l'utilisation d'une grille 2D). Chaque *thread* au sein de cette structure est identifié par des coordonnées uniques 3D (*threadIdx.x*, *threadIdx.y*, *threadIdx.z*) lui permettant d'être localisé. De la même façon, un *bloc* peut être identifié par ses coordonnées dans la *grille* (respectivement *blockIdx.x*, *blockIdx.y*, *blockIdx.z*). Les *threads* [8] exécutent le même *kernel* mais traitent des données différentes selon leur localisation spatiale (identifiant).

Techniques d'implémentation Dans [4], nous avons réalisé un état de l'art de l'utilisation du GPGPU dans les SMA et identifié deux approches permettant d'implémenter un modèle multiagent sur GPU : (1) *tout-sur-GPU* qui consiste à exécuter entièrement la simulation sur la carte graphique et (2) *hybride* pour laquelle la simulation est partagée entre le CPU et le GPU. [4] montre que l'approche hybride

7. www.lirmm.fr/∼hermellin/Website/Reynolds_Boids_With_TurtleKit.html

8. Le terme thread s'apparente ici à la notion de tâche : un *thread* peut être considéré comme une instance du *kernel* qui s'effectue sur une partie restreinte des données en fonction de son identifiant, c'est-à-dire suivant sa localisation dans la grille globale.

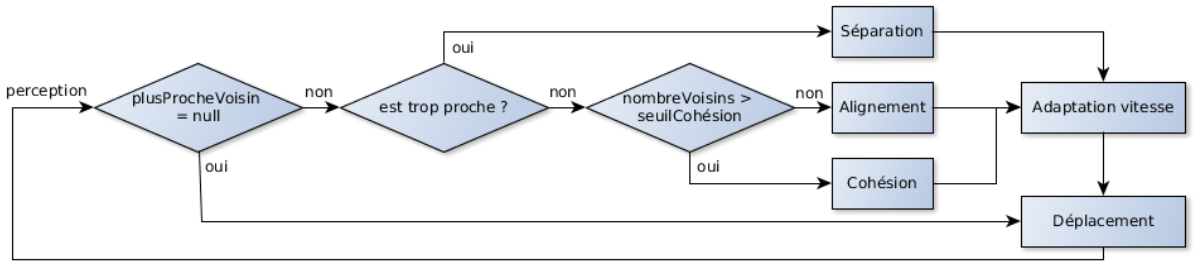


FIGURE 1 – Flocking : comportement global

Algorithme 1 : Séparation

entrées : $myHeading$, $nearestBird$, $maximumRotation$
sortie : $myHeading$ (la nouvelle orientation de l'agent)

$collisionHeading \leftarrow$ headingToward($nearestBird$) ;
si $myHeading$ *seSitueDansInterval*($collisionHeading, maximumRotation$) **alors**
  changeHeading($myHeading$);
fin
return $myHeading$

Algorithme 2 : Alignement

entrées : $myHeading$, $nearestBird$
sortie : $myHeading$ (la nouvelle orientation de l'agent)

$nearestBirdHeading \leftarrow$ getHeading($nearestBird$) ;
si $myHeading$ *estProche*($nearestBirdHeading$) **alors**
  adaptHeading($myHeading$);
fin
sinon
  adaptHeading($myHeading, maximumRotation$);
fin
return $myHeading$

Algorithme 3 : Cohésion

entrées : $myHeading$, $nearestNeighborsList$
sortie : $myHeading$ (la nouvelle orientation de l'agent)

$sumOfHeading, neighborsAverageHeading = 0$;
pour $bird$ *in* $nearestNeighborsList$ **faire**
  $sumOfHeading += getHeading(bird)$;
fin
$neighborsAverageHeading = sumOfHeading/sizeOf(nearestNeighborsList)$;
si $myHeading$ *estProche*($neighborsAverageHeading$) **alors**
  adaptHeading($myHeading$);
fin
sinon
  adaptHeading($myHeading, maximumRotation$);
fin
return $myHeading$

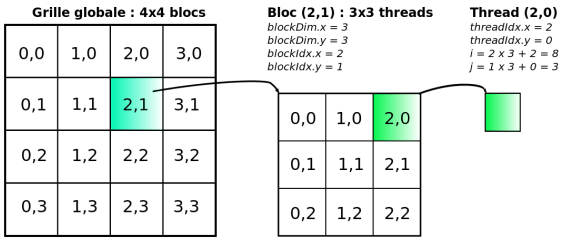


FIGURE 2 – Thread, blocks et grille

apparaît comme la plus prometteuse car elle autorise une plus grande flexibilité (il est possible de choisir ce qui va être exécuté sur le GPU) et augmente l'accessibilité des outils développés.

4.2 Délégation GPU des perceptions agents

Le principe La *délégation GPU des perceptions agents* est basée sur une approche hybride et a été proposée dans [7]. Ce principe consiste en la réalisation d'une séparation explicite entre le comportement des agents, géré par le CPU, et les dynamiques environnementales traitées par le GPU. L'idée sous-jacente est d'identifier dans les comportements des agents des calculs qui peuvent être transformés en dynamiques environnementales. Ce principe de conception a été énoncé de la manière suivante : "*Tout calcul de perception agent qui n'implique pas l'état de l'agent peut être transformé dans une dynamique endogène de l'environnement, et ainsi considéré pour une implémentation dans un module GPU indépendant*".

Travaux connexes Cette approche de *délégation GPU* peut être rapprochée d'autres travaux visant à séparer et/ou déplacer une partie des calculs effectués par les agents dans d'autres structures tels que les interactions ou l'environnement dans le but de répartir la complexité du code et de modulariser son implémentation.

Par exemple, dans le cadre des simulations multiagents, l'approche EASS (*Environment As Active Support for Simulation*)[1] vise à renforcer le rôle de l'environnement en lui déléguant la politique d'ordonnancement ainsi qu'un système de filtrage des perceptions. IODA (*Interaction Oriented Design of Agent simulations*) [11] est quant à elle centrée sur la notion d'interaction et considère que tout comportement réalisable par des agents est décrit de façon abstraite (c'est-à-dire en exprimant ce qu'il a de général) sous la forme d'une règle appelée interaction. Enfin, [10] propose une réduction de la complexité des modèles en se basant sur une approche environnement-centrée : l'environnement devient alors un espace d'échange dédié à l'exécution de dynamiques visant à faciliter la réutilisabilité et l'intégration des différents processus des agents.

Dans un contexte plus générale, l'approche des *artefacts* intègre dans l'environnement un ensemble d'entités dynamiques représentant les ressources et les outils que les agents vont pouvoir utiliser et partager [14, 17]. Ces entités, appelées artefacts, structurent et organisent l'environnement en proposant un modèle de programmation générique englobant les fonctionnalités auxquelles les agents vont avoir accès.

La délégation GPU sur un cas d'étude L'intégration du calcul sur GPU à été réalisée dans TurtleKit [9] [8] qui est une plate-forme de simulation multiagent générique, implémentée en Java, qui utilise un modèle multiagent spatialisé où l'environnement est discrétisé sous la forme d'une grille de cellules. L'approche hybride présente dans TurteKit se focalise sur la modularité et permet d'atteindre plusieurs objectifs : (1) conserver l'accessibilité du modèle agent dans un contexte GPU, (2) passer à l'échelle et travailler avec un grand nombre d'agents sur de grandes tailles d'environnement et (3) promouvoir la réutilisabilité des travaux effectués.

La *délégation GPU* n'a été appliquée pour l'instant que sur un seul modèle dans TurtleKit : un modèle d'émergence multi-niveaux (MLE) [2]. Ce modèle très simple repose sur un unique comportement qui permet de générer des structures complexes qui se répètent de manière fractale. Le comportement agent correspondant est extrêmement simple et repose uniquement sur la perception, l'émission et la réaction à des phéromones. Ainsi, dans ces travaux, des modules GPU pour la perception et la diffusion des phéromones ont été proposés.

5 Délégation GPU et flocking

5.1 Application de la délégation GPU

La *délégation GPU* permet de convertir des comportements ne faisant pas intervenir l'état de l'agent en dynamiques environnementales. Dans notre modèle de *flocking*, le comportement de cohésion se prête à l'application du principe de délégation. En effet, ce comportement

9. http ://www.turtlekit.org

Algorithme 4 : Calcul parallèle : *Kernel* Average

entrées : $width$, $height$, $fieldOfView$, $headingArray$ and $nearestNeighborsList$
sortie : $flockCentering$ (la moyenne des directions)

$i = blockIdx.x * blockDim.x + threadIdx.x$;
$j = blockIdx.y * blockDim.y + threadIdx.y$;
$sumOfHeading, flockCentering = 0$;
si $i < width$ *et* $j < height$ **alors**
  $sumOfHeading =$ getHeading($fieldOfView$, $headingArray[i, j]$);
fin
$flockCentering[i, j] = sumOfHeading/sizeOf(nearestNeighborsList)$;

consiste à réaliser la moyenne des orientations des agents en fonction du champ de vision [10]. Tous les agents doivent réaliser ce calcul qui consiste en une récupération d'une liste de voisins (*nearestNeighborsList*) et d'un parcours séquentiel de la liste pour calculer la moyenne des orientations :

pour $bird$ *in* $nearestNeighborsList$ **faire**
  $sumOfHeading+ = getHeading(bird)$;
fin
$neighborsAverageHeading =$
$sumOfHeading/sizeOf(nearestNeighborsList)$;

Ce parcours de boucle est lourd car effectué par tous les agents dans leur propre comportement et à chaque pas de simulation.

5.2 Traduction GPU du calcul de la moyenne des orientations

Le calcul de la moyenne est indépendant de l'état des agents et peut être déporté dans l'environnement. Pour ce faire, un tableau 2D (*headingArray*, correspondant à la grille de l'environnement), stocke l'orientation de tous les agents en fonction de leur position. Ce tableau est ensuite envoyé au module GPU *Average Kernel* qui se charge du calcul des orientations moyennes. La traduction GPU consiste donc à transformer le calcul de boucle séquentiel précédemment effectué dans le comportement de cohésion des agents par un calcul parallèle effectué sur le GPU et géré par l'environnement. En fonction du champ de vision de l'agent (*fieldOfView*), le module calcule de manière simultanée la moyenne pour tout l'environnement. Plus précisément, chaque *thread* du GPU calcule la moyenne des orientations d'une cellule selon sa propre position dans la grille GPU (ses identifiants *i* et *j*, algorithme 4). Une fois réalisées, les orientations moyennes sont disponibles dans tout l'environnement. Les agents n'ont donc plus qu'à récupérer dans un tableau 2D (*flockCentering*, retourné par le module GPU) la valeur correspondant à leur position et à adapter leur mouvement.

L'algorithme 4 présente une implémentation du module GPU. Après avoir initialisé les coordonnées *i* et *j* du *thread* utilisé et les variables temporaires (*sumOfheading* et *flockCentering*), on test si le *thread* ne possède pas des coordonnées supérieures à la taille de l'environnement (représenté ici par le tableau 2D *headingArray*). On ajoute ensuite dans *sumOfheading* l'ensemble des orientations des voisins se trouvant dans le champs de vision puis on divise cette valeur par le nombre de voisins pris en compte. Le module retourne ensuite le tableau *flockCentering* contenant toutes les moyennes.

Par rapport à la version séquentielle de l'algorithme, on voit que la boucle a disparu. Ainsi tout l'intérêt de la version GPU tient dans le fait que la parallélisation de cette boucle est réalisée grâce à l'architecture matérielle.

5.3 Implémentation et intégration du module Average

L'implémentation du module GPU dans TurtleKit a été réalisée en CUDA [11] et JCuda [12]. La figure 3 illustre l'application du principe sur notre modèle au sein de TurtleKit.

10. Dans TurtleKit, on appelle champ de vision le nombre de cellules (le rayon autour de la cellule sélectionnée) à prendre en compte pour le calcul de la moyenne.

11. CUDA (Compute Unified Device Architecture), version utilisée : 6.5. http ://www.nvidia.fr/object/cuda-parallel-computing-fr.html

12. La librairie JCuda autorise l'appel de *kernels* GPU, écrits en CUDA, directement depuis Java. Version utilisée : 0.6.5

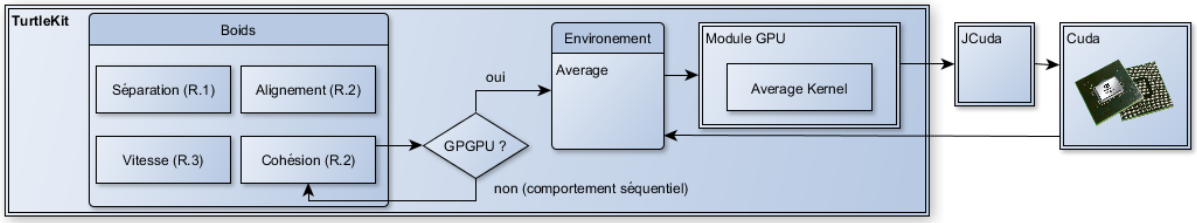


FIGURE 3 – Délégation de la moyenne dans TurtleKit

6 Expérimentation

6.1 Protocole expérimental

Afin de tester notre implémentation des boids de Reynolds et l'application du principe de délégation GPU associée, nous simulons plusieurs tailles d'environnement tout en faisant varier le nombre d'agents et exécutons successivement la version séquentielle du modèle (où la moyenne est calculée dans le comportement des agents) puis la version GPGPU (utilisant le module GPU *Average*). Pour évaluer la performance et rester cohérent avec les critères d'analyse des modèles de la section 2, nous relevons le temps de calcul moyen en millisecondes pour une itération.

6.2 Tests de performances

Pour ces tests, nous réutilisons la même configuration que celle utilisée en section 2 et composée d'un processeur Intel Core i7 (génération Haswell, 3.40GHz), d'une carte graphique Nvidia Quadro K4000 (768 cœurs CUDA) et de 16Go de RAM. La Figure 4 présente les résultats obtenus pour diverses populations d'agents dans des environnements de 256 x 256 et 512 x 512.

Dans l'environnement de 256 x 256, on observe un gain de performance atteignant jusqu'à 25%. Pour l'environnement de 512 x 512, le gain observé est de 15% au maximum. Cette différence de performances s'explique car la densité des agents dans l'environnement est plus faible. Les agents passent donc moins de temps en cohésion et plus à s'aligner et se séparer. Ainsi, selon la densité des agents présents, l'utilisation du module GPU affectera les performances du modèle. Le basculement est clairement visible dans les résultats, lorsque la densité d'agents présents dépasse 5% (respectivement 1500 et 8000 entités), l'utilisation conjointe du CPU et du GPU devient plus efficace. Ainsi, plus la densité d'agents dans l'environnement augmente et plus les gains de performances observés sont importants.

Il faut aussi prendre en compte que les performances obtenues sont intéressantes si l'on considère le matériel utilisé : notre carte graphique Nvidia Quadro K4000 n'est composée que de 768 cœurs CUDA alors que la Nvidia Tesla K40, en contient 2880 et que la Nvidia Tesla K10 en a 3072 (deux GPU de 1536 cœurs sur la même carte).

6.3 Discussion

En plus des résultats de performance observés, on remarque d'autres avantages à l'application du principe de *délégation GPU* : la traduction d'une perception calculée dans le comportement de l'agent en une dynamique de l'environnement permet d'enlever une partie du code source du comportement de l'agent, ce qui simplifie la compréhension de la règle R.2. En effet, l'agent effectue alors une perception directe dans l'environnement à la place d'un calcul séquentiel pouvant être assez lourd.

Un autre aspect intéressant vient du fait que les modules créés, grâce à cette approche, sont indépendants des modèles. Ils ne sont donc pas limités aux contextes pour lesquels ils ont été définis. Nous allons donc continuer d'appliquer le principe de *délégation GPU* pour créer de nouveaux modules GPU et ainsi augmenter le nombre de modules GPU génériques indépendants disponibles ce qui permettra de constituer à terme une librairie de modules utilisable quelque soit le modèle simulé. Cette bibliothèque de fonctions GPU améliorera l'accessibilité de l'approche et l'utilisation du GPGPU dans le cadre des simulations multiagents avec TurtleKit. Cette avancée possible en terme de généricité et d'accessibilité est importante car travailler dans un contexte GPGPU amène souvent des difficultés d'implémentations de par la spécificité de cette technologie.

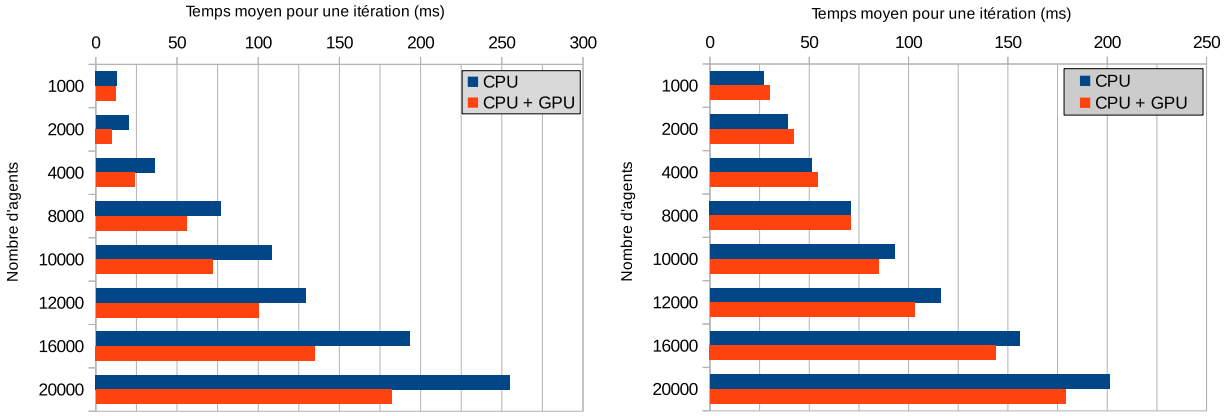


FIGURE 4 – Performance du modèle pour des environnements de taille 256 (gauche) et 512 (droite)

L'application du principe de *délégation GPU* se base sur un critère simple indépendant de l'implémentation. Cela permet de convertir le modèle et de créer le(s) module(s) GPU de manière assez rapide. TurtleKit étant encore en version alpha, nous allons continuer de travailler sur son architecture pour que la conversion d'un modèle soit la plus simple possible.

7 Conclusion et perspectives

Dans cet article, nous avons décrit comment la *délégation GPU des perceptions agents* pouvait être utilisée pour implémenter les boids de Reynolds, en utilisant le GPGPU. Notre objectif était de tester la généricité et les avantages que peut apporter cette approche. Après avoir effectué une analyse des différentes implémentations des modèles de *flocking* au sein des plates-formes multiagents, nous avons proposé notre propre version du modèle. Il a été possible d'appliquer le principe de *délégation GPU* sur le comportement de cohésion. Nous avons ensuite traduit ce comportement en une dynamique environnementale et réalisé des tests de performances.

Nos expérimentations ont montré que la *délégation GPU* a permis d'augmenter le nombre d'agents ainsi que la taille de l'environnement grâce à une accélération de la simulation pouvant atteindre 25% selon les paramètres choisis.

Le principe de délégation représente un modèle de développement qui permet de promouvoir la réutilisabilité des outils créés. Ce critère essentiel est souvent délaissé dans un contexte GPGPU [4]. D'un point de vue génie logiciel, l'utilisation de la *délégation GPU* permet une séparation explicite entre le modèle agent (les comportements de l'agent) et les dynamiques environnementales. L'application du principe autorise ainsi la création de modules GPU génériques indépendants du modèle agent.

Les deux implémentations du principe de délégation, réalisées avec MLE dans [7] et le modèle de *flocking* ici, ont montré que si l'analyse du modèle est faite en gardant à l'esprit les caractéristiques de l'approche, l'identification des comportements, la délégation des calculs et la création des modules GPU peuvent être faciles et rapides. La *délégation GPU des perceptions* nécessitant encore des compétences spécifiques, nous comptons appliquer sur d'autres modèles ce principe dans le but d'éprouver et de continuer à généraliser l'approche.

À long terme notre objectif est donc de proposer une méthodologie explicite de conception, un guide de développement consistant à rendre l'utilisation de la *délégation GPU* plus explicite et plus accessible à des utilisateurs externes. L'idéal étant au final de permettre à chacun de prendre un modèle et l'adapter pour le faire fonctionner dans un contexte GPGPU.

Références

[1] Fabien Badeig and Flavien Balbo. Définition d'un cadre de conception et d'exécution pour la simulation multi-agent. *Revue d'Intelligence Artificielle*, 26(3) :255–280, 2012.

[2] Gregory Beurier, Olivier Simonin, and Jacques Ferber. Un modele de systeme multi-agents pour l'emergence multi-niveaux. In *11eme journees Francophones sur les Systemes Multi-Agents*, Revue des

Sciences et Technologies de l'Information, pages 235–247. Hermes, 2003.

[3] Arnaud Grignard, Patrick Taillandier, Benoit Gaudou, DucAn Vo, NghiQuang Huynh, and Alexis Drogoul. GAMA 1.6 : Advancing the Art of Complex Agent-Based Modeling and Simulation. In Guido Boella, Edith Elkind, BastinTonyRoy Savarimuthu, Frank Dignum, and MartinK. Purvis, editors, *PRIMA 2013 : Principles and Practice of Multi-Agent Systems*, volume 8291 of *Lecture Notes in Computer Science*, pages 117–131. Springer Berlin Heidelberg, 2013.

[4] Emmanuel Hermellin, Fabien Michel, and Jacques Ferber. Systèmes multi-agents et GPGPU : état des lieux et directions pour l'avenir. In *Principe de Parcimonie - JFSMA 14 - Vingt-deuxièmes Journées Francophones sur les Systèmes Multi-Agents*, pages 97–106. Cepadues Editions, 2014.

[5] Sean Luke, Claudio Cioffi-Revilla, Liviu Panait, Keith Sullivan, and Gabriel Balan. MASON : A Multiagent Simulation Environment. *Simulation*, 81(7) :517–527, 2005.

[6] Mikola Lysenko and Roshan M. D'Souza. A Framework for Megascale Agent Based Model Simulations on Graphics Processing Units. *Journal of Artificial Societies and Social Simulation*, 11(4) :10, 2008.

[7] Fabien Michel. Délégation GPU des perceptions agents : intégration itérative et modulaire du GPGPU dans les simulations multi-agents. Application sur la plate-forme TurtleKit 3. *Revue d'Intelligence Artificielle*, 28(4) :485–510, 2014.

[8] Fabien Michel, Grégory Beurier, and Jacques Ferber. The TurtleKit Simulation Platform : Application to Complex Systems. In Alain Akono, Emmanuel Tonyé, Albert Dipanda, and Kokou Yétongnon, editors, *Workshops Sessions of the Proceedings of the 1st International Conference on Signal-Image Technology and Internet-Based Systems, SITIS 2005, November 27 - December 1, 2005, Yaoundé, Cameroon*, pages 122–128. IEEE, november 2005.

[9] John D. Owens, David Luebke, Naga Govindaraju, Mark Harris, Jens Kruger, Aaron E. Lefohn, and Timothy J. Purcell. A Survey of General-Purpose Computation on Graphics Hardware. *Computer Graphics Forum*, 26(1) :80–113, 2007.

[10] Denis Payet, Rémy Courdier, Nicolas Sébastien, and Tiana Ralambondrainy. Environment as support for simplification, reuse and integration of processes in spatial MAS. In *Proceedings of the 2006 IEEE International Conference on Information Reuse and Integration, IRI - 2006 : Heuristic Systems Engineering, September 16-18, 2006, Waikoloa, Hawaii, USA*, pages 127–131. IEEE Systems, Man, and Cybernetics Society, 2006.

[11] Sébastien Picault. *From multi-agent simulation to multi-level simulation. Reifying the interactions.* Habilitation à diriger des recherches, Université des Sciences et Technologie de Lille - Lille I, December 2013.

[12] Mitchel Resnick. StarLogo : An Environment for Decentralized Modeling and Decentralized Thinking. In *Conference Companion on Human Factors in Computing Systems*, CHI '96, pages 11–12, New York, NY, USA, 1996. ACM.

[13] Craig W. Reynolds. Flocks, Herds and Schools : A Distributed Behavioral Model. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, volume 21 of *SIGGRAPH Computer Graphics '87*, pages 25–34, New York, NY, USA, 1987. ACM.

[14] Alessandro Ricci, Michele Piunti, and Mirko Viroli. Environment programming in multi-agent systems : an artifact-based perspective. *Autonomous Agents and Multi-Agent Systems*, 23(2) :158–192, 2011.

[15] Paul Richmond, Dawn Walker, Simon Coakley, and Daniela M. Romano. High performance cellular level agent-based simulation with FLAME for the GPU. *Briefings in bioinformatics*, 11(3) :334–47, 2010.

[16] Elizabeth Sklar. NetLogo, a Multi-agent Simulation Environment. *Artificial Life*, 13(3) :303–311, 2007.

[17] Mirko Viroli, Andrea Omicini, and Alessandro Ricci. Engineering MAS environment with artifacts. In Danny Weyns, H. Dyke Parunak, and Fabien Michel, editors, *Environments for Multi-Agent Systems*, volume 3830 of *Lecture Notes in Computer Science*, pages 62–77. Springer Berlin Heidelberg, 2006.

Gestion des Réseaux Temporels Simples Multi-agents dynamiques

G. Casanova
Guillaume.Casanova@onera.fr

C. Lesire
Charles.Lesire@onera.fr

C. Pralet
Cedric.Pralet@onera.fr

Onera – The French Aerolab
2 Avenue Édouard Belin
31000 Toulouse, France

Résumé

La réalisation de plans d'activités par plusieurs agents est généralement soumise à un ensemble de contraintes temporelles, impliquant notamment des contraintes de synchronisation entre agents. L'ensemble des contraintes temporelles d'un plan distribué peut être représenté en utilisant une structure Multi-agent Simple Temporal Network *(MaSTN). Dans ce papier, nous considérons le problème du maintien de la cohérence temporelle des plans distribués durant l'exécution, où les contraintes temporelles peuvent être modifiées. Pour cela, nous proposons de nouveaux algorithmes incrémentaux pour gérer les MaSTN dynamiques. Nous analysons les performances de ces algorithmes lorsque les communications sont intermittentes.*

Mots-clés : *Planification Multi-Agents, Coordination, Exécution*

Abstract

The realization of plans of activities by several agents is usually subject to a set of temporal constraints, including synchronization constraints between agents. To represent the set of temporal constraints imposed on distributed plans, the framework of Multi-agent Simple Temporal Network (MaSTN) can be used. In this paper, we consider the problem of maintaining the temporal consistency of distributed plans during execution, when temporal constraints may be updated. We propose new incremental algorithms for managing dynamic MaSTNs, and we analyze the performance of these algorithms when communications are intermittent.

Keywords: *Multi-Agent Planning, Coordination, Execution*

1 Introduction

Les applications en robotique autonome telles que l'exploration automatique de larges zones dangereuses peuvent avoir des performances améliorées par l'utilisation de plusieurs robots [7], en exploitant au maximum les capacités hétérogènes des robots pour distribuer au mieux les différentes tâches de la mission. Un autre point est que les systèmes multirobots évoluent généralement dans des environnements complexes et restreints. Dans de tels environnements, la communication entre les robots est généralement chaotique, menant à des communications retardées ou même intermittentes. Dans le but d'avoir une équipe robuste à la fois aux environnements dynamiques et aux communications intermittentes, il semble indispensable d'implémenter un système autonome distribué.

Dans ce contexte, il est nécessaire d'assurer malgré les communications intermittentes une cohérence temporelle entre les plans exécutés par les robots. Pour gérer ces aspects temporels, les *Simple Temporal Network* (STN) sont souvent utilisés en robotique pour représenter des plans [5, 10], et des techniques sont disponibles pour maintenir ces STN durant l'exécution lorsque surviennent des changements tels que des retards dans la réalisation des tâches. Dans le système que nous considérons, il est cependant impossible d'avoir un STN unique pour toute l'équipe maintenu de façon centralisée. A la place, il est possible de se tourner vers le formalisme des *Multi-agent STN* (MaSTN) [1], dans lequel chaque agent connaît seulement les contraintes temporelles présentes dans son propre plan. Plusieurs algorithmes existent pour gérer les MaSTN (DIPPC et DIΔSTP [1]), cependant ils ne sont pas directement adaptés à notre besoin car ils n'ont pas été prévus pour être robustes aux communications intermittentes.

Dans ce papier, nous proposons quatre algorithmes pour maintenir la cohérence dans un MaSTN : CIP, un algorithme centralisé basé sur un agent superviseur gérant entièrement la cohérence du MaSTN ; DIP-G, un algorithme distribué basé sur le partage de toutes les informations entre agents ; DIP-L, qui reproduit la

propagation mono-agent sur le MaSTN tout en maintenant le maximum de confidentialité entre les agents ; et DIP-M, qui cherche à améliorer les performances en réduisant les impératifs de confidentialité. La section 2 fait quelques rappels sur les STN et les MaSTN. La section 3 présente les quatre algorithmes proposés. Ces algorithmes sont analysés dans la section 4. Enfin, la section 5 compare sur des scénarios générés aléatoirement les quatre algorithmes selon le nombre de messages échangés et les calculs nécessaires pour propager des perturbations.

2 Contexte

2.1 Simple Temporal Network (STN)

Un STP (*Simple Temporal Problem* [4]) est une paire $S = (V, E)$ composée d'un ensemble de variables temporelles $V = \{v_1, \ldots, v_n\}$ et d'un ensemble de contraintes temporelles E. Chaque variable $v \in V$ est associée à l'occurrence d'un évènement dans le temps ; une variable spécifique v_0 est habituellement ajoutée à V afin de représenter une position temporelle de référence. Chaque contrainte $e \in E$ prend la forme $v_j - v_i \in [l_{ij}, u_{ij}]$ avec $l_{ij} \in \mathbb{R} \cup \{-\infty\}$ et $u_{ij} \in \mathbb{R} \cup \{+\infty\}$ deux bornes spécifiant respectivement des distances temporelles minimales et maximales entre v_i et v_j. On suppose sans perte de généralité qu'il n'existe pas plusieurs contraintes dans E portant sur les mêmes variables temporelles $\{v_i, v_j\}$. Les contraintes temporelles unaires telles que $v_i \in [a, b]$ peuvent être aisément exprimées par des contraintes de distance par rapport au point de référence (contraintes $v - v_0 \in [a, b]$).

À partir de cette définition, une *solution* à un STP (V, E) est une instanciation de toutes les variables dans V telle que toutes les contraintes temporelles dans E sont satisfaites. Un STP est dit *cohérent* s'il admet une solution, *incohérent* sinon. Pour finir, un STP a une représentation graphique naturelle appelée STN (*Simple Temporal Network*), qui représente chaque variable temporelle dans V par un sommet et chaque contrainte temporelle $v_j - v_i \in [l_{ij}, u_{ij}]$ dans E par un arc $v_i \rightarrow v_j$ étiqueté par $[l_{ij}, u_{ij}]$. Les STN sont intéressants en pratique car de nombreux problèmes qui peuvent être formulés sur les STN sont solubles en temps polynomial, comme la détermination pour chaque variable temporelle v de ses dates d'occurrence au plus tôt et au plus tard dans une solution (voir [4, 3, 11, 9] pour des algorithmes).

Les algorithmes pour raisonner sur les STN ont également été étendus à des contextes dynamiques [2, 8], où les contraintes temporelles peuvent être mises à jour par deux types de modifications : (1) des *contractions*, quand une contrainte temporelle $w - v \geq d$ est mise à jour par $w - v \geq d'$ avec $d' > d$, et (2) des *relâchements*, quand une contrainte temporelle $w - v \geq d$ est mise à jour par $w - v \geq d'$ avec $d' < d$. Dans le premier cas (contraction), on effectue un raisonnement incrémental en maintenant une file de contraintes temporelles à réviser pour recalculer la cohérence du STN ou les dates au plus tôt / au plus tard associées aux variables temporelles. Dans le second cas (relâchement), on effectue un raisonnement incrémental en utilisant des informations enregistrées durant les calculs précédents, telles que les *chaînes de propagation* qui décrivent les causes des bornes actuelles des différentes variables temporelles : si une contrainte temporelle est relâchée et était la cause d'une borne d'une variable temporelle, alors cette borne est réinitialisée ainsi que toutes les bornes qui en découlent [2].

Dans ce qui suit, nous ne détaillons pas ces algorithmes de contraction et de relâchement. Nous supposons seulement que nous avons deux fonctions notées respectivement `IncrRelax`(Rlx) et `IncrPropag`(Rvs). La première prend comme paramètre un ensemble Rlx de relâchements d'arcs représentés par des triplets (v, w, b) (relâchement de la borne $b \in \{LB, UB\}$ de l'arc $v \rightarrow w$) ; elle réinitialise les bornes temporelles des sommets grâce aux chaînes de propagation, et elle renvoie les contraintes temporelles qui doivent être révisées après l'opération de relâchement. La seconde prend comme paramètre un ensemble Rvs de contraintes temporelles à réviser, chacune étant décrite par un triplet (v, w, b) ; elle renvoie un ensemble de paires (v, b) décrivant les bornes temporelles mises à jour par la révision des contraintes temporelles. Nous supposons que les appels à `IncrPropag` et `IncrRelax` actualisent tous les paramètres associés aux STN (cohérence, dates au plus tôt / au plus tard et chaînes de propagation).

2.2 Multi-agent STN (MaSTN)

Les STN ont été étendus au contexte multi-agent, dans lequel les variables temporelles ne sont pas contrôlées par un seul agent mais sont partagées par un ensemble d'agents $\mathcal{A}$. Cette extension est appelée MaSTN (*Multiagent Simple Temporal Network* [1]). Un MaSTN est défini formellement par (1) un ensemble de N STN

locaux, un par agent $A \in \mathcal{A}$, et (2) un ensemble d'arcs E_X connectant ces STN locaux. Le STN local associé à l'agent A, noté S_L^A, est défini par V_L^A l'ensemble de sommets *locaux* possédés par A, et E_L^A l'ensemble d'arcs *locaux* reliant deux sommets locaux $v, w \in V_L^A$. Chaque arc dans E_X représente une contrainte *externe* et relie deux sommets locaux d'agents différents.

En plus de ses arcs locaux, chaque agent A connaît le sous-ensemble des contraintes externes E_X^A qui s'appliquent sur un de ses sommets locaux ($E_X^A = \{\{v, w\} \in E_X | v \in V_L^A\}$). En plus de ses sommets locaux, chaque agent A connaît l'ensemble V_X^A composé des sommets non possédés par A mais impliqués dans E_X^A ($V_X^A = \{v \in V | \exists w \in V_L^A, \{v, w\} \in E_X\}$). Ainsi, l'ensemble des variables temporelles connues par l'agent A est $V^A = V_L^A \cup V_X^A$, et l'ensemble des arcs connus par A est $E^A = E_L^A \cup E_X^A$. De plus, nous définissons pour chaque agent A l'ensemble V_F^A des sommets *frontières* comme étant le sous-ensemble des sommets locaux de A connectés à au moins un sommet externe ($V_F^A = \{v \in V_L^A | \exists \{v, w\} \in E_X\}$). Par la suite, nous notons *owner*(v) l'unique agent possédant la variable temporelle v, c'est-à-dire l'agent A tel que v est compris dans V_L^A.

La figure 1 donne un exemple de MaSTN impliquant trois agents A, B et C. L'agent A (resp. B, C) possède les variables temporelles v_1^A à v_6^A (resp. v_1^B à v_8^B et v_1^C à v_8^C). Dans son plan, l'agent A doit effectuer l'acquisition *acq1*, se recharger grâce à l'agent B puis effectuer une opération de maintenance. L'agent B doit effectuer l'acquisition *acq2*, recharger l'agent A et recevoir des données de l'agent C. Un instrument de réception doit être activé (variable v_5^B) puis désactivé (variable v_8^B) avant et après la réception des données venant de C. L'agent C doit effectuer deux acquisitions (*acq3* et *acq4*) avant de transmettre à l'agent B les données collectées. Certaines contraintes temporelles définissent des bornes sur la durée des activités et sur les temps de transition entre activités. D'autres sont des exigences, comme la contrainte reliant v_2^C et v_5^C qui impose de transmettre l'acquisition *acq3* suffisamment rapidement, ou la contrainte reliant le point de référence v_0 avec la variable temporelle v_8^C qui impose une limite sur la durée du plan de C. Les contraintes externes dans E^X sont représentées par des lignes en pointillés. Elles correspondent aux points de synchronisation entre les agents.

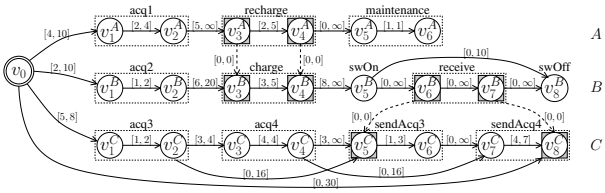


FIGURE 1 – Exemple de MaSTN impliquant trois agents.

3 Algorithmes incrémentaux pour les MaSTN dynamiques

Nous considérons maintenant les MaSTN dynamiques, dans lesquels les contraintes temporelles peuvent être mises à jour selon les informations reçues pendant l'exécution. Par exemple, la durée des activités peut être plus longue ou plus courte que prévu, les fenêtres temporelles disponibles pour la réalisation des activités peuvent changer, ou les agents peuvent replanifier et modifier leurs contraintes temporelles internes. Dans ce cas, notre but est de recalculer incrémentalement, au niveau de chaque agent A, la cohérence du MaSTN ainsi que les dates au plus tôt et au plus tard pour chaque variable possédée par A. Nous étudions quatre variantes algorithmiques pour gérer les MaSTN dynamiques. Ces variantes diffèrent par l'ensemble des contraintes temporelles prises en compte par chaque agent et par le type d'informations partagées entre les agents.

3.1 Hypothèses fondamentales

Par la suite, nous supposons que les contraintes externes dans E_X sont statiques : aucun changement sur leur existence ou leurs étiquettes. La raison à cela est que nous considérons que la modification d'une contrainte temporelle partagée par plusieurs agents doit être gérée par un processus plus complexe de re-synchronisation entre agents, ou par un processus assignant à un agent unique la responsabilité de mise à jour de la contrainte.

D'autre part, nous ne faisons pas d'hypothèse sur le protocole de communication utilisé (broadcast, unicast, multicast...), ou sur comment les données sont émises, routées et acquittées sur le réseau formé par les agents, notamment en cas de communications intermittentes. Les seules hypothèses faites sont qu'il peut exister des délais dans les transmissions et que les messages sont réceptionnés dans le même ordre que leur ordre d'envoi.

3.2 Un premier aperçu des algorithmes

La figure 2 donne un premier aperçu sur le type de connaissances manipulées par chaque agent pour les quatre algorithmes incrémentaux proposés, en reprenant le MaSTN de la figure 1. Les quatre algorithmes sont appelés CIP, DIP-G, DIP-M et DIP-L.

Dans CIP (Centralized Incremental Propagation, fig. 2(a)), un agent superviseur maintient toutes les contraintes temporelles et est responsable des calculs sur ces contraintes. Il reçoit les notifications de modifications des autres agents, et il renvoie les mises à jour concernant la cohérence du MaSTN et les bornes temporelles associées aux variables. Les autres agents ne connaissent que leur propre STN local.

Dans DIP-G (Distributed Incremental Propagation with Global information sharing, fig. 2(b)), chaque agent maintient l'ensemble de toutes les contraintes temporelles présentes dans le MaSTN, même celles qu'il n'est pas supposé connaître. Dès qu'un changement intervient sur un arc local d'un agent, cet agent envoie l'information correspondante à tous les autres agents.

Dans DIP-L (Distributed Incremental Propagation with Local information sharing, fig. 2(c)), chaque agent ne raisonne que sur les contraintes temporelles qu'il est supposé connaître, et les seules données échangées entre les agents sont les bornes temporelles sur les sommets externes.

Dans DIP-M (Distributed Incremental Propagation with Macro information sharing, fig. 2(d)), chaque agent raisonne sur ses contraintes temporelles locales et sur une vue macroscopique des contraintes temporelles des autres agents. Cette vue macroscopique donne des distances temporelles entre sommets externes. Chaque agent est responsable de l'envoi des mises à jour de sa propre vue macroscopique, et ce faisant ne révèle jamais ses contraintes internes.

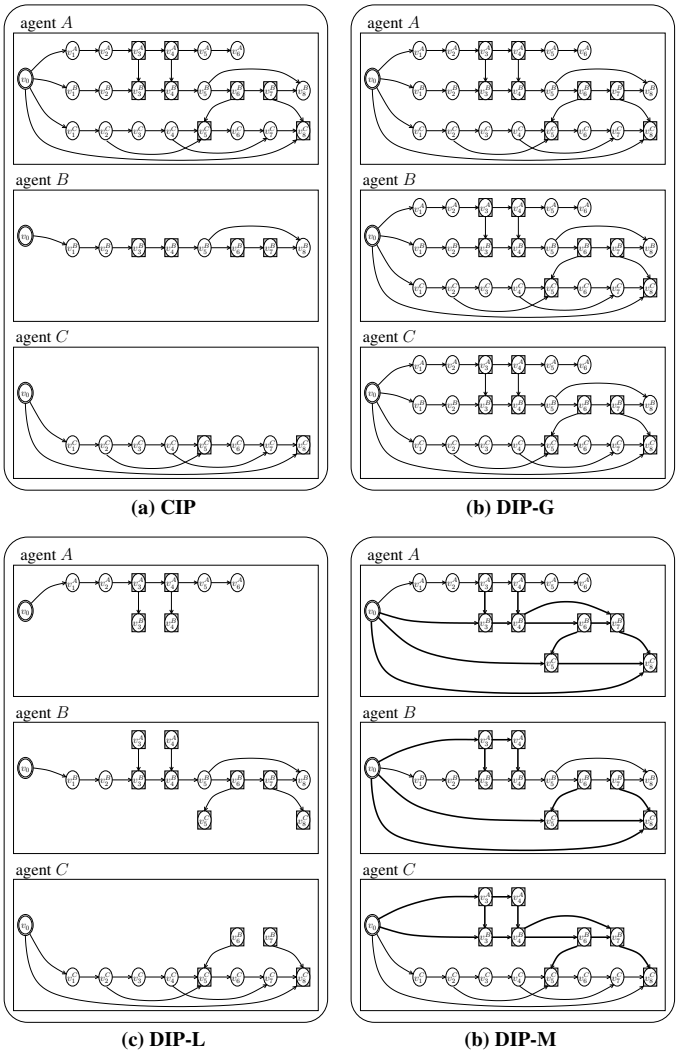


FIGURE 2 – Les quatre algorithmes proposés

3.3 Fonctions de base et structures de données utilisées

Avant de détailler les algorithmes, nous introduisons certaines fonctions et structures de données. Pour envoyer des messages, chaque agent utilise une fonction appelée $\texttt{send}(dstList, contenu)$ qui prend comme paramètre une liste $dstList$ d'agents à qui les messages sont destinés (valeur *all* quand le message est diffusé à tous les agents), et le $contenu$ du message. Cette fonction retourne un entier id qui correspond à un identificateur unique définissant un ordre strict sur les messages envoyés par les agents (l'id d'un message est strictement supérieur aux id de tous les messages envoyés précédemment). Soit src l'agent envoyant le message, alors chaque agent dans $dstList$ reçoit le message $m = (src, id, contenu)$. Le contenu des messages peut être de quatre types :

- *UPDATE*(v, w, b, d) : quand l'agent envoyant le message signale que la borne $b \in \{LB, UB\}$ de l'arc $v \rightarrow w$ a été mise à jour à la valeur d ;
- *RELAX*(v, b) (utilisé dans DIP-L) : quand l'agent émetteur ordonne de relâcher la borne b du sommet v ;
- *RELAXED*(v, b) (utilisé dans DIP-L) : quand l'agent émetteur signale que la borne b du sommet v a bien été relâchée ;
- *CONSISTENCY*(c, id) (utilisé dans CIP et DIP-L) : information de cohérence, où c prend

la valeur $vrai$ quand l'agent envoyant le message estime que le MaSTN est cohérent, et la valeur $faux$ sinon ; l'identificateur id indique que pour calculer cette cohérence, l'agent envoyant le message a pris en compte tous les messages reçus jusqu'à id inclus.

Chaque agent maintient plusieurs structures de données :
- *MsgRec* : une liste FIFO contenant les messages reçus non encore traités ;
- *Disturbs* : une liste FIFO contenant les perturbations locales non encore traitées ;
- Rvs : un ensemble de triplets (v, w, b) décrivant les contraintes temporelles à réviser sur le STN manipulé par l'agent ;
- Rlx : un ensemble de triplets (v, w, b) décrivant des bornes à relâcher ;
- de nombreux objets associés avec le STN manipulé par l'agent : la cohérence courante, les bornes actuelles des variables temporelles, les bornes actuelles des arcs, les chaînes de propagation...

En plus de la fonction `send`, chaque agent utilise plusieurs procédures élémentaires. Les procédures `setBound` et `addUpdate`, données à l'algorithme 1, sont utilisées respectivement pour mettre à jour les bornes des contraintes temporelles et pour poster les mises à jour sur le STN manipulé par l'agent. La procédure `addUpdate` peut mettre à jour l'ensemble Rlx des arcs relâchés (ligne 6) ou l'ensemble Rvs des arcs à réviser (ligne 7). Elle fixe aussi les bornes de l'arc concerné (ligne 8).

Algorithme 1 :

```
1 Procedure setBound(v,w,b,d)
2     case b = LB : l_{v,w} ← d
3     case b = UB : u_{v,w} ← d
4 Procedure addUpdate(v,w,b,d)
5     d' ← getBound(v, w, b)
6     if d < d' then Rlx ← Rlx ∪ {(v, w, b)}
7     else if d > d' then Rvs ← Rvs ∪ {(v, w, b)}
8     setBound(v, w, b, d)
```

Chaque agent utilise également trois procédures dont la définition dépend de la variante algorithmique choisie :

- `ProcessMessages`, utilisée par l'agent pour traiter les messages reçus ;
- `ProcessDisturbances`, utilisée par l'agent quand une perturbation survient sur ses propres contraintes temporelles (contraintes dans E_L^A) ;
- `ProcessUpdates`, utilisée quand l'agent effectue un raisonnement incrémental suite à des mises à jour ;

3.4 Centralized Incremental Propagation (CIP)

CIP adopte une approche centralisée dans laquelle un agent *superviseur* connaît l'ensemble du MaSTN. Il peut utiliser des méthodes mono-agent déjà étudiées pour détecter les incohérences. Les détails concernant CIP sont fournis dans l'algorithme 2 pour l'agent superviseur et dans l'algorithme 3 pour les agents esclaves.

Le superviseur traite les messages envoyés par les agents esclaves en ajoutant tout simplement l'ensemble des mises à jour reçues à l'ensemble des mises à jour à traiter (lignes 2-5 dans l'algorithme 2). Il maintient également, pour chaque agent esclave A, un champ $lastIdRec(A)$ représentant l'identificateur du dernier message envoyé par A qui a été traité. Pour traiter l'ensemble des mises à jour induites par les changements locaux ou par les messages, le superviseur considère le STN $S = (V, E)$ contenant tous les sommets et tous les arcs du MaSTN ($V = \cup_{A \in \mathcal{A}} V_A^L$ et $E = (\cup_{A \in \mathcal{A}} E_L^A) \cup E_X$). Il utilise les fonctions `IncrRelax` et `IncrPropag` introduites en à la section 2 (lignes 13 et 14), puis il transmet les mises à jour aux agents esclaves concernés (lignes 16-18). Le superviseur envoie également aux agents esclaves l'information de cohérence du MaSTN (lignes 20-24).

Concernant les agents esclaves (algorithme 3), chaque agent A reçoit du superviseur les messages et les mises à jour des bornes de ses propres variables dans V_L^A (ligne 5). Les messages de cohérence sont pris en compte seulement s'ils ont été traités connaissant le dernier message envoyé (ligne 7). De plus, chaque perturbation locale (changement dans E_L^A) est directement envoyée au superviseur et le statut de cohérence est temporairement mis à inconnu (lignes 10 à 13).

3.5 Distributed Incremental Propagation with Global information sharing (DIP-G)

DIG-G est une approche évitant une structure centralisée. Dans DIP-G, chaque agent connaît l'ensemble du MaSTN. Ainsi, chaque agent peut indépendamment vérifier la cohérence du MaSTN. Dans cette version, les agents n'ont qu'à envoyer les perturbations détectées à tous les autres agents. La description de DIP-G est donnée à l'algorithme 4 : à chaque réception de message, les mises à jour qu'il contient sont trai-

Algorithme 2 : CIP - procédures du superviseur

```
1  Procedure ProcessMessages()
2      while MsgRec ≠ ∅
3          PickNext (src, id, UPDATE(v, w, b, d)) from MsgRec
4          addUpdate(v, w, b, d)
5          lastIdRec(src) ← id
6      ProcessUpdates()

7  Procedure ProcessDisturbances()
8      while Disturbs ≠ ∅
9          PickNext UPDATE(v, w, b, d) from Disturbs
10         addUpdate(v, w, b, d)
11     ProcessUpdates()

12 Procedure ProcessUpdates()
13     Rvs ← Rvs ∪ IncrRelax(Rlx)
14     BoundUpdates ← IncrPropag(Rvs)
15     Rlx ← ∅; Rvs ← ∅
16     foreach (v, b) ∈ BoundUpdates
17         A ← owner(v)
18         lastIdSent(A) ←
           send(A, UPDATE(v0, v, b, getBound(v, b)))
19     c ← getConsistency()
20     foreach A ∈ 𝒜
21         s ← CONSISTENCY(c, lastIdRec(A))
22         if lastConsSent(A) ≠ s then
23             send(A, s)
24             lastConsSent(A) ← s
```

Algorithme 3 : CIP - procédures des esclaves

```
1  Procedure ProcessMessages()
2      while MsgRec ≠ ∅
3          PickNext m = (src, id, content) from MsgRec
4          if content = UPDATE(v0, v, b, d) then
5              setBound(v, b, d)
6          else if (content = CONSISTENCY(c, idRec))
7              if lastIdSent = idRec then setConsistency(c)

8  Procedure ProcessDisturbances()
9      if Disturbs ≠ ∅ then
10         setConsistency(unknown)
11         while Disturbs ≠ ∅
12             Pick u from MsgRec
13             lastIdSent ← send(supervisor, u)

14 Procedure ProcessUpdates() : empty
```

tées, et à chaque modification d'un arc local, ce changement est transmis à tous les autres agents (pas de confidentialité). Chaque agent traite les mises à jour de manière analogue au superviseur dans CIP (voir lignes 13 à 15). Dans DIP-G, les agents n'échangent pas de message de cohérence, ils échangent uniquement les informations brutes sur les arcs locaux.

3.6 Distributed Incremental Propagation with Local information sharing (DIP-L)

Dans DIP-L (algorithme 5), chaque agent A ne connaît que son propre STN local $S^A = (V_L^A \cup V_X^A, E_L^A \cup E_X^A)$. Quand un agent détecte

Algorithme 4 : Procédures DIP-G

```
1  Procedure ProcessMessages()
2      while MsgRec ≠ ∅
3          PickNext m = UPDATE(v, w, b, d) from MsgRec
4          addUpdate(v, w, b, d)
5      ProcessUpdates()

6  Procedure ProcessDisturbances()
7      while Disturbs ≠ ∅
8          PickNext u = UPDATE(v, w, b, d) from Disturbs
9          addUpdate(v, w, b, d)
10         send(all, u)
11     ProcessUpdates()

12 Procedure ProcessUpdates()
13     Rvs ← Rvs ∪ IncrRelax(Rlx)
14     BoundUpdates ← IncrPropag(Rvs)
15     Rlx ← ∅; Rvs ← ∅
```

une perturbation sur certains de ses arcs locaux dans E_L^A, il la propage au niveau de son STN local puis envoie uniquement les informations relatives aux sommets frontières modifiés aux agents concernés. Ces derniers vont ensuite propager les changements sur leur propre partie du problème, et éventuellement envoyer de nouvelles mises à jour à l'agent original ou aux autres agents. Ainsi, les contraintes temporelles sont propagées dans le MaSTN de manière distribuée et locale. Les agents partagent également leur vision de la cohérence de leur propre partie du problème.

La principale difficulté pour définir DIP-L vient du fait qu'effectuer des contractions et des relâchements de contraintes en parallèle sur les STN peut mener à des résultats faux. Sur les MaSTN, le problème est amplifié car on peut montrer qu'effectuer des contractions et des relâchements en parallèle peut mener à un cycle infini d'envois de messages. Lorsque les mises à jour surviennent simultanément à l'intérieur du réseau d'agents, il est nécessaire d'ajouter des mécanismes empêchant les agents de propager les mises à jour correspondant à des contractions avant que tous les relâchements soient terminés. Pour cela, nous maintenons, au niveau de chaque agent A, une structure de donnée appelée *liste de relâchements en attente*. Pour chaque sommet w et pour chaque type de borne b, nous maintenons un ensemble $RlxWait(w, b)$ pour représenter l'ensemble de sommets frontières v de A contenus dans l'arbre de chaînes de propagation de racine w pour la borne b, et dont le relâchement doit être terminé avant qu'une contraction puisse être faite sur la borne b de w. Dans ce cas, w est appelé la source du relâchement de v pour la borne b, noté par $w =$

$rlxSrc(v, b)$. Ces aspects ne sont pas pris en compte dans l'algorithme DIPPC [1], qui gère uniquement les contractions de contraintes.

Dans DIP-L, quatre types de messages sont échangés entre les agents : *UPDATE*(v_0, v, b, d) (ligne 4) quand un agent reçoit une mise à jour sur la borne d'un sommet externe, *RELAX*(v, b) (ligne 6) quand un agent reçoit une requête de relâchement sur la borne b d'un sommet externe v, *RELAXED*(v, b) (ligne 8) quand un agent reçoit la confirmation que tous les sommets appartenant à l'arbre de chaînes de propagation de racine v ont été relâchés pour la borne b, et *CONSISTENCY*(c, lid) (ligne 14) quand un agent reçoit l'information de cohérence c du STN local d'un autre agent, et quand le dernier message pris en compte par l'autre agent a pour identificateur lid. En particulier, quand une liste de relâchements en attente devient vide (ligne 11), un message de confirmation de relâchement est envoyé à l'agent concerné (ligne 13).

La procédure `ProcessDisturbances` (lignes 19 à 23) ajoute toutes les perturbations locales à l'ensemble des mises à jour à traiter.

La procédure `ProcessUpdates` se décompose en deux parties : une partie est dédiée aux relâchements de contraintes (lignes 25 à 37), et une autre est dédiée aux contractions de contraintes (lignes 38 à 49). La partie relâchement envoie les requêtes de relâchement quand les sommets frontières sont relâchés (lignes 31-32), alors que la partie contraction envoie les mises à jour aux agents voisins quand les bornes temporelles des sommets frontières changent (lignes 41 à 43), ainsi que les informations de cohérence (lignes 45 à 49).

Algorithme 5 : Procédures DIP-L

```
1  Procedure ProcessMessages()
2      while MsgRec ≠ ∅
3          PickNext m = (src, id, content) from MsgRec
4          if content = UPDATE(v0, v, b, d) then
5              addUpdate(v0, v, b, d)
6          else if content = RELAX(v, b) then
7              Rlx ← Rlx ∪ {(v0, v, b)}
8          else if content = RELAXED(v, b) then
9              w ← rlxSrc(v, b)
10             RlxWait(w, b) ← RlxWait(w, b) \ {(v, src)}
11             if (RlxWait(w, b) = ∅) ∧ (w ∈ V_X^this) then
12                 A ← owner(w)
13                 lastIdSent(A) = send(A, RELAXED(w, b))
14         else if content = CONSISTENCY(c, idRec) then
15             if lastIdSent(src) = idRec then
16                 consistency(src) ← c
17         lastIdRec(src) ← id
18     ProcessUpdates()

19 Procedure ProcessDisturbances()
20     while Disturbs ≠ ∅
21         PickNext UPDATE(v, w, b, d) from Disturbs
22         addUpdate(v, w, b, d)
23     ProcessUpdates()

24 Procedure ProcessUpdates()
25     foreach (v, b) ∈ Rlx
26         RlxFront ← V_F^this ∩ getPropagTree(v, b)
27         if RlxFront ≠ ∅ then
28             foreach w ∈ RlxFront
29                 rlxSrc(w, b) ← v
30             RlxWait(v, b) ← {(w, A) | w ∈ RlxFront ∩ V_X^A}
31             foreach (w, A) ∈ RlxWait(v, b)
32                 lastIdSent(A) ← send(A, RELAX(w, b))
33         else if v ∈ V_X^this then
34             A ← owner(v)
35             lastIdSent(A) ← send(A, RELAXED(v, b))
36     Rvs ← Rvs ∪ IncrRelax(Rlx)
37     Rlx ← ∅
38     if ∀(v, b) ∈ V^A × {LB, UB}, RlxWait(v, b) = ∅ then
39         BoundUpdates ← IncrPropag(Rvs)
40         Rvs ← ∅
41         foreach (v, b) ∈ BoundUpdates | v ∈ V_F^this
42             foreach A ∈ A | v ∈ V_X^A
43                 send(A, UPDATE(v0, v, b, getBound(v, b)))
44         c ← getConsistency()
45         foreach A ∈ A
46             s ← CONSISTENCY(c, lastIdRec(A))
47             if lastConsSent(A) ≠ s then
48                 lastIdSent ← send(A, s)
49                 lastConsSent(A) ← s
```

3.7 Distributed Incremental Propagation with Macro-information sharing (DIP-M)

La principale limitation de la méthode précédente est que comme chaque agent connaît uniquement son STN local, de nombreux messages peuvent être échangés entre les agents avant d'obtenir les bornes des variables temporelles.

Dans DIP-M, des informations supplémentaires sont partagées entre les agents tout en respectant une certaine confidentialité : chaque agent A construit une vue globale de son propre STN local et partage cette vue avec tous les autres agents. Cette vue globale est appelée le *macro-STN local* de A. De manière formelle, c'est un STN $S_M^A = (V_M^A, E_M^A)$ où V_M^A est l'ensemble composé de tous les sommets frontières de A plus le point de référence v_0 ($V_M^A = V_F^A \cup \{v_0\}$), et où E_M^A est un ensemble d'arcs tel que toute solution de S_M^A peut être étendue à une solution du STN local S_L^A, et réciproquement toute solution de S_L^A peut être projetée en une solution de S_M^A.

Le macro-STN local associé à A fournit une vue globale, restreinte aux sommets frontières,

de l'ensemble de solutions du STN local de A. Cette vue globale permet aux autres agents de prédire comme les variables temporelles externes possédées par A réagissent aux perturbations.

Le *macro-STN* $S_M = (V_M, E_M)$ associé à un MaSTN impliquant un ensemble d'agents $\mathcal{A}$ est ensuite défini comme l'union de tous les macro-STN locaux de tous les agents, plus l'ensemble des arcs externes ($V_M = \cup_{A\in\mathcal{A}} V_M^A$ et $E_M = E_X \cup (\cup_{A\in\mathcal{A}} E_M^A)$). La figure 3(a) donne la structure du macro-STN associé au MaSTN de la figure 1.

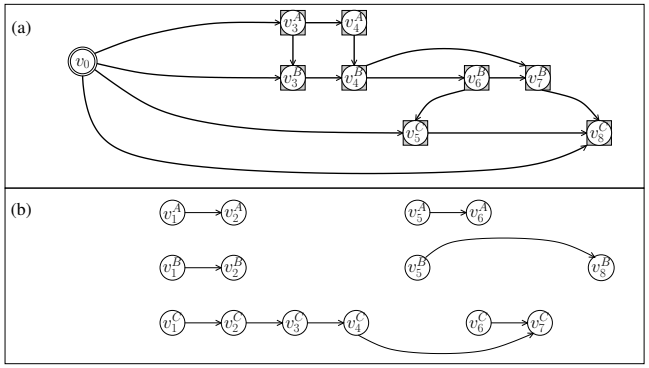


FIGURE 3 – (a) Structure du macro-STN obtenu sur l'exemple de la figure 1 ; (b) ensemble des composantes du macro-STN

Afin de construire le macro-STN local S_M^A associé à chaque agent A, nous utilisons un algorithme semblable à l'algorithme *all-pairs shortest paths* qui élimine un par un les sommets non-frontières du STN local S_L^A. Éliminer un sommet non-frontière consiste à calculer tous les intervalles possibles de distance entre les sommets dans le voisinage de v, c'est-à-dire entre les sommets liés directement à v par une contrainte temporelle. Plus précisément, pour tous sommets u, w dans le voisinage de v, éliminer v signifie mettre à jour les bornes au plus tôt et au plus tard de $e = \{u, w\}$ par :

$$l_e = \max(l_e, l_{\{u,v\}} + l_{\{v,w\}}) \quad (1)$$

$$u_e = \min(u_e, u_{\{u,v\}} + u_{\{v,w\}}) \quad (2)$$

À partir de ce processus d'élimination, nous pouvons construire le macro-STN local S_M^A et le maintenir incrémentalement durant les changements de contraintes temporelles. Initialement, l'ensemble des variables non-frontières $V_L^A \backslash V_F^A$ possédées par A sont partitionnées en un ensemble de composantes C^A tel que tout chemin dans S_L^A entre deux variables v, w appartenant à des composantes distinctes passe par un sommet frontière dans V_F^A ; formellement, C^A est l'ensemble de l'ensemble de composantes connexes obtenu en supprimant les sommets frontières de S_L^A (voir la figure 3(b)) ; nous calculons ensuite, pour chaque composante $c \in C^A$ et pour chaque paire de sommets frontières v, w connectés à c par une contrainte temporelle, la distance temporelle minimale et maximale $l_{\{v,w\},c}$ et $u_{\{v,w\},c}$ entre v et w selon les contraintes impliquées dans c ; ces distances sont calculées par élimination de tous les sommets appartenant à la composante c ; les bornes étiquetant les macro-arcs entre v et w sont alors obtenues en combinant toutes les bornes $l_{\{v,w\},c}/u_{\{v,w\},c}$ de toutes les composantes c reliées à v et w, plus éventuellement l'arc direct entre v et w.

Dans le cas de changements concernant certains arcs locaux, il suffit d'appliquer à nouveau la procédure d'élimination, mais restreinte aux composantes touchées par les changements locaux ; ces composantes sont celles contenant des variables temporelles impliquées dans la mise à jour de contraintes temporelles.

Grâce au macro-STN, chaque agent peut raisonner sur l'ensemble du MaSTN dans le cas de changements sur son STN local, sans avoir à attendre des messages de la part des autres agents (voir la figure 2(d)). Les procédures utilisées dans DIP-M sont données à l'algorithme 6. La principale différence avec DIP-G est qu'au lieu de transmettre des mises à jour brutes sur les arcs internes, DIP-M transmet des mises à jour portant uniquement sur les macro-arcs (ligne 5). Sur ce point, DIP-M utilise une fonction appelée `IncrComputeMacroEdges` qui met à jour de manière incrémentale son propre macro-STN local suite aux perturbations survenant sur ses arcs locaux (ligne 3). Cette fonction renvoie l'ensemble des modifications sur les macro-arcs dans E_M^A. En terme de confidentialité, chaque agent ne révèle que les contraintes de distance entre ses sommets externes.

4 Analyse Théorique

Nous considérons deux métriques principales pour évaluer les quatre algorithmes : a) l'*efficacité des messages*, correspondant au nombre de messages envoyés divisé par le nombre de perturbations (relâchements ou contractions) et b) l'*efficacité des calculs*, correspondant au nombre total de vérifications de contraintes effectuées par tous les agents divisé

Algorithme 6 : Procédures DIP-M

```
1  Procedure ProcessMessages() : identical to DIP-G

2  Procedure ProcessDisturbances()
3      MacroUp ← IncrComputeMacroEdges(Disturbs)
4      foreach u ∈ MacroUp
5          send(all, u)
6      while Disturbs ≠ ∅
7          PickNext u = UPDATE(v, w, b, d) from Disturbs
8          addUpdate(v, w, b, d)
9      ProcessUpdates()

10 Procedure ProcessUpdates() : identical to DIP-G
```

par le nombre de vérifications de contraintes qui seraient effectuées en appliquant la fonction `IncrPropag` sur l'ensemble du MaSTN (ce qui équivaudrait à effectuer une propagation mono-agent sur un STN classique).

Les complexités sont résumées (sans démonstrations) dans la Table 1. w correspond à la taille de la plus grande composante dans le MaSTN.

métrique	CIP	DIP-G	DIP-L	DIP-M
messages	$\|V\|$	$O(1)$	$O(\|E_X\|^N)$	w^2
calculs	1	N	$O(\|E_X\|^N)$	N

TABLE 1 – Complexité des pires cas pour les quatre algorithmes

DIP-L est sensible aux communications intermittentes car les agents ne possèdent pas les informations nécessaires sur le MaSTN pour effectuer une propagation complète. Les trois autres algorithmes sont moins sensibles grâce à la redondance d'information.

5 Expérimentations

5.1 Méthodologie

Nous évaluons les quatre algorithmes proposés sur un ensemble de problèmes aléatoires. Ces problèmes ont été générés en deux étapes : génération du MaSTN, puis génération du scénario sur ce MaSTN.

La génération du MaSTN prend comme paramètres le nombre d'agents (N), le nombre de sommets par agent ($\mathcal{V}$), le nombre d'arcs locaux ($\mathcal{L}$) et externes ($\mathcal{E}$) par agent. La génération est *structurée* dans le sens où chaque STN local S_L^A est composé d'une chaîne principale, à laquelle sont ajoutés des arcs locaux entre des sommets aléatoires. Finalement, $\mathcal{E}$ arcs externes sont générés entre les agents. Pour chaque arc, nous générons une valeur pour les bornes tout en assurant que le MaSTN global reste cohérent.

Ensuite, pour chaque instance de MaSTN, un ensemble de scénarios de $N \times \mathcal{V}$ pas est créé. Chaque pas est décrit par la nouvelle valeur d'un arc appartenant à un agent. La plupart du temps, cette nouvelle valeur est prise entre les bornes de l'arc (contraction de contrainte), mais occasionnellement nous tirons une valeur hors de ces bornes (relâchement de contrainte). Dans nos expériences, nous tirons en moyenne un relâchement pour neuf contractions.

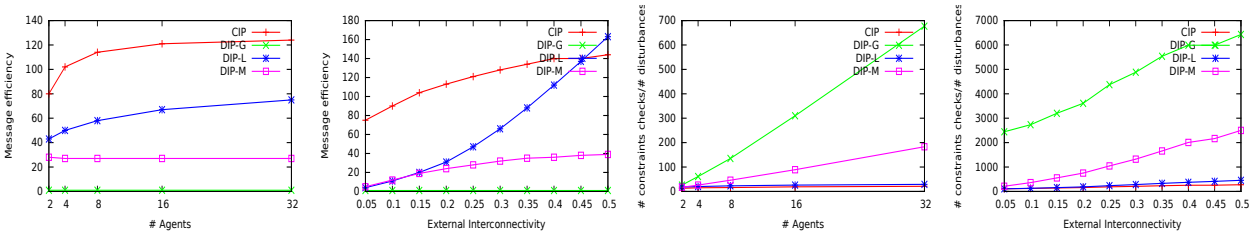
Pour chaque ensemble de paramètres, nous générons dix MaSTN et pour chacun d'eux un scénario correspondant. Nous exécutons ensuite chaque algorithme (CIP, DIP-G, DIP-L, DIP-M) sur chaque scénario en utilisant un processus par agent. Chaque agent génère ses propres perturbations (suivant le scénario) et propage les nouvelles contraintes selon l'algorithme utilisé. Nous mesurons ensuite le nombre de messages échangés entre les agents et le nombre de contraintes vérifiées par chaque agent pour chaque propagation.

5.2 Résultats

La figure 4 présente les résultats permettant de comparer les quatre algorithmes. L'efficacité messages est le nombre moyen de messages envoyés par perturbation. L'interconnectivité externe est $\frac{\mathcal{E}}{\mathcal{V}}$, c.-à-d. le ratio du nombre de sommets frontières par le nombre de sommets (0 signifie que les STN locaux sont indépendants ; 0.5 signifie que la moitié des sommets sont partagés).

Les figures 4a et 4c montrent l'efficacité en termes de messages et en termes de vérifications de contraintes. Au niveau des messages, les quatre algorithmes conservent de bonnes performances lors du passage à l'échelle, et les différences sont peu significatives. La légère augmentation pour CIP et DIP-L vient de l'apparition de plus longues chaînes de propagation dans les MaSTN. Concernant le nombre de vérifications de contraintes, CIP surpasse nettement les autres algorithmes. DIP-L est également très performant grâce à la faible redondance des calculs (chaque agent est responsable de ses propres contraintes). Pour DIP-G et DIP-M, une augmentation linéaire peut être observée à cause de la redondance des calculs effectués sur plusieurs agents.

DIP-G envoie exactement le même nombre de



(a) Messages/Agents (b) Messages/Inter. Externe (c) Calculs/Agents (d) Calculs/Inter. Externe

FIGURE 4 – Résultats pour l'efficacité messages (a, b) et l'efficacité calculs (c, d)

messages quelle que soit l'interconnectivité externe, comme montré à la figure 4b, et il surpasse significativement les autres algorithmes pour les interconnectivités externes les plus hautes. L'augmentation du nombre de messages envoyés par CIP est une conséquence de l'augmentation des dépendances entre les agents, car plus le couplage externe est élevé, plus le nombre de variables temporelles affectées par une mise à jour d'arc est grand. DIP-L et DIP-M présentent des comportements plus intéressants : les performances de DIP-L sont directement affectées par le nombre de contraintes externes. DIP-M n'est cependant que partiellement affecté par le couplage externe et se stabilise rapidement. Cependant, la figure 4d montre que l'efficacité des calculs de DIP-M empire pour les MaSTN plus denses (couplage externe élevé), à cause de l'effort calculatoire requis pour calculer les macro-arcs de composantes plus larges. Les trois autres algorithmes ont un taux d'augmentation similaire, causé par des propagations plus longues dans les réseaux plus denses.

En conclusion, DIP-L est particulièrement adapté aux MaSTN creux (couplage externe faible) où les agents n'utilisent que peu d'informations concernant leurs voisins, tandis que DIP-M offre dans l'ensemble les meilleures performances grâce au partage de macro-informations.

6 Conclusion et Perspectives

Dans ce papier, nous avons proposé quatre algorithmes pour maintenir la cohérence des MaSTN : CIP, DIP-G, DIP-L et DIP-M. Ces algorithmes ont été analysés à la fois d'un point de vue théorique et sur un ensemble de tests générés aléatoirement. Nous concluons que DIP-L et DIP-M sont les algorithmes les plus adaptés pour les cas d'utilisation réels, le premier étant plus adapté aux MaSTN creux tandis que le second est plus adapté aux MaSTN denses.

Dans le futur, nous comptons étendre le cadre macro-STN aux autres variantes de STN telles que les STNU [6]. Ces extensions apporteront des modèles plus précis d'exécution de plans, tout en intégrant les concepts de variables temporelles incontrôlables. Nous préparons également une expérimentation réelle avec une équipe de robots terrestres et aériens pour une mission d'exploration.

Références

[1] J. C. Boerkoel, L. Planken, R. Wilcox, and J. A. Shah. Distributed Algorithms for Incrementally Maintaining Multiagent Simple Temporal Networks. In *ICAPS*, Rome, Italy, 2013.

[2] R. Cervoni, A. Cesta, and A. Oddi. Managing Dynamic Temporal Constraint Networks. In *AIPS*, Chicago, IL, USA, 1994.

[3] A. Cesta and A. Oddi. Gaining Efficiency and Flexibility in the Simple Temporal Problem. In *TIME*, Key West, FL, USA, 1996.

[4] R. Dechter, I. Meiri, and J. Pearl. Temporal Constraint Networks. *Artificial Intelligence*, 49(1-3) :61–95, 1991.

[5] M. Di Rocco, F. Pecora, and A. Saffiotti. When Robots Are Late : Configuration Planning for Multiple Robots with Dynamic Goals. In *IROS*, Tokyo, Japan, 2013.

[6] P. H. Morris, N. Muscettola, and T. Vidal. Dynamic Control Of Plans With Temporal Uncertainty. In *IJCAI*, Seattle, WA, USA, 2001.

[7] L. E. Parker. Distributed intelligence : Overview of the field and its application in multi-robot systems. *Journal of Physical Agents*, 2, 2008.

[8] L. Planken, M. de Weerdt, and N. Yorke-Smith. Incrementally Solving STNs by Enforcing Partial Path Consistency. In *ICAPS*, Toronto, Canada, 2010.

[9] L. R. Planken, M. M. de Weerdt, and R. P. van der Krogt. P3C : A New Algorithm for the Simple Temporal Problem. In *ICAPS*, Sydney, Australia, 2008.

[10] C. Pralet and C. Lesire. Deployment of mobile wireless sensor networks for crisis management : A constraint-based local search approach. In *CP*, Lyon, France, 2014.

[11] L. Xu and B. Y. Choueiry. A New Efficient Algorithm for Solving the Simple Temporal Problem. In *TIME*, Cairns, Queensland, Australia, 2003.

Personnalisation par un système multi-agent de la navigation au sein d'un corpus documentaire

Zina El Guedria[a]
zina.el_guedria@insa-rouen.fr

Laurent Vercouter[a]
laurent.vercouter@insa-rouen.fr

[a]Laboratoire d'Informatique, du Traitement de l'Information et des Systèmes,
Normandie Université, Institut National des Sciences Appliquées de Rouen, France

Résumé

La recherche documentaire dans un corpus numérique s'apparente à un processus de navigation guidé par un besoin d'information d'un utilisateur. Cette navigation nécessite l'usage d'outils classiques de recherche d'information pour sélectionner des documents pertinents en fonction d'une requête, mais ils doivent être complétés par des mécanismes de personnalisation et d'adaptation capable de faire évoluer la représentation du besoin en fonction des spécificités d'un utilisateur, de sa navigation en cours ou du corpus considéré. Nous proposons dans cet article un modèle multi-agent pour la navigation personnalisée permettant d'intégrer des facteurs hétérogènes de personnalisation.Ce travail s'inscrit dans le cadre d'un projet régional (PlaIR 2.0) ayant pour objectif le développement d'une plateforme d'accès personnalisé à un corpus numérique de documents juridiques.

Mots-clés : *Coordination, Evolution, Adaptation, Systèmes mixtes*

Abstract

Documentary research in a digital corpus is like a crawling process guided by a user need of information. Such crawlings require the use of traditional information retrieval tools to select relevant documents based on a query, but they must be complemented by personalization and adaptation mechanisms able to change the representation of need according to the specificities of a user, his current crawling or considered corpus. We propose in this article a multi agent model for personalized crawling that take into account heterogeneous personalisation factors. This work is part of a regional project (PlaIR 2.0) whose aim is developing a personalized access platform to a digital corpus of legal documents.

Keywords: *Coordination, Evolution, Adaptation, Mixed systems*

1 Introduction

L'accès aux documents d'un corpus numérique soulève des problèmes liés à la recherche d'information, à la visualisation des résultats d'une requête et à la navigation entre les documents. Certains de ces problèmes sont similaires à ceux rencontrés dans le domaine de la recherche d'information sur le Web (par exemple le calcul de la pertinence de documents par rapport à une requête) mais d'autres sont spécifiques au fait que l'on s'intéresse à un corpus fermé concernant des types de documents, requêtes et utilisateurs restreints que l'on peut représenter de manière plus fine que dans le cadre général du Web. Dans ce cadre il est réaliste de pouvoir identifier et chercher à reconnaître des pratiques ou usages spécifiques à l'accès à une base documentaire donnée et d'adapter en conséquence le comportement de la plateforme logicielle.

Nous nous intéressons dans cet article à la personnalisation de la navigation dans un corpus de documents. Les facteurs à prendre en compte dans cette personnalisation sont multiples : un profil propre à l'utilisateur, un cas d'usage reconnu pour la navigation, une proximité sémantique entre documents ou une recommandation construite sur l'historique des navigations. Nous proposons de traiter cette diversité d'influence par un système multi-agent. Le besoin informationnel et l'état courant d'une navigation sont représentés dans notre modèle par un environnement virtuel partagé que les agents perçoivent et sur lequel ils peuvent agir. Cet environnement, implémenté dans une couche d'artefacts, est ainsi un objet construit et adapté par l'activité collective des agents et de l'utilisateur, qui représentent la couche décisionnelle. Les contributions décrites dans cet article sont :

— une modélisation sous la forme d'agents et d'artefacts d'un système multi-agent pour réaliser une navigation personnalisée ;
— une représentation d'un profil de navigation incluant un processus de reformula-

tion de requêtes ;
— un processus de recommandation de documents à base de profils proches.

Cet article est organisé comme suit : La section 2 donne un aperçu de quelques travaux en RI personnalisée. Plus particulièrement, nous décrivons quelques travaux utilisant les systèmes multi-agents pour la recherche d'information ainsi que des travaux en recherche d'information personnalisée. La section 3 présente notre modélisation multi-agent de la plateforme de navigation personnalisée décrite selon deux couches : une couche Navigation et une couche Décision. La section 4 présente la problématique applicative. Nous y présentons le corpus documentaire utilisé, la réalisation de la couche Navigation par des artefacts ainsi qu'un scénario d'exécution. Enfin, la conclusion résume les travaux décrits dans cet article ainsi que nos perspectives de travaux futurs.

2 Recherche d'information et systèmes multi-agent

2.1 Système multi-agent pour la recherche d'information

Les premières approches multi agents apparues dans le domaine de recherche d'information se placent dans le cadre de la modélisation et résolution des systèmes de recherche d'information. Généralement, les tâches du processus sont affectées à différents agents qui peuvent les exécuter en séquence ou en concurrence. Dans ces modèles multi-agent les auteurs ont utilisé des agents utilisateurs, des agents ressources et des agents de fonctionnements et des agents d'ontologie etc. Parmi les premiers systèmes multi agents pour la recherche d'information nous pouvons citer à titre d'exemple : InfoSleuth [1] , le système NetSA [2], la librairie digitale UMDL [3], le modèle RETSINA [8] et le système AgentSeek [9].

Dans le contexte de personnalisation de la recherche d'information interviennent les systèmes multi agents tel est le cas de SARIPOD (Système multi-Agent de Recherche Intelligente POssibiliste de Documents Web) [4]. Ce modèle est un modèle multi-agent qui offre une collaboration entre les différents acteurs et la mise en oeuvre de toutes les fonctionnalités du système de recherche d'information. Il est à base de deux Réseaux Petits Mondes Hiérarchiques (RPMH) et d'un Réseau Possibiliste (RP) : Le premier RPMH consiste à structurer les documents retrouvés en zones denses de pages Web thématiquement liées les unes aux autres. Le second RPMH consiste à ne pas chercher seulement le mot-clé dans les pages Web mais aussi les substantifs qui lui sont sémantiquement proches. Les Réseaux Possibilistes combinent les deux RPMH afin d'organiser les documents recherchés selon les préférences de l'utilisateur.

Dans le même cadre nous citons la plateforme SWAPP (Search based Web AdaPtive Platform), la mise en place des systèmes interactifs auto-adaptatifs par systèmes multi-agents auto-organisateurs appliqués à la personnalisation de l'accès à l'information [10]. Ce système utilise l'approche de AMAS (Adaptive Multi-Agent System) pour proposer une évaluation adaptative et personnalisée du feedback implicite de l'utilisateur en utilisant l'UIM (User Interest Manager) ainsi que la construction adaptative de son profil à base de l'UPM (User Profile Manager) en utilisant des documents textuels représentant ses intérêts.

2.2 Recherche d'information personnalisée

Le besoin de la personnalisation de la recherche d'information commence par la difficulté de l'expression de besoin. En effet, pour affiner leurs besoins d'information les utilisateurs n'arrivent pas à trouver les termes exacts exprimant leurs besoins [11] et même en faisant des efforts, le besoin reste toujours ambigu [12]. Cette tâche est difficile même pour les utilisateurs expérimentés [13] et se termine par le choix de la pertinence, sous ses différents types [14, 15, 16] et pour un seul type de pertinence différents facteurs existent [14].

Avec l'apparition de la problématique de personnalisation dans la recherche d'information plusieurs travaux sont apparus. Nous pouvons distinguer la personnalisation individuelle [17, 18] où l'utilisateur est considéré de manière isolée et le modèle utilisateur est construit sur la base de ses préférences de contenu ou activités et mobilité via les applications. La personnalisation collaborative [19, 20] où l'utilisateur est considéré comme membre d'un groupe d'utilisateurs et le modèle utilisateur est construit sur la base de ses préférences de contenu ou activités via les applications ainsi que les modèles des autres utilisateurs similaires.

Il existe trois modes de personnalisation à savoir : La personnalisation explicite, implicite

et hybride. Commençons par la personnalisation explicite. Dans ce cas, la collecte des données descriptives de l'utilisateur est fournie par l'utilisateur. Par exemple les données démographiques (âge, sexe, langue maternelle [21, 22]), le choix des préférences (langue, genre de documents [23, 18]), le sujet d'intérêt [24] et les jugements qualitatifs sur l'information (j'aime, j'aime pas) [25, 17]. Les données citées ci-dessus peuvent être collectées à travers des formulaires (cases à cocher, saisie de mots clés), des interfaces élaborées (expression d'exemples, contre exemples, votes, notes, annotations) ou des questionnaires.

En ce qui concerne la personnalisation implicite, les données sont collectées à partir des données démographiques qui peuvent être déduites des interactions et des activités des utilisateurs [26] ou des requêtes [27]. Nous citons aussi les données sociales de l'utilisateur telles que les annotations, les posts, les blogs, les messages, les signaux, etc. [30, 28, 29].

L'activité de l'utilisateur et une source primordiale pour la personnalisation implicite. Cette activité peut être déterminée d'une part à partir des pages visitées, des requêtes passées [31, 17], des données de navigation [33, 32], des applications utilisées [33, 34], des favoris [34] et de l'historique de localisations [35] de l'utilisateur. D'autre part des interactions de l'utilisateur telles que les mouvements des yeux ; les données de clics [36] ; les actions sur les documents (ouverture, fermeture, impression, temps de lecture etc.) [37] ; les messages (e-mails) envoyés ou reçus [38] ; les annotations sociales, et les bookmarks [28, 29].

Enfin la personnalisation hybride utilise à la fois la personnalisation implicite et la personnalisation explicite. Cette personnalisation hybride peut commencer par la collecte explicite puis implicite (système WAIR[39]) ou la collecte implicite puis explicite tout en validant la collection implicite par des retours de l'utilisateur.

Un autre axe de personnalisation traite la durée de la session de recherche à savoir, la personnalisation basée sur le contexte courant (court-terme) et la personnalisation basée le contexte passé (long-terme). Pour la personnalisation à court terme, a mémoire de la personnalisation est égale à la durée de la session. Dans ce cas, on rencontre un problème de délimitation de la session de recherche. Soit une délimitation basée sur le temps (par exemple 30 min dans [40]), une délimitation basée sur la similitude de requêtes successives [41, 42] ou une délimitation basée sur la similitude des résultats de recherche [43]. La personnalisation basée sur le contexte passé (long-terme) où la mémoire de personnalisation est égale à la durée du compte utilisateur. Dans ces systèmes, on peut délimiter les sessions de recherche [18, 7] ou ne pas les délimiter [17].

3 Système multi-agent pour une navigation personnalisée

Cette section décrit le système multi-agent proposé pour réaliser une navigation personnalisée dans un corpus numérique de documents. Une approche multi-agent a été adoptée de manière à représenter l'hétérogénéité des facteurs de personnalisation à appliquer. Chaque agent exerce ainsi une influence différente sur la sélection de documents à présenter à l'utilisateur. La première partie de cette section présente l'architecture globale du système avant que soient détaillés l'environnement partagé des agents, représentant la navigation en cours, puis les spécifications des agents pour la couche décisionnelle.

3.1 Architecture multi-agent

L'objectif de notre plateforme est de permettre à des utilisateurs de naviguer dans un corpus fermé en visualisant séquentiellement différents documents et en affinant ou adaptant leur requête au fur et à mesure de leur session de navigation. Pour cela, il est nécessaire dans un premier temps d'utiliser des mécanismes classiques de recherche d'information pour sélectionner des documents pertinents en fonction d'une requête. Nous proposons de compléter ces outils par des mécanismes de personnalisation et d'adaptation pour faire évoluer la représentation du besoin. Cette évolution est dynamique car réalisée pendant une navigation en fonction du profil des utilisateurs, de leurs actions et des navigations précédemment observées. Pour cette raison, nous utilisons le terme de *besoin informationnel* pour désigner l'objectif global que la plateforme doit satisfaire plutôt que celui de *requête*, propre à une recherche ponctuelle d'information.

Nous représentons le besoin informationnel au sein d'un environnement virtuel, partagé par les agents. Ce besoin, initialement exprimé par une requête composée d'un ensemble de termes, est modifié par le biais d'actions des agents sur cet

environnement. L'évolution du besoin informationnel est ainsi le résultat d'un processus de co-construction impliquant les agents et l'utilisateur dans le but d'intégrer différentes sources de personnalisation et un contrôle par l'utilisateur. De plus, l'environnement partagé des agents inclut des outils nécessaires pour la navigation (index, moteur de recherche d'information, interface avec l'utilisateur), ainsi que l'ensemble des documents jugés pertinents à l'instant courant de la navigation.

Afin de distinguer la couche décisionnelle et la couche opérationnelle de la plateforme, nous avons opté pour une approche Agent-Artefact [44]. Les décisions d'évolution du besoin informationnel sont prises par le système multi-agent, parfois en interaction avec l'utilisateur. Le stockage des informations liées à la navigation et l'exécution de requêtes ponctuelles sont à la charge d'artefacts de l'environnement des agents. Cette architecture est représentée par la figure 1.

Au niveau de la couche *Navigation*, cinq types d'artefacts sont utilisés. Un artefact *Interface* encapsule les fonctions d'interaction directe avec l'utilisateur. Chaque instance d'artefacts de type *Navigation* représente le besoin informationnel d'une session de navigation pour laquelle une instance d'artefact *Recherche* est créée pour exécuter une recherche d'information sur ce besoin. Le résultat d'une recherche est stocké dans un artefact de type *Document*. Enfin, les artefacts de type *Profil* collectent des informations sur le comportement des utilisateurs.

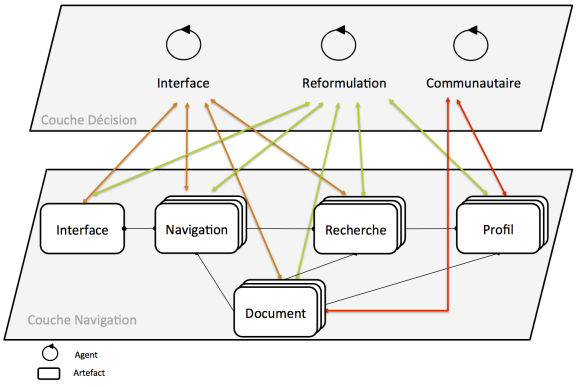


FIGURE 1 – Architecture Agent-Artefact proposée

La couche *Décision* contient les agents qui vont faire évoluer le besoin informationnel et les documents présentés pendant la navigation. L'agent *Interface* sert de point d'entrée à l'utilisateur pour créer une requête initiale et la modifier pendant la navigation. L'agent *Reformulation* propose l'ajout de termes au besoin informationnel sur la base des documents sélectionnés par les recherches précédentes. Enfin l'agent *Communautaire* propose d'ajouter des documents jugés pertinents d'après l'historique des navigations passées similaires.

3.2 Couche navigation

Les cinq artefacts de la couche navigation sont spécifiés dans cette sous-section.

Les navigations ont lieu au sein d'un corpus numérique de documents. Nous notons $\mathcal{D}$ l'ensemble des documents du corpus. Les requêtes sont effectuées par un ensemble de mots-clés parmi l'ensemble $\mathcal{K}$. Parmi ces mots-clés, un sous-ensemble $\mathcal{T} \subset \mathcal{K}$ regroupe l'ensemble des termes utilisés pour indexer les documents du corpus. Enfin, nous faisons l'hypothèse que les termes indexant un document sont accessibles par une fonction *Index* telle que :

$$Index : \mathrm{D} \mapsto P(\mathrm{T})$$

Artefact Navigation.
Un artefact Navigation encapsule un besoin informationnel d'un utilisateur. Soit $\mathcal{N} = \{\mathrm{N}_1, ..., \mathrm{N}_m\}$ l'ensemble de toutes les navigations, pour chaque navigation N_i, un utilisateur u_{N_i} a une requête courante K_{N_i} composée de plusieurs termes $k^j_{N_i}$.

$$\mathrm{N}_i = (\mathrm{K}_{N_i}, \mathrm{u}_{N_i})$$

Avec

$$\mathrm{K}_{N_i} = \{\mathrm{k}^1_{N_i}, ..., \mathrm{k}^l_{N_i}\}$$

Artefact Document. Un artefact Document contient une référence aux documents jugés pertinents pour une navigation donnée N_i. Cet ensemble est amené à évoluer en cours de navigation en fonction des actions des agents et de l'utilisateur. L'ensemble de documents D_i correspondant à la navigation N_i est l'union des documents résultant d'une recherche d'information $\mathrm{D}_{i,RI}$ et de ceux recommandés par l'agent Communautaire $\mathrm{D}_{i,REC}$:

$$\mathrm{D}_i = (\mathrm{D}_{i,RI}, \mathrm{D}_{i,REC})$$

Avec :

$$\mathrm{D}_{i,RI} = \left\{\mathrm{d}^1_{i,RI}, ..., \mathrm{d}^n_{i,RI}\right\},$$

$$\mathrm{D}_{i,REC} = \left\{\mathrm{d}^1_{i,REC}, ..., \mathrm{d}^p_{i,REC}\right\}$$

Artefact Profil. Le profil P_i d'un utilisateur impliqué dans une session de navigation N_i est composé par sa requête initiale $\mathrm{K}^{INIT}_{P_i} \subset \mathcal{K}$, les termes proposés en reformulation $\mathrm{T}^{ACC}_{P_i} \subset \mathcal{T}$ (acceptés par l'utilisateur) et $\mathrm{T}^{REJ}_{P_i} \subset \mathcal{T}$ (rejetés par l'utilisateur) et enfin les références des documents jugés pertinents par l'utilisateur $\mathrm{D}^{REL}_{P_i} \subset \mathrm{D}_i$.

$$\mathrm{P}_i = \left(\mathrm{K}^{INIT}_{P_i}, \mathrm{T}^{ACC}_{P_i}, \mathrm{T}^{REJ}_{P_i}, \mathrm{D}^{REL}_{P_i}\right)$$

avec

$$\mathrm{K}^{INIT}_{P_i} = \left\{\mathrm{k}^1_{P_i}, ..., \mathrm{k}^n_{P_i}\right\}$$

$$\mathrm{T}^{ACC}_{P_i} = \left\{\mathrm{t}^{ACC,1}_{P_i}, ..., \mathrm{t}^{ACC,p}_{P_i}\right\}$$

$$\mathrm{T}^{REJ}_{P_i} = \left\{\mathrm{t}^{REJ,1}_{P_i}, ..., \mathrm{t}^{REJ,q}_{P_i}\right\}$$

$$\mathrm{D}^{REL}_{P_i} = \left\{\mathrm{d}^{REL,1}_{P_i}, ..., \mathrm{d}^{REL,r}_{P_i}\right\}$$

et

$$\mathrm{K}^{INIT}_{P_i} \cap \mathrm{T}^{ACC}_{P_i} \cap \mathrm{T}^{REJ}_{P_i} = \emptyset$$

Artefact Recherche.
L'artefact Recherche sert de moteur de recherche d'information. Nous utilisons pour notre plateforme le moteur *Lucene* [1] qui est une application open source gratuite pour la recherche plein texte et l'analyse de contenu textuel. L'objectif de l'artefact est de prendre en entrée un ensemble de termes issus d'une requête et de fournir les références de documents jugés pertinents en sortie.

$$Recherche : (\mathrm{K}) \mapsto P\,(\mathrm{D})$$

Artefact Interface.
Le rôle de l'artefact Interface est de gérer les interactions avec l'utilisateur. Il récupère les requêtes ainsi que les retours de l'utilisateur pour le compte des agents Interface et Reformulation, et affiche les résultats des recherches effectuées par les agents.

1. http ://lucene.apache.org/core/

3.3 Couche Décision

La couche décision est composée de trois agents qui ont la capacité de percevoir et d'agir sur leur environnement partagé représentant un processus de navigation en cours.

Agent Interface.
Il existe un agent Interface par utilisateur, en charge du suivi de sa navigation. Un utilisateur u commence une navigation en saisissant une requête $\mathrm{K}_u = \left\{\mathrm{k}^1_u, ..., \mathrm{k}^n_u\right\}$ composée d'un ensemble de mots-clés. Si l'utilisateur n'avait pas de navigation en cours, une nouvelle session de navigation est créée avec cette requête.

S'il existe déjà des navigations pour cet utilisateur, l'agent interface doit déterminer s'il s'agit du début d'une nouvelle navigation ou si l'insertion de nouveaux termes correspond à la continuité d'une session en cours. Dans ce but, l'agent calcule une mesure de similarité $Sim(\mathrm{K}_u, \mathrm{K}_{N_i})$ entre les termes de la requête et ceux de toutes les navigations précédentes de l'utilisateur. Nous faisons ici l'hypothèse qu'une telle fonction existe sans nous restreindre à un calcul précis de similarité (pour avoir des exemples de similarité entre requêtes, voir [41, 42]). Si la similarité est en dessous d'un seuil θ pour toutes les navigations passées, l'agent considère qu'une nouvelle navigation commence, sinon il considère être dans la continuité des plus similaires.

$$\exists \mathrm{N}_i \in \mathcal{N}/\mathrm{u} = \mathrm{u}_{N_i}, Sim(\mathrm{K}_u, \mathrm{K}_{N_i}) > \theta,$$
$$\mathrm{K}_{N_i} = \mathrm{K}_{N_i} \cup \mathrm{K}$$

Sinon, un nouvel artefact de navigation est créé :

$$\nexists \mathrm{N}_i \in \mathcal{N}/\mathrm{u} = \mathrm{u}_{N_i}, Sim(\mathrm{K}_u, \mathrm{K}_{N_i}) < \theta,$$
$$\mathrm{N}_m = (\mathrm{K}, u)$$

À chaque fois qu'un artefact de navigation est créé ou mis à jour (notons-le N_j) l'agent effectue une recherche par l'artefact correspondant $Recherche(\mathrm{N}_j)$ avec pour résultat de créer ou de mettre à jour l'artefact Document avec un ensemble $\mathrm{D}_{N_j,RI}$.

Enfin, l'agent Interface envoi l'ensemble de documents D_j à l'artefact Interface pour les visualiser.

Agent Reformulation.
Un agent Reformulation est associé a chaque utilisateur de la plateforme. Le fonctionnement de l'agent Reformulation est basé sur l'observation des artefacts. Il réagit à la perception qu'une navigation N_i et un ensemble de documents D_i ont été créés. Dans ce cas la première action de l'agent Reformulation est de créer un artefact Profil :

$$P_i = (K_{N_i}, \emptyset, \emptyset, \emptyset)$$

La deuxième partie du fonctionnement de l'agent Reformulation se base sur le retour de pertinence de l'utilisateur depuis l'artefact Interface. Dans notre système nous avons choisi le feedback explicite en demandant à utilisateur d'appuyer sur un bouton "c'est pertinent" pour exprimer son jugement de pertinence. L'ensemble des documents sélectionnés comme pertinents constitue l'ensemble $D_{P_i}^{REL}$ qui est ajouté à P_i.

La proposition de reformulation se base sur les termes fréquents indexant les documents jugés pertinents qui sont obtenus par $Index(D_{P_i}^{REL})$. L'ensemble $Proposition(N_i)$ contenant un ensemble de termes proposés par l'agent Reformulation à l'utilisateur est construit tel que :

$$\forall d \in D_{P_i}^{REL}, \forall t \in Index(d)/ \\ t \notin K_{P_i}^{INIT} \cup T_{P_i}^{ACC} \cup T_{P_i}^{REJ}, \\ t \in Proposition(N_i)$$

Cet ensemble est proposé à l'utilisateur qui peut choisir de les accepter (ensemble T_{acc}) ou de les rejeter (ensemble T_{rej}). Le profil est mis à jour par l'agent Reformulation avec

$$T_{P_i}^{ACC} = T_{P_i}^{ACC} \cup T_{acc}$$
$$T_{P_i}^{REJ} = T_{P_i}^{REJ} \cup T_{rej}$$

Si $T_{P_i}^{ACC}$ a été modifié, l'agent Reformulation lance une nouvelle recherche en utilisant la nouvelle requête :

$$Recherche\left(K_{P_i}^{INIT} \cup T_{P_i}^{ACC}\right) \mapsto D'$$

puis met à jour l'ensemble des documents pertinents à base de la recherche d'information.

$$D_i = \left(D_{i,RI} \cup D', D_{i,REC}\right)$$

Ce processus de reformulation, ici décrit après le premier résultat d'une requête, sera réitéré à chaque fois que l'utilisateur sélectionnera des documents proposés comme étant pertinents (ceux-ci pouvant avoir été issus d'une requête déjà re-formulée).

Agent Communautaire.
La couche décision contient un unique agent Communautaire dont le rôle est de proposer des documents qui ont été jugés pertinents pour des profils de navigation proche. À la création d'un profil P_i, l'agent Communautaire analyse tous les autres Profils P_j pour comparer leurs termes initiaux et acceptés à ceux de P_i. Si tous les termes de P_i sont inclus dans ceux d'un profil P_j, les documents jugés pertinents pour P_j seront recommandés pour P_i. La construction de l'ensemble D_{AJ} représentant l'ensemble des documents à ajouter à ceux recommandés se fait comme suit :
$\forall P_j \in \mathcal{P}/K_{P_i}^{INIT} \subset K_{P_j}^{INIT} \cup T_{p_j}^{ACC}$,
$D_{P_j}^{REC} \subset D_{AJ}$

$$D_j = (D_{j,RI}, D_{j,REC} \cup D_{AJ})$$

4 Application à la navigation dans un corpus de document

La plateforme multi-agent de navigation personnalisée, modélisée dans la section précédente, a pour but d'être implémentée pour un prototype d'accès à un corpus numérique du droit international du transport. Nous décrivons dans cette section, le corpus utilisé pour le prototype, l'implémentation en cours de réalisation avec la plateforme Cartago [45], puis un scénario d'utilisation.

4.1 Le corpus de l'IDIT

Un prototype de plateforme de navigation sur un corpus numérique dans le domaine du droit international du transport est en développement dans le cadre du projet PlaIR 2.0 (Plateforme d'Indexation Régionale) en collaboration avec l'Institut du Droit International du Transport (IDIT).

Le corpus, constitué par l'IDIT [5], contient autour de 40 000 références incluant des documents classés en quatre groupes : jurisprudence,

articles, règlementation, fonds documentaires. Ces documents sont indexés à l'aide d'une terminologie du domaine et par une analyse plein texte des documents. La personnalisation en fonction des utilisateurs et des navigations est très intéressante dans ce type de corpus. D'une part les utilisateurs n'ont pas les mêmes besoins pour une même requête du fait de niveaux d'expertises différents. D'autre part, des navigations "typiques" ont lieu, sans être précisément formalisées, selon des usages fréquents comme des études comparatives de cas, de jurisprudence, des recherches précises, ...

4.2 Les artefacts

La modélisation des artefacts du système pour leur implémentation dans Cartago [45] est représentée par la figure 2. Un artefact est défini par ses opérations et ses propriétés observables. À titre d'exemple, nous citons quelques opérations de l'artefact Profil qui permettent aux agents de mettre à jour les informations qu'il contient : $SetOwner$, $UpdateQuery$, $SetTacc$ et $SetTrej$ permettent respectivement de modifier l'utilisateur, les termes de son besoin informationnel courant, les termes acceptés et rejetés. Par ailleurs, les propriétés observables donnent des informations aux agents. Par exemple pour l'artefact Recherche $currentQuery$ et $currentResult$ contiennent respectivement les mots-clés d'une requête et les documents obtenus en résultat.

La notion de liens entre les artefacts (Links dans Cartago) permet le partage des opérations entre les artefacts. Dans l'artefact Profil, l'agent peut accéder et ajouter les documents pertinents ainsi que les besoins d'informations de l'utilisateur $AddRelevantD$, $AddQuery$. Par manque d'espace, nous ne pouvons pas détailler ici tous les artefacts du modèle.

La couche décisionnelle est implémentée en Jason, ce qui nous permet d'utiliser la plateforme intégrée *JaCa* (Jason+Cartago).

4.3 Scénario d'exécution

Dans cette partie, nous décrivons un scénario de navigation au sein du corpus de l'IDIT.

Requête initiale. Supposons qu'un utilisateur, Paul, initie une navigation en cherchant toutes les décisions juridiques ayant impliquées des camions. Sa requête initiale est $K_{N_1}^{INIT} = \{camions\}$. L'agent Interface crée un artefact Navigation défini par $N_1 = (\{camions\}, Paul)$.

L'agent Interface lance, via l'artefact Recherche, une première requête : $Recherche(camions)$ qui renvoie deux documents $D_{N_1,RI} = \{Decision_1, Decision_{10}\}$. Il s'ensuit la création d'un artefact Document $D_1 = \{\{Decision_1, Decision_{10}\}, \emptyset\}$.

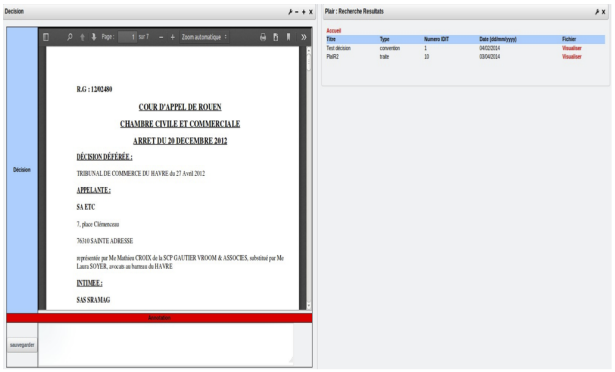


FIGURE 3 – Retour de pertinence

Reformulations. L'utilisateur choisit de visualiser les deux documents (Fig3). Mais, il juge la décision$_{10}$ comme un résultat pertinent. Ce retour de pertinence, ainsi que les premiers résultats, entraîne la création d'un artefact Profil par l'agent Reformulation initialisé à $P_1 = (\{camions\}, \emptyset, \emptyset, \{Decision_{10}\})$. L'agent Reformulation extrait les termes de l'index de la décision$_{10}$. $Index(Decision_{10})$ qui renvoie {véhicules motorisés, camions, fruits, aliments, accidents, motocyclettes, automobiles}.

Les termes ayant le poids le plus important et ne contenant pas les termes du besoin d'information initial sont : *véhicules motorisés*, *accidents* et *fruits* . L'agent Reformulation propose alors ces termes à l'utilisateur qui a le choix de les accepter ou les rejeter. Supposons, que Paul n'accepte que les deux derniers, son profil de navigation est alors devenu : $P_1 =$ {{camions}, {accidents , fruits}, {véhicules motorisés}, $\{Decision_{10}\}$}.

L'agent Reformulation lance une nouvelle recherche :
$Recherche(camions, accidents, fruits)$
qui renvoie comme résultats les décisions $(1, 10, 15, 20$ et $19)$. L'utilisateur consulte les décisions $(20, 19$ et $15)$ et juge les décisions 15 et 19 comme pertinentes. L'agent Reformulation met alors à jour le profil de l'utilisateur $Paul$.

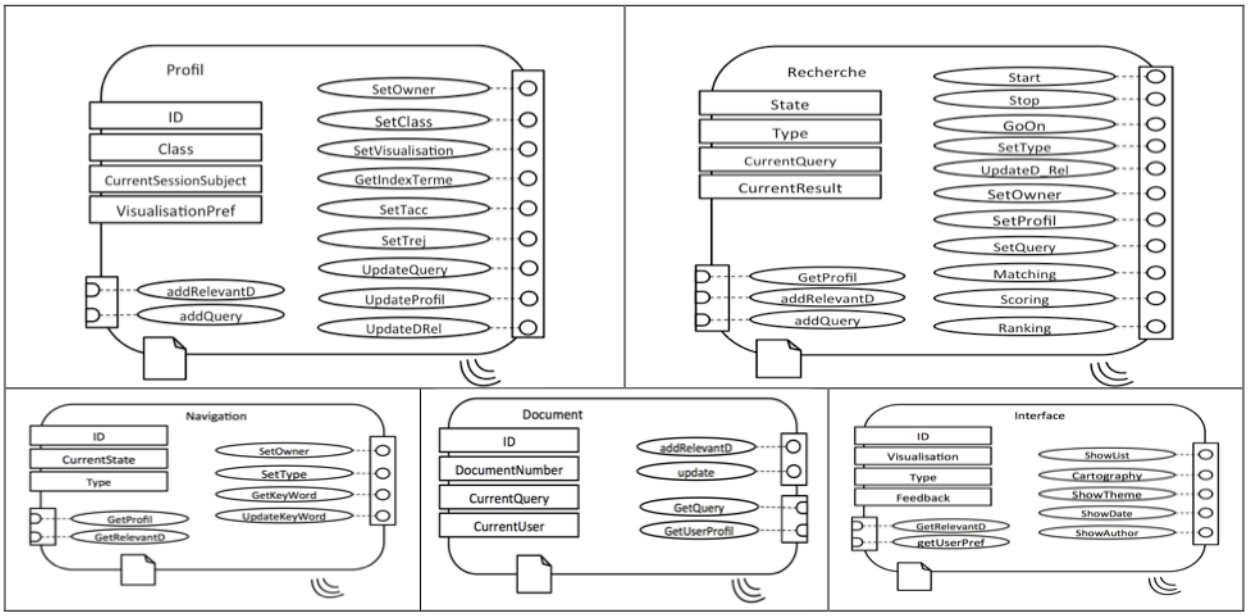


FIGURE 2 – Les artefacts

$P_1 = (\{\text{camions}\}, \{\text{accidents, fruits}\}, \{\text{véhicule motorisés}\}, \{Decision_{10}, Decision_{15}, Decision_{19}\})$.

Les termes de l'index de la décision$_{15}$ et la décision$_{19}$ sont : *fruits*, *camions*, *dommages*, *marchandises*, *poids lourd*, *pourcentage de dommages* et *délai de livraison*. À partir de la proposition de ces termes et qui ne font pas partie du besoin d'information initial et des termes déjà acceptés ou rejetés, l'utilisateur sélectionne les termes qui lui semblent utiles pour affiner son besoin d'information, ce qui amène à la proposition de nouveaux documents que l'utilisateur pourra à nouveau juger comme pertinents. Ce processus continue jusqu'à ce que l'utilisateur n'accepte plus de termes pour affiner son besoin d'information ou s'il exprime un nouveau besoin qui caractérise une nouvelle navigation.

Recommandation communautaire. Supposons qu'un utilisateur, Jean, utilise la plateforme pour une navigation ultérieure à Paul. L'artefact de navigation associé est $N_2 = (\{\text{fruits, délai de livraison}\}, Jean)$. Parallèlement aux documents proposés par les agents Interface et Reformulation, l'agent Communautaire compare ce besoin d'information initial avec les autres profils connus. Comme résultat il trouve $\{\text{fruits, délai de livraison}\} \subset \{K^{INIT} \cup T^{ACC}, Paul\}$. Il propose comme documents recommandés les documents jugés pertinents par $Paul$ pour ce besoin d'information à savoir les décisions $(27, 15, 19$ et $10)$. Le résultat à afficher à l'utilisateur est composé des résultats de l'agent Interface et des recommandations de l'agent Communautaire : Décision$_1$, Décision$_{51}$, Décision$_{11}$, Décision$_{19}$ et Décision$_{27}$, Décision$_{15}$, Décision$_{10}$, Décision$_{19}$.

5 Conclusion et Perspectives

Nous avons présenté à travers cet article un modèle multi-agent pour accompagner la navigation dans un corpus numérique de documents. Le corpus de documents du Droit International du Transport, considéré dans le projet PlaIR 2.0, est représentatif d'un cas d'accès à l'information où la personnalisation est un aspect crucial à prendre en compte et qui comporte différents points de vue à savoir un profil propre à l'utilisateur, un cas d'usage ref pour la navigation, une proximité sémantique entre documents ou une recommandation construite sur l'historique des navigations.

Nous avons proposé une modélisation sous la forme d'une couche d'artefacts pour représenter par un environnement partagé la navigation des utilisateurs, et sous la forme d'une couche décisionnelle, les agents qui vont, conjointement avec l'utilisateur influencer la navigation en faisant évoluer leur environnement. Un scénario de navigation illustre de quelle manière les

agents vont pro-activement proposer des reformulations de requêtes ou des documents jugés pertinents lors de navigations passées. En perspectives, nous envisageons une validation expérimentale sur un panel d'utilisateurs.

6 Remerciements

Les travaux menés dans cet article bénéficient d'un financement du Grand Réseau de Recherche : Logistique, Mobilité, Numérique de la Région Haute-Normandie (projet PlaIR 2.0, 2013-2016).

Références

[1] Nodine, M., Fowler, J., Ksiezyk, T., Perry, B., Taylor, M., et Unruh, A. (2000). Active information gathering in infosleuthTM. International Journal of Cooperative Information Systems, 9(01n02), 3-27.

[2] Marc Côté, Nader Troudi, *NetSA : Une Architecture Multiagent pour la Recherche sur Internet*, Expertise Informatique, Vol 3(3), 1998.

[3] B. Chaib-draa, I. Jarras et B. Moulin, *Systèmes multi agents : Principes généraux et applications*, Edition Hermès (2001).

[4] Elayeb Bilel, *SARIPOD : Système multi-Agent de Recherche Intelligente POssibiliste de Documents Web*, Thèse de doctorat en informatique, Université de Toulouse,Toulouse (France), 2009.

[5] *Institut du Droit International des Transports*, Site : http ://www.idit.asso.fr/.

[6] Gowan J., *A multiple model approach to personalised information access*, Master thesis in computer science, Faculty of science, Université de College Dublin, February, 2003.

[7] Sieg A., Mobasher B., Lytinen S., Burke R., *Using Concept Hierarchies to Enhance User Queries in Web-based Information Retrieval* , Artificial Intelligence and Applications(AIA), 2004.

[8] Katia Sycara and Keith Decker and Anandeep Pannu and Mike Williamson and Dajun Zeng, *Distributed Intelligent Agents*, IEEE Expert, Vol. 11, pp. 36-46, 1996.

[9] Grey D.J., Dunne G. and R.I. Ferguson, *A Mobile Agent Architecture for Searching the WWW*, In Proc. Workshop on Agents in Industry, 4th International Conference of Autonomous Agents, Barcelona, 2000.

[10] Lemouzy, Sylvain, *Systèmes interactifs auto-adaptatifs par systèmes multi-agents auto-organisateurs : application à la personnalisation de l'accès à l'information*, Thèse de doctorat en informatique, Université Paul Sabatier - Toulouse III, Toulouse (France), 2011.

[11] Jansen, B.J., Spink, A. and Saracevic, T. (2000). *Real life, real users, and real needs : A study and analysis of user queries on the web*. Information Processing and Management, 36(2), 207-227

[12] Spink, A. et Jansen, B. (2004). *Web search : public searching of the Web*. Netherlands : Kluwer Academic Publishers.

[13] Leake, D. B., et Scherle, R. (2001, January 14 - 17). *Towards context-based search engine selection*. Paper presented at the International Conference on Intelligent User Interfaces, Santa Fe, New Mexico, United States.

[14] T. Saracevic. *Relevance : A review of the literature and a framework for thinking on the notion in information science*. part iii : Behavior and effects of relevance. J. Am. Soc. Inf. Sci. Technol., 58(13) :2126-2144, 2007.

[15] Borlund, P. (2003). *The concept of relevance in IR*. Journal of the American Society for information Science and Technology, 54(10), 913-925.

[16] Mizzaro, S. *Relevance : The whole history*. JASIS, 1997, vol. 48, no 9, p. 810-832.

[17] Liu, F., Yu, C., et Meng, W. (2004). *Personalized web search for improving retrieval effectiveness*. Knowledge and Data Engineering, IEEE transactions on, 16(1), 28-40.

[18] Sieg, A., Mobasher, B., et Burke, R. (2007, November). *Web search personalization with ontological user profiles*. In Proceedings of the sixteenth ACM conference on Conference on information and knowledge management (pp. 525-534). ACM.

[19] Teevan, J., Morris, M. R., et Bush, S. (2009, February). *Discovering and using groups to improve personalized search*. In Proceedings of the Second ACM International Conference on Web Search and Data Mining (pp. 15-24). ACM.

[20] Morris, M. R., Teevan, J., et Bush, S. (2008, November). *Enhancing collaborative web search with personalization : groupization, smart splitting, and group hit-highlighting*. In Proceedings of the 2008 ACM conference on Computer supported cooperative work (pp. 481-484). ACM.

[21] Hupfer, M. E., et Detlor, B. (2006). *Gender and Web information seeking : A self-concept orientation model*. Journal of the American Society for Information Science and Technology, 57(8), 1105-1115.

[22] Weber, I., et Castillo, C. (2010, July). *The demographics of web search*. In Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval (pp. 523-530). ACM.

[23] Martin, I., et Jose, J. M. (2004, April). *Fetch : A Personalised Information Retrieval Tool*. In RIAO (Vol. 4, pp. 405-419).

[24] Cheverst, K., Davies, N., Mitchell, K., Friday, A., et Efstratiou, C. (2000, April). *Developing a context-aware electronic tourist guide : some issues and experiences*. In Proceedings of the SIGCHI conference on Human Factors in Computing Systems (pp. 17-24). ACM.

[25] Chen, L., et Sycara, K. (1998, May). *WebMate : a personal agent for browsing and searching*. In Proceedings of the second international conference on Autonomous agents (pp. 132-139). ACM.

[26] Hu, J., Zeng, H. J., Li, H., Niu, C., et Chen, Z. (2007, May). *Demographic prediction based on*

user's browsing behavior. In Proceedings of the 16th international conference on World Wide Web (pp. 151-160). ACM.

[27] Jones, R., Kumar, R., Pang, B., et Tomkins, A. (2007, November). *I know what you did last summer : query logs and user privacy*. In Proceedings of the sixteenth ACM conference on Conference on information and knowledge management (pp. 909-914). ACM.

[28] Karweg, B., Huetter, C., et Böhm, K. (2011, October). *Evolving social search based on bookmarks and status messages from social networks*. In Proceedings of the 20th ACM international conference on Information and knowledge management (pp. 1825-1834). ACM.

[29] Hecht, B., Teevan, J., Morris, M. R., et Liebling, D. J. (2012). *SearchBuddies : Bringing Search Engines into the Conversation*. ICWSM, 12, 138-145.

[30] Vallet, D., Cantador, I., et Jose, J. M. (2010). *Personalizing web search with folksonomy-based user and document profiles*. In Advances in Information Retrieval (pp. 420-431). Springer Berlin Heidelberg.

[31] Shen, X., Tan, B., et Zhai, C. (2005, August). *Context-sensitive information retrieval using implicit feedback*. In Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval (pp. 43-50). ACM.

[32] Speretta, M., et Gauch, S. (2005, September). *Personalized search based on user search histories*. In Web Intelligence, 2005. Proceedings. The 2005 IEEE/WIC/ACM International Conference on (pp. 622-628). IEEE.

[33] Yau, S. S., Liu, H., Huang, D., et Yao, Y. (2003, November). *Situation-aware personalized information retrieval for mobile internet*. In Computer Software and Applications Conference, 2003. COMPSAC 2003. Proceedings. 27th Annual International (pp. 639-644). IEEE.

[34] Dumais, S., Cutrell, E., Cadiz, J. J., Jancke, G., Sarin, R., et Robbins, D. C. (2003, July). *Stuff I've seen : a system for personal information retrieval and re-use*. In Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval (pp. 72-79). ACM.

[35] Ying, J. J. C., Lu, E. H. C., Lee, W. C., Weng, T. C., et Tseng, V. S. (2010, November). *Mining user similarity from semantic trajectories*. In Proceedings of the 2nd ACM SIGSPATIAL International Workshop on Location Based Social Networks (pp. 19-26). ACM.

[36] Sun, J. T., Zeng, H. J., Liu, H., Lu, Y., et Chen, Z. (2005, May). *Cubesvd : a novel approach to personalized web search*. In Proceedings of the 14th international conference on World Wide Web (pp. 382-390). ACM.

[37] Pretschner, A., et Gauch, S. (1999). *Ontology based personalized search*. In Tools with Artificial Intelligence, 1999. Proceedings. 11th IEEE International Conference on (pp. 391-398). IEEE.

[38] Bellotti, V., Begole, B., Chi, E. H., Ducheneaut, N., Fang, J., Isaacs, E., ... et Walendowski, A. (2008, April). *Activity-based serendipitous recommendations with the Magitti mobile leisure guide*. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (pp. 1157-1166). ACM.

[39] Zhang, B. T., et Seo, Y. W. (2001). *Personalized web-document filtering using reinforcement learning*. Applied Artificial Intelligence, 15(7), 665-685.

[40] Silverstein, C., Marais, H., Henzinger, M., et Moricz, M. (1999, September). *Analysis of a very large web search engine query log*. In ACm SIGIR Forum (Vol. 33, No. 1, pp. 6-12). ACM.

[41] He, D., Göker, A., et Harper, D. J. (2002). *Combining evidence for automatic web session identification*. Information Processing et Management, 38(5), 727-742.

[42] Jones, R., et Klinkner, K. L. (2008, October). *Beyond the session timeout : automatic hierarchical segmentation of search topics in query logs*. In Proceedings of the 17th ACM conference on Information and knowledge management (pp. 699-708). ACM.

[43] Daoud, M., Lechani, L. T., et Boughanem, M. (2009). *Towards a graph-based user profile modeling for a session-based personalized search*. Knowledge and Information Systems, 21(3), 365-398.

[44] Omicini, A., Ricci, A., Viroli, M. *Artifacts in the A&A meta-model for multi-agent systems. Autonomous Agents and Multi-Agent Systems*. Autonomous agents and multi-agent systems, 17(3), 432-456.(2008)

[45] A. Ricci, M. Viroli, and A. Omicini. *CArtAgO : An infrastructure for engineering computa- tional environments in MAS*. In D. Weyns, H. V. D. Parunak, and F. Michel, editors, Proc. of E4MAS, pages 102-119, Hakodate, Japan, 2006.

Approche décentralisée pour un apprentissage constructiviste en environnement continu : application à l'intelligence ambiante.

S. Mazac[a,b] sebastien.mazac@ubiant.com
F. Armetta[a] frederic.armetta@univ-lyon1.fr
S. Hassas[a] salima.hassas@univ-lyon1.fr

[a]Université Lyon 1, LIRIS, UMR5205, F-69622, France

[b]ubiant, France

Résumé

Le paradigme constructiviste de l'apprentissage en intelligence artificielle (IA) se développe largement à travers des concepts tels que l'IA incarnée, l'IA énactive ou la Robotique développementale. L'objectif commun à ces approches est de créer des agents autonomes car dotés d'une capacité d'adaptation à leur environnement et même d'apprentissage, à l'image des organismes biologiques. Un vaste champ d'application inclut tous les systèmes en interaction avec un environnement complexe, dont les objectifs sont variés et non prédéfinis. Suivant ce positionnement, nous nous intéressons ici au problème de l'intelligence ambiante. Nous proposons un modèle décentralisé d'apprentissage constructiviste pour un système d'AmI basé sur une architecture multi-agent. Avec ce travail nous abordons notamment le problème de l'amorçage de l'apprentissage sensorimoteur pour des environnements réels continus sans modélisation de l'environnement à priori.

Mots-clés : *Apprentissage Développemental, Constructivisme, Intelligence ambiante, Intelligence artificielle, Apprentissage, Systèmes multi-agents*

1 Introduction

Commençons par donner un exemple minimal permettant d'illustrer la problématique. Le paradigme de l'intelligence ambiante (AmI) vise à exploiter grâce à l'intelligence artificielle (IA) les possibilités offertes par le développement des environnements connectés, afin d'offrir aux habitants/usagers divers services (voir [Friedewald et al., 2005] et [Augusto, 2007]). Pour cela l'apprentissage est un élément crucial, en particulier la reconnaissance d'activités qui suppose l'apprentissage de motifs d'interactions variés comme présenté dans [Aztiria et al., 2010]. Dans ce contexte, nous définissons un système d'apprentissage S comme constitué d'un ensemble de capteurs $\{c_1; c_2 ... c_n\}$ et d'un ensemble d'actionneurs $\{a_1; a_2 ... a_n\}$. S est considéré comme un *agent* [1] *autonome* qui doit découvrir et apprendre des relations entre ces différentes variables, par exemple un lien entre a_x, c_y et c_z. Par la suite l'agent peut alors utiliser cette connaissance, par exemple dans le cadre de la réalisation d'un objectif défini par l'utilisateur. Supposons que l'actionneur a_x soit une prise actionnable sur laquelle est branchée un appareil non identifié D, que c_y soit le capteur de la consommation électrique de cette prise et c_z un capteur de luminosité dans la pièce. Si D est une lampe, S doit déterminer qu'il peut agir sur la consommation et la luminosité en activant et désactivant la prise. Il en va de même avec la température si jamais on remplace D par un chauffage électrique. De telles régularités dans l'expérience de l'agent peuvent sembler triviales et l'apprentissage relativement aisé. C'est éventuellement le cas si le concepteur du système le dote d'une représentation à priori, telle que la caractérisation des évènements et la façon d'appréhender leurs liens temporels. L'élaboration d'une telle représentation permet en effet de focaliser la tâche d'apprentissage sur un objectif précis et de restreindre l'espace de recherche en ayant préconstruit une partie de la solution, notamment en limitant les façons possibles de « percevoir » les données brutes. Cependant nous nous positionnons ici dans le cas d'un apprentissage plus complet et considérons le cas où l'agent doit construire seul sa propre représentation à partir des données brutes. Dans ce cas, l'agent doit réaliser un double apprentissage puisqu'il doit en parallèle : apprendre à percevoir efficacement, de façon à : apprendre des régularités ou motifs récurrents. Avant d'introduire plus en

1. Un agent intelligent en IA ([Russell and Norvig, 2009])

détail ce *problème d'amorçage*, et de présenter notre contribution, les paragraphes suivants décrivent succintement les avancées des approches constructivistes et de la robotique développementale sur ces questions.

2 Le problème des environnements continus

Dans le cadre d'environnements réels tels que celui de l'AmI ou de la robotique, le système doit donc faire face à la complexité du flot continu de données brutes provenant du matériel. C'est un problème de *discrétisation* de l'expérience, qui doit être réalisée de manière adaptée à la relation système-environnement. Par exemple l'agent doit pouvoir précisément identifier un évènement dans le temps et l'*espace* (ensemble des domaines de valeur des variables), mais la tâche qui consiste à délimiter dans la trame de l'expérience ce qui peut être considéré comme un évènement pertinent est particulièrement difficile. Le cerveau humain est capable de focaliser son attention sur les détails significatifs et d'ignorer tout le reste. Cette propriété de la cognition n'est pas évidente à reproduire en IA et certains problèmes fondamentaux bien connus en dépendent (*perceptual aliasing*, *frame problem*, etc.). De façon plus générale ces problèmes sont donc liés à la difficulté de gérer la complexité des informations du monde réel sans passer par une modélisation simplificatrice, connue en IA comme le *paradoxe de Moravec*. Ce paradoxe pointe du doigt le fait plutôt surprenant que les *capacités sensori-motrices* basiques des organismes vivants sont plus difficiles à reproduire artificiellement que les capacités cognitives de plus haut niveau tel que le raisonnement, comme le souligne [Brooks, 1991]. Dans le cadre de l'apprentissage constructiviste appliqué à des systèmes réels, ces problèmes sont bien sûr cruciaux.

3 Apprentissage constructiviste et robotique développementale

En psychologie, la théorie constructiviste de l'apprentissage développée en particulier par [Piaget, 1954] propose une vision de l'apprentissage comme un processus actif de construction et d'adaptation d'une représentation par le sujet, en interaction avec son environnement, plutôt que comme la simple acquisition d'un modèle figé du monde. A la suite de [Drescher, 1991] qui s'inspira directement de la théorie de Piaget pour proposer un modèle d'apprentissage artificiel constructiviste, certains chercheurs se sont intéressés à l'apprentissage de schémas sensorimoteurs (voir : [Guerin, 2011]). Dans ce modèle la brique élémentaire de représentation est appelée un schéma en référence au concept de *schème* sensorimoteur de la théorie de Piaget. Un schéma est un triplet : contexte C, action A, résultat R qui signifie pour l'agent : « si j'exécute A en observant C, alors j'observe R ». De nouveaux schémas sont appris de façon incrémentale alors que l'agent exécute aléatoirement des actions à la manière d'un enfant qui tâtonne. Le schéma est une structure intéressante car elle permet de relier une action de l'agent aux conséquences qu'il perçoit dans l'environnement, donc basé sur sa propre expérience. Afin de permettre l'abstraction, le mécanisme d'apprentissage de schémas propose également le concept d'*item synthétique*, afin de représenter une notion nouvelle à partir des besoins liés à l'expérience, comme présenté dans [Perroto et al., 2010]. Cependant, ces modèles sont intéressants d'un point de vue théorique mais difficilement applicables tels quels à des problèmes réels en environnement continu à cause de la complexité des algorithmes utilisés. C'est pourquoi la plupart des travaux dans ce domaine concernent des environnements simulés discrets. Le domaine de *robotique développementale* (voir [Lungarella et al., 2003]) nous intéresse donc particulièrement puisqu'il s'inscrit tout à fait dans cette approche, et doit gérer les environnements réels. De plus, habituellement le concepteur ne définit pas une tâche particulière mais le robot doit plutôt explorer librement son environnement. Il faut donc prévoir des mécanismes de *motivation* afin de susciter et guider l'exploration ([Oudeyer et al., 2007]). La capacité de prédire est une composante fondamentale du mécanisme d'apprentissage duquel émerge la cognition, comme énoncé par [Von Glasersfeld, 1984] [2]. Les fonctions cognitives de plus haut niveau, telles que par exemple la planification, peuvent être considérées comme la prédiction basée sur l'expérience, d'une série d'évènements. C'est pour-

2. *"Quite generally, our knowledge is useful, relevant, viable, or however we want to call the positive end of the scale of evaluation, if it stands up to experience and enables us to make predictions and to bring about or avoid, as the case may be, certain phenomena (i.e., appearances, events, experiences). If knowledge does not serve that purpose, it becomes questionable, unreliable, useless, and is eventually devaluated as superstition."*

quoi la notion d'évènement est très importante dans le cadre des environnements continus. Sans modélisation préconstruite, l'agent doit en effet être capable de déterminer ce qu'il est pertinent de considérer comme un évènement. Et le seul critère qui permet d'en juger pour l'agent est l'utilisation de cet évènement dans le cadre d'une prédiction.

4 Le problème d'amorçage

Il existe des pistes intéressantes pour rendre plus efficaces les modèles existants et permettre des applications à des problèmes réels aux environnements continus. Par exemple nous pouvons mentionner [Chaput et al., 2003] qui proposent d'utiliser des cartes auto-organisatrices (SOM) pour améliorer l'apprentissage de schémas. Poursuivant dans cette direction, [Provost et al., 2006] proposent un système qui utilise un processus de discrétisation basé sur les SOM couplé à un apprentissage par renforcement dans un problème de navigation de robot. Dans le même esprit, [Linaker and Jacobsson, 2001] conçoivent un système composé d'un mécanisme d'extraction d'évènement (système de classification) qui convertit les données brutes (multidimensionnelles et horodatées) en des séries d'évènements sur une plus grande échelle de temps, ainsi que d'un mécanisme d'apprentissage par renforcement. Le problème majeur d'une telle approche réside dans le fait que le processus de classification à bas niveau est unique et défini à priori. En effet, pour une même variable, les classes d'évènements générées peuvent être différentes en fonction des patterns dans lesquels cette variable est impliquée, et d'autre part une classification réussie peut devenir inadaptée plus tard, suite à un changement de matériel par exemple.

Une solution possible est proposé par [Mugan and Kuipers, 2007] qui présentent un mécanisme d'extraction d'évènements couplé à un système d'apprentissage de règles prédictives, similaires aux schémas. A partir de variables continues, les évènements sont spécifiés comme des transitions entre des états qui ne sont pas définis initialement. Des prédictions sont apprises pour associer ces évènements et une boucle de rétroaction permet de guider le processus en favorisant la création de nouveaux états, pour apprendre de plus précis et de plus nombreux évènements et règles. Ce dernier travail illustre la mise en place d'une dynamique entre le processus de discrétisation de l'expérience et l'apprentissage de motifs sensori-moteurs, qui est au centre du problème d'amorçage. Un élément de représentation ou un motif peut être considéré comme pertinent s'il contribue à la construction de nouveaux motifs, et donc appartient à une représentation qui le justifie. En d'autres termes, l'agent doit apprendre à percevoir de manière adaptée pour pouvoir apprendre efficacement. Cette définition autoréférentielle, caractéristique de l'étude des systèmes vivants, illustre le *problème d'amorçage* énoncé par [Kuipers et al., 2006] qui réside dans l'apprentissage des motifs sensorimoteurs à partir des données brutes de capteurs inconnus (voir également [Guerin, 2011]). Pour les organismes biologiques, ce double apprentissage est probablement réalisée à la fois durant la phylogénèse et durant l'ontogénèse, mais il s'agit d'un seul et même problème dans le cas des systèmes artificiels. Dans la suite, nous proposons un modèle systémique, se focalisant sur les dynamiques entre les processus de traitement de l'information qui permettent de construire des motifs sensorimoteurs, et autorise de multiples implémentations de ces processus.

5 Présentation du modèle

5.1 Interactions Agent-Environnement

L'apprentissage doit donc reposer sur l'interaction agent-environnement, c'est-à-dire sur les signaux échangés à la frontière entre l'agent et son environnement. Dans notre modèle, cette frontière est modélisée par un ensemble de variables continues qui représentent l'ensemble des capteurs et des actionneurs : $V = \{v_1; v_2...v_n\}$. Tel que représenté sur la Figure 1-(a), ce flot continu de données brutes peut être interprété par l'agent comme une séquence d'évènements. Par la suite nous appelons expérience $E = \{e_1; e_2...e_n\}$ l'ensemble des évènements construits par l'agent, au sein de laquelle nous considérons qu'il existe des régularités. Une régularité est un motif récurrent que l'agent doit être capable d'isoler et d'identifier dans son espace de recherche. Du point de vue du système proposé, il n'y a pas de différence entre les variables représentant les capteurs et les actionneurs. Donc les motifs appris peuvent concerner indifféremment capteurs et actionneurs. Il faut cependant que des actions soient exécutées pour pouvoir apparaître dans des patterns sensori-moteurs. Dans le domaine de l'AmI, il est davantage contraignant qu'en robotique d'utiliser les interactions aléatoires

comme point de départ du développement. En effet un comportement par tâtonnement comme dans [Mugan and Kuipers, 2007] pourrait être trop long et ses conséquences trop gênantes pour les utilisateurs. Heureusement, nous pouvons utiliser à notre avantage une propriété des systèmes ambiants : les utilisateurs peuvent être à l'origine des actions possibles. D'ailleurs pour des raisons de sécurité et d'éthique, la conception d'un système ambiant devrait toujours faire en sorte que l'utilisateur garde le contrôle de son environnement. Donc nous considérons comme contrainte de conception que chaque action qui pourrait être décidée et réalisée par le système, doit aussi pouvoir être provoquée directement par l'utilisateur. D'autre part nous faisons l'hypothèse que l'agent est capable de percevoir une telle action produite par l'utilisateur, comme une forme de *proprioception*[3], de sorte que si cette action est impliquée dans un motif, l'agent l'apprenne et soit capable de la réaliser par lui-même plus tard. [Najjar and Reignier, 2013] font la même hypothèse et nomment cela « actions observées ». La phase initiale d'exploration aléatoire des systèmes constructivistes peut donc dans le cas des systèmes ambiants être remplacée par des actions provoquées par des utilisateurs, ce qui est intéressant car ainsi l'utilisateur est impliqué dès le début du processus d'apprentissage comme une forme de motivation pour l'agent.

5.2 Structure élémentaire de représentation

Le rôle de la structure élémentaire de représentation est d'exprimer le motif régulier que le système est capable d'apprendre au plus bas niveau. Plus généralement il s'agit d'exprimer une boucle élémentaire d'interaction entre l'agent et son environnement telle que celle exprimée par le *cercle fonctionnel* de [Von Uexküll, 1992]. Notons ici qu'une autre approche possible afin de s'affranchir du problème d'amorçage est de considérer que le schéma (ou toute autre élément représentationnel du même type) constitue une structure élémentaire indivisible intrinsèque au système. Par exemple [Georgeon and Aha, 2013] définissent une interaction sensorimotrice comme une primitive. Cela suppose néanmoins dans le cas de systèmes réels, d'implémenter ces primitives, et donc de connaître parfaitement les capacités sensori-motrices de l'agent. La même remarque s'applique à l'approche de [Brooks, 1991] avec l'architecture de subsomption pour la robotique où les comportements de bas niveau sont préalablement codés. Donc à l'image d'un schéma, une structure élémentaire de représentation est construite en définissant des éléments de représentation (ex. contexte, action, ...) et des liens logiques pour unir ces éléments (ex. *"et"*, *"pendant"*, *"consécutif"*, etc.). Dans le cas des environnements réels continus il y a une infinité de possibilités pour définir ces structures à partir des données brutes. Le problème d'amorçage se pose donc dès lors que le concepteur laisse tout ou partie de ce travail de discrétisation et d'abstraction à la charge de l'agent. Pour cela, nous choisissons de définir un élément de représentation générique *"évènement"*, que l'on peut distinguer en deux sous-catégories génériques. Le premier type d'évènement représente un moment ciblé de l'évolution d'une variable. Rappelons que comme une variable peut représenter aussi bien l'état d'un capteur que d'un actionneur, un évènement peut donc représenter une action. Le deuxième type d'évènement, appelé *association*, permet de décrire une relation entre deux ou plusieurs évènements. Les différentes implémentations possibles de ces éléments et des moyens de les produire constituent donc l'espace de recherche de la discrétisation de l'expérience continue en une série d'évènements. Par exemple, on peut définir un évènement comme étant une variation notable d'une variable, et une association comme étant une durée identifiable entre deux évènements. D'autre part on peut aussi spécifier qu'un évènement est, non pas une variation, mais un état stable et que l'association n'est pas une durée ciblée mais une inclusion, etc. Par combinaison de ces différents types d'évènements il est ainsi possible d'exprimer une régularité de type schéma entre autres. Nous présenterons plus loin les choix d'implémentations que nous utilisons pour réaliser ces structures.

5.3 L'activité du système multi-agent

Notre proposition est basée sur l'utilisation d'un système multi-agent[4] (SMA) pour réaliser l'apprentissage sensori-moteur décrit précédemment. Le système multi-agent proposé se compose de trois populations d'agents distinctes qui exécutent des fonctions complémentaires et

3. i.e. perception de ses propres actions. Dans le corps humain : sensibilité permettant de percevoir ses propres mouvements, la position du corps, etc.

4. Pour éviter toute confusion, le terme « agent » fera désormais référence à une entité du SMA et non au système global apprenant, que l'on désignera simplement comme « le système »

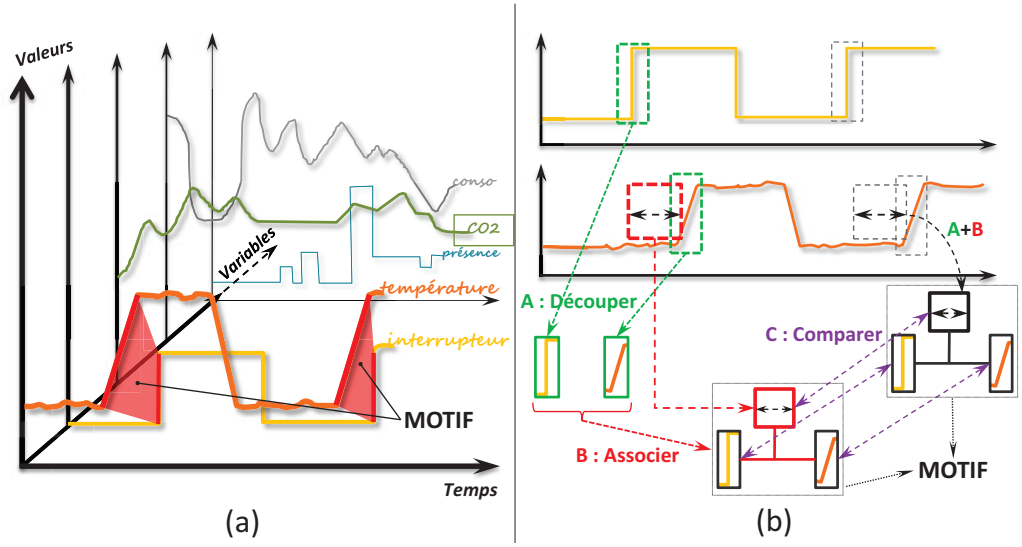


FIGURE 1 – (a) Expérience continue : ensemble de variables.
(b) Les 3 types d'opérations. *A* : Découpage temporel (création d'un évènement). *B* : Trouver corrélation entre évènement (création d'une association). *C* : Comparer des éléments de représentation.

interagissent afin de construire la représentation. Ils sont guidés par des boucles de rétroaction provenant de l'évaluation de la construction réalisée. En effet, en lien avec la notion de structure élémentaire de représentation présentée précédemment, nous définissons trois opérations élémentaires qui interviennent pour l'amorçage et l'apprentissage de motifs sensorimoteurs à partir de l'expérience continue, tel qu'illustré sur la Figure 1-(b). Tout d'abord il faut découper l'expérience continue des variables en des moments finis que l'on peut considérer comme des évènements. Ensuite il faut pouvoir associer ces évènements pour créer des évènements plus complexes (évènements *associations*), et enfin il est nécessaire de pouvoir comparer deux éléments de représentation, par exemple pour reconnaître différentes instances d'un même évènement. Un agent implémente donc une fonction correspondant à l'une de ces trois catégories d'opération. Le comportement d'une fonction peut varier en fonction d'un *paramètre* ajustable par l'agent, constituant ainsi un espace de recherche supplémentaire. Une certaine fonction peut convenir pour apprendre une régularité de l'expérience de l'agent à un certain niveau, mais échouer pour un autre aspect de l'expérience. L'exploration conjointe par les agents des différentes implémentations possibles de ces opérations et de leurs espaces de recherche respectifs fournit des propositions de discrétisations et de représentations multiples qui constituent l'espace de recherche global. L'évaluation des solutions oriente la recherche et renforce les zones intéressantes. La Figure 2 offre une vue synthétique sous forme de diagramme d'activité de ce processus de construction décentralisé.

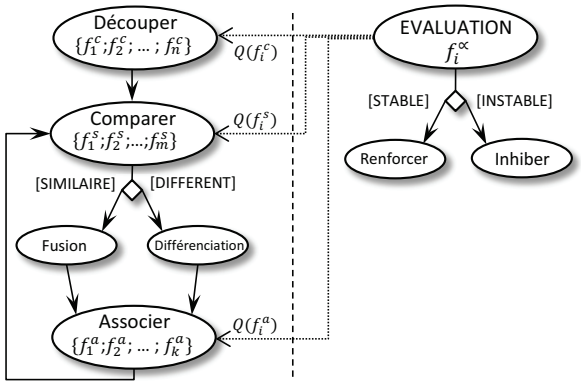


FIGURE 2 – Activité globale du système.

Nous pouvons formaliser ces 3 populations d'agents de la manière suivante :

– Nommons *"découper"* l'opération permettant de générer des évènements à partir des variables. L'ensemble des fonctions possibles de type "découper" est défini comme : $F^d = \{f_1^d; f_2^d; \ldots; f_n^d\}$. Les différentes fonctions implémentant cette opération peuvent être défi-

nie par :

$$f_i^d : v_i \rightarrow \{e_1^d; e_2^d; ...; e_n^d\}$$
$$V \rightarrow E$$

Ce type de fonction est donc implémenté par les *agents "découper"* (D) qui sont connectés à une variable. Par exemple, une implémentation possible est un agent qui utilise des fenêtres glissantes avec une durée paramétrable. L'exploration possible liée à cette fonction est donc la recherche d'une taille de fenêtre pertinente, adaptée à la variable et aux éventuels motifs concernés.

– Deuxièmement l'opération *Association* permet de « relier » les évènements existants. Une *fonction association* ($F^a = \{f_1^a; f_2^a; ...; f_n^a\}$) peut être définie par :

$$f_i^a : e_x, e_y \rightarrow e_z^a$$
$$E \times E \rightarrow E$$

Par exemple une implémentation possible de cette fonction est un *agent Association* (A) qui, à partir d'un élément de référence e_1, génère une représentation des durées entre les occurrences successives de e_1 et d'autres éléments e_i du système. L'espace de recherche de cet agent réside dans la manière dont l'agent sélectionne les durées des occurences des éléments.

– Le troisième type d'opération, *Similarité* permet de « comparer », étape nécessaire à toute manipulation des éléments de représentation. Une telle fonction ($F^s = \{f_1^s; f_2^s; ...; f_n^s\}$) de comparaison peut être définie par :

$$f_i^s : e_x, e_y \rightarrow s$$
$$E \times E \rightarrow [0, 1]$$

où e_x, e_y sont des éléments comparables car provenant d'une même source ($s = 1$ signifie totalement similaire, $s = 0$ totalement différent). L'implémentation de l'*agent Similarité* (S) va dépendre des structures de données utilisées pour les éléments de représentation. Dans cette version nous utilisons des histogrammes, donc une première implémentation d'un agent S est une fonction de comparaison d'histogrammes telle que l'intersection. Le paramétrage conditionne l'échantillonnage de l'histogramme, l'agent pouvant ainsi être plus ou moins précis pour évaluer la similarité de deux éléments.

Le traitement opéré par les agents S est de trier les instances d'évènements produits par les deux autres types d'agents et d'identifier les classes d'évènements intéressantes. Pour cela, les agents forment des organisations qui sont toujours composées d'au moins un agent S, comme illustré sur la Figure 3 qui montre les formes d'organisations possibles sous la forme d'un diagramme Agent-Groupe-Rôle ([Ferber and Gutknecht, 1998]). Au sein d'une organisation, un agent producteur d'évènements (D ou A) reçoit des données sélectionnées à un niveau inférieur, et les traite pour créer un nouvel évènement. Ces nouveaux évènements sont classés et sélectionnés par l'agent S, et diffusés à leur tour dans le système. Comme illustré sur la Figure 4, les agents peuvent appartenir à différentes organisations, dans lesquelles ils jouent différents rôles, à différents niveaux.

Comme on peut le voir sur la Figure 2, ce mécanisme est un processus circulaire, amorcé par la création d'évènements à partir des variables. A chaque fois qu'un nouvel évènement est créé par un agent D, il est comparé avec les précédents. Il peut alors être fusionné avec un élément similaire (la classe d'évènement se voit ainsi renforcée), ou à défaut initier une nouvelle distinction, c'est-à-dire une nouvelle classe d'évènements possibles. Ces évènements peuvent ensuite être associés, comparés, puis les associations sont utilisées à leur tour comme des évènements et ainsi de suite. Ces opérations génériques et la dynamique de leur interaction avec l'évaluation d'une représentation prédictive permettent de définir un modèle de base pour le mécanisme d'extraction de motifs sensori-moteurs.

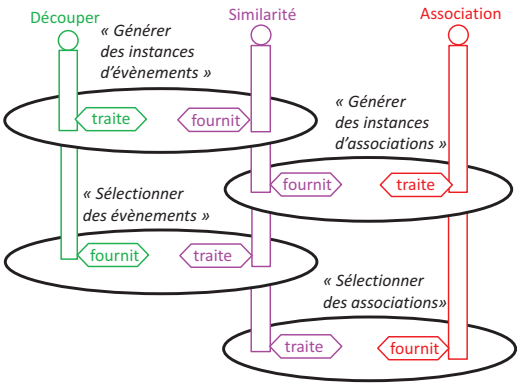


FIGURE 3 – Organisations possibles entre les types d'agents.

6 L'évaluation des organisations

L'espace de recherche du système est donc l'ensemble des fonctions possibles $F = F^D \cup F^A \cup$

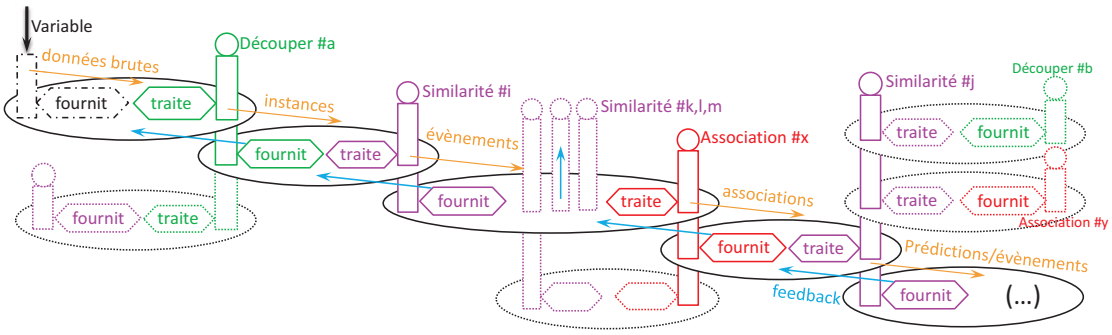


FIGURE 4 – Exemples d'interactions entre instances d'agents.

F^S. La pertinence d'une fonction peut être évaluée d'après sa participation à la construction de motifs, qui se fait nécessairement en collaboration avec d'autres types de fonctions. Nous pouvons représenter la rétroaction par une fonction d'évaluation $Q : f \in F \rightarrow [0,1]$ qui s'applique aux structures générées, et influence les agents participants. Ainsi le but du système est de trouver les combinaisons de fonctions qui permettent de construire des motifs pertinents, en maximisant l'évaluation. Cet ensemble peut être défini comme $F' \subset F = \{f' \in F' | Q(f') > T_Q\}$ où T_Q est un seuil pour conserver uniquement les structures ayant le meilleur score. Pour guider l'exploration des agents en fonction de cette évaluation, un marquage est effectué dans une grille partagée (espace de marquage) à chaque nouvelle évaluation d'une structure de représentation. La position du marquage dans la grille correspond aux paramètres des différentes fonctions employées pour construire la structure évaluée. L'évaluation des motifs peut correspondre à leur faculté à produire des prédictions fiables, comme présenté dans [Mugan and Kuipers, 2007].

Mais dans le cas du problème d'amorçage, il n'est pas possible d'évaluer la capacité prédictive du système, tant que l'on n'a aucun motif à considérer comme une structure prédictive. Pour pallier cela, nous introduisons une mesure d'*intérêt* pour orienter la recherche vers les zones intéressantes de l'espace de recherche lors de la phase d'amorçage. L'intérêt des éléments de représentation est défini par leur *spécificité* et leur *poids*. Si les processus de construction sont trop précis, une trop grande quantité d'évènements spécifiques va être générée, mais ayant un poids faible. Si au contraire, le processus est trop général, peu d'éléments seront générés et auront un poids fort mais ne seront pas spécifiques, c'est-à-dire qu'ils ne se démarqueront pas suffisamment du reste de l'expérience. Cette mesure repose donc sur l'idée de trouver un compromis entre précision et stabilité, et permet de guider le système avant que des motifs puissent être évalués, et ainsi surmonter le problème d'amorçage. La spécificité d'un élément peut être exprimée comme la différence entre cet élément et la classe d'élément la plus générale possible, appelée *référence*. Par exemple, pour une association qui contient les durées entre les occurrences de 2 évènements e_1 et e_2 , la référence est l'association qui contient toutes les durées entre les occurrences de e_1 et des évènements aléatoires e_i. La *spécificité* d'un évènement est : $s(e) = 1 - f^s(e, ref)$ définie entre 0 et 1. Notons $|e|$ le nombre d'occurences de l'évènement $e \in E^\alpha$, et E^α l'ensemble des classes d'évènements générées par l'agent S à partir de toutes les occurences produites.

Le *poids* d'un évènement est défini en fonction de la répartition des occurences de cet évènement par rapport aux autres évènements de spécificité équivalente : $w(e) = \frac{|e|}{\sum_{|s(e_i)-s(e)|<\epsilon; e_i \in E^\alpha} |e_i|}$.

L'intérêt d'un évènement est $i(e) = s(e) * w(e)$.

Une fois qu'un motif est identifié comme intéressant, il peut être considéré comme une structure prédictive entre un évènement e_1 et un évènement e_2, qui est évaluée de manière classique en fonction de son taux de succès (confiance) et de sa précision. Nous définissons pour cela un score $s = acc * rel$. La confiance (rel) est définie comme le rapport du nombre de prédictions réussies sur le nombre d'évènements e_1 observés : $rel = nb(predictions)/nb(e_1)$. Soit tol la tolérance de la prédiction, qui est l'écart type de la durée moyenne entre e_1 et e_2 lors des prédictions réussies. La précision (acc) est définie comme $acc = 1 - (tol * freq(e_2))$ ($acc = 0$ si

$tol < 0$), où $freq(e_2)$ est la fréquence d'apparition de e_2.

7 Résultats

Nous proposons d'abord des résultats portant sur des données simulées. Dans cette expérience nous simulons l'interaction simple entre des variables, du type : activations d'interrupteurs provoquant des variations progressives de variables bruitées avec plus ou moins de latence. Ces activations sont répétées après un intervalle de temps aléatoire borné. L'intérêt principal du simulateur est de pouvoir tester le système sur des échelles de temps plus courtes. Dans cette expérience nous ajoutons 7 variables $V1; ...; V7$, certaines étant liées par des régularités, comme illustré par la Figure 5. L'activation ou la désactivation de $V1$ entraîne des variations de $V2$ et $V3$. Il en va de même pour $V6$ et $V7$ avec des délais et des amplitudes différentes, tandis que $V4$ et $V5$ s'activent régulièrement et aléatoirement sans lien avec aucune autre variable.

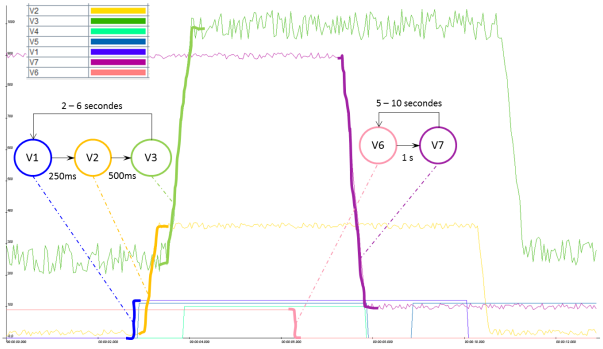


FIGURE 5 – Simulation de régularités entre variables continues.

Ce scénario a pour objectif de montrer que ces régularités émergent de l'activité du système sous forme d'associations qui restent stables. Pour cela nous visualisons les agents et les structures générées au sein du système sous la forme d'un graphe, grâce à la bibliothèque *Graphstream* ([Pigné et al., 2008]). Dans cette expérience le système est initialement composé de trois agents D par variable, des agents S sont générés à la demande, et les agents A apparaîssent lorsqu'un évènement créé est suffisamment intéressant. L'état initial est représenté sur la Figure 6-A. Sans rentrer dans les détails, l'implémentation utilisée est la suivante. Agent D : génération d'histogrammes contenant la variance sur des fenêtres glissantes. Agent A : génère des histogrammes contenant la durée entre les deux dernières occurrences d'évènements. Agent S : comparaison des histogrammes par intersection.

Lorsqu'une organisation est évaluée avec un score suffisamment élevé, les agents exploitent cette position : des évènements ou des associations sont générés (Figure 6-B [5]). Puis éventuellement les prédictions sont activées pour une association suffisamment intéressante (Figure 6-C). Lorsque la prédiction atteint un certain score, elle est confirmée. Lorsque l'évaluation n'est pas probante, les agents explorent d'autres possibilités en modifiant leur paramétrage. Une vidéo que le lecteur pourra consulter permet de mieux visualiser l'activité du système [6].

Un motif est donc constitué par un ensemble d'agents et un ensemble d'éléments de représentation. Ces éléments contiennent des données qui décrivent précisement la nature de cette régularité. Comme le montre la Figure 6-C, nous constatons donc que dans le cas de la simulation le système est capable de retrouver et de stabiliser toutes les régularités qui existent entre les variables, pour un temps d'apprentissage variable entre une quinzaine et une trentaine d'occurences des variations simulées, suivant les paramètres. D'autre part rares sont les régularités non existantes (faux positifs) qui apparaîssent. Cela correspond généralement à des coïncidences apprises qui ne peuvent être renforcées et disparaissent rapidement. Le nombre d'observations nécessaires à l'apprentissage d'un phénomène peut être réduit d'une expérience sur l'autre si on mémorise les espaces de marquage des agents, car de cette façon le système converge plus rapidement vers des choix de discrétisation adaptés. La figure 7-(a) montre les évènements du motif qui prédit une variation de V3 à partir de l'activation de V1. De gauche à droite, on visualise l'évènement initial $e1$, l'évènement association, et l'évènement prédit $e2$. Pour les évènements $e1$ et $e2$ les histogrammes montrent la répartition de la variance sur les fenêtres temporelles sélectionnées par rapport à la répartition en général de toutes les fenêtres. Ces évènements ont été détectés comme intéressants car ils sont spécifiques et suffisamment récurrents. Mais en définitive, c'est uniquement la participation de l'évènement à un pattern permettant de réaliser des prédictions effectives (cf. score de la prédiction), qui autorise à les considérer comme des évènements pertinents. L'his-

5. Par souci de lisibilité, l'affichage des différents éléments est épuré à chaque étape sur l'image

6. http://liris.cnrs.fr/sycosma/wiki/doku.php?id=iadev-intamb

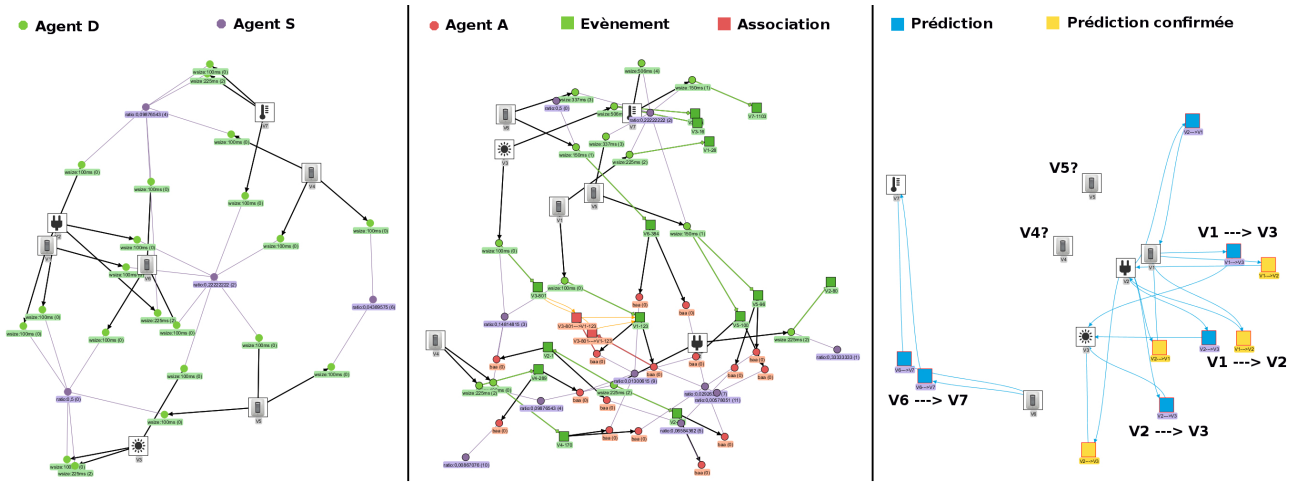


FIGURE 6 – A : Etat initial. B : Création d'associations. C : Motifs intéressants.

togramme de l'évènement association, montre quant à lui la répartition des durées observées entre les instances de $e1$ et $e2$ par rapport à la répartition des durées entre $e2$ et d'autres évènements choisis aléatoirement.

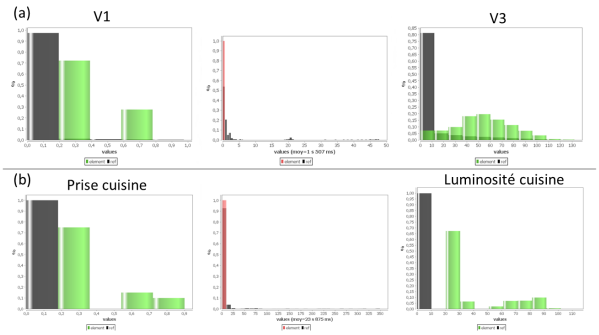


FIGURE 7 – Exemples des données des éléments perception et association d'un motif.

8 Conclusion et perspectives

Dans ce travail nous proposons une architecture qui permet de simuler et mettre en évidence la dynamique de l'amorçage de l'apprentissage de motifs sensorimoteurs à partir de signaux continus dans le cadre d'une approche constructiviste, sans s'attacher à une implémentation spécifique, et de manière décentralisée. Ce modèle autorise la création et le traitement parallèle d'une variété d'agents différents pour implémenter les différents rôles spécifiés, à l'image du concept de « société de l'esprit » développé par [Minsky, 1991]. Outre les expériences en simulation, quelques expériences préliminaires ont été réalisées à ce jour sur un système réel comprenant une trentaine de variables parmi lesquelles (température, luminosité, CO2, humidité, ouverture de porte, prise pilotable, etc.), dans trois pièces différentes. Une configuration préalable des plages de paramètres des agents en fonction des variables nous permet de borner un peu les espaces de recherche et d'accélérer le processus d'apprentissage. Au bout d'environ une journée (ce qui est relativement court par rapport aux échelles de temps de l'AmI), il est déjà possible d'observer quelques motifs. Par exemple le lien entre une prise pilotable et la luminosité dans une pièce, comme mentionné dans l'exemple d'introduction. Ce motif est illustré en Figure 7-(b), et les caractéristiques de la prédiction ainsi que son évaluation dans le Tableau 8. Ce travail d'implémentation est toujours en cours, en partenariat avec la société *ubiant* [7].

EXEMPLE MOTIF SIMULATION	e1	e2	Prédiction
Fréquence	5 s 823 ms	5 s 823 ms	-
Durée moyenne	304 ms	1 s 716 ms	298 ms
Tolérance	-	-	41 ms
Précision	-	-	0,994
Nombre d'observations	17	18	17
Confiance	-	-	1
Score	-	-	0,985
EXEMPLE MOTIF REEL	e1	e2	Prédiction
Fréquence	17 m 35 s 93 ms	16 m 2 s 4 ms	-
Durée moyenne	522 ms	3 s 461 ms	25 s 293 ms
Tolérance	-	-	3 s 207 ms
Précision	-	-	0,996
Nombre d'observations	17	20	17
Confiance	-	-	1
Score	-	-	0,993

FIGURE 8 – Evaluation des motifs.

7. http://www.ubiant.com

Pour illustrer la dynamique du système, nous montrons une instanciation minimale des différentes opérations du modèle, reposant sur des agents réactifs simples. L'analyse qualitative des résultats présentés ici démontre la possibilité d'obtenir des motifs satisfaisants de manière efficace par cette méthode, à partir des seules données brutes, en temps réel. Par la suite, nous allons enrichir le système en ajoutant d'autres implémentations d'agents et en faisant varier certains paramètres tels que le nombre d'agents, les choix et la vitesse d'exploration, divers seuils utilisés dans les évaluations, etc. De plus amples tests sont à effectuer pour étudier quantitativement les influences de ces divers changements sur l'efficacité du système. Il est clair que plus les possibilités de discrétiser et d'associer des évènements augmentent, plus le système peut être expressif, mais plus l'espace de recherche croît. Grâce à la nature décentralisée du système, les calculs sont cependant hautement parallélisables. La gestion de la mémoire pourrait être un défi non négligeable, même si très peu de données brutes sont stockées. Enfin nous allons poursuivre le travail commencé avec le système d'AmI réel sur des temps d'expérimentations plus longs, les premiers résultats étant très encourageants.

Références

[Augusto, 2007] Augusto, J. (2007). Ambient intelligence : the confluence of ubiquitous/pervasive computing and artificial intelligence. *Intelligent Computing Everywhere*, pages 213–234.

[Aztiria et al., 2010] Aztiria, A., Izaguirre, A., and Augusto, J. (2010). Learning patterns in ambient intelligence environments : a survey. *Artificial Intelligence Review*, 34(1) :35–51.

[Brooks, 1991] Brooks, R. A. (1991). Intelligence without representation. *Artificial Intelligence*, 47(1 - 3) :139 – 159.

[Chaput et al., 2003] Chaput, H. H., Kuipers, B., and Miikkulainen, R. (2003). Constructivist learning : A neural implementation of the schema mechanism. In *In Proceedings of the Workshop on SelfOrganizing Maps (WSOM03*.

[Drescher, 1991] Drescher, G. (1991). *Made-up minds : a constructivist approach to artificial intelligence*. The MIT Press.

[Ferber and Gutknecht, 1998] Ferber, J. and Gutknecht, O. (1998). A meta-model for the analysis and design of organizations in multi-agent systems. In *Multi Agent Systems, 1998. Proceedings. International Conference on*, pages 128–135.

[Friedewald et al., 2005] Friedewald, M., Costa, O. D., Punie, Y., Alahuhta, P., and Heinonen, S. (2005). Perspectives of ambient intelligence in the home environment. *Telematics and Informatics*, pages 221–238.

[Georgeon and Aha, 2013] Georgeon, O. and Aha, D. (2013). The Radical Interactionism Conceptual Commitment. *Journal of Artificial General Intelligence*, 4(2) :31–36.

[Guerin, 2011] Guerin, F. (2011). Learning like a baby : a survey of artificial intelligence approaches. *The Knowledge Engineering Review*, 26 :209–236.

[Kuipers et al., 2006] Kuipers, B. J., Beeson, P., Modayil, J., and Provost, J. (2006). Bootstrap learning of foundational representations. *Connection Science*, 18(2) :145–158.

[Linaker and Jacobsson, 2001] Linaker, F. and Jacobsson, H. (2001). Mobile robot learning of delayed response tasks through event extraction : A solution to the road sign problem and beyond. In *International Joint conference on artificial intelligence*, volume 17, pages 777–782. Lawrence erlbaum associates ltd.

[Lungarella et al., 2003] Lungarella, M., Metta, G., Pfeifer, R., and Sandini, G. (2003). Developmental robotics : a survey. *Connection Science*, 15(4) :151–190.

[Minsky, 1991] Minsky, M. (1991). Logical versus analogical or symbolic versus connectionist or neat versus scruffy. *AI magazine*, 12(2) :34.

[Mugan and Kuipers, 2007] Mugan, J. and Kuipers, B. (2007). Learning distinctions and rules in a continuous world through active exploration. In *Proceedings of the Seventh International Conference on Epigenetic Robotics (EpiRob-07)*, pages 101–108.

[Najjar and Reignier, 2013] Najjar, A. and Reignier, P. (2013). Constructivist Ambient Intelligent Agent for Smart Environments. In *PerCom - IEEE International Conference on Pervasive Computing and Communications*, San Diego, États-Unis.

[Oudeyer et al., 2007] Oudeyer, P.-Y., Kaplan, F., and Hafner, V. (2007). Intrinsic motivation systems for autonomous mental development. *Evolutionary Computation, IEEE Transactions on*, 11(2) :265–286.

[Perroto et al., 2010] Perroto, F., Alvarez, I., and Buisson, J.-C. (2010). Constructivist Anticipatory Learning Mechanism (CALM) : Dealing with Partially Deterministic and Partially Observable Environments. In *International Conference on Epigenetic Robotics (EpiRob)*, pages 110–120. Lund University Cognitive Science.

[Piaget, 1954] Piaget, J. (1954). *The Construction of Reality in the Child*. Basic Books.

[Pigné et al., 2008] Pigné, Y., Dutot, A., Guinand, F., and Olivier, D. (2008). GraphStream : A Tool for bridging the gap between Complex Systems and Dynamic Graphs.

[Provost et al., 2006] Provost, J., Kuipers, B. J., and Miikkulainen, R. (2006). Developing navigation behavior through self-organizing distinctive-state abstraction. *Connection Science*, 18(2) :159–172.

[Russell and Norvig, 2009] Russell, S. and Norvig, P. (2009). *Artificial Intelligence : A Modern Approach*. Prentice Hall Press, Upper Saddle River, NJ, USA, 3rd edition.

[Von Glasersfeld, 1984] Von Glasersfeld, E. (1984). An introduction to radical constructivism. *The invented reality*, pages 17–40.

[Von Uexküll, 1992] Von Uexküll, J. (1992). A stroll through the worlds of animals and men : A picture book of invisible worlds. *Semiotica*, 89(4) :319–391.

Méthodes de distribution pour les simulations de mobilité des voyageurs

Matthieu Mastio[a] matthieu.mastio@ifsttar.fr
Mahdi Zargayouna[a] hamza-mahdi.zargayouna@ifsttar.fr
Omer Rana[b] ranaof@cardiff.ac.uk
Gerard Scemama[a] gerard.scemama@ifsttar.fr

[a] Université Paris Est, IFSTTAR, GRETTIA, 10-14 Boulevard Newton, 77447 Champs sur Marne, France

[b] School of Computer Science & Informatics, Cardiff University, United Kingdom

Résumé

Avec la généralisation de l'information voyageurs en temps réel, la dynamique des réseaux de transport est de plus en plus difficile à analyser et à prévoir. Il devient nécessaire de développer des outils de simulation pour les décideurs de politiques de mobilité, tenant compte de ce nouvel environnement informationnel. En effet, informer un très grand nombre de voyageurs guidés individuellement peut avoir des conséquences importantes sur l'état du trafic. Il est utile d'évaluer cet impact par la simulation. Or, les simulations multi-agents existantes pour la mobilité de voyageurs ne peuvent prendre en considération qu'une partie du volume réel de voyageurs. En distribuant ces simulations, il serait possible de prendre en compte des volumes réels de voyageurs connectés et d'analyser et de prévoir l'état des réseaux de transport. Dans cet article, nous proposons une comparaison entre deux méthodes pour la distribution des simulations multi-agents de mobilité des voyageurs, permettant la prise en compte de flux réalistes et de zones géographiques étendues.

Mots-clés : *Simulation, Calcul haute performance, Systèmes distribués, Systèmes Multi-agents, Transport*

Abstract

With the generalization of real-time traveler information, the behavior of modern transport networks becomes harder to analyze and to predict. It is now critical to develop simulation tools for mobility policies decision makers, taking into account this new information environment. Indeed, the spread of individualized information may highly influence the traffic network. However, existing mobility multiagent and micro-simulations can only consider a sample of the real volumes of travelers, especially for large areas. With distributed simulations, it would become possible to analyze and predict the status of current and future networks, with informed and connected travelers. In this paper, we propose a comparison between two methods for distributing multiagent travelers mobility simulations, allowing for the consideration of realistic travelers flows and wide geographic areas.

Keywords: *Simulation, High performance computing, Distributed systems, Multiagent, Transport*

1 Introduction

Les systèmes de transport se complexifient de jour en jour et ils doivent évoluer pour intégrer de nouvelles entités connectées (appareils mobiles, véhicules connectés, etc.). Nous pouvons maintenant non seulement fournir des itinéraires optimaux pour les voyageurs, mais nous sommes également en mesure de mettre à jour ces itinéraires en temps réel en fonction de l'état du réseau (congestions, accidents, véhicule de transport en commun ou covoiturage annulé, etc). Donner aux utilisateurs du réseau des informations sur l'état du trafic est généralement bénéfique et permet l'amélioration globale de l'écoulement des flux de voyageurs dans les réseaux de transport.

Toutefois, la diffusion massive de l'information via des panneaux d'affichage, des annonces à la radio ou par le biais d'un guidage individuel peut avoir des effets pervers et créer de nouveaux embouteillages. En effet, avec la généralisation de l'information voyageurs en temps réel, le comportement des réseaux de transport modernes devient plus difficile à analyser et à prévoir. Il devient alors important de modéliser finement et de simuler un nombre réaliste de voyageurs en interaction avec les sources d'information afin d'observer précisément ces effets dans un environnement virtuel contrôlé. La possibilité d'exécuter un simulateur de trafic à l'échelle d'une ville, d'une région ou d'un pays permettrait d'observer les conséquences de différentes stratégies d'information sur l'état du

trafic multimodal avant de les appliquer dans le monde réel. Sur la base des résultats de simulation, il est possible de comparer les prévisions réalisées par le simulateur avec des données mesurées sur le réseau réel afin d'affiner itérativement le modèle.

FIGURE 1 – Le simulateur SM4T [25].

Nous avons récemment développé SM4T (*Multiagent MultiModal Mobility of Travelers*), une plateforme de simulation de mobilité multimodale [26] (cf. Figure 1). Cette plateforme simule le comportement de voyageurs qui interagissent dans un environnement complexe, dynamique et ouvert dans lequel ils n'ont qu'une perception partielle. Elle est fondée sur le paradigme multi-agent, particulièrement adapté pour ce type de simulation [2]. Chaque agent essaye de trouver l'itinéraire le plus efficace pour atteindre sa destination dans un réseau évoluant dynamiquement. Un agent peut potentiellement être informé de l'état du réseau et tirer parti de cette information pour modifier l'itinéraire qu'il pensait initialement prendre. Ainsi, il est intéressant de pouvoir exécuter une simulation sur l'ensemble du réseau, car des décisions prises à l'échelle microscopique par des agents à un endroit et un moment donné peuvent avoir une influence globale sur l'état du réseau à un tout autre endroit plus tard dans la simulation.

Cependant, la simulation du nombre réel de voyageurs d'une grande ville (plusieurs millions de voyageurs) nécessite d'une part une puissance de calcul considérable et d'autre part une architecture permettant la distribution des traitements sur un grand nombre d'hôtes. La version actuelle de SM4T, comme la majorité des simulateurs de mobilité de voyageurs actuels, ne permet pas une telle distribution. Cela induit une limite quant au nombre de voyageurs, de moyens de transports et à la taille des réseaux considérés. Par conséquent, la prévision des effets des réglementations et des stratégies d'information sur de grands réseaux en présence de voyageurs connectés et informés en temps réel devient très difficile. Notre principal objectif est donc de tester le passage à l'échelle de ce type de simulateur. A cet effet, nous cherchons à partager la simulation de manière à répartir équitablement la charge de travail entre plusieurs serveurs. Nous souhaitons pour cela proposer des schémas génériques de distribution qui pourront être appliqués pour passer à l'échelle les simulateurs existants. Nous avons ainsi développé un simulateur de trafic donc le fonctionnement est très simple pour se concentrer exclusivement sur la problématique de la distribution.

La suite de cet article est structurée comme suit. Nous présentons tout d'abord dans la section 2 les précédentes propositions concernant la simulation de mobilité de voyageurs ainsi que les plates-formes multi-agents distribuées existantes. La section 3 présente une définition formelle de notre environnement multi-agent. Dans la section 4, nous décrivons deux méthodes permettant de distribuer le simulateur sur plusieurs hôtes. Enfin, la section 5 fournit une comparaison de ces méthodes ainsi qu'une description de notre configuration expérimentale. Nous concluons et présentons les problématiques sur lesquelles nous travaillons actuellement en section 6.

2 État de l'art

Il existe plusieurs simulateurs multi-agents de mobilité des voyageurs. Par exemple, MATSim [18] est une plate-forme bien connue de micro-simulation de mobilité. Toutefois, les entités mobiles dans MATSim sont passives et leur état est modifié par des modules centraux, ce qui limite sa flexibilité et sa capacité à intégrer de nouveaux types d'agents (proactifs). Transims [19] simule des voyages multimodaux et évalue les impacts des changements de politique dans les caractéristiques du trafic tandis que Miro [8] reproduit les dynamiques urbaines d'une ville française et propose un prototype de simulation multi-agent capable de tester la planification de scénarios en précisant les comportements des individus. AgentPolis [15] est également une plate-forme multi-agent pour

le transport multimodal. SUMO [6] et VISSIM [12] sont des simulateurs microscopiques largement utilisés et principalement axés sur la simulation de trafic. Cependant, aucune de ces propositions ne considère le problème de la distribution. A notre connaissance, aucun patron ni méthode de distribution qui soit spécifique aux simulateurs de trafic multi-agent, prenant en compte ses caractéristiques propres, telles que la présence d'un environnement physique, n'a été proposée.

Certaines plates-formes multi-agents polyvalentes ont été spécifiquement développées ces dernières années pour la simulation à grandes échelles. RepastHPC [10], une version distribuée de Repast Simphony [21], utilise les mêmes concepts de projections et de contextes que Repast et les adapte pour les environnements distribués. Pandora [1] est proche de RepastHPC et génère automatiquement le code nécessaire pour les communications interserveurs. GridABM [14] est basé sur Repast Simphony mais considère une autre approche et propose au programmeur des *templates* généraux qui peuvent être adaptés à la topologie des communications de son application. FLAMME [9] permet au programmeur de générer des simulations HPC (High Performance Computing) à partir d'automates à états finis.

Cependant, ces plates-formes distribuées n'offrent pas de contrôle précis sur la façon dont sont effectuées les communications entre les hôtes. En effet, la couche de communication est transparente pour le programmeur, ce qui rend plus facile pour lui la tâche de développer des simulations distribuées, mais ne lui permet pas d'optimiser les dites communications. La meilleure manière de gérer les communications dépend de l'application et l'utilisation de plates-formes générales pour un simulateur de mobilité ne produit pas de résultats optimaux, puisqu'il existe très souvent des connaissances et des contraintes propres à l'application qui peuvent en optimiser la distribution.

Des travaux plus théoriques étudient des méthodes générales pour résoudre ce problème. Dans [17] et [23], les auteurs proposent d'assouplir certaines contraintes de synchronisation pour obtenir un meilleur passage à l'échelle, en réduisant le temps passé par les hôtes à s'attendre les uns les autres. Cet assouplissement implique nécessairement une perte de précision qui n'est pas viable dans le cas d'un simulateur de trafic. Dans [20] et [22], les auteurs discutent des questions liées à la simulation multi-agent dans un environnement virtuel distribué. Ces deux travaux décrivent des méthodes permettant de partitionner l'environnement virtuel en plusieurs régions afin de paralléliser l'exécution de la simulation. Dans cet article, nous adaptons les approches proposées par [22] pour notre problème qui présente une structure de graphe. Ces méthodes générales, dont nous testerons l'efficacité, pourront être appliquées pour distribuer les simulations de mobilité de voyageurs de la litérature.

3 L'environnement multi-agent

3.1 Le modèle

L'environnement multi-agent d'une simulation de mobilité est constitué du réseau de transport dans lequel les agents voyageurs évoluent. Nous modélisons ce réseau de transport avec un graphe orienté $G(V, E)$ où $E = \{e_1, ..., e_n\}$ est un ensemble d'arcs représentant les routes et $V = \{v_1, ..., v_n\}$ est un ensemble de sommets représentant les intersections. Un ensemble d'agents A se déplace dans ce réseau depuis des origines vers des destinations en essayant de minimiser leur temps de parcours. Le temps de parcours d'un arc au temps t dépend du nombre d'agents présents sur cet arc à ce moment t.

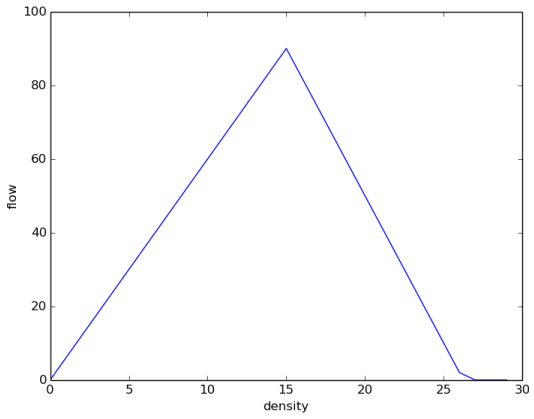


FIGURE 2 – Diagramme fondamental avec $\alpha = 6$, $\beta = 8$ et $k_c = 15$.

Pour le calculer, nous utilisons le diagramme fondamental triangulaire du trafic qui donne une relation entre le flux q (véhicules/heure) et la densité k (véhicules/km). Le diagramme fondamental suggère que si l'on dépasse une densité critique de véhicules k_c, plus il y a de véhicules empruntant une route, plus ces véhicules seront

ralentis. Voici l'équation que nous utilisons pour modéliser ce phénomène :

$$q = \begin{cases} \alpha k & \text{if} \quad k \leq k_c \\ -\beta(k - k_c) + \alpha k_c & \text{if} \quad k > k_c \end{cases} \qquad (1)$$

Cette équation est paramétrée par α la vitesse en régime fluide, β la vitesse de propagation de la congestion et k_c la densité critique. Comme $v = \frac{q}{k}$ on a :

$$v = \begin{cases} \alpha & \text{if} \quad k \leq k_c \\ \frac{-\beta(k-k_c)+\alpha k_c}{k} & \text{if} \quad k > k_c \end{cases} \qquad (2)$$

Ainsi nous pouvons définir une fonction de coût qui retourne un temps de parcours par unité de distance $(1/v)$ en fonction du nombre d'agents présents sur un arc.

$$cost(|A_e|) = \begin{cases} \frac{1}{\alpha} & \text{if} \quad |A_e| \leq k_c \\ \frac{|A_e|}{-\beta(|A_e|-k_c)+\alpha k_c} & \text{if} \quad |A_e| > k_c \end{cases} \qquad (3)$$

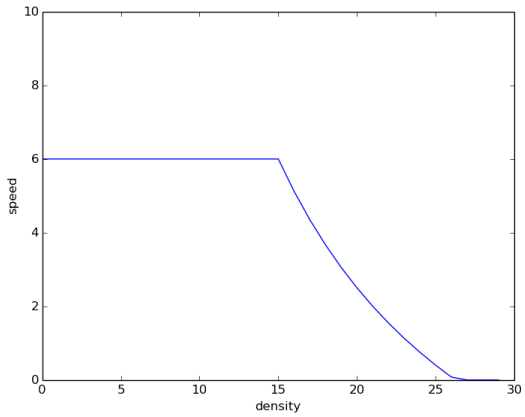


FIGURE 3 – Vitesse en fonction de la densité.

Les arcs et les sommets sont tous deux valués par des poids positifs évoluant dynamiquement avec le nombre d'agents présents. Avec A_v l'ensemble d'agents sur un sommet v et A_e l'ensemble d'agents présent sur un arc e, nous avons $|v| = |A_v|$ qui représente le poids d'un somment et $|e| = |A_e|$ le poids d'un arc. $|V| = \sum_{v \in V} |v|$ est le poids d'un sous-ensemble de sommets. De la même manière, si $|e|$ désigne le poids d'un arc, $|E| = \sum_{e \in E} |e|$ est le poids d'un sous ensemble d'arcs.

3.2 Le simulateur

Nous avons développé un simulateur de référence où chaque agent représente un voyageur évoluant dans un environnement multi-agent comme décrit dans le paragraphe précédent. Les agents sont placés sur le réseau de transport au début de la simulation avec un sommet d'origine et un sommet de destination choisis de manière aléatoire. Ils calculent leur plus court chemin sur la base de l'état actuel du réseau avant de commencer à voyager. Ils recalculent leurs itinéraires à chaque fois qu'ils atteignent un sommet, afin de vérifier s'il existe un nouveau chemin plus court, suivant la dynamique du réseau. A chaque pas de temps de la simulation, si les agents sont actuellement sur un arc, ils avancent sur cet arc autant que possible. La distance parcourue dépend du nombre total d'agent présents sur l'arc et est calculée grâce au diagramme fondamental comme décrit dans la section précédente. Lorsqu'un agent a atteint sa destination, il est supprimé du système. La simulation se termine lorsque tous les voyageurs ont atteint leurs destinations ou quand un certain temps (paramètre) est écoulé.

Ce simulateur garde les propriétés générales de tout simulateur de trafic multi-agent : des agents évoluant dans un graphe et cherchant à minimiser leur temps de parcours. Cependant, il n'est pas aussi complexe que d'autres simulateurs de la littérature dont la vocation est d'obtenir une simulation réaliste en termes de trafic généré. En effet, notre objectif est de ne représenter que les caractéristiques partagées par les simulateurs de trafic qui influent sur la complexité et l'efficacité du processus de distribution.

4 Approches proposées

Pour lancer notre simulation à l'échelle d'une ville, nous avons besoin d'une capacité de mémoire et d'une puissance de calcul importante. C'est pourquoi nous cherchons à la déployer sur plusieurs hôtes. Une telle simulation s'exécutant sur un cluster est typiquement SPMD (*Simple Program Multiple Data*), c'est à dire que tous les processeurs, reliés par un réseau, exécutent le même programme, mais ne possèdent qu'une partie des données du programme dans leurs mémoires privées. Les communications sont explicitement déclarées par le programmeur. L'avantage de cette approche est sa grande évolutivité ; il peut être mis en oeuvre sur la plupart des architectures parallèles et nous

pouvons déployer le même simulation sur des systèmes plus puissants si nécessaire.

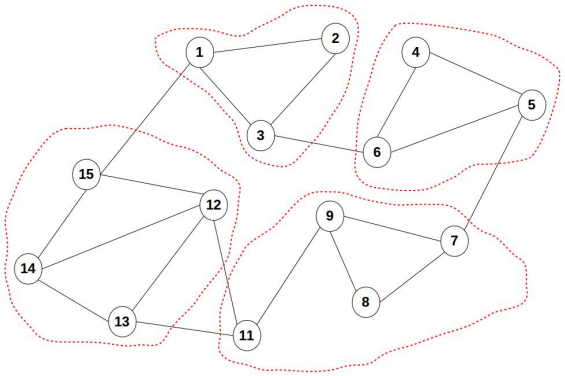
Dans les simulations de mobilité, les agents voyageurs se déplacent sur un réseau de transport. Pour distribuer ces simulations, nous devons diviser efficacement la charge de travail entre les hôtes disponibles. Dans notre modèle, la charge de travail principale est générée par le calcul du plus court chemin. Le chemin d'un agent doit être recalculé chaque fois que cet agent atteint une nouvelle intersection (parce que les temps de déplacement évoluent dynamiquement). Ainsi, un agent peut être considéré comme une unité de travail. Par conséquent, afin de distribuer ce modèle, nous devons partager le plus uniformément possible les agents entre les hôtes. Pour ce faire, nous pouvons distribuer soit l'environnement, soit les agents.

4.1 Distribution des agents

Une première approche à envisager est de diviser l'ensemble des agents en k parts égales (avec k le nombre de serveurs disponibles), de distribuer chaque sous-ensemble sur un serveur et d'exécuter la simulation. Comme les temps de parcours dépendent du nombre d'agents sur chaque arc, tous les agents ont besoin de savoir à chaque étape le nombre d'agents présents sur chacun des arcs pour calculer leur plus court chemin. Dans notre implémentation, à chaque pas de temps, si un agent géré par un serveur quitte un arc ou arrive sur un arc, ce serveur communique l'information à tous les autres serveurs. Ainsi, à n'importe quel pas de temps, tous les serveurs doivent connaître l'état de l'ensemble du réseau. Il s'agit de la seule communication nécessaires à la simulation pour cette approche. En effet, les agents ne sont pas mobiles : ils évoluent pendant tout leur cycle de vie au sein d'un même hôte. En revanche, leur avancement sur le réseau dépend des mouvements des autres agents, gérés par d'autres hôtes. D'où la communication continue entre serveurs pour informer de la dynamique locale de chacun d'eux. Ainsi le coût de communication total est de $k.|E|.I$, avec I représentant la taille d'un entier [1].

4.2 Distribution de l'environnement

Le second patron de distribution s'efforce de garder sur le même serveur les agents qui sont géographiquement proches sur le réseau. Au lieu de distribuer les agents, nous distribuons donc les sommets et les arcs sortants (et donc les agents situés sur ces sommets et ces arcs) de sorte que les agents qui sont situés au même endroit soient sur le même serveur.



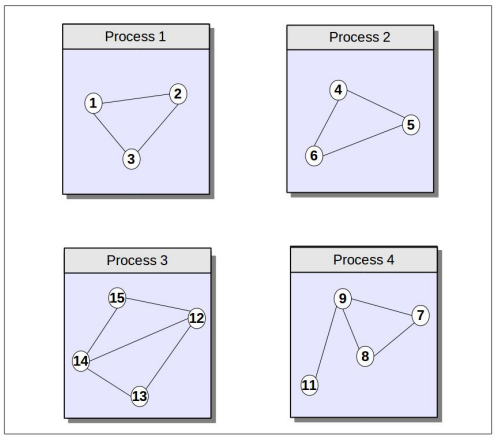


FIGURE 4 – Le graphe est partitionné et chacune des parties est distribuée entre les processus disponibles.

Chaque serveur connaît uniquement ce qui se passe sur la partie du graphe qu'il gère. Ainsi, à chaque pas de temps, avant que les agents n'agissent, tous les serveurs doivent se synchroniser avec les autres serveurs. Il y a maintenant deux types de communications : les serveurs doivent communiquer le poids de leurs arcs (le nombre d'agents présents sur ceux ci) et, quand un agent se déplace sur un sommet qui n'est pas sur son serveur actuel, il doit être transféré vers le serveur gérant ce sommet. Soit C le coût de communication des arcs et M le coût de migration des agents. A chaque étape, le coût global des communications est donnée par $T = C + M$. Le coût des communications du poids des arcs est donné par : $C = |E| \times I$. Un agent peut être

1. L'entier en question renseigne le nombre d'agents quittant ou arrivant sur un arc

codé avec trois entiers (ID, son emplacement actuel et sa destination). Soit n le nombre de migrations pour un pas de temps. Ainsi, le coût de la migration des agents est : $M = 3.I.n$. Il y a en moyenne $|A|/|E|$ agents par sommet. Donc, avec E_c l'ensemble des arcs, nous avons en moyenne $n = |A|/|E| \times |E_c|$. Ainsi T = $I(|E|+3|A|/|E| \times |E_c|)$. Ainsi, moins il y a d'arcs entre deux serveurs et plus le coût de communication est bas.

Pour que la méthode de distribution de l'environnement soit efficace, il faut répartir les sommets dans k ensembles disjoints de sorte que chaque ensemble ait approximativement le même nombre de sommets et que le poids de coupe - le poids total des arcs coupés par la partition - soit minimal. Ce problème est connu sous le nom $(k, 1+\epsilon)$-*balanced partitioning problem*, dont l'objectif est de trouver une collection de sous-ensembles disjoints $V_1, \ldots, V_k$ recouvrant V, c'est à dire $V = V_1 \cup ... \cup V_k$ tel que chaque partie contient au plus $(1 + \epsilon)\frac{|A|}{k}$ et $|E_c|$ soit minimisée.

Algorithme 1 Algorithme *Differential Greedy* modifié

Entrées: Graphe $G = (V, E)$, taille k de la partition
Sorties: Partition P
$P \leftarrow P_0, ..., P_{k-1}$
$V' \leftarrow V$
Pour $p \in [O, k-1]$ **faire**
  $v \leftarrow$ un sommet aléatoire de V'
  $P_p \leftarrow v$
  $V' \leftarrow V' \setminus v$
fin pour
Tant que $|V'| > 0$ **faire**
  $p \leftarrow$ indice de la partition la plus légère
  $m = min_{v \in V'}(1 + \epsilon)$(nombre de v voisins $\in P_p$) − (nombre de v voisins $\notin P_p$)
  mv = un sommet aléatoire de $v \in V'|(1 + \epsilon)$(nombre de v voisins $\in P_p$) − (nombre de v voisins $\notin P_p$) $= m$
  $P_p \leftarrow P_p \cup mv$
  $V' \leftarrow V' \setminus mv$
Fin tant que
Retourner P

Le problème du partitionnement de graphe a été largement étudié dans la littérature. Comme démontré dans [7], il s'agit d'un problème NP-complet. C'est à dire que la tentative de trouver une solution optimale avec, par exemple, un programme en nombre entier n'est pas une option pour des graphes de grande taille. C'est pourquoi certaines heuristiques ont été proposées pour résoudre ce problème dans un temps raisonnable. Outre la méthode spectrale [11] qui a été beaucoup utilisée dans le passé, des approches constructives comme les algorithmes Min-Max [5] et DG [13] se sont révélées efficaces pour diviser de grands graphes. Plus récemment, la méthode de partitionnement multi-niveau, originellement créée pour améliorer les techniques existantes [4] a été reconnue comme étant une méthode très puissante qui offre une vision plus globale des graphes que les techniques traditionnelles. Comme la complexité du problème de partitionnement dépend de la taille du graphe, l'idée simple du partitionnement multi-niveau est de regrouper les sommets et de travailler avec des groupes de sommets plutôt qu'avec des sommets indépendants. La partition multi-niveau a été formalisée dans un cadre générique par Walshaw dans [24]. Nous distribuons notre réseau grâce à un partitionnement multi-niveau utilisant une version légèrement modifiée de l'algorithme DG (Algorithm 1). Nous avons modifié cet algorithme pour pouvoir l'utiliser avec des sommets pondérés et produire des partitions plus connectées.

5 Expérimentations

5.1 Implémentation

Pour tester l'efficacité de ces approches, nous avons implémenté notre modèle et l'avons déployé sur un cluster réel. Nous avons choisi Python comme langage de développement pour son efficacité dans le prototypage rapide. Python est un langage portable, mature, disposant de nombreuses bibliothèques scientifiques éprouvées. Il est avec C et Fortran l'un des langages les plus utilisés pour le calcul haute performance [16]. Comme le but de nos travaux est de fournir une étude de l'efficacité relative de différentes méthodes de distribution, et non une performance absolue, le choix de Python nous semble pertinent.

Pour les communications inter-processus, nous utilisons MPI (Message Passing Interface), qui est le standard *de facto* pour le calcul parallèle avec une importante communauté d'utilisateurs. MPI offre un modèle de communication simple entre les différents processus d'un programme et a de nombreuses implémentations efficaces qui s'exécutent sur une grande variété de machines [2]

2. De plus, nous bénéficions de MPI4PY, qui est une interface efficace qui permet d'utiliser MPI avec Python.

5.2 Résultats

Nous avons exécuté les simulations distribuées sur le cluster de l'Université de Cardiff. Pour nos tests, nous avons utilisé huit hôtes sous CentOS Linux (noyau 2.6.32-220) sur un processeur Intel Xeon E5-2620 CPU (12 cœurs à 2 GHz) avec 32 Go de mémoire. Nous avons exécuté la simulation sur trois configurations : la première est une version séquentielle du programme s'exécutant sur un seul hôte (conf1), la seconde est une version distribuée sur les huit hôtes (conf2), et la dernière est exécutée sur les huit hôtes utilisant les 12 cœurs de chacun d'entre eux (conf3). La simulation est effectuée pour 100 pas de temps sur un réseau invariant d'échelle de 200 noeuds (figure 5) généré avec le modèle Barabasi-Albert [3]. Nous avons comparé les deux méthodes de distribution (distributions des agents et distribution de l'environnement) sur les différentes configurations avec un nombre croissant d'agents (de 1000 à 40000).

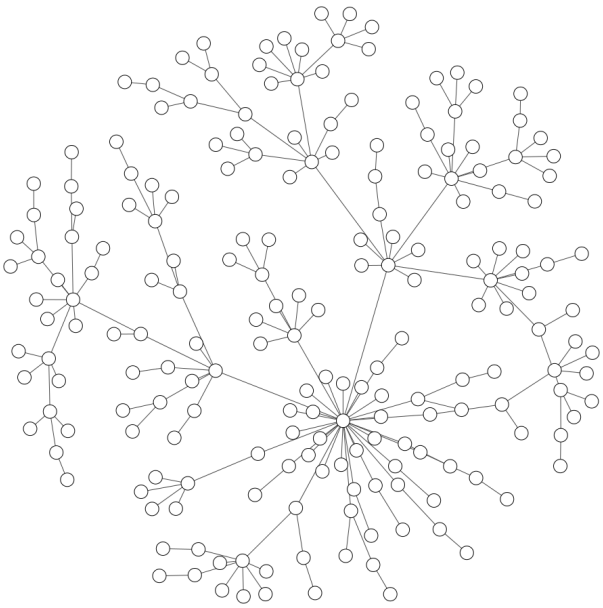


FIGURE 5 – Un graphe de 200 sommets généré avec le modèle de Barabasi-Albert.

Les résultats (table 1) montrent que la distribution des agents est plus efficace que la distribution de l'environnement avec le modèle proposé [3] (figure 6). En effet, par souci

3. Les temps de calcul ne sont pas strictement croissant avec le nombre d'agents pour la distribution de l'environnement. Cela est probablement dû au caractère aléatoire des origines/destinations des agents. La simulation peut en effet parfois être plus complexe alors qu'elle comporte moins d'agents.

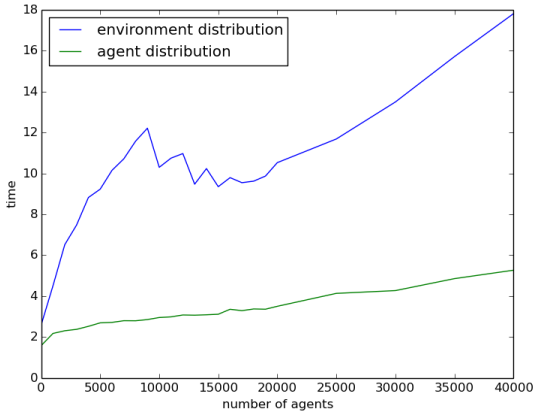


FIGURE 6 – Comparaison des temps d'exécution entre les différentes méthodes de distribution sur conf3.

de généricité, nous n'avons pas défini d'interactions locales entre les agents dans notre simulateur de référence actuel. Par conséquent, nous sommes dans un cas idéal pour une distribution fondée sur une répartition des agents, puisque le nombre de communications inter-serveurs est limité. Avec l'implémentation future d'interactions locales (en intégrant un modèle de poursuite ainsi que des communications inter-véhicules), la distribution de l'environnement sera avantagée grâce à la cohabitation sur le même serveur d'agents physiquement proches. En outre, la distribution de l'environnement ne bénéficie pas à l'heure actuelle de mécanisme d'équilibrage de charge dynamique. Si un nombre important d'agents est concentré sur la même partie du réseau, ils seront sur le même serveur. Ce serveur prendra plus de temps pour calculer tous les plus courts chemins, et tous les autres serveurs devront l'attendre pour passer au pas de temps suivant. Ainsi, un serveur surchargé peut ralentir l'ensemble de la simulation.

Les améliorations de temps d'exécution mesurées entre l'exécution séquentielle et les deux méthodes de distribution sur conf3 sont indiquées sur les figures 7 et 8. L'amélioration mesure combien de fois la simulation est plus rapide sur conf3 (96 cœurs) que sur conf1. Comme nous pouvons le constater sur ces figures, pour 40.000 agents, la simulation est 8 fois plus rapide avec la distribution de l'environnement et 28 fois plus rapide avec la distribution des agents. Ces deux méthodes améliorent largement le temps d'exécution de notre simulateur de mobilité multi-agent. Comme ex-

number of agents	1000	5000	10000	20000	30000	40000
conf1 (1 core)	10	27	43	67	104	140
conf2 agent distribution (12 cores)	3	6	8	12	17	23
conf3 agent distribution (96 cores)	2	3	3	4	4	5
conf2 environment distribution (12 cores)	6	13	17	22	31	41
conf3 environment distribution (96 cores)	5	10	11	11	15	17

TABLE 1 – Temps d'exécution (en secondes) pour une simulation de 100 pas de temps sur un réseau invariant d'échelle de 200 noeuds.

pliqué ci-dessus, la méthode de distribution des agents montre de meilleurs résultats en raison de la généricité et de la relative simplicité du modèle actuel. Nous pensons que l'ajout de communications inter-agents fera que la différence de performance entre les deux méthodes devrait largement diminuer, voire s'inverser.

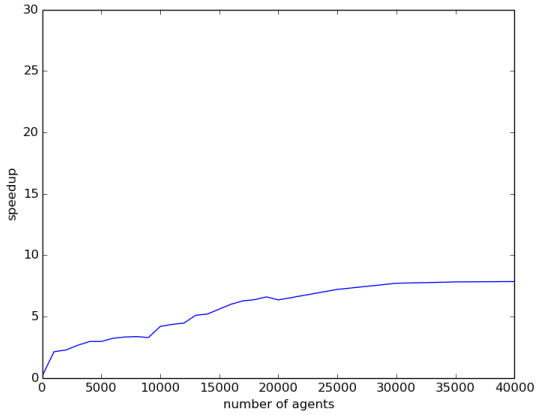


FIGURE 7 – Amélioration de temps d'exécution entre conf1 et conf3 pour la distribution de l'environnement.

6 Conclusions et perspectives

Dans cet article, nous avons présenté deux méthodes de distribution sur plusieurs hôtes d'un simulateur de mobilité multi-agent. Les deux méthodes sont efficaces et améliorent le passage à l'échelle du simulateur. Avec notre modèle de simulation actuel, la méthode de distribution des agents est plus efficace que la méthode de distribution de l'environnement. Notre modèle de simulation est très générique, nos propositions et conclusions sont applicables à tous les simulateurs de mobilité actuels.

Nos travaux futurs porteront sur deux aspects. Nous aborderons tout d'abord des modèles de simulation de mobilité plus spécifiques, où

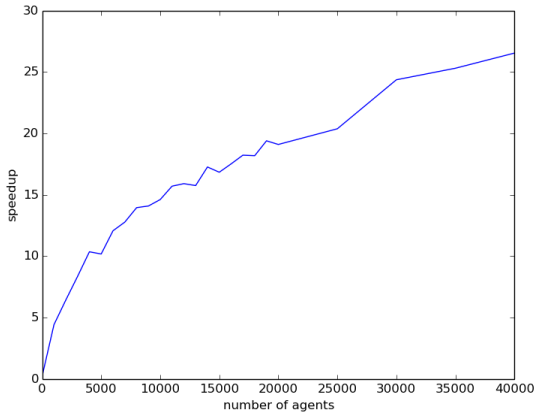


FIGURE 8 – Amélioration de temps d'exécution entre conf1 et conf3 pour la distribution des agents.

tous les types de communications sont présents (locaux, globaux, communautaires, etc.). Nos agents qui se contentaient ici du calcul de plus court chemin auront alors a gérer des situations nécessitant plus d'interactions locales (par exemple pour la négociation de l'ordre de passage dans les intersections). Nous prévoyons que la méthode de distribution de l'environnement affiche de meilleures performances comparatives qu'avec le modèle actuel. De plus, la distribution de l'environnement se fait actuellement de manière statique au début de la simulation et nous pensons que les temps de calcul pourrait être raccourcis en ajoutant des mécanismes d'équilibrage de charge dynamique. Ce point d'amélioration sera notre deuxième axe de travail.

Références

[1] E.S. Angelotti, E.E. Scalabrin, and B.C. Avila. PANDORA : a multi-agent system using paraconsistent logic. In *Fourth International Conference on Computational Intelligence and Multimedia Applications,*

2001. ICCIMA 2001. Proceedings, pages 352–356, 2001.

[2] Fabien Badeig, Flavien Balbo, Gérard Scemama, and Mahdi Zargayouna. Agent-based coordination model for designing transportation applications. In *Intelligent Transportation Systems, 2008. ITSC 2008. 11th International IEEE Conference on*, pages 402–407. IEEE, 2008.

[3] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286(5439) :509–512, October 1999.

[4] Stephen T. Barnard and Horst D. Simon. Fast multilevel implementation of recursive spectral bisection for partitioning unstructured problems. *Concurrency : Pract. Exper.*, 6(2) :101–117, April 1994.

[5] Roberto Battiti and Alan Bertossi. Greedy, prohibition, and reactive heuristics for graph partitioning. *IEEE Transactions on Computers*, 48 :361–385, 1998.

[6] Michael Behrisch, Laura Bieker, Jakob Erdmann, and Daniel Krajzewicz. SUMO - simulation of urban MObility - an overview. In *SIMUL 2011, The Third International Conference on Advances in System Simulation*, pages 55–60, 2011.

[7] Thang Nguyen Bui and Curt Jones. Finding good approximate vertex and edge partitions is NP-hard. *Information Processing Letters*, 42(3) :153–159, May 1992.

[8] Sébastien Chipeaux, Fabrice Bouquet, Christophe Lang, and Nicolas Marilleau. Modelling of complex systems with AML as realized in MIRO project. In *LAFLang 2011, workshop of the Int. Conf. WI/IAT (Web Intelligence and Intelligent Agent Technology)*, pages 159–162, Lyon, France, 2011. IEEE Computer Society.

[9] S. Coakley, M. Gheorghe, M. Holcombe, S. Chin, D. Worth, and C. Greenough. Exploitation of high performance computing in the FLAME agent-based simulation framework. In *2012 IEEE 14th International Conference on High Performance Computing and Communication 2012 IEEE 9th International Conference on Embedded Software and Systems (HPCC-ICESS)*, pages 538–545, 2012.

[10] Nicholson Collier and Michael North. Repast HPC : A platform for large-scale agent-based modeling. In Werner Dubitzky, Krzysztof Kurowski, and Bernhard Schott, editors, *Large-Scale Computing*, pages 81–109. John Wiley & Sons, Inc., 2011.

[11] W. E. Donath and A. J. Hoffman. Lower bounds for the partitioning of graphs. *IBM J. Res. Dev.*, 17(5) :420–425, September 1973.

[12] Martin Fellendorf and Peter Vortisch. Microscopic traffic flow simulator vissim. In *Fundamentals of Traffic Simulation*, pages 63–93. Springer, 2010.

[13] C.M. Fiduccia and R.M. Mattheyses. A linear-time heuristic for improving network partitions. In *19th Conference on Design Automation, 1982*, pages 175–181, June 1982.

[14] Laszlo Gulyas, Gabor Szemes, George Kampis, and Walter de Back. A modeler-friendly API for ABM partitioning. In *ASME 2009 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, pages 219–226, 2009.

[15] Michal Jakob, Zbynek Moler, Antonín Komenda, Zhengyu Yin, Albert Xin Jiang, Matthew Paul Johnson, Michal Pechoucek, and Milind Tambe. Agentpolis : towards a platform for fully agent-based modeling of multi-modal transportation (demonstration). In *International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2012, Valencia, Spain, June 4-8, 2012 (3 Volumes)*, pages 1501–1502, 2012.

[16] Hans Petter Langtangen and Xing Cai. On the efficiency of python for high-performance computing. In *Modeling, Simulation and Optimization of Complex Processes*, pages 337–357. Springer Berlin Heidelberg, January 2008.

[17] D. Mengistu and M. v Lowis. An algorithm for optimistic distributed simulations. In *Modelling, Simulation, and Identification / 658 : Power and Energy Systems / 660, 661, 662*. ACTA Press, 2011.

[18] Maciejewski Michal and Nagel Kai. Towards multi-agent simulation of the dynamic vehicle routing problem in matsim. In *Proceedings of the 9th International Conference on Parallel Processing and Applied Mathematics - Volume Part II*, PPAM'11, pages 551–560, Berlin, Heidelberg, 2012. Springer-Verlag.

[19] Kai Nagel and Marcus Rickert. Parallel implementation of the transims

micro-simulation. *Parallel Computing*, 27(12) :1611–1639, 2001.

[20] Beatrice Ng, Antonio Si, Rynson W.H. Lau, and Frederick W.B. Li. A multi-server architecture for distributed virtual walkthrough. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*, VRST '02, pages 163–170. ACM, 2002.

[21] Michael J. North, Nicholson T. Collier, Jonathan Ozik, Eric R. Tatara, Charles M. Macal, Mark Bragen, and Pam Sydelko. Complex adaptive systems modeling with repast simphony. *Complex Adaptive Systems Modeling*, 1(1) :3, 2013-03-13.

[22] Omar Rihawi, Yann Secq, and Philippe Mathieu. Effective distribution of large scale situated agent-based simulations. In *ICAART 2014 6th International Conference on Agents and Artificial Intelligence*, volume 1, pages 312–319. SCITEPRESS Digital Library, 2014.

[23] Matthias Scheutz and Paul Schermerhorn. Adaptive algorithms for the dynamic distribution and parallel execution of agent-based models. *Journal of Parallel and Distributed Computing*, 66(8) :1037–1051, 2006.

[24] Chris Walshaw. Multilevel refinement for combinatorial optimisation : Boosting metaheuristic performance. In *Hybrid Metaheuristics*, number 114 in Studies in Computational Intelligence, pages 261–289. Springer Berlin Heidelberg, January 2008.

[25] Mahdi Zargayouna, Besma Zeddini, Gérard Scemama, and Amine Othman. Simulating the impact of future internet on multimodal mobility. In *The 11th ACS/IEEE International Conference on Computer Systems and Applications AICCSA'2014*. IEEE Computer Society, 2014.

[26] Mahdi Zargayouna, Besma Zeddini, Gérard Scemama, and Amine Othman. Agent-based simulator for travelers multimodal mobility. *Frontiers in Artificial Intelligence and Applications*, 252 :pp 81–90, January 2013.

Institution artificielle située pour une aide à la régulation dans le cadre de la gestion de crises

Maiquel De Brito[a] maiquel.b@posgrad.ufsc.br
Lauren Thevin[b] lauren.thevin@imag.fr
Catherine Garbay[b] Catherine.Garbay@imag.fr
Olivier Boissier[c] olivier.boissier@emse.fr
Jomi F. Hübner[a] jomi.hubner@ufsc.br

[a]PPGEAS/Federal University of Santa Catarina, Florianópolis, Brazil

[b]LIG/Université de Grenoble, Grenoble, France

[c]Laboratoire Hubert Curien UMR CNRS 5516, Institut Henri Fayol, Mines Saint-Etienne, Saint-Etienne, France

Résumé

Ce document propose l'utilisation d'une institution artificielle située (SAI) au sein d'un système multi-agent normatif, interactif, et hybride pour réguler la collaboration humaine dans une situation de gestion de crises. Les normes permettent de réguler les actions des acteurs humains en fonction de la dynamique de l'environnement dans lequel ils sont situés. Cette dynamique provient à la fois de l'évolution de l'environnement et de l'activité des acteurs. Notre objectif est de coupler les normes et les caractéristiques de l'environnement pour proposer une régulation de crises ancrée dans le contexte. En ajoutant un niveau constitutif entre l'environnement et les normes, l'utilisation de SAI permet la mise en place d'un couplage lâche entre ces deux éléments. Ainsi, les normes ne se réfèrent plus à des faits environnementaux, mais aux fonctions statutaires, c'est-à-dire à l'interprétation institutionnelle des faits environnementaux selon des règles constitutives. Nous montrons comment cette modélisation déclarative et distincte permet de gérer l'interprétation des événements, tout en tenant compte du contexte organisationnel.

Mots-clés : *Institutions artificielles situées, système normatif, Interaction tangible, gestion de crises*

Abstract

This paper highlights the use of Situated Artificial Institution (SAI) within an hybrid, interactive, normative multi-agent system to regulate human collaboration in crisis management. Norms regulate the actions of human actors based on the dynamics of the environment in which they are situated. This dynamics result both from environment evolution and actors' actions. Our objective is to couple norms to environment state to provide a context aware crisis regulation. Introducing a constitutive level between the environment state and the normative one, SAI provides a loosely coupling of norms with the environment. Norms are thus no more referring to environmental facts but to status functions, i.e. institutional interpretation of environmental facts through constitutive rules. We present how this declarative and distinct SAI modelling succeeds in managing the interpretation of the events while taking into account organizational context.

Keywords: *Situated Artificial Institutions, Normative System, Tangible Interaction, Crisis Management*

1 Introduction

La gestion de crises a pour objectif d'organiser les secours lors d'accidents naturels ou artificiels afin de limiter les dégâts humains et matériels. Elle met en oeuvre une collaboration complexe et décentralisée entre de multiples acteurs et organisations (e.g. pompiers, policiers, citoyens). Tous se coordonnent dans un environnement fortement dynamique et incertain afin de prendre des actions efficaces et cohérentes en lien avec les multiples missions qui leur incombent (information, sécurité, logistique, logement).

Dans un tel contexte, des plateformes logicielles support à la gestion de crises sont de plus en plus utilisées. Habités d'une même volonté, nous sommes actuellement impliqués dans le développement d'une plateforme s'appuyant sur des surfaces tangibles afin de médier les interactions opportunistes entre les différents acteurs distants impliqués. Afin de prendre en compte et mettre en oeuvre les politiques et normes de ges-

tion de crises utilisées pour coordonner les actions collectives des acteurs, nous avons proposé l'utilisation de systèmes multi-agents normatifs pour définir un système socio-technique [20] dans lequel des humains et des agents logiciels coopèrent (i.e. *système hybride*) en combinant des interactions physiques, digitales et virtuelles (i.e. *interaction mixte*) qui sont régulées par des normes et des organisations (i.e *système normatif*). Ces trois dimensions sont bien adaptées pour répondre aux verrous soulevés par les systèmes de gestion de crises tels que nous les avons décrits ci-dessus.

Ce papier aborde plus particulièrement une caractéristique supplémentaire et importante à considérer pour développer de tels systèmes. Elle concerne le couplage entre la régulation par les normes et l'environnement physique dans lequel la crise se déroule. Ancrer les normes et la régulation dans l'environnement doit être réalisé d'une manière flexible et facilement modifiable afin de faire face à des situations de crises changeantes et complexes. En fait, deux problèmes peuvent se produire : (*i*) divergences dans l'interprétation des événements issus de l'environnement, dépendant du contexte, du rôle de l'acteur ou de l'organisation à laquelle il appartient, (*ii*) incohérences dans les interventions humaines du fait de normes d'organisations différentes et donc sources possibles de conflits. C'est la raison pour laquelle, comme proposé dans [7], nous nous tournons vers les *Institutions Artificielles Situées* (SAI) qui offrent, comme nous l'expliquons et l'illustrons dans ce qui suit, des abstractions et primitives adaptées pour résoudre ce problème. Dans ce papier, nous ne présentons pas une version complète d'une application réalisée via une SAI. Nous nous sommes focalisés sur la production d'un prototype montrant ce que cette approche (ancrage des normes dans l'environnement au travers de règles constitutives) peut apporter au développement d'applications réelles de gestion de crises.

La section 2 présente le contexte applicatif en décrivant rapidement un scénario de gestion de crises et en listant les besoins qui sous-tendent l'application que nous développons. La section 3 définit une Institution Artificielles Située. La section 4 présente comment la notion de SAI a été utilisée et mise en place pour répondre aux besoins de la section 2. La section 5 décrit l'exécution d'un scénario sur l'implémentation réalisée. En section 6, avant de conclure, nous comparons notre proposition vis-à-vis des approches existantes.

2 Besoins et Démarche suivie

Afin de motiver la démarche suivie pour notre proposition, nous présentons une étude de cas extraite de l'application de gestion de crises envisagée. A partir des exigences et besoins ainsi illustrés, nous donnons les dimensions qui soutiennent notre démarche consistant en une approche située pour soutenir l'activité de collaboration médiée par la machine.

2.1 Etude de cas

Le fil conducteur retenu pour la présentation de notre proposition est un exemple simplifié d'évacuation de zone dans un contexte de crue. Les acteurs, dans cette activité, sont organisés en trois groupes : le *Poste de Commandement Communal (CCP)*, placé sous la responsabilité du *Maire*, la *cellule logistique (LC)* placée sous la responsabilité du *CCP*, et les *pompiers (FF)*. Ces acteurs interviennent au cours de deux phases successives – prévention et urgence – qui sont régies par des politiques spécifiques. En phase *préventive*, le *Maire* demande à *LC* d'évacuer les *zones sûres*. *FF* est au contraire responsable de l'organisation de l'évacuation lorsqu'une *situation risquée* survient, et durant la phase d'*urgence*, au cours de laquelle la zone est considérée comme *risquée*. Une zone *sûre* est viable au plan de la sécurité, car des personnels non professionnels (*Maire* et *LC*) peuvent effectuer l'évacuation. Les zones *risquées*, non viables au plan de la sécurité, impliquent l'intervention de professionnels comme *FF* pour réaliser l'évacuation.

Ces acteurs travaillent selon des politiques et des normes variées. Une de ces politiques spécifie notamment qu'il doit y avoir un seul groupe d'acteurs à un instant donné pour gérer l'évacuation. Selon la phase (*prévention* ou *urgence*), et selon le statut de la zone (*sûre*, *risquée*), un seul acteur, *LC* ou *FF*, sera en charge d'intervenir. Cependant, qualifier une zone de *sûre* ou *risquée* peut être source de conflit. Pour les membres du *CCP*, ou le *Maire*, une zone est considérée comme *sûre* dès lors que la phase est préventive et que le nombre d'habitants est inférieur à un certain seuil. Au contraire, pour les membres du groupe *FF*, une zone est considérée comme *sûre* en phase préventive et si aucun risque électrique n'est observé, indépendamment du nombre d'habitants. Supposons que nous soyons en phase préventive, que le nombre d'habitants excède le seuil critique et qu'il n'y ait pas de risque électrique. Le *Maire* va consi-

dérer que c'est à *FF* d'intervenir, alors que *FF* va considérer le contraire : c'est au *Maire* de diriger l'évacuation. Ce court exemple montre que des divergences sont possibles dans l'interprétation d'événements en provenance de l'environnement, selon le contexte, les rôles ou l'organisation des acteurs ; il révèle également des incohérences dans l'intervention humaine issues d'incohérences dans les politiques des organisations impliquées.

Supposons en outre qu'en raison d'une évolution de la crue, la situation est modifiée : de phase *préventive* on passe en phase d'*urgence*. Il va en résulter une évolution des politiques d'intervention. En conséquence, la validité de certains faits comme "demander l'*évacuation* d'une zone" va changer. Ceci illustre un enjeu supplémentaire lié à l'évolution possible des normes à considérer.

2.2 Dimensions de l'étude

Comme nous l'avons vu, la gestion de crises est une activité collaborative où les acteurs humains doivent intervenir de manière efficace mais flexible pour faire face aux évolutions non prévisibles des situations. L'analyse des approches existantes en gestion de crises, collecticiels et systèmes multi-agents (MAS), montre qu'un système d'aide à la gestion de crises doit être conçu en répondant à trois dimensions fondamentales, comme *hybride*, *mixte* et *normatif*.

– **Système Hybride** : la gestion de crises est une activité collaborative complexe impliquant la participation d'acteurs et d'organisations multiples. Ils doivent agir et se coordonner pour réaliser de manière efficace leurs multiples missions (e.g. information, sécurisation, ravitaillement, hébergement), au sein d'environnement hautement dynamiques et incertains. Etant donné le caractère distribué et décentralisé inhérent à la gestion de crises, une approche multi-agent apparaît bien adaptée : les acteurs humains et artificiels sont considérés comme des agents interagissant au sein d'un environnement partagé sous le contrôle de politiques de régulation et de coordination qui sont dépendantes des organisations et du contexte.

– **Système Mixte** : Pour faire face au caractère distribué de la gestion de crises, le système est déployé sur un réseau de tables TangiSense [13] à travers lesquelles les acteurs humains interagissent. Ces tables peuvent détecter et localiser des objets tangibles équipés de tags RFID. Leur surface est en outre équipée d'un écran LCD qui permet l'affichage de simulations ainsi que des retours virtuels liés aux objets tangibles. Le choix de cette technologie est motivé par sa capacité à soutenir une activité flexible et opportuniste. Pour garantir le *partage des contextes organisationnels* [10], c'est-à-dire la possibilité pour les acteurs de percevoir les rôles, missions et normes des autres acteurs, nous exploitons les retours virtuels pour mettre en lumière les incohérences et conflits potentiels des actions tangibles effectuées vis-à-vis des normes de régulation et de coordination.

– **Système Normatif** : Le manque de ressources, les changements de situations, le caractère distribué et inter-organisationnel de l'activité de gestion de crises peut rendre la collaboration difficile [8]. Des politiques cohérentes sont nécessaires pour réguler correctement ces activités. Les normes et les systèmes normatifs, tels que définis dans [2] apportent le niveau d'abstraction et les mécanismes appropriés pour exprimer ces politiques et réguler l'activité décentralisée d'agents oeuvrant au sein d'environnements dynamiques et non prédictibles. Outre la régulation des *activités de coordination* entre humains, les normes dans notre approche sont utilisées pour réguler l'activité des agents du système. Il s'agit de gérer le degré d'autonomie des agents, et de réguler l'allocation des tâches entre acteurs humains et artificiels. Ceci peut être utile pour faire évoluer le rôle du système, d'un rôle plutôt éducatif (la plupart des tâches étant conférées aux acteurs humains), à un rôle de supervision (la plupart des tâches sont effectuées avec le soutien du système). Il est également nécessaire de diriger l'interaction humain-machine, c'est-à-dire de décrire quelles sont les interactions autorisées pour les acteurs humains (*activité de production*) et comment générer les retours virtuels (*activité de communication*). Ces trois types d'activités font référence aux espaces classiquement envisagés dans le domaine des collecticiels [12].

2.3 Démarche

Selon les trois dimensions présentées précédemment, la gestion de crises sera soutenue par un système multi-agent hybride et normatif médiant et régulant des interactions mixtes entre acteurs humains et artificiels. Trois spécifications normatives doivent être considérées pour tenir compte des trois espaces d'activité concernés : espaces de *production*, de *coordination* et de *communication* [12]. Notre objectif est d'ancrer ces spécifications dans l'environnement tout en préservant leur indépendance vis-à-vis du monde physique. Dans ce but, nous introduisons un niveau intermédiaire formalisé

Espace/Niveau	*Faits*	*Règles d'interprétation*	*Faits interprétés*	*Normes*
Production	Entrée tangible	Règles de reconnaissance de pattern	Patron	Validités des Patrons
Coordination	Patron	Règles constitutives	Faits institutionnels	Validité des Faits institutionnels
Communication	Validité des Patrons et des faits institutionnels	Règles de virtualisation	Fait virtuel	Génération des retours virtuels

FIGURE 1 – Ancrage des normes de production, de coordination et de communication dans l'environnement

par des règles d'interprétation et des faits interprétés, associant les mondes physiques (i.e. *Faits*) et organisationnels (i.e. *Normes*). Nous aboutissons ainsi à la décomposition présentée dans le tableau de Fig. 1 pour chaque espace d'activité. La première étape (cf. Fig. 1 – ligne Production) consiste à interpréter les entrées tangibles comme des patrons d'activité qui peuvent être valides ou non au sens de la spécification de l'activité de production (e.g. cette entrée tangible est-elle une interaction valide ?). Ces patrons sont ensuite interprétés (cf. Fig. 1 – ligne Coordination) comme des faits interprétés qui peuvent être valides ou non au sens de la spécification de l'activité de coordination (e.g. cette activité est-elle conforme aux missions et rôles actuels de l'acteur ?). Enfin, à partir de ces deux interprétations, un fait virtuel est généré (cf. Fig. 1 – ligne Communication) et transmis selon les règles de retour virtuel (où et sous quelle forme transmettre un retour virtuel ?).

Pour illustrer la capacité déclarative de notre modèle, focalisons-nous sur l'espace de coordination (cf. Fig. 1 – ligne Coordination) et considérons les divergences possibles entre l'interprétation par le *Maire* ou les *FF* de la notion de zone sûre. Ceci pourra être modélisé par le biais de deux règles d'inteprétation, exprimant des relations possibles entre des propriétés physiques de l'environnement et le fait interprété *zone sûre*, tout en préservant la possibilité d'une spécification normative indépendante et partagée par les deux acteurs, relative à la gestion des zones sûres. En ce qui concerne maintenant l'évolution potentielle des normes, d'une phase *préventive* à une phase d'*urgence*, la modélisation proposée permet au contraire de faire évoluer la spécification normative partagée, sans toucher aux règles d'interprétation des faits physiques.

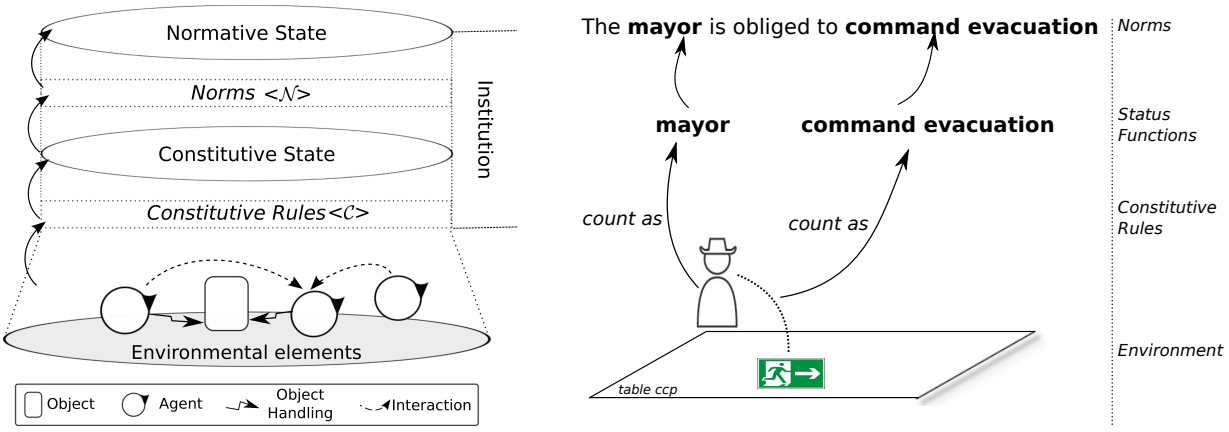
Nous examinons dans la section suivante comment formaliser et implémenter ces spécifications, en nous focalisant sur l'espace de *coordination*.

3 Institution artificielle située

Le modèle d'institution artificielle située (SAI) est constitué d'*éléments environnementaux*, de *fonctions statutaires*, de *règles constitutives* et de *normes*, visant à permettre une régulation d'un SMA à partir de faits issus de l'environnement (cf. Fig. 2(a)) [7]. Comme dans la dimension "système normatif", les normes définissent les obligations, permissions et interdictions des agents. Elles font référence ici à un niveau abstrait qui ne fait pas directement référence à l'environnement. Par exemple, une norme qui stipule que "*le maire est obligé d'ordonner l'évacuation*" ne spécifie ni qui est le maire qui est obligé de suivre la norme, ni ce que le maire doit concrètement faire pour la satisfaire. La réalisation effective d'une norme dépend de sa connexion à l'environnement puisque sa dynamique (activation, satisfaction, etc) est le résultat de faits qui s'y produisent. Nous verrons ci-dessous comment SAI aborde ces aspects.

Eléments environnementaux Les éléments environnementaux sont représentés par $\mathcal{X} = \{\mathcal{A}_\mathcal{X}, \mathcal{E}_\mathcal{X}, \mathcal{S}_\mathcal{X}\}$ tels que (*i*) $\mathcal{A}_\mathcal{X}$ est l'ensemble des agents qui peuvent agir dans le système, (*ii*) $\mathcal{E}_\mathcal{X}$ est l'ensemble des événements qui peuvent se produire dans l'environnement et (*iii*) $\mathcal{S}_\mathcal{X}$ est l'ensemble des propriétés utilisées pour décrire les états possibles de l'environnement.

Fonctions statutaires Les fonctions statutaires d'une SAI sont formellement représentées par $\mathcal{F} = \mathcal{A}_\mathcal{F} \cup \mathcal{E}_\mathcal{F} \cup \mathcal{S}_\mathcal{F}$, où (*i*) $\mathcal{A}_\mathcal{F}$ est l'ensemble des fonctions statutaires agents (i.e. qui peuvent être affectées à des agents), (*ii*) $\mathcal{E}_\mathcal{F}$ est l'ensemble des fonctions statutaires événements (i.e. qui peuvent être affectées à des événements), et (*iii*) $\mathcal{S}_\mathcal{F}$ est l'ensemble des fonctions statutaires états (i.e. qui peuvent être affectées à des états).
Les fonctions statutaires sont des fonctions que des éléments de l'environnement (agents, événements, états) remplissent selon une perspective institutionnelle [18]. Un agent devient le maire si l'institution le considère comme tel



(a) Vision générale abstraite

(b) Vision générale appliquée à la gestion de crises

FIGURE 2 – Vision générale de SAI

(ceci ne dépend pas uniquement des compétences de l'agent). De manière similaire, l'institution peut considérer l'événement issu d'un agent "signal d'alarme" comme étant un "ordre d'évacuation" ou une "alerte incendie" dans le contexte d'une crise. Il en est de même pour les états de l'environnement (e.g. "plus de 500 personnes dans une zone" peut signifier, dans l'institution, que la zone est à haut risque en cas de crise).

Règles constitutives Une règle constitutive $c \in \mathcal{C}$ est un n-uplet $\langle x, y, t, m \rangle$ signifiant que $x \in \mathcal{F} \cup \mathcal{X} \cup \{\varepsilon\}$ "counts as" [1] (i.e. a la fonction statutaire) $y \in \mathcal{F}$ quand l'événement $t \in \mathcal{E}_{\mathcal{F}} \cup \mathcal{E}_{\mathcal{X}} \cup \top$ a eu lieu et tant que l'état m est vrai dans l'environnement ou dans l'institution. [2] Si une fonction statutaire y est affectée à x, nous disons que x *constitue* y.

Normes Les normes dans une SAI sont des n-uplet de la forme $\langle c, a, d, i, o, \rho \rangle$ où (*i*) c est la condition d'activation de la norme, exprimée par des fonctions statutaires événements ou états ; (*ii*) a est la fonction statutaire agent pointant sur les agents ciblés par la norme ; (*iii*) $d \in \{obliged, prohibited, permitted\}$ est l'opérateur déontique de la norme ; (*iv*) i est l'objet de la norme exprimé par une fonction statutaire soit événement, soit état ; (*v*) o est une fonction statutaire événement ou état, optionnelle, qui est constituée comme conséquence de la non conformité d'une norme et (*vi*) ρ est une fonction statutaire optionnelle événement ou état pointant sur la date limite de la satisfaction de la norme.

4 SAI pour la gestion de crises

L'étude de cas décrite dans la section 2 est réalisée avec le SMA normatif hybride déployé sur un réseau de tables tangibles soutenant les interactions mixtes [3]. L'environnement dans lequel les agents interagissent correspond donc à des événements et états produits par les actions des acteurs humains sur les tables. Étant donné que l'action des agents sur l'environnement n'a pas en soi un sens dans la gestion de crises, les règles constitutives d'une SAI permettent d'*institutionnaliser* des faits survenus dans l'environnement, et de leur donner leur propre sens dans l'application (par exemple, le tangible B dans la position (C, D) counts-as une commande d'évacuation du centre-ville). Cette institutionnalisation est importante pour la régulation du scénario qui est, en définitive, la régulation des activités des agents dans l'environnement (cf. Fig. 2(b)). La section 4.1 décrit les aspects pertinents de l'environnement dans le cas d'étude proposé. Les sections 4.2 à 4.4 donnent la spécification de SAI pour notre cas d'étude (cf. Fig. 3). Ces éléments exprimés avec le langage de définition de SAI proposé dans [7], permettent de donner du sens aux interactions tangibles et de les réguler.

1. Par la suite nous préférons utiliser l'expression "count-as" dans sa version originelle plutôt que sa traduction.

2. ε exprime le fait que l'élément n'est pas présent dans la règle constitutive. Dans ce cas, la règle constitutive réalise un assignement libre (*freestanding assignment*) [18, 7]. Lorsque $t = \top$, l'affectation ne dépend d'aucun événement.

3. La plateforme logicielle utilisée est une version de JaCaMo [3], étendue à SAI et aux surfaces tangibles.

4.1 Environnement

L'environnement est composé de l'ensemble (éventuellement distribué) des équipements tangibles impliqués dans l'application. Du point de vue de la SAI, les agents appartiennent également à l'environnement. En dépit de la possible complexité de l'environnement, les éléments pertinents sont dans notre cas les événements survenus et les états reposant sur ces événements.

Parmi tous les événements qui peuvent se produire dans l'environnement, ceux qui sont pertinents ici sont (*i*) `checkin(AgentID, TableID)`, déclenché lorsque l'agent `AgentID` s'identifie sur la table `TableID`, et (*ii*) `putTangible(TableID, TangiID, X, Y, AgentID)`, déclenché lorsque l'agent `AgentID` met un tangible `TangiID` sur les coordonnées (X, Y) de `TableID`.

Les propriétés environnementales pertinentes qui composent l'état de l'environnement, fournies par des bases de données, GIS, etc., sont (*i*) `nbInhabitants(ZoneID, X)` qui établit que `ZoneID` a X habitants et (*ii*) `security_phase(ZoneID, Phase)` qui établit que `ZoneID` est en phase de sécurité, où $Phase \in \{preventive, emergency\}$.

4.2 Fonctions statutaires

Les dynamiques environnementales décrites en section 4.1 animent la dynamique institutionnelle en faisant appel à la constitution des fonctions statutaires. La spécification en Fig. 3 spécifie les fonctions statutaires pertinentes du cas d'étude présenté, comme suit :
– Les **fonctions statutaires agents** définissent que les agents agissent dans le scénario en tant que (*i*) *maire* de la ville (`mayor`), (*ii*) membre du CCP (`ccp_member`), (*iii*) membre du LC (`logistic_cell`) et (*iv*) `firefighter`.
– Les **fonctions statutaires événements** définissent que les événements survenant dans l'environnement peuvent signifier dans l'institution (*i*) de demander à un `Evacuator` d'évacuer une `Zone` (`ask_for_evacuation(Zone,Evacuator)`), et (*ii*) l'évacuation d'une `Zone` (`evacuate(Zone)`).
– Les **fonctions statutaires états** définissent que le système peut être dans des états où, du point de vue de l'institution, (*i*) une `Zone` est considérée sûre pour des procédures de sécurité (`secure(Zone)`), (*ii*) une `Zone` est risquée (`insecure(Zone)`), et

```
status_functions:
  agents: mayor, ccp_member, logistic_cell, firefighter.
  events: ask_for_evacuation(Zone,Evacuator), evacuate(Zone).
  states: secure(Zone), insecure(Zone), electric_risky(Zone).
norms:
/*The mayor is prohibited to ask Logistic Cell to evacuate insecure zones*/
1: insecure(Zone): mayor prohibited ask_for_evacuation(Zone,logistic_cell).
/*The mayor is permitted to ask Logistic Cell to evacuate secure zones*/
2: secure(Zone): mayor permitted ask_for_evacuation(Zone,logistic_cell).
/*The firefighter is obliged to evacuate insecure zones*/
3: insecure(Zone): firefighters obliged evacuate(Zone).
/*Firefighters are prohibited to evacuate secure zones*/
4: secure(Zone): firefighter prohibited evacuate(Zone).
constitutive_rules:
/*** Agent-Status Functions constitutive rules ***/
/*Actors carry the status functions according to their check in the tables*/
1: Agent count-as mayor when checkin(table_ccp,Agent)
   while not(Other is mayor)|Other==Agent.
2: Agent count-as logistic_cell when checkin(table_logistic_cell,Agent).
3: Agent count-as firefighter when checkin(table_fire_brigade,Agent).
4: mayor count-as ccp_member.
/*** Event-Status Functions constitutive rules ***/
/*Mayor putting tangibleObject1 on (15,20) means asking to LC to evacuate
   the downtown*/
5: putTangible(_,tangibleObject1,15,20,Actor)
   count-as ask_for_evacuation(downtown,logistic_cell) while Actor is mayor.
/*LC putting tangibleObject2 on (15,20) means LC is evacuating the downtown*/
6: putTangible(_,tangibleObject2,15,20,Actor)
   count-as evacuate(downtown) while Actor is logistic_cell.
/*FF putting tangibleObject3 on (15,20) means FF is evacuating the downtown*/
7: putTangible(_,tangibleObject3,15,20,Actor)
   count-as evacuate(downtown) while Actor is firefighter.
/*** State-Status Functions constitutive rules ***/
/*A zone in preventive phase is secure if it does not pose electrical risks
   and it has at most 500 inhabitants*/.
8: security_phase(Zone,preventive) count-as secure(Zone)
   while not(electric_risky(Zone)) &
         ((nbInhabit(Zone,X)& X<=500) |
          (phase(Zone,preventive) is secure(Zone)))
/*A zone in preventive phase is insecure if it poses electrical risks*/.
9: security_phase(Zone,preventive)
   count-as insecure(Zone) while electric_risky(Zone).
/*A zone in emergency phase insecure*/
10: security_phase(Zone,emergency) count-as insecure(Zone).
/*The downtown is electric risky if the firefighter puts the tangible
   tangibleObject4 on (15,20)*/
11: count-as electric_risky(downtown)
    when putTangiNote(_,tangibleObject4,15,20,Actor)
    while Actor is firefighter.
```

FIGURE 3 – Spécification normative et constitutive de l'étude de cas

(*iii*) une `Zone` présente un risque électrique (`electrical_risky(Zone)`).

4.3 Règles constitutives

Comme pour les fonctions statutaires, trois ensembles de règles constitutives sont considérés :
- Règles constitutives des fonctions statutaires agents Les règles 1 à 3 (cf. Fig. 3) précisent que les fonctions statutaires agents de `mayor`, `logistic_cell` et *firefighter* sont constituées par les agents qui se sont inscrits sur la table produisant l'événement `checkin(Table, Agent)`. La clause **while** de la règle 1 assure que la fonction statutaire de `mayor` n'est attribuée qu'à un seul agent à la fois, car elle définit que l'agent `mayor` conserve cette fonction statutaire tant qu'elle n'est pas attribuée à un autre agent ou à l'agent lui-même. La règle 4 postule que l'agent portant la fonction statutaire de `mayor` porte également la fonction statutaire de membre du (`ccp_member`).
- Règles constitutives des fonctions statutaires événements Les règles 5 à 7 (cf. Fig. 3) définissent que certaines interactions tangibles signifient, d'un point de vue institutionnel, une

demande d'évacuation et une évacuation. Ce sens est conditionné par l'objet tangible utilisé durant l'interaction et ainsi que par l'acteur qui exécute l'interaction.

- Règles constitutives des fonctions statutaires états Selon la règle 8 (cf. Fig. 3), la propriété `security_phase(Zone,preventive)` définit les cas où la zone est assez sécuritaire pour que les non professionnels puissent intervenir. A travers la première partie de la clause **while**, cette relation entre l'état de l'environnement (zone préventive) et état institutionnel (zone sure) est présente dans le cas où la zone ne comporte pas de risque électrique. En outre, à travers la partie restante de la clause **while**, cette relation existe dans les cas où la zone a, au plus, 500 habitants ou si elle est déjà valide en termes de sécurité. Ainsi (i) si `security_phase(Zone,preventive)` devient effective lorsque la zone a plus de 500 habitants, la zone n'est pas considérée comme valide pour la sécurité et (ii) la zone reste sure même si le nombre de ses habitants change, dépassant ainsi le seuil. Notez que, si `security_phase(Zone,preventive)` ne devient pas active dans l'environnement, on ne peut pas définir la fonction statutaire `secure(Zone)`. Les règles 9 et 10 définissent une zone `insecure(zone)` du point de vue institutionnel. La règle 11 définit ce qui constitue une zone à risque électrique. Elle réalise un assignement libre puisqu'il n'y a pas d'élément concret dans l'environnement pour porter les fonctions statutaires.

4.4 Normes

Les normes décrites en Fig. 3 définissent des autorisations, interdictions et obligations liées aux demandes d'évacuation et aux évacuations elles-mêmes. Notez que les normes ne se réfèrent pas directement à l'environnement, mais au contraire, à des fonctions statutaires. Par exemple, la norme 2 permet à tout agent porteur de la fonction statutaire de *maire* d'exécuter des actions dont la signification dans l'institution correspond à un événement de demande à la *LC* d'effectuer une évacuation.

5 Contributions de la régulation avec SAI gestion de crises

Pour illustrer l'utilisation pratique de SAI dans notre application de gestion de crises, nous nous plaçons dans un système composé de trois tables tangibles : `ccp`, `logistic_cell`, et `fire_brigade`, éventuellement distibuées, utilisées respectivement par le *CCP*, la *cellule logistique*, et les *pompiers*. Sauf indication contraire, les règles de régulation de l'application suivent les spécifications illustrées en Fig. 3.

Les acteurs humains représentant le `mayor`, `logistic_cell`, et `firefighter` se sont identifiés dans le système, mettant alors en œuvre les fonctions statutaires agents appropriées selon les règles constitutives 1 à 4. Ils collaborent sur une zone contenant 300 habitants. Lors du démarrage, ils sont en phase préventive. Les exemples suivants illustrent comment l'utilisation de SAI permet une régulation située, tout en gérant des divergences de règles constitutives, des incohérences de normes, des évolutions environnementales.

5.1 Exemple 1 : Modification des règles constitutives (sans modifier les normes)

Dans le scénario de gestion de crise présenté, alors que les différents acteurs et les organisations peuvent avoir des visions différentes sur le même fait institutionnel, ils ont besoin d'avoir la même interprétation de chaque situation pour distribuer efficacement leurs efforts. Supposons que, pour le *Maire*,une zone est sûre chaque fois qu'elle est en phase préventive et son nombre d'habitants est inférieur à un certain seuil. Pour les *Pompiers*, par contre, une zone est *sûre* chaque fois qu'elle est en phase préventive et ne présente pas de risque, par exemple électrique. Alors, une zone *sûre* est constituée différemment selon les perspectives du *Maire* et des *Pompiers* :

```
/* Mayor's view */
security_phase(Zone,Phase) count-as secure(Zone)
while Zone is preventive & ((nbInhabit(Zone,X)& X<500) |
(security_phase(Zone,Phase) is secure(Zone))).
/* Firefighters' view  */
security_phase(Zone,Phase) count-as secure(Zone)
while not(electric_risky(Zone)) & Zone is preventive &
(security_phase(Zone,Phase) is secure(Zone))).
```

Dans cet exemple, comme les conditions des règles constitutives de "Mayor's view" et de "Firefighters' view" `nbInhabit(Zone, X)& X<500)` et `not(electric_risky(Zone))` ne se recoupent pas entièrement, les incohérences d'interprétation se produiront, car l'un considérera la zone comme `secure(zone)` et l'autre comme `insecure(zone)`. Cela va entraîner des incohérences d'action car la même action peut être simultanément autorisée et interdite (normes 1 et 2) selon le point de vue des acteurs. Ces incohérences peuvent être résolues en

agrégeant ces deux règles constitutives, générant ainsi la règle constitutive 8.

5.2 Exemple 2 : Modifier les normes (sans changer les règles constitutives)

Le contraire est également possible : les normes peuvent changer sans changer les règles constitutives. Considérons par exemple la norme 1. Elle indique que le *Maire* a *interdiction* de demander à la *cellule logistique* d'évacuer une zone *risquée*, ce qui signifie pour lui : soit qu'il est en phase d'*urgence* soit qu'il y a un *risque électrique* soit qu'il y a plus de 500 habitants (règles constitutives 8-10). Les règles institutionnelles pourraient évoluer pour tenir compte du risque électrique comme seule condition pour interdire au *Maire* de demander l'*évacuation*. Pour tenir compte de cette évolution, les règles constitutives pourraient rester telles quelles et la norme 1 peut être changée en :

```
electric_risky(Zone): mayor prohibited
ask_for_evacuation(Zone, logistic_cell).
```

5.3 Exemple 3 : Contextualiser l'évolution des normes actives

Cet exemple montre comment les normes peuvent évoluer automatiquement, en fonction du contexte, ainsi fournir un règlement situé. Comme déjà mentionné, en phase préventive, le *Maire* est *autorisé* à demander l'*évacuation*. Lorsque la phase change en phase d'*urgence*, le *Maire* a *interdiction* de s'occuper de l'*évacuation* et il est *obligatoire* pour les *pompiers* le faire. Cela est exprimé comme suit :

En phase *préventive*, la propriété environnementale `security_phase(Zone, preventive)` est toujours maintenue. Si la `Zone` n'est pas `electric_risky` et a ((nbInhabit(Zone,X)& X<=500), alors la fonction statutaire `secure(Zone)` est constituée par la règle constitutive 8. En conséquence, la norme 2 devient active. Lors d'une évolution en phase d'*urgence*, la fonction statutaire précédente est modifiée en `security_phase(Zone, emergency)`. La fonction statutaire `secure(Zone)` n'est plus constituée. Ainsi, la norme 2 est désactivée tandis que la norme 1 est activée. Comme on le voit, en changeant le contexte (de *préventif* à *urgence*), même si les faits sont interprétés avec le même ensemble de règles de comportement, les normes actives changent.

5.4 Exemple 4 : Renforcer la proactivité dans la gestion de crises

Comme mentionné dans la section Section 2, le rôle du système peut changer entre un rôle purement éducatif, où les acteurs prennent en charge toutes les actions, jusqu'à une aide plus active du système en gestion de crise, où le système peut être plus autonome et automatiser certaines actions. Supposons que le *Maire* a été informé d'une inondation dans une zone donnée. Dans ce contexte, le *Maire* a pour obligation de réaliser la constitution de la fonction statutaire événement :

```
putTangible(_,tangibleObject1,X_zone,Y_zone,Actor)
count-as ask_for_evacuation(Zone, logistic_cell)
while Actor is mayor.
```

Ceci exprime le fait que le *Maire*, grâce à un objet tangible, devrait explicitement entreprendre les actions nécessaires. Dans un contexte où le système est plus autonome, au contraire, la tâche `ask_for_evacuation(Zone, logistic_cell)` serait entreprise par le système si la règle constitutive 5 est définie comme suit :

```
get_information(flood, Zone, Agent)
count-as ask_for_evacuation(Zone, logistic_cell)
while Agent is mayor.
```

6 Travaux liés et discussions

Dans ce qui suit, nous rappelons brièvement les particularités de la collaboration humaine dans le cadre de la gestion de crise et dressons quelques réponses issues du domaine du Computer Supported Collaborative Work (CSCW). Nous montrons ensuite les liens avec des préoccupations centrales de la cognition distribuée, sociale et située. Nous terminons par une discussion sur les valeurs ajoutées des SMA normatifs et plus particulièrement des Institutions Artificielles Situées.

La gestion de crise est une activité collaborative complexe qui implique la participation de multiples acteurs et organisations souvent distribués spatialement et temporellement, avec des perceptions locales, des buts et des politiques qui peuvent être divergents [14]. Les participants doivent disposer de procédures claires pour se coordonner et agir dans ces environnements dégradés, exposés à des contraintes critiques. La non connaissance mutuelle de ces règles rend

difficile la mise en place par les acteurs d'une réponse cohérente.

Les plateformes actuelles proposent souvent des outils de communication simple (e.g. Google Wave ou Wiki) donnant une réponse dans des contextes clairement définis et fermés. Leur adaptation dans un cadre de gestion de crises est possible dans le cas de routines de secours bien définies et ne tolèrent souvent pas d'exceptions [9]. Dans les travaux du CSCW appliqués à cette problématique [15], une attention particulière a été accordée à la prise en compte du contexte avec un focus sur les politiques qui régulent le travail distribué [17], sur le partage d'un environnement physique commun [19].
Comme ces travaux, notre proposition s'appuie également sur un environnement physique partagé mais aussi sur un partage de normes liées à des organisations. L'utilisation de supports tangibles permet la mise en place d'une activité flexible et opportuniste. Des retours virtuels mettent en avant les manques possibles, les incohérences entre politiques et favorisent la mise en place d'une conscience organisationnelle partagée [20]. Une approche multi-agent normative, hybride et mixte est bien adaptée à la mise en place de telles dimensions. Les différents espaces de modélisation que nous proposons prennent en compte la dimension physique de production et de communication ainsi que la dimension organisationnelle de coordination.
Une contribution supplémentaire de notre travail est de situer ces différentes normes dans l'environnement physique, i.e. de coupler la sémantique de l'organisation et de ses normes avec les éléments de l'environnement physique [1]. Dans [6], il est proposé de lier des faits environnementaux à la dynamique des éléments de régulation plutôt qu'à des concepts institutionnels. Bien que ceci puisse permettre de spécifier qu'une interaction tangible "counting as" par exemple une violation de norme, il n'est pas possible, comme nous le proposons ici, de spécifier qu'une telle interaction "count as" une évacuation. Une autre proposition [5, 16, 4] consiste à traiter ce problème de positionnement dans l'environnement comme un problème d'interopérabilité entre l'environnement et l'institution. Ceci se limite à concevoir des interfaces permettant à l'environnement d'informer les éléments de régulation sur ce qui devrait se passer dans l'institution. Une troisième approche, proposée dans [1], en lien avec [11], relie les éléments de l'environnement avec des concepts institutionnels mais pas à la sémantique de ces concepts. Dans ce cas, par exemple, alors qu'il est possible d'établir qu'un élément de l'environnement "counts as" une *évacuation*, il n'est dit nulle part si une *évacuation* est un événement, un agent ou autre chose. A la différence de cette approche, l'approche SAI donne une sémantique institutionnelle aux éléments de l'environnement et les relie également à la sémantique des normes.

7 Conclusions

Ce papier a mis en avant l'utilisation des Institutions Artificielles Situées (SAI) au sein d'un SMA normatif, hybride et mixte afin de réguler la collaboration d'humains dans le cadre de la gestion de crise. La conception proposée s'appuie sur l'analyse de différents domaines de recherche. Une première version démonstratrice de notre approche est en cours de mise en place sur une version de la plateforme JaCaMo [3] étendue à SAI et les tables tangibles.
Comme nous l'avons montré dans les exemples de notre application, l'approche proposée répond à deux préoccupations rencontrées dans la conception d'un outil de gestion de crises : une expression claire de la coordination [8] et une préservation de la flexibilité, exigences toutes deux nécessaires mais souvent non conciliables [9]. Notre proposition permet plus précisément de répondre aux questions de divergences d'interprétations, d'incohérences de normes, d'évolution de contexte et de niveau d'autonomie au sein du système. Ceci est mis en place facilement grâce à deux niveaux de modélisation distincts [1], exprimés de manière déclarative : le niveau constitutif, et le niveau normatif. Plus généralement, la modélisation que nous proposons rend le traitement des normes sensible au contexte, résolvant ainsi le compromis flexibilité-déclarativité : les changements dans l'environnement physique vont lancer le déclenchement de règles constitutives, activant à leur tour les normes correspondantes.

Nos travaux futurs concerneront la modélisation de l'ensemble des normes impliquées dans les activités de production et de communication permettant ainsi de mettre en place la totalité des interactions mixtes dans le SMA normatif, hybride et situé au centre duquel se place la conscience organisationnelle partagée.

Remerciements

Les auteurs remercient les membres de l'IRMa (Institut des Risques Majeurs de Grenoble) pour leur collaboration et également les soutiens accordés par CAPES, CNPq (contrats

448462/2014-1 et 306301/2012-1) et l'ARC 6 Région Rhône-Alpes (ARC-13-009716-01 et ARC-13-010152-01).

Références

[1] Huib Aldewereld, Sergio Álvarez Napagao, Frank Dignum, and Javier Vázquez-Salceda. Making norms concrete. In Wiebe van der Hoek, Gal A. Kaminka, Yves Lespérance, Michael Luck, and Sandip Sen, editors, *AAMAS 2010*, pages 807–814, 2010.

[2] Guido Boella, Leendert van der Torre, and Harko Verhagen. Introduction to the special issue on normative multiagent systems. *Autonomous Agents and Multi-Agent Systems*, 17(1) :1–10, 2008.

[3] Olivier Boissier, Rafael H. Bordini, Jomi Fred Hübner, Alessandro Ricci, and Andrea Santi. Multi-agent oriented programming with jacamo. *Sci. Comput. Program.*, 78(6) :747–761, 2013.

[4] Maiquel Brito, Jomi F. Hübner, and Rafael H. Bordini. Programming institutional facts in multi-agent systems. In Huib Aldewereld and Jaime S. Sichman, editors, *COIN VIII*, volume 7756 of *LNCS*, pages 158–173. 2013.

[5] Jordi Campos, Maite López-Sánchez, Juan Rodríguez-Aguilar, and Marc Esteva. Formalising Situatedness and Adaptation in Electronic Institutions. In Jomi Hübner, Eric Matson, Olivier Boissier, and Virginia Dignum, editors, *COIN IV*, volume 5428 of *LNCS*, pages 126–139. 2009.

[6] Mehdi Dastani, Leendert Torre, and Neil Yorke-Smith. Monitoring interaction in organisations. In Huib Aldewereld and Jaime S. Sichman, editors, *COIN VIII*, volume 7756 of *LNCS*, pages 17–34. 2013.

[7] Maiquel de Brito, Jomi Fred Hübner, and Olivier Boissier. A conceptual model for situated artificial institutions. In Nils Bulling, Leendert van der Torre, Serena Villata, Wojtek Jamroga, and Wamberto Vasconcelos, editors, *CLIMA XV*, volume 8624 of *LNCS*. 2014.

[8] Julie Dugdale, Nargès Bellamine-Ben Saoud, Bernard Pavard, and Nico Pallamin. Simulation and Emergency Management. In B. Van de Walle, M. Turoff, and S. R. Hiltz, editors, *Information Systems for Emergency Management*, volume 16 of *Advances in Management Information Systems*. M.E. Sharpe, 2010. Part IV. Systems Design and Technology (Chap. 10).

[9] J. Franke and F. Charoy. Design of a collaborative disaster response process management system. In *9th International conference on the design of Cooperative Systems*, 2010.

[10] Catherine Garbay, Fabien Badeig, and Jean Caelen. Normative multi-agent approach to support collaborative work in distributed tangible environments. In Steven E. Poltrock, Carla Simone, Jonathan Grudin, Gloria Mark, and John Riedl, editors, *CSCW 12*, pages 83–86. ACM, 2012.

[11] Davide Grossi, Huib Aldewereld, Javier Vázquez-Salceda, and Frank Dignum. Ontological aspects of the implementation of norms in agent-based electronic institutions. *Computational & Mathematical Organization Theory*, 12(2-3) :251–275, 2006.

[12] Yann Laurillau and Laurence Nigay. Clover architecture for groupware. In *CSCW 2002, Proceeding on the ACM 2002 Conference on Computer Supported Cooperative Work, New Orleans, Louisiana, USA, November 16-20, 2002*, pages 236–245, 2002.

[13] Yoann Lebrun, Emmanuel Adam, Sébastien Kubicki, and René Mandiau. A multi-agent system approach for interactive table using RFID. In *Advances in Practical Applications of Agents and Multiagent Systems, 8th International Conference on Practical Applications of Agents and Multiagent Systems, PAAMS 2010, Salamanca, Spain, 26-28 April 2010, Yves Demazeau, Frank Dignum, Juan M. Corchado, Javier Bajo editors*, pages 125–134, 2010.

[14] A. H. J Oomes. Oragnization awareness in crisis management. In *ISCRAM*, 2004.

[15] Volkmar Pipek, SophiaB. Liu, and Andruid Kerne. Crisis informatics and collaboration : A brief introduction. *CSCW*, 23(4-6) :339–345, 2014.

[16] Michele Piunti, Olivier Boissier, Jomi Fred Hübner, and Alessandro Ricci. Embodied organizations : a unifying perspective in programming agents, organizations and environments. In *(MALLOW 2010)*, volume 627 of *CEUR*, 2010.

[17] J. M. Robert. Cognition située, cognition distribuée et cognition socialement partagée. Cours, 2012.

[18] John Searle. *Making the Social World :The Structure of Human Civilization*. Oxford University Press, 2009.

[19] Orit Shaer and Eva Hornecker. Tangible user interfaces : Past, present, and future directions. *HCI*, 3 :1–137, 2010.

[20] Lauren Thévin, Fabien Badeig, Julie Dugdale, Olivier Boissier, and Catherine Garbay. Un système multi-agent normatif pour la collaboration et l'interaction mixte. In *JFSMA 18*, pages 1–10, 2014.

Benchmarking de performance pour l'exploration multi-robots

Zhi Yan Luc Fabresse Jannik Laval Noury Bouraqadi

prénom.nom@mines-douai.fr
Département Informatique et Automatique,
École des Mines de Douai, France

Résumé

Le benchmarking de performance est devenu un thème important dans la robotique. C'est en effet un moyen essentiel de comparer des solutions dans des conditions différentes. Dans cet article, nous nous intéressons sur l'analyse comparative des performances des systèmes multi-robots qui explorent et cartographient des terrains inconnus. Nous proposons une sélection de métriques pour comparer objectivement différents algorithmes de coordination de systèmes multi-robots pour effectuer l'exploration. Nous identifions également des paramètres qui influent sur la performance de la flotte robotique. En faisant varier les paramètres, nous pouvons identifier les forces et les limites d'un algorithme. Ce travail est aussi une première étape concrète pour résoudre le problème général de la comparaison quantitative de différents algorithmes de coordination multi-robots. Nous illustrons ces contributions avec des simulations réalistes d'une stratégie d'exploration basée sur les frontières. Ces simulations ont été mises en œuvre à l'aide du middleware ROS, ce qui permet de découpler le code des contrôleurs des robots du moteur physique du simulateur. Nous pouvons donc utiliser le même code aussi bien en simulation qu'avec de vrais robots.

Mots-clés : *benchmarking de performance, système multi-robots, exploration collaborative*

Abstract

Performance benchmarking has become an important topic within robotics. It is indeed, a critical way to compare different solutions under different conditions. In this paper, we focus on performance benchmarking of multi-robot systems which explore and map unknown terrains. We summarize metrics to objectively compare different algorithms that can be applied to collaborative multi-robot exploration. We also identify parameters that impact robotic fleet performances. By varying the parameters, we can identify strengths and limits of an algorithm. This work is also a first concrete step to address the general problem of objectively comparing different multi-robot coordination algorithms. We illustrate these contributions with realistic benchmark simulations of the frontier-based exploration strategy. The simulations were implemented in ROS.

Keywords: *performance benchmarking, multi-robot system, collaborative exploration*

1 Introduction

De nombreuses applications robotiques peuvent bénéficier de l'utilisation d'une flotte de plusieurs robots au lieu de compter sur un seul robot [15]. En effet, avoir plusieurs robots signifie une augmentation de robustesse grâce à la redondance. En outre, plusieurs robots peuvent effectuer des tâches en parallèle et donc accélérer l'exécution, ce qui peut finalement augmenter les avantages des applications telles que la recherche et le sauvetage après les tremblements de terre, la recherche du feu à l'intérieur des bâtiments, l'exploration minière, et le déminage.

Cependant, l'utilisation des systèmes multi-robots soulève le défi de la coordination [21]. Pour vraiment profiter du parallélisme potentiel d'une flotte robotique, nous devons avoir des stratégies pour l'organisation de l'activité des robots qui garantissent la plus haute performance. A titre d'exemple, considérons l'exploration d'un environnement inconnu [18] qui est une tâche commune à de nombreuses applications. Une stratégie de coordination doit attribuer à chaque robot un ensemble de zones à explorer de façon à réduire à la fois le temps nécessaire pour construire une carte complète, et l'énergie totale consommée par la flotte robotique [20]. Malheureusement, l'élaboration de stratégie optimale ou quasi-optimale pour la coordination n'est pas facile. C'est pourquoi la communauté de recherche fait des efforts considérables pour traiter ce problème [4, 7, 12, 18, 19, 22].

L'abondance des algorithmes d'exploration multi-robots collaborative est en soit un problème quand on doit choisir le plus approprié à utiliser pour une situation donnée. Les auteurs évaluent leurs solutions avec différents robots ou simulateurs, dans différents environnements et conditions. Par conséquent, les résultats présentés ne peuvent pas être comparés facilement. De plus, la reproductibilité des expériences est presque impossible.

Une évaluation mathématique des algorithmes - comme l'analyse de la complexité - est pratiquement infaisable, à cause de la complexité des systèmes multi-robots et de leurs environnements. Il y a trop de paramètres qui peuvent influer sur les performances. Des exemples de ces paramètres sont : le nombre de robots, la puissance de calcul disponible, la mémoire sur chaque robot, le fait que la flotte est homogène ou hétérogène.

Dans cet article, nous adoptons une approche alternative qui est une évaluation empirique. Elle consiste au benchmarking des algorithmes qui peuvent être appliqués à l'exploration multi-robots [3, 6]. Pour comparer efficacement les différents algorithmes, nous avons sélectionné cinq métriques qui permettent une évaluation quantitative de la performance. Ils permettent de mesurer le temps, le coût et l'efficacité de l'exploration, ainsi que la complétude et la qualité des cartes construites. Nous identifions également des paramètres qui influent sur la performance d'exploration d'un système multi-robots. Ces paramètres visent à faciliter la reproductibilité des expériences. Ils favorisent la définition des environnements standards et des plans d'expérience qui peuvent être partagées par la communauté, et donc faciliter la comparaison des résultats obtenus par différents chercheurs.

Enfin, nous illustrons nos métriques et paramètres en utilisant une simulation 3D avec des robots basés sur ROS (abréviation anglaise de *Robot Operating System*). ROS est un méta-système d'exploitation, quelque chose entre le système d'exploitation et le middleware. Il est aujourd'hui reconnu comme une plateforme logicielle standard et est utilisé par de nombreuses institutions. Nos simulations visent la stratégie de Yamauchi [19] pour l'exploration multi-robots basée sur le concept de frontière, avec deux algorithmes de fusion de carte. Nous montrons que nos métriques peuvent sensiblement refléter l'impact des différents algorithmes sur la performance du système. En outre, pour montrer l'influence des paramètres, nous avons mesuré les performances de différentes tailles de flotte sur différents terrains.

Le reste du papier est organisé comme suit : la Section 2 donne un aperçu des travaux connexes ; la Section 3 présente notre sélection de métriques de performance pour le benchmarking ; la Section 4 discute les paramètres d'un système d'exploration pour une flotte de robots mobiles ; la Section 5 décrit les simulations de benchmark et les résultats ; le papier est conclu dans la Section 6.

2 Etat de l'art

Bautin *et al.* [2] ont comparé leur algorithme d'affectation des frontières pour l'exploration multi-robots nommé MinPos avec deux algorithmes bien connus dans l'état de l'art : celui de Yamauchi [19] et celui de Burgard [4], dans trois environnements différents. Ils ont mesuré le temps d'exploration donné par les étapes de simulation, tout en faisant varier le nombre de robots dans différentes méthodes. La comparaison a été éffectué sur deux simulateurs développés par eux-mêmes. L'un est discret, l'autre est plus réaliste qui permet d'utiliser exactement les mêmes codes en simulation et sur les vrais robots.

Faigl *et al.* [9] ont proposé un framework d'évaluation pour étudier les stratégies d'exploration de façon indépendante sur les ressources de calcul disponibles et comparé cinq algorithmes de répartition des tâches dans les scénarios d'exploration multi-robots. Ils ont expérimenté des simulations discrètes dans quatre environnements qui représentent des espaces ouverts et des bureaux. Le métrique de performance utilisé est le nombre d'étapes de simulation nécessaire pour explorer tout l'environnement par le simulateur.

Couceiro *et al.* [5] ont présenté plusieurs expériences de simulation pour évaluer cinq algorithmes d'exploration et de cartographie multi-robots. Ils ont utilisé deux métriques de performance : 1) le rapport d'exploration d'environnement au fil du temps, calculé par la division de la surface de la carte construite par les robots à un instant donné par la surface totale du terrain ; 2) l'aire sous la courbe (*area under the curve – AUC*) qui est obtenu en calculant la moyenne de 500 itérations du rapport d'exploration. Dans leurs expériences, le temps est représenté par le nombre d'itérations d'exploration. Les expériences ont été effectuées en utilisant le simula-

teur MRSim [1], qui est un simulateur discret basé sur Matlab.

Frank *et al.* [10] ont comparé quatre stratégies différentes de sélection de frontières pour l'exploration multi-robots et analysé la performance en termes du (1) temps (compté en nombre de pas de simulation) nécessaire pour explorer tout l'environnement, ainsi que de (2) la zone explorée par pas de temps. Les expériences de simulation ont été réalisées en Matlab. Les expériences sont réalisées dans un cadre discret et idéal : elles négligent les problèmes de localisation et seulement les environnements convexes sans obstacles sont considérés.

Amigoni [1] a expérimentalement comparé quatre stratégies d'exploration afin d'évaluer leurs forces et leurs faiblesses. Les expériences ont été effectuées sur un simulateur discret avec un seul robot. Seulement deux métriques ont été prises en compte pour comparer les performances des stratégies d'exploration : (1) le nombre d'opérations de détection nécessaires pour réaliser l'exploration et (2) la distance totale parcourue par le robot pendant l'exploration.

Scrapper *et al.* [17] se sont concentrés sur le développement de méthodes et techniques de test standard pour évaluer quantitativement la performance des systèmes de cartographie robotique par rapport aux exigences définies par l'utilisateur. Ils ont défini la qualité de la carte comme le rapport entre le nombre de points caractéristiques valables trouvés sur la carte construite par le robot et le nombre de points caractéristiques dans la véritable carte. Toutefois, cette métrique n'évalue pas si les caractéristiques présentent la même forme.

Lass *et al.* [13] ont examiné plusieurs métriques d'évaluation pour les systèmes multi-agents. Ils ont classé les métriques le long de deux axes : la performance et les types de données. Les métriques de performance quantifient la consommation des ressources du système, telles que l'utilisation de bande passante, la consommation d'énergie, et le temps d'exécution. Les types de données incluent le nominal, l'ordinal, l'intervalle, et le rapport. Ce travail peut être utilisé comme une référence, car les systèmes multi-robots peuvent être considérés comme un cas particulier de systèmes multi-agents.

1. http://www.mathworks.com/matlabcentral/fileexchange/38409-mrsim-multi-robot-simulator--v1-0-

Nous sélectionnons donc les métriques indépendants du matériel et du logiciel, comme le temps d'exploration, le rapport d'exploration, et la distance totale parcourue, tout en donnant une définition détaillée. Le métrique comme le nombre d'opérations de détection nécessaires pour réaliser l'exploration est ainsi rejeté, puisque différents matériels ont différentes capacités de détection. En outre, nous utilisons un simulateur réaliste afin de rendre les résultats de benchmarking plus significatives.

3 Métriques

La mesure de performance est une pierre angulaire de l'analyse et la comparaison quantitative. Elle est particulièrement nécessaire pour l'exploration robotique notamment pour des applications critiques comme les opérations de sauvetage après un tremblement de terre et un incendie. Une comparaison quantitative de plusieurs méthodes d'exploration permet de choisir la méthode la plus efficace afin de localiser les victimes plus rapidement.

Dans cette section, nous présentons cinq métriques de performance que nous avons sélectionnées pour quantifier les résultats d'exploration et donnons clairement leurs définitions. Ils sont le *temps d'exploration*, le *coût d'exploration*, l'*efficacité d'exploration*, l'*complétude de carte*, et la *qualité de carte*. Elles présentent l'avantage d'être applicables à une large variété de problèmes d'exploration : avec des flottes robotiques de caractéristiques différentes qui opérent dans différents types d'environnements.

3.1 Temps d'exploration

L'un des objectifs concernant l'optimisation de l'exploration multi-robots est de minimiser le temps global d'exploration [18, 20, 19, 4]. Le défi est de faire que chaque robot se déplace vers une direction différente afin de maximiser la surface découverte de l'environnement et de réduire ainsi le nombre de zones visitées par plus d'un robot.

Nous définissons le temps d'exploration comme le temps total nécessaire pour achever une mission d'exploration pour une flotte de robots. Dans notre définition, le décompte commence lorsqu'au moins un robot de la flotte commence l'exploration, et se termine lorsqu'au moins un robot a exploré un pourcentage cible (par exemple, 99%) de l'ensemble du terrain (avec ou sans échange de la carte explorée). Le temps

est mesuré en temps réel, nous montrant combien de jours, heures, minutes, et secondes la flotte a été passé à l'exploration.

3.2 Coût d'exploration

La définition du coût d'exploration dépend fortement des besoins des utilisateurs. Il pourrait être l'énergie consommée par les actionneurs et les ressources de calcul (par exemple, CPU, RAM, et bande passante de réseau), ou le prix des robots, ou leurs coûts de manutention et d'entretien.

Cependant, la dépense énergetique est la seule à être directement impactée par les algorithmes d'exploration. Par ailleurs, l'énergie consommée pour le calcul peut être négligée face à celle consommée par les moteurs pour les déplacements. Or, la distance parcourue par les robots est une bonne approximation du coût énergetique des moteurs. Elle est aussi simple à mesurer, ce qui explique sa large utilisation dans la litterature [18, 20, 4, 22].

Aussi, nous définissons le coût d'exploration comme la somme des distances parcourues par chaque robot de la flotte :

$$coutDExploration(n) = \sum_{i=1}^{n} d_i \qquad (1)$$

où n est le nombre de robots dans la flotte, et d_i est la distance parcourue par le robot i.

3.3 Efficacité d'exploration

L'efficacité d'exploration est directement proportionnelle à la quantité d'informations extraites de l'environnement, et est inversement proportionnelle aux coûts d'exploration de la flotte de robot [22] :

$$EfficaciteDExploration(n) = \frac{M}{coutDExploration(n)} \qquad (2)$$

où n est le nombre de robots dans la flotte, et M est la surface de la zone explorée totale en mètres carrés. Par exemple, si la valeur de l'efficacité d'exploration est 1.6, cela signifie que chaque robot qui se déplace d'$1m$ découvre en moyenne $1.6m^2$ du terrain. Inspiré de l'analyse de rapport coûts-avantages en économie, les utilisateurs peuvent considérer qu'un algorithme est digne d'être utilisé si il a une valeur supérieure ou égale à 1.

3.4 Complétude de carte

La construction de carte est une tâche étroitement couplée à l'exploration. La complétude de la carte est l'un des problèmes principaux [5, 10, 17]. Mesurer la complétude nécessite une connaissance préalable du terrain exploré et cartographié. Nous définissons la complétude de carte comme le rapport entre la superficie de zone explorée M et la superficie totale de la carte réelle P :

$$completudeDeCarte = \frac{M}{P} \qquad (3)$$

3.5 Qualité de carte

Construire une carte exacte par des robots autonomes est encore un problème ouvert. Les raisons des erreurs d'une carte peuvent être la précision des capteurs ou les algorithmes utilisés pour *Simultaneous Localization And Mapping (SLAM)*. Pour identifier ces erreurs, nous avons besoin d'une carte réelle du terrain. Comme la carte quadrillée d'occupation [2] est largement utilisée pour représenter l'espace inconnu, occupé et libre dans l'exploration, nous définissons d'abord l'erreur de carte comme le nombre de cellules c dans la carte explorée qui ont une valeur différente de la cellule p correspondante dans la carte réelle :

$$erreurDeCarte = \sum_{i} c_i, c_i \neq p_i \qquad (4)$$

où c_i et p_i sont les cellules avec l'indice i dans la carte 2D.

Les résultats obtenus de cette formule sont affectés par la résolution de la carte. Une exigence de haute résolution crée un grand nombre de cellules. En utilisant les mêmes capteurs et algorithmes, l'erreur est susceptible d'être plus importante dans une carte à haute résolution que dans une faible. Une bonne performance d'exploration doit afficher un compromis entre l'erreur de carte et sa résolution.

Contrairement à la métrique de qualité de carte définie dans [5, 17] qui se concentre principalement sur la complétude de la carte construite, nous sommes préoccupés par sa précision topologique.

Nous ensuite la qualité de carte comme le rapport du chevauchement de la carte exploré et la

2. http://en.wikipedia.org/wiki/Occupancy_grid_mapping

carte réalité de terrain à la superficie totale de la carte réalité de terrain P :

$$qualiteDeCarte = \frac{M - A(erreurDeCarte)}{P} \quad (5)$$

où M est la surface de la zone explorée totale en mètres carrés, et $A(erreurDeCarte)$ est la superficie occupée par les cellules d'erreur.

4 Paramètres

Pour évaluer un algorithme pour l'exploration collaborative ou pour comparer plusieurs d'entre eux, il faut choisir l'environnement à explorer et les robots à utiliser. Cela est une spécification du benchmark. Différentes parties d'une telle spécification peuvent varier et avoir un impact sur les performances d'exploration. En effet, la modification de ces paramètres peut affecter de manière significative une ou plusieurs métriques. Nous énumérons ci-dessous, les paramètres que nous avons identifiés, regroupés en trois familles : *Robot*, *Flotte*, and *Environnement*.

Notre objectif est de fournir à la communauté un cadre de référence pour définir les configurations de benchmark. Chaque configuration correspond à une combinaison de paramètres et définit donc un plan d'expérience. Cette idée a déjà été adoptée dans d'autres domaines tels que les bases de données d'images pour la reconnaissance d'objets. Elle a été partiellement abordée dans la *RoboCup Rescue* [3] avec différentes arènes.

4.1 Robot

– Propriétés de locomotion. Elles couvrent les caractéristiques du robot telles que le modèle de mouvement (holonome ou non).
– Capacités de calcul : CPU, RAM, fréquence d'horloge. Lors du choix d'un algorithme d'exploration, il faut prendre en considération les ressources disponibles pour le calcul.
– La précision, la fréquence et la portée des capteurs. Les caractéristiques des capteurs influent sur la localisation et la construction de carte, et peuvent avoir un impact sur la qualité de la carte construite.

3. http://wiki.robocup.org/wiki/Robot_League#Field_Description

4.2 Flotte

– Nombre de robots. Intuitivement on pourrait penser qu'avoir plus de robots peut conduire à l'exploration plus rapide. Mais, cela dépend en fait de la stratégie de coordination.
– Homogénéité de la flotte. L'utilisation d'une flotte hétérogène comme l'association de robots aériens et de robots terrestres pourrait améliorer la performance d'exploration.
– Positions initiales des robots. Selon l'environnement et les obstacles, la performance d'exploration peut être significativement affectée par la position des robots au démarrage de l'exploration [20].
– Bande passante des canaux de communication. Certains algorithmes nécessitent que les robots échangent de grandes quantités de données. Leur performance pourrait baisser considérablement lors de l'utilisation des robots avec des interfaces réseau qui offrent une bande passante limitée.
– Portée de communication. L'exploration multi-robots nécessite souvent une communication habituellement réalisée par des réseaux sans fil. Selon le réseau sans fil utilisé, la portée de communication peut varier. Cette portée influence la collaboration et la performance d'exploration. En effet, sur de grands terrains ou en fonction des densités et des matériaux d'obstacles, les transmissions sans fil pourraient être ralenties. Les robots pourraient être déconnectés et être incapables de communiquer ou de coopérer. Toutefois, cette question peut être atténuée par la prise en compte de la connectivité du réseau dans la planification [7, 14].

4.3 Environnement

– Taille du terrain. A nombre de robots constants, l'exploration d'un grand terrain nécessite généralement plus de temps que pour un petit terrain.
– Densité et forme des obstacles. Dans un environnement avec beaucoup d'obstacles, il y a moins d'espace à explorer. Cependant, la navigation peut être plus compliquée, surtout avec des obstacles concaves où les blocages peuvent survenir lorsque plusieurs robots sont situés dans la même zone.
– Morphologie géographique. L'exploration d'un terrain avec un seul grand espace prend sans doute moins de temps qu'un environnement décomposé en un certain nombre d'espaces ouverts reliés avec des couloirs étroits. Dans ce dernier, il est probable que

les robots se gênent.
– Dynamicité. Si l'environnement change (*e.g.* effondrements de construction) ou qu'il y a d'autres entités mobiles (*e.g.* sauveteurs humains ou d'autres robots), le temps d'exploration et les coûts associés peuvent varier entre différentes expérimentations.

5 Illustration avec des simulations

Pour illustrer nos métriques et nos paramètres de benchmark, nous avons effectué une série de simulations en utilisant la stratégie d'exploration multi-robots de Yamauchi basée sur les frontières [19]. Dans cette stratégie décentralisée, chaque robot décide de manière autonome où aller en fonction de sa carte. L'échange de la carte est la seule tâche coopérative. Une fois qu'un robot met à jour sa carte, il sélectionne la frontière la plus proche et se déplace vers elle. Nous évaluons l'impact de deux algorithmes de fusion de carte sur la performance d'exploration.

L'expérimentation est réalisée avec différentes tailles de flotte allant de 1 à 30 robots, et quatre terrains présentés dans la figure 1. Ces terrains, inspirés par la *RoboCup Rescue*, ont la même taille, mais différentes surfaces d'exploration :

1. Le terrain *loop* a une densité d'obstacles faible et une forme de rocade sans aucune fourche.
2. Le terrain *cross* correspond à un carrefour. Ilcontient cinq fourches de la route mais la densité d'obstacles est faible.
3. Le terrain *zigzag* n'a pas de bifurcation mais contient plus d'obstacles. En outre, il a un long chemin à planifier pour le robot.
4. Le terrain *maze* est le plus complexe qui contient de nombreux obstacles et impasses.

5.1 Banc d'essai

Afin de faciliter une comparaison analytique des différents algorithmes dans des conditions différentes, nous avons construit un banc d'essai pour la collecte des données. La figure 2 illustre l'architecture de notre banc d'essai. Il consiste en :
– Le simulateur robotique MORSE [8] fournit un moteur physique réaliste permettant la simulation en 3D.
– ROS méta-système [16] est utilisé pour construire le logiciel de commande du robot.

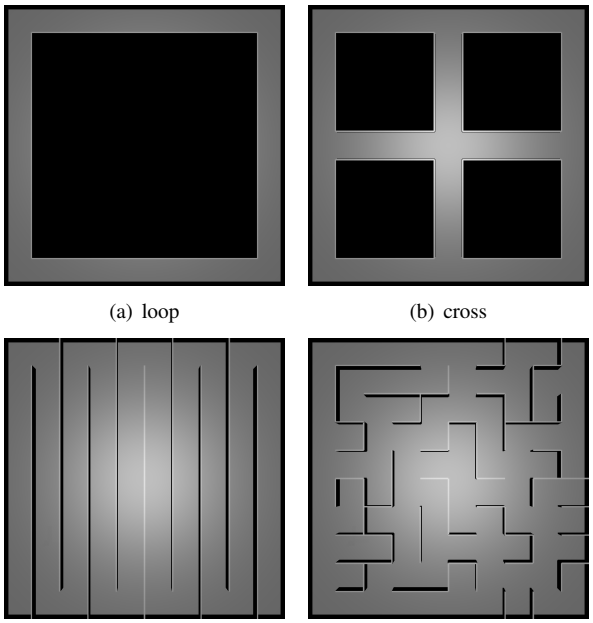


FIGURE 1 – Terrains créés pour le benchmarking de performance.

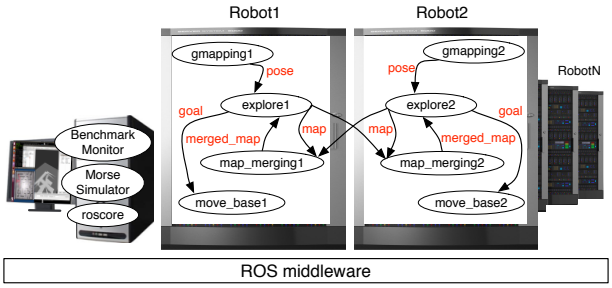


FIGURE 2 – Architecture de banc d'essai.

– Une grappe de serveurs est utilisé pour fournir l'infrastructure de calcul distribué pour répondre aux exigences de la simulation des robots.

Chaque contrôleur de robot intègre aux paquetages ROS que nous avons développés d'autres pré-existants. Les plus importants sont représentés sur la figure 2 et décrits ci-dessous.

– *gmapping* : Il effectue un SLAM à base de laser [11]. Nous l'utilisons pour extraire la position du robot, qui alimente le package *explore*.
– *explore* : Il met en œuvre la stratégie de Yamauchi pour l'exploration basée sur les frontières.
– *map_merging* : Il fusionne la carte de robots en connaissant les positions initiales relatives du robot.
– *move_base* : Il met en œuvre l'algorithme de Dijkstra pour la planification de chemin.

Nous avons publié nos packages développés sur notre site Web à l'adresse `http://car.mines-douai.fr/`. Les packages sont en *open-source* avec l'intention de fournir à la communauté un système reproductible, afin d'accélérer les comparaisons de résultats. Nous avons également évalué notre système basé sur ROS avec deux robots robuLAB-10 pour l'exploration intérieure. Une vidéo avec ces deux robots réels est également disponible sur notre site Web.

5.2 Algorithmes de fusion de cartes

Nous sommes intéressés par l'impact sur la performance d'exploration de deux algorithmes de fusion de cartes respectivement utilisés dans [19] et [4]. Le premier [19] (Algorithme 1) est un algorithme glouton qui se concentre simplement sur l'espace inconnu. Le deuxième [4] (Algorithme 2) est un algorithme probabiliste dans lequel le robot construit la carte fusionnée en calculant la probabilité pour que chaque cellule dans la carte explorée soit occupée. Ces deux algorithmes nécessitent la connaissance des positions initiales des robots.

Algorithme 1 Algorithme Glouton

1: $\delta \leftarrow PoseInitialeRelative(robot_a, robot_b)$
2: **for all** $cellule$ in $robot_a.carte\_quadrillee\_explore$ **do**
3: **if** $cellule == ESPACE\_INCONNU$ **then**
4: $cellule \leftarrow robot_b.carte\_quadrillee\_explore(coordonnee(cellule) + \delta)$
5: **end if**
6: **end for**

Algorithme 2 Algorithme Probabiliste

1: $\delta \leftarrow PoseInitialeRelative(robot_a, robot_b)$
2: **for all** $cellule$ in $robot_a.carte\_quadrillee\_explore$ **do**
3: $Probabilite(cellule) \leftarrow$ *nouvelle probabilite selon les formules dans* [4].
4: **if** $cellule \geq LIMITE\_DE\_CONFIANCE$ **then**
5: $cellule \leftarrow ESPACE\_OCCUPE$
6: **else**
7: $cellule \leftarrow ESPACE\_LIBRE$
8: **end if**
9: **end for**

5.3 Paramètres fixés

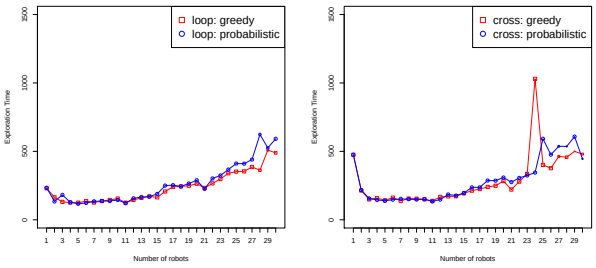
Le Tableau 1 reprend les paramètres pour lesquels nous avons choisi des valeurs fixées pour nos expériences. Nous pouvons voir qu'une équipe homogène de robots Pioneer 3-DX avec 2 processeurs et 2 Go de mémoire est simulée. Chaque robot est équipé d'un laser SICK LMS500 qui fournit 180 points d'échantillonnage avec 180 degrés de champ de vision et une portée maximale de 30 mètres. Conformément au robot réel, la vitesse maximale de chaque robot simulé a été fixée à 1,2 mètre par seconde pour un mouvement linéaire et 5,24 radian par seconde pour un mouvement de rotation. Un bruit a été ajouté aux données d'odométrie. L'écart type est de 0,022 mètres pour le bruit de position (x et y) et 0,02 radians pour le bruit de rotation. Ce bruit est proche de celui du vrai robot Pioneer 3-DX.

TABLE 1 – Les paramètres fixés dans l'expérience de référence.

Robot	
Capacité de calcul	2 processeurs, 2Go de mémoire
Vitesse maximale	1,2m/s, 5,24rad/s
Bruit d'odométrie	0,022m, 0,02rad
Télémètre laser	SICK LMS500
Flotte	
Homogénéité	homogène (Pioneer 3-DX)
Positions initiales des robots	à gauche, de haut en bas, tous les 2 mètres
Débit réseau	1 Gigabit/seconde
Portée de communication	200 mètres
Environnement	
Taille du terrain	80 mètres × 80 mètres
Hauteur d'obstacle	2 mètres
Largeur de couloir	8 mètres

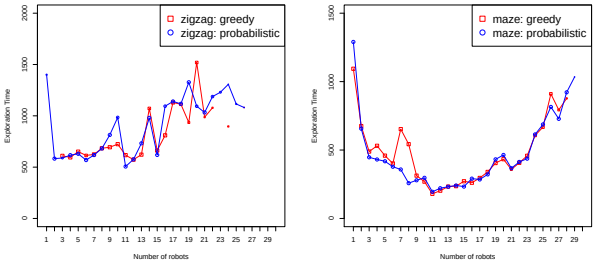
Les robots sont initialement placés le long d'une ligne verticale à partir du coin en haut à gauche vers le coin en bas à gauche du terrain. La distance entre les positions initiales des robots est réglée à 2 mètres. Les robots communiquent entre eux à travers un réseau dont le débit est 1 Gigabit par seconde. La portée maximale de communication entre les robots est fixée à 200 mètres en fonction de leur position relative dans l'environnement simulé. L'impact des obstacles sur la communication est ignoré. Ce réglage ne permet pas une comparaison des algorithmes avec une communication réaliste. Mais il n'a pas d'impact sur notre évaluation, car nous évaluons des algorithmes de fusion de carte. En fait, ce réglage est une conséquence directe du fait que le simulateur MORSE que nous avons utilisé ne supporte pas cette fonction. Néanmoins, nous avons prévu d'aborder ce point important dans nos futurs travaux et ajouter différents modèles de communication à notre banc d'essai.

Tous les terrains font une taille de 80 mètres de long et 80 mètres de large. La hauteur des obstacles est réglé sur 2 mètres et la largeur des couloirs est fixée à 8 mètres.



(a) *loop* terrain

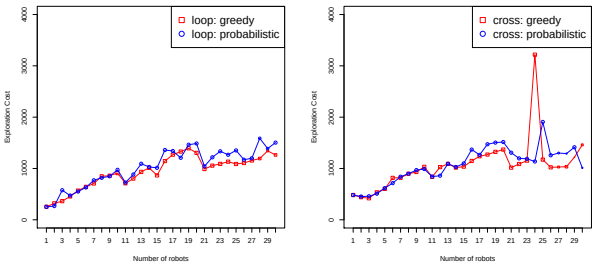
(b) *cross* terrain



(c) *zigzag* terrain

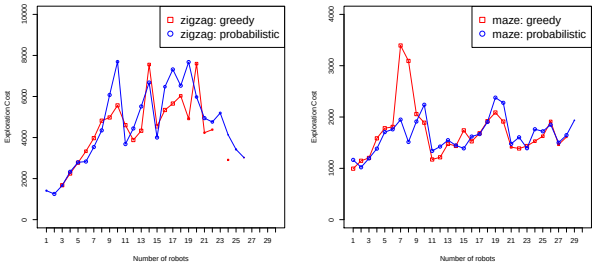
(d) *maze* terrain

FIGURE 3 – Résultats avec le métrique *temps d'exploration*.



(a) *loop* terrain

(b) *cross* terrain



(c) *zigzag* terrain

(d) *maze* terrain

FIGURE 4 – Résultats avec le métrique *coût d'exploration*.

5.4 Résultats

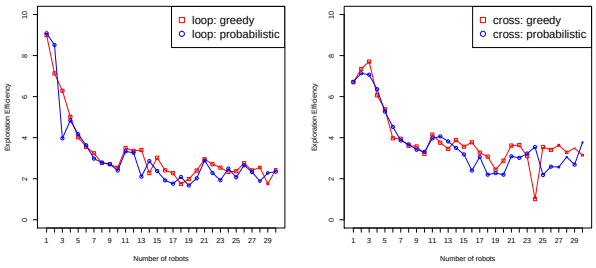
Les résultats sont représentés sur les figures 3, 4, 5 et 6. Quatre métriques sont sélectionnées pour les tests de référence. Nous n'avons pas mesuré la *complétude de carte*, puisque c'ést une condition préalable pour comparer les différents algorithmes de fusion de carte. Chaque figure contient quatre graphes, correspondant chacun à un terrain. Dans chaque graphe, l'abscisse représente le nombre de robots dans la flotte, et l'ordonnée indique les mesures de métriques. Le carré rouge représente l'algorithme glouton, et le cercle bleu représente l'algorithme probabiliste.

Comme, (1) Il existe des composants non-déterministes dans l'environnement simulé, tels que les bruits de scan de laser et d'odométrie ; (2) l'algorithme mis en oeuvre pour SLAM est un algorithme probabiliste ; (3) nous utilisons une grappe de serveurs partagée avec une bande passante variable suivant la charge ; nous avons effectué cinq essais pour chaque taille de flotte, et affiché la valeur moyenne de ces essais. Un moniteur est déployé sur une station de travail, qui termine chaque essai lorsque 99% du terrain est découvert (essai réussi) ou lorsque le temps d'exploration dépasse 2000 secondes (essai échoué). Une vidéo complète d'un essai d'exploration est disponible sur notre site Web.

La valeur moyenne n'est pas affichée si les cinq essais ont échoué. Ces situations se produisent par exemple lorsque la taille de la flotte est supérieur à 26 robots dans le terrain *zigzag*. Les causes de l'échec peuvent provenir de deux aspects : l'utilisation de contrôleurs de robot basé sur ROS et l'incertitude de trafic réseau :

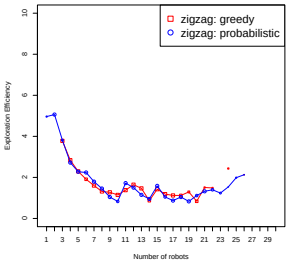
– ROS est un système basé sur une architecture distribuée encore en développement pour la partie multi-robots. La perte de localisation de robot et l'échec de planification de chemin de longue distance de temps en temps sont occasionnellement apparus dans nos expériences.
– Comme mentionné précédemment, la grappe de serveurs est à partagée. Lorsque l'expérience est exécutée pendant les périodes de pointe du réseau, un essai peut échouer, ou déboucher sur des valeurs aberrantes (*cf.* 24 robots dans le terrain *cross* mettant en oeuvre l'algorithme glouton en termes de *temps d'exploration*, *coût d'exploration*, et *efficacité d'exploration*).

Les figures montrent que les différences de résultats entre l'algorithme glouton et l'algorithme probabiliste ne sont pas significatives en termes de *temps d'exploration*, *coût d'exploration*, et *efficacité d'exploration*. En général, le *temps d'exploration* et le *coût d'exploration* augmentent et l'*efficacité d'exploration* diminue lorsque le nombre de robots augmente, sauf pour le terrain *maze*. La raison principale est que plus il y a de robots plus il faut de *temps* pour éviter la collision avec les autres. Ainsi, les robots doivent généralement replanifier leurs

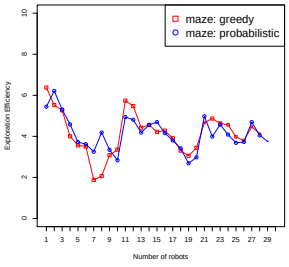


(a) *loop* terrain

(b) *cross* terrain

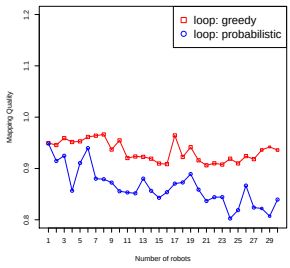


(c) *zigzag* terrain

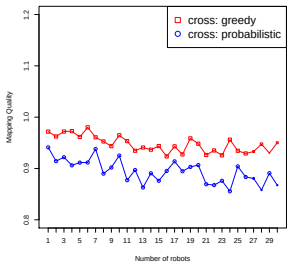


(d) *maze* terrain

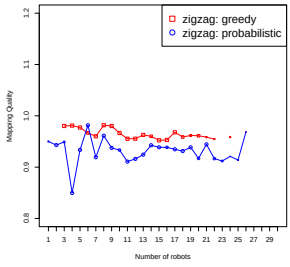
FIGURE 5 – Résultats avec la métrique *efficacité d'exploration*.



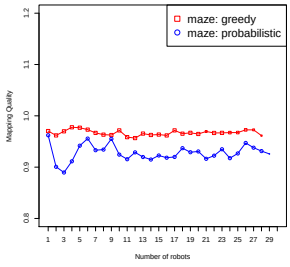
(a) *loop* terrain



(b) *cross* terrain



(c) *zigzag* terrain



(d) *maze* terrain

FIGURE 6 – Résultats avec la métrique *qualité de carte*.

trajectoires, le *coût d'exploration* est donc augmenté. Comme la simulation s'arrête lorsque 99% de la superficie totale a été explorée, alors le terme M dans l'expression de l'*efficacité d'exploration* est une valeur constante. Ainsi l'*efficacité d'exploration* est inversement proportionnelle au *coût d'exploration*. Un aspect intéressant de ces résultats est que la taille optimale de la flotte de robot peut être évaluée pour un terrain donné. Par exemple, lors de l'exploration du terrain *maze*, la flotte idéal est de 11 robots. Effectivement, cette taille de la flotte minimise le *temps d'exploration* et le *coût d'exploration* tout en assurant une bonne *efficacité d'exploration*.

La figure 6 montre que l'algorithme glouton et l'algorithme probabiliste influencent la qualité de la carte. Ce résultat indique que ces deux algorithmes n'impactent significativement que la métrique qualité de la carte.

En outre, les résultats montrent que, avec le terrain *zigzag*, la position initiale des robots influence largement les mesures de performance. Les simulations ont montré que l'exploration est principalement effectuée par un seul robot. En effet, il n'y a qu'une seule frontière et elle est toujours proche du même robot.

Basé sur ces résultats expérimentaux, il est clair que la taille de la flotte de robot et la forme de terrain ont un fort impact sur le *temps d'exploration*, le *coût d'exploration* et l'*efficacité d'exploration*, tandis que l'algorithme de fusion de carte a un impact sur la *qualité de carte*.

6 Conclusions

Dans cet article, nous avons considéré le benchmarking de performance pour l'exploration multi-robots. Notre préoccupation est de savoir comment comparer quantitativement différents algorithmes et d'effectuer une évaluation objective sur des paramètres prédéfinis communs. Il n'est pas facile de répondre à cette question en raison de la complexité des systèmes multi-robots et des environnements qu'ils explorent.

Pour résoudre ce problème, nous avons sélectionné cinq métriques pour quantifier la performance d'exploration, à savoir : le *temps d'exploration*, le *coût d'exploration*, l'*efficacité d'exploration*, la *complétude de carte*, et la *qualité de carte*. Ces métriques peuvent être utilisées dans les systèmes simulés et réels grâce à l'utilisation de ROS.

Nous avons également identifié plusieurs paramètres qui influent sur la performance de l'exploration. Nous considérons ces paramètres comme une première étape vers la définition d'environnements standards et des plans d'expérience de référence qui peuvent être partagés par la communauté de recherche.

Pour illustrer notre contribution, nous avons comparé deux algorithmes de fusion de cartes

utilisés pour la stratégie d'exploration multi-robots basée sur les frontières de Yamauchi. Nous nous sommes appuyés sur des simulations effectuées à l'aide du simulateur 3D MORSE, avec des contrôleurs de robot basés sur ROS. Alors que la plupart des paramètres avaient des valeurs fixes, nous avons fait varier quelques-uns d'entre eux : le nombre de robots et les dispositions de terrain. En utilisant les métriques définies, nous avons montré ainsi l'impact de différents algorithmes sur la performance d'exploration.

Dans le futur, nous aimerions définir un ensemble de valeurs de référence pour les paramètres que nous avons identifiés. Notre objectif est de fournir à la communauté une base de données qui sera utilisée pour comparer des algorithmes. La prochaine étape sera d'utiliser cette base de données pour comparer les algorithmes existants. Cela devrait finalement nous donner un aperçu sur comment comparer les solutions existantes, et quel algorithme devrait être utilisé pour un problème particulier.

Une autre direction pour les travaux futurs sera de réaliser des simulations encore plus réalistes en introduisant un système de gestion de propagation des ondes radio et d'absorption des obstacles dans le simulateur.

Remerciements

Ce travail fait partie du projet Sucré qui est soutenu par Région Nord Pas-de-Calais.

Références

[1] Francesco Amigoni. Experimental evaluation of some exploration strategies for mobile robots. In *Proceedings of ICRA'08*, pages 2818–2823, 2008.

[2] Antoine Bautin, Olivier Simonin, and François Charpillet. MinPos : A novel frontier allocation algorithm for multi-robot exploration. In *Proceedings of ICIRA'12*, pages 496–508, 2012.

[3] Fabio Bonsignorio, Elena Messina, and Angel P. del Pobil. Fostering progressin performance evaluation and benchmarking of robotic and automation systems. *Robotics and Automation Magazine*, 21(1) :22–25, march 2014.

[4] W. Burgard, M. Moors, D. Fox, R. G. Simmons, and S. Thrun. Collaborative multi-robot exploration. In *Proc. ICRA'00*, pages 476–481, April 2000.

[5] Micael S. Couceiro, Patricia A. Vargas, Rui P. Rocha, and Nuno M.F. Ferreira. Benchmark of swarm robotics distributed techniques in a search task. *Robotics and Autonomous Systems*, 62(2) :200–213, February 2014.

[6] Angel P. del Pobil. Why do we need benchmarks in robotics research ? In *Proceedings of IROS'06*, Beijing, China, 2006.

[7] Arnaud Doniec, Noury Bouraqadi, Michael Defoort, Van Tuan Le, and Serge Stinckwich. Distributed constraint reasoning applied to multi-robot exploration. In *Proceedings of ICTAI'09*, pages 159–166, 2009.

[8] Gilberto Echeverria, Séverin Lemaignan, Arnaud Degroote, Simon Lacroix, Michael Karg, Pierrick Koch, Charles Lesire, and Serge Stinckwich. Simulating complex robotic scenarios with morse. In *proceedings of SIMPAR'12*, pages 197–208, 2012.

[9] Jan Faigl, Olivier Simonin, and François Charpillet. Comparison of task-allocation algorithms in frontier-based multi-robot exploration. In *Proceedings of EUMAS'14*. Springer.

[10] Sebastian Frank, Kim Listmann, Dominik Haumann, and Volker Willert. Performance analysis for multi-robot exploration strategies. In *proceedings of SIMPAR'10*, pages 399–410. Springer, 2010.

[11] Giorgio Grisetti, Cyrill Stachniss, and Wolfram Burgard. Improved techniques for grid mapping with rao-blackwellized particle filters. *Robotics, IEEE Transactions on*, 23(1) :34–46, 2007.

[12] A. Howard. Multi-robot simultaneous localization and mapping using particle filters. *The International Journal of Robotics Research*, 25 :1243–1256, December 2006.

[13] Robert N Lass, Evan A Sultanik, and William C Regli. Metrics for multiagent systems. In *Performance Evaluation and Benchmarking of Intelligent Systems*, pages 1–19. Springer, 2009.

[14] Van Tuan Le, Noury Bouraqadi, Serge Stinckwich, Victor Moraru, and Arnaud Doniec. Making networked robot connectivity-aware. In *Proceedings of ICRA'09*, Kobe, Japan, May 2009.

[15] Lynne E Parker. Multiple mobile robot systems. *Springer Handbook of Robotics*, pages 921–941, 2008.

[16] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng. ROS : an open-source robot operating system. In *Proc. IEEE ICRA'09 Workshop on Open Source Software*, Kobe, Japan, May 2009.

[17] Chris Scrapper, Raj Madhavan, Rolf Lakaemper, A Censi, A Godil, A Wagan, and A Jacoff. Quantitative assessment of robot-generated maps. In *Performance Evaluation and Benchmarking of Intelligent Systems*, pages 221–248. Springer, 2009.

[18] Cyrill Stachniss. *Robotic Mapping and Exploration*. Springer, 2009.

[19] B. Yamauchi. Frontier-based exploration using multiple robots. In *Proceedings of Agent'98*, 1998.

[20] Zhi Yan, Luc Fabresse, Jannik Laval, and Noury Bouraqadi. Team size optimization for multi-robot exploration. In *Proceedings of SIMPAR'14*, Bergamo, Italy, October 2014.

[21] Zhi Yan, Nicolas Jouandeau, and Arab Ali Cherif. A survey and analysis of multi-robot coordination. *International Journal of Advanced Robotic Systems*, 10, December 2013.

[22] R. Zlot, A. Stentz, M.B. Dias, and S. Thayer. Multi-robot exploration controlled by a market economy. In *Proceedings of ICRA'02*, volume 3, pages 3016–3023, 2002.

Un simulateur multiagent de trafic coopératif

M. Guériau[a,b] maxime.gueriau@ifsttar.fr
R. Billot[b] romain.billot@ifsttar.fr
F. Armetta[a] frederic.armetta@univ-lyon1.fr
S. Hassas[a] salima.hassas@univ-lyon1.fr
N-E. El Faouzi[b] nour-eddin.elfaouzi@ifsttar.fr

[a] Laboratoire d'InfoRmatique en Image et Systèmes d'information, Université de Lyon, France

[b] Laboratoire d'Ingénierie CIrculation Transports, IFSTTAR-ENTPE, Université de Lyon, France

Résumé

Cet article présente un modèle multiagent de trafic coopératif et son implémentation, vue comme une extension du simulateur de trafic MovSim. La dynamique physique des véhicules est décrite avec le point de vue du domaine de la théorie du trafic en proposant un modèle microscopique, multi-anticipatif et bilatéral de suivi de véhicules. Ce modèle est enrichi par des paradigmes agent qui influencent le comportement des véhicules grâce aux mécanismes de perception, de communication et au concept de confiance. Les résultats en simulation, utilisant des données réelles, mettent en valeur l'impact positif des véhicules connectés sur l'homogénéisation du flux de trafic, même en présence de capteurs défaillants. Le travail présenté se positionne aussi comme un outil d'aide à la décision pour le déploiement futur de véhicules coopératifs et de leur infrastructure, et pour la conception de stratégies de contrôle de tels systèmes.

Mots-clés : *Systèmes coopératifs, Simulation multiagent, Modélisation du trafic, Véhicules connectés*

Abstract

The paper presents an agent-based cooperative traffic model implemented as an extension of the Multi-model Open-source Vehicular-traffic SIMulator (MovSim). Connected vehicles dynamics are described from a traffic flow theory standpoint using a bilateral multi-anticipative microscopic model. Then, agent-based concepts allow to influence vehicles dynamics through perception and trust-based mechanisms. Sensors failures and information reliability are well captured by trust concepts. This framework is implemented into the simulator and tested with real data. Simulation results shed light on how connected vehicles improve traffic flow homogenization and ensure safety even with unreliable sensors. The proposed tool presents the potential to act as a decision support tool within the future deployment of connected vehicles and the design of cooperative traffic management strategies.

Keywords: *Cooperative Systems, Multiagent simulation, Traffic modeling, Connected Vehicles*

1 Introduction

Les Systèmes Coopératifs de Transport Intelligents (C-ITS) profitent des avancées récentes des technologies de la communication inter-véhiculaire pour permettre une optimisation de l'usage de l'infrastructure routière. Plus généralement, la motivation principale des Systèmes de Transport Intelligents était de résoudre les problèmes de congestion, de plus en plus critiques dans les zones urbaines à forte densité. Les conducteurs sont déjà assistés dans leur mobilité quotidienne : de plus en plus de véhicules sont équipés de Systèmes Avancés d'Assistance à la Conduite. Les nouveaux capteurs et périphériques communicants contribuent au développement des véhicules connectés qui s'avèrent prometteurs pour l'amélioration du flux de trafic. Avant de présenter les verrous scientifiques identifiés dans ce domaine et nos contributions, nous allons clarifier certains termes relatifs au domaine des C-ITS et des véhicules connectés.

1.1 Les Systèmes Coopératifs dans un environnement connecté

Les véhicules connectés sont capables d'échanger des informations avec d'autres véhicules (communication véhicule à véhicule – V2V) et avec les Unités de Bord de Route (UBR, communication de véhicule à infrastructure / d'infrastructure à véhicule – V2I / I2V). Les capteurs embarqués sur les véhicules leur four-

nissent des connaissances leur permettant de construire une représentation locale et partielle de leur environnement de navigation. Dans ce travail, nous nous intéressons à des véhicules connectés et partiellement automatisés, équipés de quelques capteurs et d'un périphérique de communication. Le comportement longitudinal (accélération / freinage) est délégué à une loi coopérative, qui tire profit des informations issues de la perception et de la communication du véhicule. En accord avec les technologies actuelles, les véhicules connectés sont équipés d'un télémètre laser à l'avant (LIDAR [1]), d'un dispositif de localisation (*e.g.* un récepteur GPS) et d'un émetteur/récepteur pour la communication (utilisant le protocole 802.11p [2]), comme illustré par la figure 1.

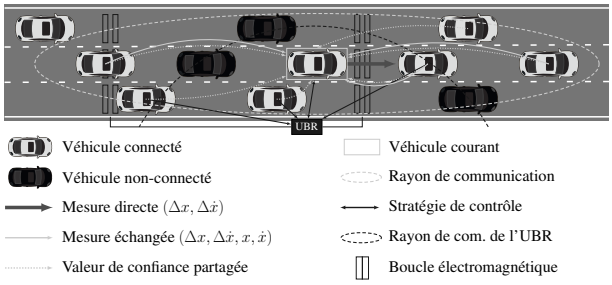


FIGURE 1 – Architecture des Systèmes Coopératifs

Il est attendu que les véhicules connectés permettent d'améliorer le trafic en termes de performance du réseau, de sécurité, de confort ou d'économie d'énergie (*e.g.* émissions de CO2). Ces quatre axes d'actions peuvent être abordés par des stratégies globales ou individuelles de contrôle du trafic. Les véhicules connectés orientent l'étude vers le point de vue individuel, où le contrôle peut se faire à travers la définition de comportements "coopératifs" spécifiques. C'est la raison pour laquelle la syntaxe "véhicule coopératif" est aussi employée pour désigner ces véhicules connectés. La modélisation de ces véhicules doit prendre en compte la complexité des différentes dynamiques qui entrent en jeu dans le système complet : les interactions physiques, la communication et la fiabilité de l'information.

1.2 Travaux connexes

La gestion intelligente du trafic s'appuie sur des systèmes complexes (*i.e.* les Systèmes de Transports Intelligents - ITS) qui peuvent être modélisés suivant le paradigme agent. C'est pourquoi la modélisation multiagent a été largement appliquée au domaine des transports, à toutes les échelles (voir [4] pour une synthèse). A l'échelle microscopique, les concepts agent ont été utilisés pour intégrer la communication dans les ITS [5] ou pour étudier le routage dynamique des véhicules dans un réseau urbain [8, 2]. A l'échelle d'une ville, le contrôle des feux de signalisation et des intersections sont les applications les plus courantes des systèmes multiagents dans les transports, avec une tendance récente pour les intersections coopératives [7]. Cependant, l'utilisation principale de ces concepts se retrouve à l'échelle macroscopique où les agents sont utilisés comme des entités applicatives de l'infrastructure pour la gestion du trafic et pour la conception de systèmes d'aide à la décision pour le contrôle du trafic [12, 27].

Pour les véhicules connectés, l'échelle de modélisation considérée est l'échelle microscopique : les agents sont des véhicules qui communiquent entre eux (V2V) et/ou potentiellement avec l'infrastructure (V2I/I2V). Les approches classiques de modélisation (voir [11, 26] pour une synthèse) ne sont pas adaptées car elles ne prennent en compte ni la communication ni les interactions avec l'infrastructure. Concernant le trafic coopératif autoroutier, l'objectif est d'obtenir une auto-organisation d'agents coopératifs à travers l'homogénéisation du flux de trafic. Une structure multiagent auto-organisationnelle [13] est assez flexible pour atteindre ce but lorsque l'objectif est d'obtenir une forme d'organisation à travers des processus internes et autonomes. Une attention spécifique doit être accordée à l'influence mutuelle qui s'exerce entre les niveaux informationnels et comportementaux [1]. Les systèmes basés sur les concepts de confiance et de réputation [24] sont souvent utilisés à bon escient pour lier ces deux niveaux, puisque le flux d'informations doit être organisé afin d'utiliser l'information de manière efficace et assurer sa fiabilité [3]. Dans le cas d'un trafic autoroutier composé de véhicules connectés non fiables (*i.e.* aux capteurs défaillants), l'objectif collectif est d'atteindre un état de trafic alliant sécurité et comportements homogènes. Ce travail constitue l'amélioration de l'architecture proposée dans [18], où une preuve de concept incluant un modèle basé sur la confiance fut proposée. Le travail présenté dans cet article va plus loin en s'appropriant le modèle de confiance avec la spécificité d'utiliser

1. *LIght Detection And Ranging*
2. aussi appelé *Wireless Access in Vehicular Environments (WAVE)*

une couche de confiance, qui peut être décrite comme la couche d'apprentissage, qui contrôle les dynamiques physiques et de communication.

1.3 Contributions et organisation de l'article

Notre article présente un modèle d'architecture générique appliqué aux véhicules connectés et testé en simulation. Le but est d'évaluer efficacement les bénéfices apportés par la stratégie coopérative présentée, en termes de performance et de sécurité, tout en intégrant la fiabilité de l'information. L'article est organisé comme suit : l'architecture multiagent complète est présentée d'un point de vue conceptuel dans la section 2. Il s'agit d'une approche systémique générique capable de réaliser le couplage entre trois dynamiques : dynamique physique (un modèle de suivi de véhicule multi-anticipatif bilatéral qui intègre la communication V2V), dynamique informationnelle (communication inter-véhiculaire V2V) et dynamique de contrôle (la fiabilité de l'information est estimée à travers le concept de confiance). Ensuite, dans la section 3 le détail de l'implémentation originale est présenté sous la forme d'une extension de MovSim [26] (*the Multi-model Open-source Vehicular-traffic SIMulator*). Les résultats des simulations, réalisées à partir de paramètres extraits d'un jeu de données réelles calibrées et de règles de communication réalistes, confirment le potentiel des véhicules connectés pour améliorer le trafic et valident la pertinence de notre approche de modélisation (section 5).

2 Une architecture multiagent en trois couches

Concernant la gouvernance du système, l'objectif est de tirer profit des informations fiables pour influencer les mouvements des véhicules et ainsi homogénéiser le flux de trafic. La figure 2 présente les interactions entre les trois couches. La couche physique représente les règles qui permettent de modéliser le comportement physique des véhicules. La couche de communication gère les échanges d'informations par des règles de proximité et de fiabilité. La couche confiance propose une représentation de la fiabilité de l'information et permet aux agents d'évaluer la confiance qu'ils accordent aux autres. Les couches communication et confiance influencent la couche physique, et donc le comportement des véhicules.

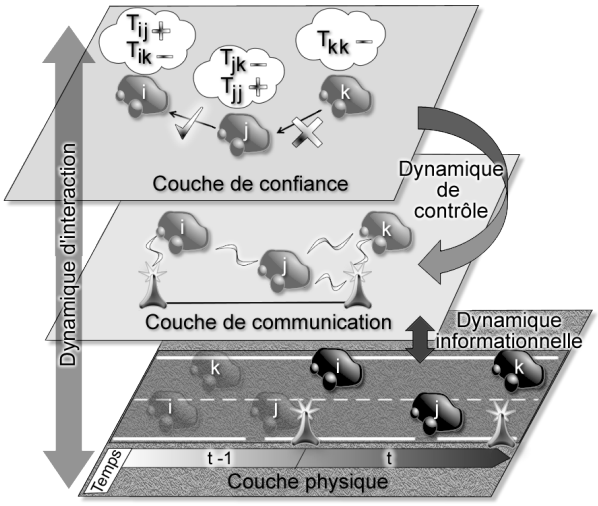


FIGURE 2 – Architecture en 3 couches

2.1 Modèle Physique

Les véhicules connectés peuvent être vus comme des véhicules semi-autonomes dont le comportement longitudinal est délégué à un contrôle par une unité embarquée, et dont le conducteur n'est responsable que des changements de voie. Ainsi, la perception directe des véhicules connectés provient de leurs capteurs embarqués alors que la perception des véhicules classiques (non connectés) provient du conducteur lui-même. Les véhicules connectés sont équipés d'un dispositif de localisation et d'un télémètre laser à l'avant (capable de mesurer l'inter-distance et la vitesse relative de son prédécesseur immédiat) ; cette information est complétée par la communication. La perception des véhicules non connectés est limitée à leur prédécesseur immédiat (distance/vitesse relative), sans capacité de communication.

Ces deux modèles physiques peuvent être décrits comme suit :

Véhicules non connectés : un modèle de suivi de véhicule générique donne l'accélération du véhicule courant i suivant la formule $\ddot{x}_i = f(\dot{x}_i, \Delta x_i, \Delta \dot{x}_i)$ où f est une fonction non linéaire générale, $\dot{x}_i$ la vitesse du véhicule courant, Δx_i l'inter-distance (distance relative au prédécesseur immédiat $i+1$) et $\Delta \dot{x}_i$ la vitesse relative entre les véhicules courant et précédent. Nous avons choisi le modèle IDM (*Intelligent Driver Model*), qui est un modèle sans collisions et capable de reproduire les instabilités d'un flux de trafic [26]. Ce modèle exploite une variable de freinage et une variable d'accélération complétés par des paramètres facilement interpré-

tables. La fonction IDM f_{IDM} s'écrit

$$\ddot{x}_i = a\left[1 - \left(\frac{\dot{x}_i}{v_0}\right)^{\gamma} - \left(\frac{s^*(\dot{x}_i, \Delta\dot{x}_i)}{\Delta x_i}\right)^2\right] \quad (1)$$

où s est l'inter-distance entre le véhicule considéré et son prédécesseur, $s^*(\dot{x}_i, \Delta\dot{x}_i) = s_0 + \max\left(0, \dot{x}_i T + \frac{\dot{x}_i \Delta\dot{x}_i}{2\sqrt{ab}}\right)$ est l'inter-distance désirée, a et b sont respectivement l'accélération maximale et la décélération désirée, v_0 est la vitesse désirée, s_0 est la distance minimale à l'arrêt, T est l'intervalle de sécurité temporel désiré, et γ est un facteur d'agressivité de l'accélération (généralement fixé à 4). De plus, les véhicules peuvent effectuer un changement de voie opportuniste, avant de mettre à jour leur accélération, en suivant la stratégie MOBIL [16]. Cette stratégie consiste à déterminer si un créneau disponible est suffisamment sûr, en estimant le créneau temporel disponible sur la voie adjacente. La stratégie MOBIL modélise le comportement de changement de voie humain et inclut par exemple un facteur de politesse.

Véhicules Connectés : au niveau microscopique, la communication est modélisée par une loi de poursuite multi-anticipative capable de décrire les interactions entre les véhicules (V2V) et d'apporter plus de stabilité au trafic [10]. Nous utilisons le modèle bilatéral multi-anticipatif (BMA [20]), définit comme la combinaison des influences pondérées des véhicules environnants, et qui s'écrit

$$\ddot{x}_i = f_{IDM}\left(\dot{x}_i, \sum_j a_{ij}\Delta x_{i+j}, \sum_j a_{ij}\Delta\dot{x}_{i+j}\right) \quad (2)$$

où a_{ij} est le coefficient d'interaction qui pondère l'information entre l'agent i et l'agent j qu'il perçoit, et f_{IDM} correspond à l'équation 1. Le coefficient d'interaction a_{ij} est calculé à partir de la règle de proximité dans la couche communication et à partir de la représentation de la confiance associée à chaque véhicule :

$$a_{ij} = p_{ij} \cdot T_{ij}, \quad (3)$$

où p_{ij} représente la règle de proximité (eq. 4), et T_{ij} représente la confiance que i a en j (eq. 5). Ces variables sont décrites en détail dans la section 2.2.

La formulation du modèle multi-anticipatif exécuté par les véhicules connectés proposée dans cette approche se base sur IDM (avec lequel nous nous comparons). Le choix de ce modèle pour les véhicules non-connectés est motivé par les résultats validés obtenus dans le cadre d'un scénario de trafic autoroutier [26], reproduit en simulation dans la partie expérimentations (section 5). La stabilité de notre modèle a également fait l'étude d'une analyse [20] dans d'autres conditions et avec différentes lois de poursuites, comme *Optimal Velocity with Relative Velocity* (OVRV [15]).

2.2 Communication

Deux types de messages sont échangés par les véhicules : les messages de mesure et les messages de confiance. A chaque fois qu'un capteur d'un véhicule perçoit une nouvelle information, l'agent correspondant la partage avec les autres véhicules présents dans le rayon de communication de son périphérique communiquant. Les messages de mesure contiennent les valeurs des capteurs (télémètre et localisation) : vitesse et position dans un repère global, position et vitesse relative provenant des véhicules coopératifs voisins. Une fois par pas de simulation, les véhicules connectés utilisent les messages reçus afin d'étendre leur perception en construisant une représentation locale des véhicules de leur voisinage. Cette projection depuis l'environnement global jusqu'à une perception locale est rendue possible par l'architecture multiagent adoptée. Un véhicule connecté peut ensuite estimer et attribuer un poids à chaque véhicule perçu et présent sur la même voie. Cette pondération détermine l'importance que prendra le véhicule j perçu dans la loi multi-anticipative bilatérale (eq. 2). Ainsi, la règle de proximité p_{ij} s'écrit

$$P_{ij} = \frac{|\Delta\dot{x}_{i+j}|}{(\Delta x_{i+j})^{\delta}}, \text{ then } p_{ij} = \frac{P_{ij}}{\sum_{k\in\mathcal{P}_i} P_{ik}}. \quad (4)$$

où $\mathcal{P}_i$ est l'ensemble des agents perçus par i et δ permet d'ajuster l'interaction avant/arrière de l'aspect bilatéral (les valeurs choisies sont respectivement 2 et 0.5 afin d'assurer la stabilité de la file de véhicules et respecter les propriétés physiques du flux de trafic), afin de donner plus d'importance aux véhicules les plus proches et/ou aux véhicules qui présentent une différence de vitesse plus importante. Ainsi, les véhicules connectés peuvent anticiper pour adopter le bon comportement et la sécurité/homogénéité du flux est assurée par la règle de proximité. Dans le cas d'un problème de communication qui empêcherait un véhicule de recevoir tous les messages, le modèle BMA (eq. 2) ainsi sim-

plifié restitue alors exactement le modèle non-coopératif de suivi de véhicule IDM (eq. 1).

2.3 Confiance et fiabilité de l'information

La notion de confiance est liée au concept de délégation qui est un aspect critique dans les Systèmes MultiAgents [6]. Le trafic coopératif autoroutier est un parfait exemple de système basé sur la délégation puisque les actions d'un véhicule dépendent directement des actions des autres véhicules. Il existe de nombreux modèles de confiance et de réputation dans la littérature (voir [23] pour une revue). Nous proposons d'introduire le concept de confiance comme un mécanisme capable de faire face aux problèmes de fiabilité des capteurs. Nous avons adapté le modèle présenté dans [21, 18], qui utilise une structure de données correspondant à un réseau de confiance et nommé *TrustNet* [25]. Ce modèle fournit une représentation numérique de la confiance calculée à partir des expériences non-agrégées acquises par les agents. Dans ce modèle, la confiance des agents correspond à l'opinion de la majorité ; ce qui pourrait être néfaste dans certains systèmes mais pas pour le trafic où la fiabilité moyenne est supposée digne de confiance. Nous ne considérons pas les comportements égoïstes ou de triche dans cette approche. Seule la fiabilité de l'information basée sur les erreurs des capteurs embarqués dans les véhicules a un impact sur la confiance qu'un agent accorde aux autres (et à lui-même). Pendant leur processus de décision, les agents comparent les informations dont ils disposent (encapsulées dans les messages de confiance). Chaque agent i mémorise et met à jour une opinion locale, envers chacun des agents j précédemment perçus, sous la forme d'une valeur de confiance T_{ij} :

$$T_{ij} = \frac{T_{ii} DT_{ij} + IT_{ij}}{T_{ii} + 1}. \qquad (5)$$

Cette valeur de confiance est calculée à chaque pas de temps de simulation t. Elle s'exprime comme une moyenne pondérée de trois composantes $\in [0,1]$:

1. **La confiance directe** DT_{ij} **:** elle représente une erreur de mesure. Cela consiste à évaluer le pourcentage d'erreur induit par les capteurs en termes de position relative, depuis les mesures du véhicule j. Elle s'écrit :

$$DT_{ij} = \frac{|\Delta x_i + x_i - x_j|}{\Delta x_i + x_i}, \qquad (6)$$

où Δx_i et x_i correspondent aux mesures du véhicule courant i (donnés respectivement par son télémètre laser et son périphérique de localisation) ; x_j est la position de j dans l'espace global (donnée par son propre périphérique de localisation).

2. **La confiance indirecte** IT_{ij} **:** elle concerne les valeurs de confiance partagées par les autres agents dans les messages de confiance, en d'autres termes les "témoignages" d'autres agents. Elle s'écrit

$$IT_{ij} = \frac{\Sigma_{k \in \mathcal{W}_i} T_{ik} T_{kj}}{\Sigma_{k \in \mathcal{W}_i} T_{ik}}, \qquad (7)$$

où $\mathcal{W}_i$ est l'ensemble des agents k qui ont partagé leur valeur de confiance en j avec l'agent i (les agents k sont situés dans le rayon de communication de i). Cela implique que k doit avoir précédemment comparé ses mesures directes avec j, ce qui permet de réduire de manière significative la complexité des calculs.

3. **La confiance en soi** T_{ii} **:** elle correspond à la moyenne des valeurs de confiance que les autres agents ont en l'agent i, au temps t ; elle prend donc en compte les valeurs de confiance gardées en mémoire. Elle s'écrit

$$T_{ii} = \frac{\Sigma_{k \in \mathcal{C}_i} T_{kk} T_{ki}}{\Sigma_{k \in \mathcal{C}_i} T_{kk}}, \qquad (8)$$

où $\mathcal{C}_i$ est l'ensemble des agents k qui partagent leur valeur de confiance en i (les agents k sont situés dans le rayon de communication de i). Cette confiance en soi représente la confiance qu'un agent a en les informations qu'il produit.

3 Implémentation

Le modèle présenté dans cet article a été implémenté dans MovSim [26] (*the Multi-model Open-source Vehicular-traffic SIMulator*). Ce simulateur, écrit en Java, est conçu autour d'une architecture générique et propose plusieurs modèles de trafic déjà implémentés. Nous avons réalisé une une extension complète pour rejoindre les concepts agent et intégrer notre architecture de modélisation du trafic coopératif. L'objectif est de garantir aux véhicules une perception locale et partielle de leur environnement (donnée par leurs capteurs respectifs), de modéliser leurs interactions complexes (communication V2V/V2I) et de leur assurer un processus de décision autonome.

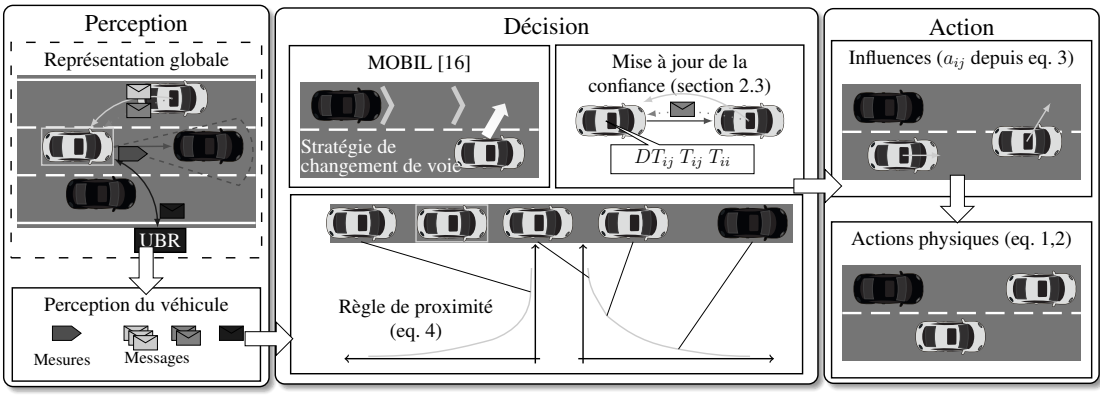


FIGURE 3 – Schéma d'un pas de simulation

3.1 Architecture

L'architecture repose sur l'implémentation des trois couches présentées précédemment (section 2) et s'articule autour d'une boucle perception-decision-action (figure 3). Après une étape d'initialisation qui permet de créer et d'instancier les structures de données avec des paramètres tirés d'un jeu de données calibrées, la boucle principale de la simulation est exécutée. D'abord, pendant l'étape de perception, chaque véhicule perçoit son environnement local et met à jour les valeurs de ses capteurs. Ensuite, tous les véhicules connectés présents dans le rayon de communication du véhicule courant lui communiquent leurs mesures comparables et les valeurs de confiance qu'ils ont calculées antérieurement. Enfin, chaque véhicule connecté met à jour ses valeurs de confiance. Pendant l'étape d'action (qui correspond conceptuellement à la couche physique de la section 2.1), un véhicule peut réaliser un changement de voie (ou non) et mettre à jour son accélération.

3.2 Perception

Capteurs. Les véhicules connectés utilisent les mesures de leurs capteurs de façon altruiste. Ils sont équipés de leurs propres capteurs qui leur fournissent des informations partielles de leur environnement de navigation direct. Étant donné que les capteurs réels peuvent être défectueux, nous avons modélisé leur fiabilité en ajoutant un bruit configurable lors de la perception de chaque capteur. Le bruit de mesure du véhicule i percevant un véhicule j est un bruit blanc Gaussien dont l'écart-type correspond au pourcentage d'erreur du capteur.

Communication. La version originale du simulateur MovSim n'inclut aucune forme de communication ; ces nouvelles formes d'échange d'information ont donc dû être implémentées. Chaque entité du système (ce qui inclut les véhicules mais aussi les Unités de Bord de Route – UBR) est associée à un identifiant unique. Conformément aux spécifications du standard Wi-Fi pour les ITS, nous ne considérons pas les échanges d'informations partiels, les messages modifiés, les pertes ou la duplication d'informations. Tous les messages qui sont envoyés sont reçus par les véhicules équipés qui sont à portée de l'émetteur, après une latence de 100 ms (conformément aux valeurs observées [22]). Nous définissons un message comme un objet contenant les informations suivantes : l'identifiant de l'émetteur, la date d'envoi, la date d'expiration et les données échangées. Deux types de données sont échangées dans deux types de messages :

Message de mesure : contient toutes les mesures capteurs partagées par un véhicule avec ses voisins connectés ;

Message de confiance : contient le triplet (i, j, T_{ij}), *i.e.* chaque valeur de confiance partagée T_{ij} qu'un agent i a en un autre agent j ;

3.3 Décision

Dans un second temps, chaque agent décide d'une action à réaliser immédiatement en réaction à la perception de l'environnement calculée lors de la phase de perception.

D'une perception globale à une perception locale. Les véhicules connectés partagent les mesures de leurs capteurs dans les messages dédiés (position et vitesse relative perçue par leur

télémètre, position et vitesse intrinsèque globale donnée par leur périphérique de localisation). Le modèle d'environnement de la version originale de MovSim (*i.e.* le réseau routier, qui comprend les informations relatives à l'infrastructure et aux véhicules) correspond au formalisme EuroRoad [9]. Ainsi, le modèle d'environnement se compose de deux parties liées qui forment la définition de chaque section routière : une trajectoire linéaire (longueur, connections ...) et la géométrie associée (position, courbure ...). Cette définition détaillée permet donc de déterminer la position d'un véhicule dans l'espace en 2 dimensions en plus de sa position longitudinale sur sa section. Les coordonnées en 2 dimensions sont utilisées pour calculer la perception de chaque véhicule depuis la position du véhicule courant et les caractéristiques propres à chaque capteur. Grâce à cela, le modèle prend en compte les problèmes tels que les virages prononcés ou les routes à plusieurs voies parallèles.

Durant la phase de perception, le véhicule reçoit de manière indirecte la position et la vitesse des véhicules sur la même voie dans son rayon de communication (devant et derrière lui). Une projection est réalisée pour permettre la fusion des données issues de la communication et de la perception. Cette fusion permet de convertir toutes les informations dans un espace local, en exprimant les informations des véhicules perçus dans un repère local propre au véhicule courant. La perception des véhicules est partielle puisqu'ils ne peuvent pas faire la distinction entre un véhicule perçu connecté ou non connecté. Les seules informations qu'ils accumulent grâce aux messages de mesure correspondent à la position et la vitesse des véhicules de leur environnement immédiat.

Des perceptions aux influences. Une fois la perception des véhicules connectés étendue grâce à la communication, chaque véhicule peut pondérer l'importance des véhicules présents dans sa perception en suivant la stratégie coopérative, c'est-à-dire en appliquant les règles de proximité et de confiance. Les agents mémorisent les coefficients calculés sous la forme d'un vecteur d'influence. Cette structure de données encapsule l'information concernant les véhicules perçus (devant et derrière) jusqu'à la dernière étape d'action de la boucle de simulation.

3.4 Action

La dernière étape de l'architecture est en charge du comportement physique des véhicules (cf. couche physique, détaillée en section 2.1). La position et la vitesse des véhicules est modifiée pour correspondre aux valeurs nouvellement calculées, après application du comportement propre à chaque véhicule (étape de décision). Les véhicules non connectés appliquent une loi de poursuite classique IDM, eq. 1), alors que les véhicules connectés exhibent un comportement coopératif plus complexe (BMA, eq. 2).

4 Indicateurs de performance

Afin d'évaluer la performance du modèle, il est nécessaire de sélectionner des indicateurs pertinents. Dans cette approche, l'objectif est de montrer que le modèle coopératif en trois couches et capable d'améliorer significativement le flux de trafic en termes d'homogénéité et de sécurité. Un trafic homogène peut être facilement mis en évidence en traçant les trajectoires et les vitesses des véhicules dans un diagramme espace-temps. Pour la sécurité du trafic, nous avons choisi l'indicateur *Post-Enroachement Time* (PET). Il correspond à la différence de temps entre deux véhicules passant par une même zone spatiale. Ainsi, le PET traduit un risque direct de collision (qui survient lorsque le PET prend une valeur de 0 secondes). Cet indicateur est par exemple utilisé pour évaluer la sécurité des systèmes automatiques de régulation de vitesse à contrôle de distance [17].

5 Résultats expérimentaux

Le trafic simulé consiste en un trafic mixte (*i.e.* composé à la fois de véhicules connectés et non connectés) qui évoluent sur une portion rectiligne d'une autoroute péri-urbaine à trois voies de circulation pendant 20 minutes. La densité sur la section est initialisée à 40 véhicules/km qui circulent à 90 km/h, ce qui correspond à une situation de trafic dense où des ondes de congestion sont susceptibles de se former et de se propager dans le flux. Le débit d'entrée est maintenu à 1800 véhicules/heure/voie pour s'assurer que la congestion ne se réduira pas à cause d'une diminution de la demande. Tous les paramètres du modèle physique sont générés aléatoirement depuis un jeu de paramètres calibrés (nous invitons le lecteur à consulter [19] où la méthodologie de calage détaillée et le

jeu de données initial sont présentés, accompagnés d'une analyse statistique de la distribution des paramètres). Les objectifs des simulations sont d'évaluer l'influence du taux de véhicules connectés et celui de la fiabilité de l'information échangée. Nous avons mis en place deux expérimentations :

Expérimentation 1 : influence de la stratégie coopérative sur l'homogénéité du trafic.

Expérimentation 2 : validation de l'utilisation de la confiance pour évaluer la fiabilité des capteurs. Comparaison sans la confiance et avec le modèle FIRE.

5.1 Expérimentation 1

En traçant les trajectoires et les vitesses des véhicules, la formation, la propagation et la diminution des ondes de congestion sur la section simulée sont clairement visibles. Nous souhaitons montrer l'impact de l'introduction d'une faible proportion de véhicules connectés dans un flux composé principalement de véhicules non connectés.

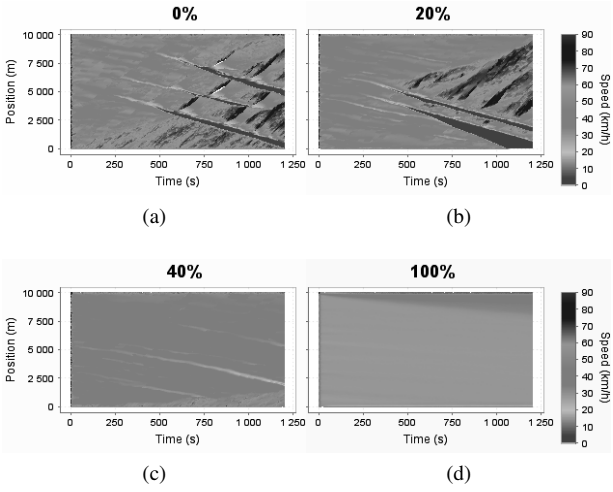


FIGURE 4 – Trajectoires et vitesses sur la voie de droite, pour plusieurs fractions de véhicules connectés parmi des véhicules non connectés (a : 0%, b : 20%, c : 40% et d : 100%).

Les résultats, présentés dans des diagrammes espace-temps (figure 4) montrent que même un faible pourcentage de véhicules connectés (environ 40%, fig. 4.c) permet de prévenir la propagation des ondes de congestion. Le cas 0% (fig. 4.a) correspond à la simulation témoin et reflète le comportement d'un flux composé intégralement de véhicules non connectés. Les portions en rouge représentent des véhicules arrêtés sur la chaussée (vitesse nulle) alors que les portions homogènes en vert témoignent de vitesses homogènes. Ainsi, les résultats montrent l'impact de la stratégie coopérative sur l'homogénéisation du flux de trafic, même avec une faible proportion (*e.g.* 20%, fig. 4.b) de véhicules connectés.

5.2 Expérimentation 2

La seconde expérimentation met en évidence l'effet de la confiance sur la gestion des capteurs défaillants. Dans cette simulation, certains véhicules connectés sont équipés de capteurs défaillants qui fournissent des mesures entachées d'erreurs importantes (+/- 50%) pour l'interdistance et la vitesse relative. La figure 5 présente les distributions des valeurs de PET pour trois modèles différents : sans confiance, modèle FIRE et modèle TrustNet. Nous avons choisi de comparer notre approche avec le modèle FIRE [14], car tous deux sont des modèles de confiance calculatoires, adaptés à un calcul par des agents, et qui utilisent un mécanisme sans agrégation pour convertir les interactions entre agents en valeurs de confiance. Les agents qui utilisent FIRE stockent les résultats r_i (représentés sous la forme d'une notation) de leurs expériences passées dans leur mémoire (limitée). Chacun de ses résultats est l'évaluation d'une interaction entre deux agents à un temps donné, ce qui est similaire à la confiance directe de notre modèle lors de l'évaluation de l'erreur de mesure. Nous avons fixé la taille de la mémoire à 6000 résultats d'interaction (ce qui correspond à 10 minutes de temps de simulation avec un pas de temps de 0,1s). Chaque valeur de confiance est calculée à partir de la somme pondérée des expériences passées, et les poids décroissent en fonction du temps de la simulation. L'objectif étant d'accorder plus d'importance aux interactions récentes tout en gardant une mémoire des expériences passées (le module de réputation de FIRE n'est pas implémenté ici). Chaque valeur de confiance T_{ij} s'écrit donc

$$T_{ij} = \sum_{r_i} \omega(r_i) \cdot \nu_i, \qquad (9)$$

où $\omega(r_i) = \exp -\frac{\Delta t(r_i)}{\lambda}$ est la fonction de pondération qui estime la pertinence ou la fiabilité associées à la notation du résultat et ν_i est la valeur de la note associée au résultat r_i. En suivant la description du modèle FIRE [14], nous avons choisi $\lambda = -\frac{60}{\ln(0.5)}$, ce qui implique que

$\omega(r_i) = 1$ après 0 seconde et $\omega(r_i) = 0$ après 10 minutes de simulation (ce qui correspond à la taille maximum de la mémoire de l'agent). Dans

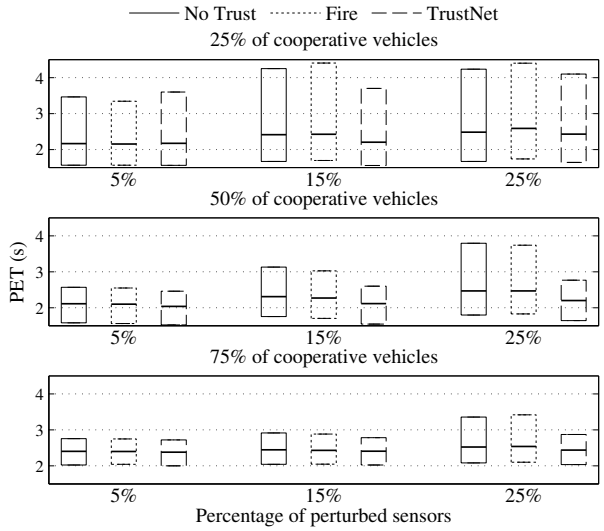


FIGURE 5 – Distributions des PET (médiane, 1er et 9ème déciles) pour trois variantes de modèles de confiance. Du haut vers le bas : augmentation du % de véhicules équipés. De gauche à droite : augmentation du % de capteurs défaillants.

la figure 5, nous rappelons que le pourcentage de capteurs défaillants concerne uniquement les véhicules connectés. Donc, si cette valeur est fixée à 25% avec une proportion de véhicules équipés de 25%, cela veut dire que tous les véhicules connectés sont équipés de capteurs défaillants. Le but pour le trafic coopératif serait d'obtenir des valeurs de PET homogènes et suffisamment importantes (des valeurs trop faibles indiqueraient une sécurité réduite). De haut en bas, la figure 5 révèle une augmentation des performances de notre modèle lorsque le pourcentage de véhicules connecté augmente. De manière plus générale, le modèle coopératif multi-anticipatif bilatéral (BMA) permet une homogénéisation des valeurs de PET, quel que soit le modèle de confiance employé, ainsi qu'une augmentation significative des valeurs de PET (ce qui traduit une sécurité accrue). Les résultats obtenus montrent que la couche de confiance renforce cet effet d'homogénéisation. Notre modèle TrustNet donne de meilleurs résultats que les approches avec lesquelles il est comparé. De gauche à droite, les performances de notre modèle sont d'autant plus visibles, comparées aux autres approches, que la fraction de véhicules équipés de capteurs défaillants augmente. Les résultats obtenus par FIRE dans notre architecture sont tout de même à souligner compte tenu de la relative simplicité de notre implémentation du modèle.

Les résultats présentés montrent l'effet bénéfique du modèle de confiance sur le comportement physique des véhicules lorsque leurs capteurs leur fournissent des informations erronées. Nous souhaitons par la suite compléter ces résultats par une analyse de l'évolution des valeurs de confiance afin d'étudier les phénomènes de percolation (densité de véhicules connectés nécessaire pour que le réseau de confiance soit efficace) et l'effet mémoire (rémanence des valeurs de confiance dans le temps).

6 Conclusion

Nous avons proposé une architecture multiagent permettant de modéliser les interactions complexes qui entrent en jeu dans les systèmes coopératifs de trafic. Grâce à un minimum d'échange d'information, la couche de confiance intégrée devrait faire évoluer le système vers une forme d'auto-organisation résultant en une harmonisation des vitesses. L'originalité de notre approche est d'utiliser des concepts récents issus de la théorie du trafic : la communication inter-véhiculaire a été intégrée à des modèles de trafic classiques à travers le concept de multi-anticipation. Puis, cette couche physique est influencée par des règles de communication utilisant des mécanismes basés sur le concept de confiance. Les résultats montrent que le modèle en trois couches est robuste au manque de fiabilité de l'information. Le modèle présenté a été implémenté en tant qu'extension d'un simulateur de trafic et est comparé avec un modèle de confiance issu de la littérature. Nous projetons par la suite de compléter notre approche par le développement d'une stratégie d'apprentissage par renforcement pour les véhicules connectés à partir des informations perçues par l'infrastructure (UBR, capteurs, etc.) afin de diffuser des consignes complémentaires de régulation.

Remerciements

Ce projet est soutenu par la Région Rhône-Alpes.

Références

[1] Sherief Abdallah and Victor Lesser. Multiagent reinforcement learning and self-organization in a network of agents. In *Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*, AAMAS '07, page 39, 2007.

[2] Jeffrey L Adler, Goutam Satapathy, Vikram Manikonda, Betty Bowles, and Victor J Blue. A multi-agent approach to cooperative traffic management and route guidance. *Transportation Research Part B : Methodological*, 39(4) :297–318, 2005.

[3] Hani Alzaid, Manal Alfaraj, Sebastian Ries, Audun Josang, Muneera Albabtain, and Alhanof Abuhaimed. Reputation-based trust systems for wireless sensor networks : A comprehensive review. In *Trust Management VII*, volume 401 of *IFIP Advances in Information and Communication Technology*, pages 66–82. Springer Berlin Heidelberg, 2013.

[4] Ana L. C. Bazzan and Franziska Klügl. A review on agent-based technology for traffic and transportation. *The Knowledge Engineering Review*, 29(3) :375–403, 2013.

[5] Birgit Burmeister, Afsaneh Haddadi, and Guido Matylis. Application of multi-agent systems in traffic and transportation. In *Software Engineering. IEE Proceedings*, volume 144, pages 51–60. IET, 1997.

[6] Cristiano Castelfranchi and Rino Falcone. Principles of trust for mas : cognitive anatomy, social importance, and quantification. In *Proceedings of the International Conference on Multi Agent Systems*, pages 72–79, 1998.

[7] Bo Chen and Harry H Cheng. A review of the applications of agent technology in traffic and transportation systems. *Intelligent Transportation Systems, IEEE Transactions on*, 11(2) :485–497, 2010.

[8] Hussein Dia and Sakda Panwai. Modelling drivers' compliance and route choice behaviour in response to travel information. *Nonlinear Dynamics*, 49(4) :493–509, 2007.

[9] Marius Dupuis and Han Grezlikowski. Opendrive®–an open standard for the description of roads in driving simulations. In *Proceedings of the Driving Simulation Conference*, pages 25–36, 2006.

[10] Hongxia Ge, S. Q. Dai, and Li-Yun Dong. An extended car-following model based on intelligent transportation system application. *Physica A : Statistical Mechanics and its Applications*, 365(2) :543–548, 2006.

[11] Dirk Helbing. Traffic and related self-driven many-particle systems. *Reviews of modern physics*, 73(4) :1067, 2001.

[12] Josefa Z Hernández, Sascha Ossowski, and Ana Garcıa-Serrano. Multiagent architectures for intelligent traffic management systems. *Transportation Research Part C : Emerging Technologies*, 10(5) :473–506, 2002.

[13] Richard Holzer, Hermann de Meer, and Christian Bettstetter. On autonomy and emergence in self-organizing systems. In *Self-Organizing Systems*, pages 157–169. Springer, 2008.

[14] Trung Dong Huynh, Nicholas R Jennings, and Nigel R Shadbolt. An integrated trust and reputation model for open multi-agent systems. *Autonomous Agents and Multi-Agent Systems*, 13(2) :119–154, 2006.

[15] Rui Jiang, Qingsong Wu, and Zuojin Zhu. Full velocity difference model for a car-following theory. *Phys. Rev. E*, 64 :017101, Jun 2001.

[16] Arne Kesting, Martin Treiber, and Dirk Helbing. General lane-changing model mobil for car-following models. *Transportation Research Record : Journal of the Transportation Research Board*, 1999(1) :86–94, 2007.

[17] Michiel M Minderhoud and Piet HL Bovy. Extended time-to-collision measures for road traffic safety assessment. *Accident Analysis & Prevention*, 33(1) :89–97, 2001.

[18] Julien Monteil, Romain Billot, Jacques Sau, Frédéric Armetta, Salima Hassas, and Nour-Eddin El Faouzi. Cooperative highway traffic : multi-agent modelling and robustness assessment to local perturbations. *Transportation Research Record : Journal of the Transportation Research Board*, 2013.

[19] Julien Monteil, Romain Billot, Jacques Sau, Christine Buisson, and Nour-Eddin El Faouzi. Calibration, estimation and sampling issues of car following model. *Proceedings of the 93rd annual meeting of the Transportation Research Board. Transportation Research Board of the National Academies, Washington, DC*, 2014.

[20] Julien Monteil, Romain Billot, Jacques Sau, and Nour-Eddin El Faouzi. Linear and weakly nonlinear stability analyses of cooperative car-following models. *Intelligent Transportation Systems, IEEE Transactions on*, PP(99) :1–13, 2014.

[21] Quang-Anh Nguyen Vu, Richard Canal, Benoit Gaudou, Salima Hassas, and Frédéric Armetta. Trustsets : using trust to detect deceitful agents in a distributed information collecting system. *Journal of Ambient Intelligence and Humanized Computing*, 3(4) :251–263, 2012.

[22] Hiro Onishi and Fanny Mlinarsky. Wireless technology assessment for automotive applications. In *Proc. ITS World Congress*, 2012.

[23] Isaac Pinyol and Jordi Sabater-Mir. Computational trust and reputation models for open multi-agent systems : a review. *Artificial Intelligence Review*, 40(1) :1–25, 2013.

[24] Jordi Sabater and Carles Sierra. Review on computational trust and reputation models. *Artificial intelligence review*, 24(1) :33–60, 2005.

[25] Michael Schillo, Petra Funk, Im Stadtwald, and Michael Rovatsos. Using trust for detecting deceitful agents in artificial societies. *Applied Artificial Intelligence Journal*, Special Issue on Trust, Deception and Fraud in Agent Societies, 2000.

[26] Martin Treiber and Arne Kesting. *Traffic flow dynamics : data, models and simulation*. Springer, 2013.

[27] Fei-Yue Wang. Agent-based control for networked traffic management systems. *Intelligent Systems, IEEE*, 20(5) :92–96, 2005.

Calibration de simulations multi-agents à l'aide d'une méthode semi-automatique d'analyse du comportement

K. Darty[a,b] kevin.darty@limsi.fr
J. Saunier[c] julien.saunier@insa-rouen.fr
N. Sabouret[b] nicolas.sabouret@limsi.fr

[a]IFSTTAR, France

[b]LIMSI-CNRS, UPR 3251, Univ. Paris-Sud, Orsay, France

[c]LITIS, INSA de Rouen, France

Résumé

L'un des principaux problèmes en simulation multi-agent est la définition des paramètres du modèle agent et leur calibration. Ce problème est encore plus difficile lorsqu'on considère des environnements virtuels immersifs, dans lesquels les agents intelligents doivent reproduire des comportements humains et apparaître « réalistes » aux yeux des utilisateurs. Dans cet article, nous proposons d'enregistrer et d'analyser les comportements des agents pour évaluer leur similarité avec ceux des humains dans un environnement virtuel immersif. Nous utilisons des méthodes de classification pour construire une abstraction des comportements individuels. Les types de comportement sont étudiés selon la composition de ces classes pour déterminer les manques, capacités et erreurs dans le modèle agent. Cette méthode nous permet 1) d'écarter les jeux de paramètres invalides, 2) de calibrer des simulations valides et 3) d'expliquer les manques du modèle agent pour l'améliorer.

Mots-clés : *Simulation multi-agent, Calibration*

Abstract

In the context of agent-based simulation, a major issue is to define relevant parameters of the agent model and calibrate them. This issue is yet harder in immersive virtual environments, where intelligent agents reproduce human behaviour and interact with users. In this paper, we propose to log and analyse agents behaviour to evaluate their similarity to humans behaviour in an immersive virtual environment. Clustering is then used to get an abstraction of individual behaviours. The behaviour archetypes are studied in terms of cluster members in order to identify agent lacks, capacities and errors. This study enables to 1) dismiss invalid parameter sets, 2) calibrate valid simulations and 3) explain lacks in the agent models for further improvement.

Keywords: *Multi-agent simulation, Calibration*

1 Introduction

Dans le contexte des simulations à base d'agents (voir par exemple [14, 1, 7]), les agents doivent souvent reproduire des comportements similaires à ceux qu'adopteraient des humains dans la même situation. Une difficulté de ces modèles de simulation est d'identifier les jeux de paramètres qui permettent d'obtenir des comportements « valides ». D'une part, le comportement des agents doit être crédible, *i.e.* les valeurs des paramètres doivent conduire à des comportements qu'un humain pourrait adopter. De l'autre, l'ensemble des comportements produits par les agents doit être représentatif de la population simulée, *i.e.* les jeux de paramètres doivent permettre d'obtenir une certaine variabilité dans les comportements des agents.

La majorité des travaux de recherche dans le domaine s'intéressent surtout au niveau individuel. Cela conduit à définir des jeux de paramètres qui correspondent à des comportements moyens ou normatifs. Si ce choix est pertinent pour des simulations de niveau macroscopique, les *SMA* s'intéressent généralement à des phénomènes microscopiques pour lesquels les comportements normatifs ne sont pas adaptés [12, 1]. Dans ce contexte, plusieurs méthodes ont été proposées pour l'analyse semi-automatique des comportements d'agents en fonction des paramètres du modèle. Par exemple, Taillandier *et al.* [17] proposent une méthode pour évaluer les ensembles caractéristiques d'une méthode d'apprentissage supervisé qui contrôle un *SMA* géographique. Caillou *et al.* [2] proposent un modèle de classification des agents suivant leur

comportement pour étudier l'impact des paramètres sur la dynamique du *SMA*. Plus récemment, nous avons proposé dans [5, 4] une méthode semi-automatique d'analyse des comportements des agents dans une simulation immersive en comparant des classifications de comportements produits par des agents et par des humains mis dans la même situation. La crédibilité du comportement des agents est alors étudiée en termes de capacités à reproduire des comportements humains, de manques (*i.e.* de comportements non reproduits) et d'erreurs (*i.e.* de comportements produits par des agents mais aucun humain). Toutefois, aucun de ces modèles ne propose de méthode pour calibrer les paramètres de chaque agent de la simulation à partir de l'analyse des comportements.

Dans cet article, nous proposons d'étendre notre approche proposée dans [5] pour l'évaluation et la calibration de simulations multi-agents. Pour cela, nous analysons les comportements produits par les agents indépendamment du modèle sous-jacent. Nous considérons donc les agents comme des boîtes noires et collectons des données sur leur comportement en fonction des paramètres du modèle. Nous comparons alors les traces de simulation à l'aide de mesures calibrées sur les comportements produits par les humains en simulation participative. L'agrégation des traces de comportement à l'aide de méthodes de classification permet d'obtenir des archétypes de comportement [4]. La composition de chaque classe (agents, humains ou mixte) permet d'identifier et d'expliciter les comportements agents en fonction des paramètres. Enfin, les classes nous permettent de définir la population d'agents pour la simulation.

La prochaine section présente les travaux connexes dans le domaine de l'évaluation des comportements, qui est à la base de notre approche, et nous présentons brièvement la méthode de classification sur laquelle nous nous appuyons. La section 3 présente notre méthode d'évaluation des comportements et de calibration des paramètres. La section 4 montre la mise en œuvre de cette méthode dans le contexte de la simulation de conduite.

2 Travaux connexes

2.1 Analyse du comportement

La notion de comportement couvre différents aspects, des actions bas niveau sélectionnées par un agent lors d'un cycle d'exécution, à des éléments plus complexes comme les mouvements de foule [1] ou les décisions macro-économiques [14]. Cependant, tous ces domaines partagent un même objectif : produire des résultats valides à l'aide de simulations multi-agents pour l'analyse et la prédiction du comportement humain.

La validité est vue dans la majorité des modèles à un niveau macroscopique : les méthodes d'analyse statistique de données sont alors bien adaptées pour déterminer la validité de la simulation [8, 1]. Il s'agit de vérifier, à travers des données quantitatives, que les agents se comportent de manière similaire à ce qui est observé dans une situation « réelle ». Cependant, le fait que le comportement global soit valide ne garantit pas que les comportements individuels soient réalistes. C'est pourquoi, dans nos travaux, nous nous intéressons au réalisme du comportement au niveau microscopique : chaque agent devrait adopter un comportement ressemblant à celui d'un humain, tout en maintenant la cohérence au niveau macroscopique.

La comparaison de traces au niveau microscopique se heurte à un problème important [2] : les données recueillies ne peuvent être analysées directement puisque celles-ci sont souvent bruitées et de nature temporelle, alors que les comportements recherchés sont de plus haut niveau. Ceci implique un recours à des traitements de données de façon à donner un sens aux traces de bas niveau.

Dans le domaine des agents virtuels, plusieurs travaux se sont intéressés à la crédibilité des agents et à leur ressemblance aux humains du point de vue de leur réaction affective [3], du comportement non verbal [15] ou de la décision [1]. Ces méthodes s'appuient sur l'évaluation de la crédibilité du comportement des agents par un observateur externe [13]. Bien adaptées pour étudier des agents virtuels, elles ne peuvent pas être utilisée pour traiter des grands nombres d'agents, comme ceux que nous rencontrons dans les simulations multi-agents. En pratique, il est impossible de traiter et tester tous les paramètres sur des centaines d'agents via des méthodes qui nécessitent que le comportement de chaque agent soit observé et analysé par plusieurs humains. Il existe peu de travaux qui s'intéressent à l'analyse objective (par opposition aux approches subjectives à base de jugement humain) et automatique. Des travaux [6, 11] proposent des approches à base d'apprentissage automatique selon des variables de bas niveau, alors que Caillou *et al.* [2] pro-

posent de définir, avec l'aide d'experts du domaine, des variables de haut niveau qui sont ensuite utilisées pour décrire les comportements analysés via un algorithme de classification automatique.

La principale limite de ces approches objectives est que, si elles permettent d'identifier les différences entre des catégories de comportements extraites à partir des traces de simulation (nous parlerons de *classes de traces*), elles ne fournissent aucune information au-delà des variables de bas niveau qui ont été utilisées. En particulier, elles ne permettent pas de donner un sens aux classes obtenues : les comportements détectés restent implicites. Au contraire, les approches subjectives, parce qu'elles s'appuient sur des analyses de haut niveau faites par des humains à travers des questionnaires validés par des experts, permettent d'obtenir des classes de comportement. Le modèle présenté dans [5] propose de combiner les approches objectives et subjectives pour tirer parti des deux méthodes. Nous présentons brièvement son fonctionnement dans la prochaine section.

2.2 Approche mixte

La méthode présentée dans [4] permet d'évaluer les comportements des agents dans le contexte d'environnements virtuels (*EV*) immersifs, en combinant l'analyse des traces d'interactions (approche objective) et les annotations d'observateurs (approche subjective). Les données de la simulation sont classifiées en classes de traces. Les questionnaires permettent de définir des catégories d'utilisateurs pour caractériser les comportements des humains et des agents. Nous évaluons alors le comportement des agents en étudiant la composition des classes de traces et en les comparant aux catégories de comportement des humains.

Cette méthode comprend 5 étapes :

1. la collecte des données de comportement des agents –virtuels– dans la simulation ;
2. la collecte des données de comportement des participants –humains– dans la même situation à l'aide de simulations participatives ;
3. l'annotation de ces données par des observateurs humains (appelés annotateurs) ;
4. le traitement des données et la classification automatique, ce qui conduit à des classes de comportements d'humains et d'agents ;
5. la comparaison des classes, *i.e.* l'analyse de leur composition et l'explicitation du comportement.

La principale limite de cette approche est que le résultat de l'analyse ne permet pas de réévaluer les paramètres de la simulation pour améliorer la validité des comportements produits. Dans cet article, nous proposons d'étendre la dernière étape de cette méthode de telle sorte que le résultat de l'évaluation puisse permettre la calibration du modèle de simulation et l'explication, en termes de valeurs de paramètres, des capacités et des manques dans les comportements d'agents.

3 Analyse & calibration

Nous proposons d'étendre la méthode précédente de la manière suivante (voir la figure 1) :

— nous utilisons l'étape d'abstraction pour calculer des scores qui mesurent d'une part la proportion de capacités, d'erreurs et de manques dans les comportements produits par les agents, et d'autre part la proportion d'agents qui reproduisent des comportements humains, en prenant en compte la fréquence d'occurrence de ces comportements dans les traces de simulation ;

— nous établissons une corrélation entre ces comportements corrects et les paramètres des agents pour calculer une nouvelle distribution de jeux de paramètres ;

— nous appliquons cette méthode en boucle pour tester les nouveaux paramètres et les évaluer en termes de niveau de reproduction de comportements humains. En cas de comportements manquants, nous explorons l'espace des paramètres, et si le modèle d'agent peut être modifié, nous corrigeons les erreurs ou complétons les comportements des agents.

3.1 Abstractions et scores

La dernière étape de la méthode initiale consiste à comparer le comportement des agents avec celui des humains en analysant la composition des classes. Trois types de classes peuvent être identifiés :

1. les classes contenant à la fois des humains et des agents : elles correspondent à des comportements de haut niveau correctement reproduits par les agents. Nous notons $\mathbb{C}_M$ l'ensemble de ces *classes mixtes*.

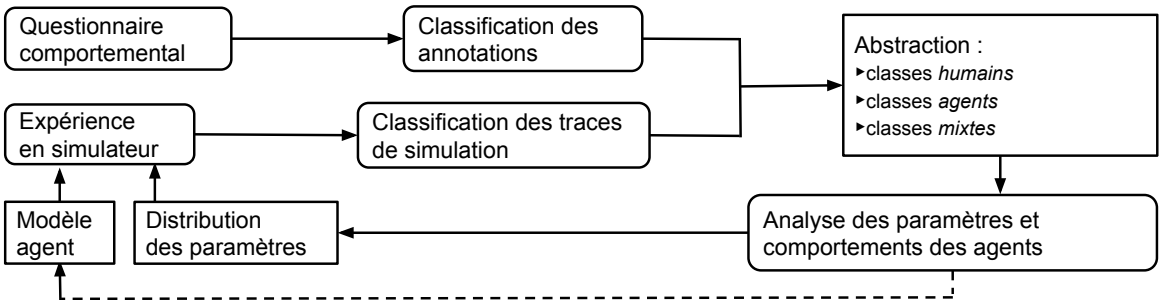


FIGURE 1 – Etapes de la méthode d'analyse et de calibration des comportements.

2. les classes contenant uniquement des agents : elles correspondent à des comportements qui ont été produits seulement par des agents. Dans la majorité des cas, cela correspond à des erreurs dans la simulation, mais cela peut aussi provenir d'un trop petit nombre de participants humains. Nous notons $\mathbb{C}_A$ l'ensemble de ces *classes agents* ;
3. les classes contenant que des humains : elles correspondent aux comportements qui n'ont pas été reproduits par les agents. Il s'agit donc soit de manques dans le modèle agent, soit d'un ensemble d'agents trop petit dans l'espace des paramètres. Nous notons $\mathbb{C}_H$ l'ensemble de ces *classes humains*.

Nous proposons de calculer trois scores d'après ces classes. Soit $|\mathbb{C}|$ la taille d'un ensemble $\mathbb{C}$. Nous définissons :

1. $S_c = \frac{|\mathbb{C}_M|}{|\mathbb{C}_M|+|\mathbb{C}_H|}$ le score des capacités ;
2. $S_m = \frac{|\mathbb{C}_H|}{|\mathbb{C}_M|+|\mathbb{C}_H|}$ le score des manques ;
3. $S_e = \frac{|\mathbb{C}_H|}{|\mathbb{C}_M|+|\mathbb{C}_A|}$ le score des erreurs.

Seuls, ces scores ne sont pas suffisants pour calibrer la simulation multi-agent. Par exemple, si un seul agent appartient à une classe contenant beaucoup d'humains, cela signifie que ce comportement ne sera reproduit que par une petite proportion des agents alors qu'il est fréquent chez les participants humains. Dans la prochaine sous-section, nous étudions donc les effectifs des classes pour détecter les cas de sous-représentation et de sur-représentation des agents.

3.2 Calibration

Dans une simulation multi-agent, la calibration du modèle implique de trouver des valeurs de paramètres permettant d'atteindre un but global ou d'obtenir un comportement spécifique [9, 10] (voir par exemple [18]). La calibration des simulations multi-agents est un problème difficilement résolu par les méthodes de calibration classiques [9] en raison du trop grand espace de paramètres, d'un temps trop élevé de simulation et des incertitudes dans la conception du modèle. Dans les simulations participatives où les agents interagissent avec des humains, une contrainte supplémentaire est que les comportements individuels soient crédibles.

Dans notre cas, nous proposons d'utiliser les traces de comportement des humains collectées pendant la phase d'évaluation du modèle : ils définissent un ensemble de comportements valides. En d'autres termes, notre but est de diminuer les valeurs de S_e et de S_m, au profit de S_c. Si l'on considère le modèle agent comme une boîte noire qui peut produire différents comportements en fonction de ses paramètres, le processus de calibration doit garantir que : 1) le comportement de chaque agent a est crédible, donc que son ensemble de paramètres $P_a = \{p_1, \ldots, p_i\}$ (avec i le nombre de paramètres du modèle) est individuellement valide ; 2) la population reproduit globalement les mêmes comportements que les humains, dans des proportions équivalentes, d'où une distribution d'ensembles de paramètres $\mathcal{P} = \{P_1, \ldots, P_n\}$ à définir, avec n le nombre d'agents.

Concrètement, les agents membres de classes de type « erreurs » ont exhibé des comportements qui ne sont pas présents chez les humains. Leurs ensembles de paramètres P_a doivent être retirés de la liste des ensembles valides. Inversement, les manques (*i.e.* les comportements exhibés uniquement par des humains) peuvent provenir de paramètres mal choisis, en supposant que le modèle agent a effectivement la capacité de produire ces comportements.

Pour obtenir une bonne proportion de comportements valides, nous prenons en compte l'appartenance des agents et des humains aux classes.

Nous pouvons donc raffiner les ensembles précédents en :

- erreurs comportementales, lorsque aucun humain n'appartient à la classe ($H(C) = 0$),
- capacités à reproduire des comportements humains dans des classes mixtes ($\mathbb{C}_M$), séparées en trois sous-ensembles :
  - $\mathbb{C}_{sur}$ l'ensemble des classes où les agents sont sur-représentés ($A(C) \gg H(C)$),
  - $\mathbb{C}_{valide}$ l'ensemble des classes où la représentation des agents est correcte, *i.e.* où la proportion d'humains et d'agents est comparable ($A(C) \approx H(C)$),
  - $\mathbb{C}_{sous}$ l'ensemble des classes où les agents sont sous-représentés ($A(C) \ll H(C)$).
- manques, lorsque aucun agent n'appartient à la classe ($A(C) = 0$).

Après une première calibration de l'ensemble des paramètres $\mathcal{P}$, les scores ne dépendent plus du nombre d'agents mais de la proportion dans la population totale. Nous avons souvent plus d'agents que d'humains (pour des raisons expérimentales liées au temps de collecte des données). De plus, l'opérateur $\approx$ qui permet de caractériser une représentation valide nécessite de définir une limite. Nous proposons de nous baser sur la taille de la classe, *e.g.* $\delta(C) = \frac{5}{100}|C|$.

Pour améliorer la calibration, nous établissons trois scores à partir des classes mixtes $\mathbb{C}_M$, en complément des scores de manque et d'erreur :

- sur-représentation $S_{sur} = \frac{|\mathbb{C}_{sur}|}{|\mathbb{C}_M|}$ avec :

  $$\mathbb{C}_{sur} = \{C_M, A(C_M) > H(C_M| + \delta|C_M|\}$$

- sous-représentation $S_{sous} = \frac{|\mathbb{C}_{sous}|}{|\mathbb{C}_M|}$ avec :

  $$\mathbb{C}_{sous} = \{C_M, A(C_M) < H(C_M) - \delta|C_M|\}$$

- représentation valide $S_{valide} = \frac{|\mathbb{C}_{valide}|}{|\mathbb{C}_M|}$ avec :

  $$\mathbb{C}_{valide} = \mathbb{C}_M \setminus (\mathbb{C}_{sur} \cup \mathbb{C}_{sous})$$

Ces scores permettent d'évaluer la qualité de la calibration courante et éventuellement de la valider. Si ça n'est pas le cas, ils permettent de modifier la proportion d'agents associés à chaque ensemble de paramètres $\mathcal{P}_i$, d'explorer de nouveaux ensembles de paramètres et de retirer ceux qui sont invalides.

3.3 Définition d'un ensemble de paramètres

Soit $\mathcal{P}_v$ l'ensemble des jeux de paramètres valides correspondant aux comportements valides $\mathcal{B}_v \subset \mathcal{B}$ ($\mathcal{B}$ désignant l'ensemble des comportements). Nous notons $simul(P_i) = b$ le comportement b produit par un ensemble de paramètres P_i et $p(b)$ la proportion d'humains exhibant ce comportement.

Puisque plusieurs jeux de paramètres peuvent produire le même comportement, la définition d'un ensemble de paramètres $P(a_i)$ avec $i \in \{1, \ldots, n\}$ pour n agents nécessite de choisir entre différents jeux de paramètres. Nous proposons de les choisir de la manière suivante : $P(a_i) = P_i \in \mathcal{P}_v$ avec une probabilité $p(P_i)$, dépendant de la proportion des comportements observés b et du nombre nb de jeux de paramètres tels que $simul(P_i) = b \in \mathcal{B}_v$ conduisent à b :

$$p(P_i) = \frac{p(b)}{nb}$$

De cette manière, les comportements qui étaient sous-représentés ont une probabilité plus élevée d'être sélectionnés, alors que les comportements sur-représentés seront moins sélectionnés.

Il est aussi possible de choisir arbitrairement l'un des ensembles de paramètres $P_i \in \mathcal{P}_v | simul(P_i) = b \in \mathcal{B}_v$ et de générer $n \cdot p(b)$ agents avec cet ensemble. Cependant, selon le type de simulation, maintenir une hétérogénéité contrôlée des agents [1] permet de produire des simulations plus réalistes.

Notons aussi que puisque nous nous limitons pour la génération des nouveaux jeux de paramètres à $\mathcal{P}_v$ (et non $\mathcal{P}$), tous les ensembles de paramètres invalides sont supprimés.

3.4 Exploration de l'espace des paramètres

Le nombre d'agents nécessaires pour couvrir tous les ensembles de paramètres possibles peut être très élevé. La première génération de jeux de paramètres s'appuie donc sur des connaissances du domaine ou sur des valeurs par défaut définies dans le modèle. En conséquence, il n'est généralement pas possible de produire tous les comportements dès le premier cycle de notre méthode d'évaluation de simulations multi-agents. Si des manques sont détectés, nous proposons alors d'utiliser une fonction d'exploration des ensembles de paramètres non

1. L'hétérogénéité est contrôlée par le fait que tous les ensembles de paramètres menant à des comportements appartenant à la même classe vont produire des comportements similaires.

encore testés :

$$P(a_i) = \begin{cases} P_i \in \mathcal{P}_v & si\ p > \gamma \\ P_k \notin \mathcal{P} & sinon \end{cases}$$

Le paramètre d'exploration γ permet de rechercher des nouveaux comportements de manière itérative, et p est une valeur aléatoire uniforme. Afin de ne pas essayer plusieurs fois des ensembles de paramètres invalides, P_k ne doit jamais avoir été sélectionné précédemment. Si P_k conduit à un comportement valide, il est ajouté à $\mathcal{P}_v$, sinon il est écarté pour la prochaine itération.

3.5 Itération de la méthode

Si tous les comportements cibles (déterminés par les classes de traces des humains) sont reproduits, un seul cycle de calibration est nécessaire. Lorsqu'il manque des comportements, l'exploration de l'espace des paramètres permet de découvrir de nouvelles classes de comportements reproduits par les agents.

Les manques et les erreurs peuvent aussi être traités par des modifications dans le modèle agent. Dans ce cas, les informations de l'étape d'annotation de la méthode d'évaluation permettent de mettre en évidence des informations sémantiques sur les comportements manquants ou erronés. Dans tous les cas, l'annotation et l'expérimentation avec des humains n'est requise qu'une seule fois, quel que soit le nombre d'itérations de la méthode de calibration des paramètres, puisque la méthode de traitement des données s'appuie sur l'agrégation des agents sur les classes de traces des humains. Ainsi, les données des agents ne modifient pas le modèle de référence constitué par les classes d'humains.

3.6 Taux de confiance

Les algorithmes de classification non supervisée peuvent produire des erreurs de classification parmi les traces humaines. En conséquence, les classes singletons peuvent représenter des comportements particuliers ou des erreurs de classification. De la même manière, l'agrégation d'agents aux classes d'humains peut, notamment à cause de l'effet de seuil, agréger un agent à une classe de comportement qui n'est pas *similaire*, ou réciproquement ne pas l'inclure dans une classe de comportement qui est *similaire*. Dans le but de vérifier cela, nous proposons de calculer un taux de confiance sur les classes résultantes.

Le taux de confiance de chaque classe dépend du nombre d'humains et d'agents dans toute la classification et dans la classe en question :
$t_A(C_M) = \frac{A(C_M)}{A(\mathbb{C})} \in [0,1]$
et $t_H(C_M) = \frac{H(C_M)}{H(\mathbb{C})} \in [0,1]$

Pour les classes mixtes, le taux de confiance est haut quand le taux de participants humains est approximativement égal à celui des agents simulés :

$$t_{conf}(C_M) = 1 - |t_A(C) - t_H(C)|$$

Par exemple, une classe contenant 9 participants sur 12 au total et seulement 2 agents sur 20 au total a un taux de $1 - \left|\frac{9}{12} - \frac{2}{20}\right| = 0{,}35$. Cette classe est alors considérée comme une classe mixte mais avec un taux de confiance bas. *A contrario*, une classe contenant 2 participants sur les 12 et 4 agents sur les 20 a un taux de $1 - \left|\frac{2}{12} - \frac{4}{20}\right| = 0{,}97$. Il y a donc une forte confiance que cette classe mixte soit réellement une capacité du modèle agent à reproduire le comportement humain adopté.

À l'inverse, pour les *classes agents*, le taux de confiance $t_{conf}(C_A)$ dépend uniquement du nombre d'agents relativement au nombre moyen d'agents $E_A(\mathbb{C}) = \frac{A(\mathbb{C})}{|\mathbb{C}_M|+|\mathbb{C}_A|}$ (respectivement $E_H(\mathbb{C}) = \frac{H(\mathbb{C})}{|\mathbb{C}_M|+|\mathbb{C}_H|}$ pour le taux de confiance $t_{conf}(C_H)$ des *classes humains*) :

$$t_{conf}(C_A) = \frac{A(C)}{E_A(\mathbb{C})}$$

$$t_{conf}(C_H) = \frac{H(C)}{E_H(\mathbb{C})}$$

Un taux de confiance élevé en une *classe d'humains* (resp. une *classe d'agents*) signifie que cette classe peut être considérée comme un manque (resp. une erreur) dans le modèle agent avec confiance.

4 Évaluation

Dans un premier temps, cette section rappelle les résultats de classification issus de [4] sur une expérimentation étudiant les comportements en simulation de conduite, puis elle les complète avec les résultats de notre extension fournissant des scores d'évaluation pour la calibration.

Le simulateur de trafic routier *ARCHISIM* [7] de l'*IFSTTAR* est évalué sur la situation spécifique suivante : sur une route bidirectionnelle, l'acteur principal rencontre un véhicule à vitesse réduite sur la file de droite et plusieurs véhicules venant en face sur la file de gauche. Le but est d'évaluer la crédibilité des comportements de conduite des agents dans le but de calibrer des simulations valides.

Pour la partie objective, nous recueillons les traces des acteurs principaux (les participants et les agents) pendant la simulation. Les indicateurs choisis par des experts du domaine sont calculés à partir de ces traces de simulation (*e.g.* la distance inter-véhiculaire ou le nombre de changements de voie). Ces indicateurs sont utilisés pour classifier les participants, et agréger les agents dans les classes. Pour la partie subjective, le questionnaire d'annotations comportemental fournit 6 sous-échelles [16] : *inexpérience*, *inattention*, *erreur de jugement*, *violation accidentelle* et *délibérée*, *risque d'accident*. 6 annotateurs ont rempli ce questionnaire à la fois pour les participants et les agents. La même méthode de traitement que celle utilisée sur les traces de simulation est appliquée sur ces scores.

22 conducteurs réguliers ont effectué l'expérimentation. Les véhicules simulés de l'*EV* étant autonomes, les situations peuvent différer. Les paramètres du modèle *ARCHISIM* sont :

- $v \in \mathbb{N}$ la vitesse désirée (en km/h),
- $reglmnt \in [0{,}100]$ la volonté de conduire selon le code de la route,
- $infra \in [0{,}100]$ la capacité de contrôle du véhicule selon l'infrastructure,
- $trafic \in [0{,}100]$ la flexibilité du temps inter-véhiculaire selon le trafic,
- $soupl \in [0{,}100]$ l'agressivité du conducteur (ne tenant pas compte des courtes variations),
- $exp \in \{0{,}1\}$ l'expérience du conducteur.

Dans cette itération, les paramètres utilisés sont les valeurs moyennes pour chaque paramètre, tandis qu'un paramètre est modifié pour chaque agent. Le paramètre v est choisi parmi $\{100, 110, 120, 130, 140\}$; les quatre paramètres suivant $(reglemnt, infra, trafic, soupl)$ sont par défaut à 50, et à 25 ou 75 sinon ; enfin exp est à 0 ou à 1.

4.1 Résultats de classification

Les classifications de traces et d'annotations sont comparées comme illustré dans la figure 2. Il y a 2 classes de comportement issues des traces : 1) *Logs cluster 1* est composée des acteurs principaux qui n'ont pas essayé de dépasser le véhicule à vitesse réduite. Étant une classe mixte, c'est une capacité du modèle agent à reproduire un comportement humain qui est celui de choisir de ne pas dépasser. 2) *Logs cluster 2* contient des acteurs principaux qui ont dépassé le véhicule à vitesse réduite. Comme cette classe est uniquement composée de participants, c'est donc un manque dans le modèle agent : les agents ne peuvent pas choisir de ne pas dépasser comme les humains le font.

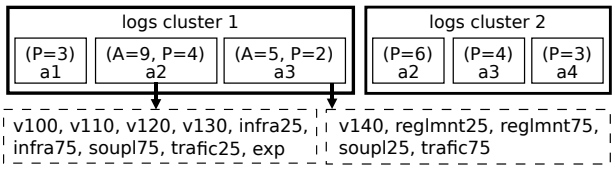


FIGURE 2 – Comparaison des acteurs principaux entre la classification de traces (avec les participants P et les agents A), et la classification d'annotations regroupés par classe avec leur numéro ($a\#$), en fonction de leurs paramètres.

Il y a 4 classes de comportements issues des annotations : 1) la classe d'annotations 1 a été considérée par les annotateurs comme ayant le style de conduite le plus dangereux, aucun agent n'a été considéré dangereux. 2) la classe d'annotations 2 a été annotée comme correspondant à des conducteurs prudents. Étant une classe mixte, le comportement normatif humain peut donc être considéré comme partiellement reproduit. 3) la classe d'annotations 3, mixte également, est une petite classe d'annotations dont les acteurs principaux ont été jugés comme étant des conducteurs ordinaires. 4) la classe d'annotations 4 contient des acteurs principaux considérés comme légèrement dangereux, comportement non reproduit par les agents.

Dans la classe *logs cluster 1*, les indicateurs n'ont pas permis de distinguer la classe d'annotations 1 du reste des acteurs principaux, tandis que ces acteurs principaux ont essayé de dépasser en vain. Pareillement, les participants de la classe d'annotations 4 n'ont pas été séparés de la classe *logs cluster 2* en une nouvelle classe.

4.2 Scores

Les scores correspondant à la classification des annotations dans cette expérimentation sont les suivants :

- le score d'erreur est $S_e = 0$ (car $|\mathbb{C}_A| = 0$),

- le score de manque est :
  $S_m = \frac{|\mathbb{C}_H|}{|\mathbb{C}_M|+|\mathbb{C}_H|} = \frac{1}{1+1} = \frac{1}{2}$.
- le score de capacité est :
  $S_c = \frac{|\mathbb{C}_M|}{|\mathbb{C}_M|+|\mathbb{C}_H|} = \frac{1}{1+1} = \frac{1}{2}$.

Le taux de confiance en la classe 1 comme étant une capacité du modèle est :

$$t_{conf}(C_1) = 1 - \left|\frac{14}{14} - \frac{9}{22}\right| \approx 0{,}41$$

$t_{conf}(C_1)$ n'étant pas proche de 0, cette classe peut donc être considérée avec confiance comme étant une capacité du modèle.

La classe de traces 2 est une classe composée uniquement d'humains, le taux de confiance est alors calculé de la manière suivante :

$$E_H(\mathbb{C}) = \frac{22}{1+1} = 11$$

$$t_{conf}(C_2) = \frac{13}{11} \approx 1{,}18$$

$t_{conf}(C_2)$ est proche de 1 signifiant que cette classe composée uniquement d'humains peut être traitée avec confiance comme étant un comportement manquant réel dans le modèle agent.

Ces scores permettent de trouver les nombres de comportements adéquats et inadéquats, et donc de produire l'ensemble des jeux de paramètres qui sont valides : $P \in \mathcal{P}_v = \{v100, v110, v120, v130, infra25, infra75, soupl75, trafic25, exp\} \cup \{v140, reglmnt25, reglmnt75, soupl25, trafic75\}$.

Il n'y a qu'une classe mixte. Cette dernière est sur-représentée, en effet pour un $\delta = \frac{5}{100}$ (*i.e.* peu de tolérance aux variations de proportions entre les comportements humains et agents) :

$$A(C_1) > H(C_1) + \delta|C_1| \Leftrightarrow 14 > 9 + \frac{5}{100} \times 23$$

Les scores de représentativité correspondants sont : $S_{sur} = \frac{|\mathbb{C}_{sur}|}{|\mathbb{C}_M|} = 1$, $S_{sous} = \frac{|\mathbb{C}_{sous}|}{|\mathbb{C}_M|} = 0$, et $S_{valide} = \frac{|\mathbb{C}_{valide}|}{|\mathbb{C}_M|} = 0$.

Les scores de type de classe ont permis de quantifier les erreurs (0), capacités (0,5), et manques (0,5). Les taux de confiance ont permis de s'assurer que la classe mixte est bien une capacité du modèle d'agent ($t_{conf}(C_1) \approx 0{,}41$) et que la classe d'humains est bien un comportement humain manquant ($t_{conf}(C_2) \approx 1{,}18$). Les scores de représentativité montrent que la calibration originale sur-représente le comportement humain (correspondant à la classe mixte) dans la population d'agents.

Une fois ces scores calculés, il est possible de calibrer une nouvelle population d'agents.

4.3 Calibration

Nous avons trouvé que seule une partie des comportements humains étaient reproduits. Dans ce cas, il existe deux possibilités pour l'utilisateur de la simulation : soit entrer dans un cycle d'exploration de l'espace des paramètres, soit calibrer le système en utilisant la première analyse. Dans cette partie, nous nous concentrons sur la seconde possibilité.

Nous avons vu que tous les jeux de paramètres étaient valides mais sur-représentés, et qu'il y avait plusieurs comportements manquants. Par conséquent, afin d'obtenir des agents reproduisant le comportement humain, nous calibrons les nouveaux ensembles de paramètres des agents en les choisissant parmi ceux valides dans la première expérimentation et en raffinant les proportions grâce aux classes d'annotations.

La première classe contient 9 traces d'agents et 4 traces de participants, tandis que la seconde classe contient 5 traces d'agents et 2 traces de participants. Nous obtenons donc, en calculant leur probabilité d'apparition dans la prochaine calibration : $p(v100) = p(v110) = p(v120) = p(v130) = p(infra25) = p(infra75) = p(soupl75) = p(trafic25) = p(exp) = \frac{4/6}{9}$ et $p(v140) = p(reglmnt25) = p(reglmnt75) = p(soupl25) = p(trafic75) = \frac{2/6}{5}$.

De cette manière, les agents représentent stochastiquement la densité de comportements humains dans la simulation. Il est à noter que cette nouvelle calibration ne change pas les scores de capacités et de manques, qui sont basés sur la proportion de comportements humains correctement reproduits. Cependant, il réduit le score d'erreur à 0 en n'utilisant que des ensembles de paramètres valides, et améliore le score de représentativité en choisissant des ensembles de paramètres pour les agents selon les comportements observés chez les humains.

Discussion

Notre méthode introduit des métriques pour mesurer les scores résultant et pour corriger les paramètres des agents selon une telle étude. Les scores d'erreurs, de manques, et de capacités permettent au concepteur de la simulation multi-agent de trouver combien d'archétypes de comportement humain ont été correctement reproduits, combien de comportements agents ne devraient pas apparaître, et combien de comportements humains sont manquants. Le taux de confiance donne des indications sur la fiabilité des classes en fonction de leurs effectifs. Ensuite, en étudiant uniquement les archétypes des comportements correctement reproduits, les scores de calibration donnent de l'information sur les proportions de chaque comportement et leurs relations à la calibration des agents.

Une des originalités de ce processus de calibration est le contexte des simulations participatives. La fonction cible du processus de calibration n'est pas comme habituellement [9] au niveau macroscopique mais à un niveau individuel. La réalité virtuelle requière que chaque agent adopte des comportements crédibles, *i.e.* des comportements pouvant être produits par des humains. Dans ce contexte, nous retirons premièrement les ensembles de paramètres qui ne produisent pas de comportements valides. Nous calibrons ensuite les proportions d'agents avec les ensembles de paramètres restants selon les données des participants humains. Un unique cycle de notre méthode assure que les comportements valides sont détectés et qu'ils sont produits en des proportions correctes, nonobstant les comportements manquants.

Dans le cas d'une boîte noire où le modèle agent est inconnu et ne peut être modifié, si des comportements sont manquants alors une solution est d'explorer l'espace des paramètres afin de trouver de nouveaux comportements d'agents. Ces nouvelles étapes ne requièrent pas une autre expérimentation avec des participants humains, puisque les données de référence sont déjà disponibles. Chaque nouveau cycle permet de trouver potentiellement de nouveaux ensembles de paramètres, soit dans des classes déjà *mixtes*, soit dans les classes de *manques* précédentes. Cela permet aussi de trouver quelles zones de l'espace des paramètres produisent des comportements invalides.

Dans le cas d'une boîte blanche où le modèle agent est connu et peut être potentiellement modifié, les données d'annotations expliquent les comportements manquants et les comportements erronés, permettant ainsi d'améliorer le modèle agent [10]. De plus, l'exploration des ensembles de paramètres peut être guidé par la connaissance du modèle [9].

5 Conclusion & perspectives

Cet article présente une méthode semi-automatique de calibration de simulations multi-agents participative basée sur la combinaison de classifications non supervisées de traces de simulation et d'annotations par des participants. Ces classifications portent à la fois sur les comportements des agents et des participants, comparés dans la même situation. L'expérimentation permet de définir un ensemble initial de traces valides qui servent de points de référence pour la calibration du modèle multi-agent. La calibration des paramètres du modèle suit une approche itérative. À chaque itération, nous obtenons les manques et les erreurs du modèle afin de raffiner l'espace des paramètres. Nous générons de nouveaux agents qui sont agrégés aux classes précédentes et nous calculons de nouveaux scores pour l'itération suivante. Notre méthode de validation a été appliquée à la simulation de trafic routier et a montré que des paramètres peuvent être correctement associés à des catégories de comportement.

L'originalité de cette approche est double. Premièrement, elle combine une analyse automatique des comportements des agents via les traces de simulation avec une analyse subjective basée sur l'évaluation humaine des comportements des agents, dans le but de définir un contexte spécifique à la situation. Cette combinaison permet une analyse de quel espace des paramètres des agents virtuels produit quel comportement perçu. Secondement, elle itère l'analyse de la classification afin de raffiner les capacités des agents à reproduire des comportements humains, tout en réduisant les manques et en supprimant les erreurs.

Plusieurs extensions doivent être examinées. Premièrement, la méthode d'agrégation dépend d'un taux de tolérance dont la valeur peut impacter la qualité des résultats : cet impact devra être vérifié. Secondement, la convergence du modèle n'a pas encore été étudiée. Notre algorithme de classification donne des scores qui pourraient être utilisés afin de stopper le processus, mais une preuve de convergence est nécessaire lorsque le cycle n'est pas utilisé pour ex-

plorer de nouveaux paramètres.

Références

[1] T. Bosse, M. Hoogendoorn, M. C. Klein, J. Treur, and C. N. Van Der Wal. Agent-based analysis of patterns in crowd behaviour involving contagion of mental states. In *Modern Approaches in Applied Intelligence*, pages 566–577. Springer, 2011.

[2] P. Caillou and J. Gil-Quijano. Simanalyzer : Automated description of groups dynamics in agent-based simulations. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 3*, pages 1353–1354, 2012.

[3] S. Campano, N. Sabouret, E. de Sevin, and V. Corruble. An evaluation of the core computational model for affective behaviors. In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, pages 745–752, 2013.

[4] K. Darty, J. Saunier, and N. Sabouret. Agents behavior semi-automatic analysis through their comparison to human behavior clustering. In *Intelligent Virtual Agents*, pages 154–163. Springer, 2014.

[5] K. Darty, J. Saunier, and N. Sabouret. Analyse des comportements agents par agrégation aux comportements humains. In *22èmes Journées Francophones sur les Systèmes Multi-Agents (JFSMA 2014)*. Cépaduès, 2014.

[6] E. Delaherche, M. Chetouani, A. Mahdhaoui, C. Saint-Georges, S. Viaux, and D. Cohen. Interpersonal synchrony : A survey of evaluation methods across disciplines. *Affective Computing, IEEE Transactions on*, 3(3) :349–365, 2012.

[7] A. Doniec, R. Mandiau, S. Piechowiak, and S. Espié. A behavioral multi-agent model for road traffic simulation. *Engineering Applications of Artificial Intelligence*, 21(8) :1443–1454, 2008.

[8] A. Drogoul, B. Corbara, and D. Fresneau. Manta : New experimental results on the emergence of (artificial) ant societies. *Artificial Societies : the computer simulation of social life*, pages 190–211, 1995.

[9] M. Fehler, F. Klügl, and F. Puppe. Techniques for analysis and calibration of multi-agent simulations. In *Engineering Societies in the Agents World V*, pages 305–321. Springer, 2005.

[10] M. Fehler, F. Klügl, and F. Puppe. Approaches for resolving the dilemma between model structure refinement and parameter calibration in agent-based simulations. In *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, pages 120–122. ACM, 2006.

[11] J. Gonçalves and R. J. F. Rossetti. Extending sumo to support tailored driving styles. *1st SUMO User Conference, DLR, Berlin - Adlershof, Germany*, 21 :205–211, 2013.

[12] B. Lacroix, P. Mathieu, and A. Kemeny. Formalizing the construction of populations in multi-agent simulations. *Engineering Applications of Artificial Intelligence*, 2012.

[13] J. C. Lester, S. A. Converse, et al. The persona effect : affective impact of animated pedagogical agents. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 359–366. ACM, 1997.

[14] P. Mathieu and O. Brandouy. A generic architecture for realistic simulations of complex financial dynamics. In *Advances in Practical Applications of Agents and Multiagent Systems*, pages 185–197. Springer Berlin Heidelberg, 2010.

[15] C. Pelachaud. Modelling multimodal expression of emotion in a virtual agent. *Phil. Trans. R. Soc. B : Biological Sciences*, 364(1535) :3539–3548, 2009.

[16] J. Reason, A. Manstead, S. Stradling, J. Baxter, and K. Campbell. Errors and violations on the roads : a real distinction ? *Ergonomics*, 33(10-11) :1315–1332, 1990.

[17] P. Taillandier and A. Drogoul. Supervised feature evaluation by consistency analysis : application to measure sets used to characterise geographic objects. In *Knowledge and Systems Engineering (KSE), 2010 Second International Conference on*, pages 63–68. IEEE, 2010.

[18] A. Veremme, É. Lefevre, G. Morvan, D. Dupont, and D. Jolly. Evidential calibration process of multi-agent based system : An application to forensic entomology. *Expert Systems with Applications*, 39(3) :2361–2374, 2012.

De l'intérêt de la cognition incarnée pour les agents logiciels

Julien Saunier
julien.saunier@insa-rouen.fr

LITIS, INSA de Rouen,
Avenue de l'Université - BP8, 76801 Saint-Étienne-du-Rouvray Cedex - France

Résumé

Alors que la recherche sur les systèmes multi-agent devient de plus en plus mature, la communauté se heurte aux mêmes difficultés que celles rencontrées par les roboticiens en terme de complexité de conception. Deux problèmes majeurs sont les environnements dynamiques et l'interface entre les agents et ces environnements.

Dans cet article, je propose de s'appuyer sur la théorie de la cognition incarnée pour réifier la notion de corps pour les agents logiciels, de la même façon que l'environnement a été montré comme étant une abstraction de premier ordre dans la conception de systèmes multi-agent. Je propose un ensemble de responsabilités pour le corps des agents, et montre comment une architecture de système multi-agent peut utiliser ce concept pour déléguer une partie des processus de l'agent à son corps, comment cela préserve l'autonomie des agents, et comment l'utiliser au delà des environnements physiques (réels ou simulés). J'illustre ensuite ce paradigme avec une architecture pour le calcul de la dynamique émotionnelle qui prend en compte des événements ponctuels, la dynamique temporelle et la contagion émotionnelle. Finalement, je discute les défis et questions ouvertes levées par l'adoption de cette approche dans les systèmes multi-agents.

Mots-clés : *Systèmes multi-agents, agent incarné, environnement, architecture*

Abstract

As the multiagent systems research becomes more mature, the autonomous agents community is now facing the same difficulty as the robotics community in terms of design complexity. Unavoidable issues are dynamic environments, and the interfaces between the agents and these environments.

In this paper, I propose to draw lessons from the embodied cognition approach to reify the concept of body for software agents, in the same way as the environment has been shown to be a first-order abstraction in multiagent systems design. I propose a set of responsibilities for the agent body, and show how an agent architecture can use this concept to delegate a part of the agent's tasks to its body, how it ensures agent autonomy, and how it may be used beyond (virtual or real) physical environments. I illustrate this paradigm with an emotion computation architecture that takes into account punctual events, temporal dynamics and emotional contagion. Finally, I discuss open questions raised by the adoption of this approach in the MAS field.

Keywords: *Multi-agent Systems, Embodied agent, Environment, Architecture*

1 Introduction

Dans la dernière décennie, un certain nombre de recherches ont mis en évidence l'importance de tout ce qui est "externe" aux agents dans la conception de systèmes multi-agents. C'est en particulier le cas des travaux du groupe de travail sur les environnements multi-agents E4MAS (Environment for Multiagent Systems) [20]. Une des idées principales est de déléguer une partie des responsabilités du système multi-agent à l'environnement. Celui-ci encapsule alors un ensemble de mécanismes tels que l'observabilité et l'accessibilité aux ressources partagées. Nous pouvons par exemple citer les travaux en cours sur la notion d'*artifacts* pour la coordination des agents, les organisations et les normes [3, 27].

Ces travaux ont également souligné l'intérêt de la notion de *situation* des agents, même lorsque ceux-ci sont purement logiciels [26], et montré comment dériver des solutions intelligentes adaptatives de concepts inspirés d'environnements physiques. Par exemple, dans [36], les auteurs montrent comment distribuer des tâches à des véhicules automatisés dans un entrepôt grâce à des champs de gradient, en combinant un fonctionnement réactif et un environnement

virtuel informé bâti en surcouche de l'environnement réel. Cette approche obtient de meilleurs résultats que les assignements de tâches classiques par protocole *contract-net*, qui reposent sur une approche symbolique centrée agent.

Décharger une partie des coûts de calcul à l'environnement et le considérer comme un composant de premier ordre du système multi-agent peut être relié au concept de cognition située et incarnée [35, 38]. L'aspect situé de la cognition implique que l'interaction entre l'agent et son environnement contraint ses comportements possibles, ce qui à son tour influence ses processus cognitifs. Le second aspect, l'incarnation de la cognition, est lié à la thèse selon laquelle la cognition est autant le produit de processus corporels que du raisonnement symbolique de haut-niveau. Ainsi, la cognition située et incarnée (que nous nommerons par la suite uniquement cognition incarnée) considère que l'intelligence émerge d'un système composé de trois éléments : l'esprit, le corps, et l'environnement. Dans le domaine des systèmes multi-agents, seuls deux de ces éléments ont été étudiés de façon approfondie : l'esprit, à travers les approches classiques des agents cognitifs fondées sur le raisonnement symbolique, et l'environnement, à travers les approches réactives et les travaux liés à E4MAS.

Dans la communauté SMA, le concept de corps n'a eu jusqu'ici qu'un impact limité. Deux articles de la littérature utilisant explicitement ce concept sont le modèle ELMS [19] et les corps logiciels (*soft-bodies*) [23]. Dans ces travaux, le corps est considéré comme contrôlé par l'environnement. Il encapsule également plusieurs responsabilités dont l'observabilité et l'accessibilité de l'état public des agents, et la médiation de la perception. D'autres travaux introduisent des médiateurs entre les agents et environnements, tels que les objets d'interaction [13], les artifacts [27] et les instances environnementales du système conatif [31], qui peuvent être vus comme fonctionnellement similaires à la notion de corps.

Dans cet article, j'argue d'une part que la cognition incarnée est un paradigme approprié pour la conception de systèmes multi-agents, et d'autre part qu'il est besoin de plus de travaux sur le sujet des corps logiciels, de la même façon que cela a été fait pour l'environnement, pour définir les responsabilités de ces composants, comment ils s'interfacent avec l'esprit (agent) et l'environnement, et leurs propriétés (autonome, actif...). Cet article est donc une synthèse de travaux antérieurs et un exposé de position.

Dans la section 2, je motive l'importance de la cognition incarnée et introduit deux vues de ce paradigme, forte et faible. Un exemple de processus qui peut être typiquement partagé entre corps, esprit et environnement – le calcul des émotions – est exposé en section 3. Ensuite, comme première base de discussion, je souligne en section 4 les principes d'une approche agent incarnée en termes d'architecture du système et de composants. Finalement, je discute en section 5 quelques questions ouvertes concernant l'autonomie des agents, et les problématiques opérationnelles levées par l'adoption de cette approche dans le domaine des systèmes multi-agents.

2 La cognition n'est pas (que) du calcul

Depuis la fin des années 80, la cognition incarnée, issue des domaines de la philosophie et des sciences cognitives [35], a eu une influence majeure sur l'intelligence artificielle. L'approche cognitiviste traditionnelle [16] est fondée sur une représentation symbolique du monde, et sont traitement logique par des méthodes de résolution de problèmes. Elle s'appuie ainsi sur le principe du penseur désincarné : l'intelligence est une émanation de l'esprit, tandis que le corps est considéré comme une interface imprécise avec le monde, composé de capteurs et d'actionneurs vus comme un dispositif d'entrée/sortie déconnecté du processus cognitif de haut-niveau.

Bâtie à l'origine en opposition au cognitivisme, l'approche incarnée de la cognition [38] considère que l'esprit, le corps et l'environnement jouent un rôle dans le processus cognitif global. Au lieu de souligner le rôle de la manipulation de symboles et des opérations formelles, la cognition incarnée met l'accent sur le fait que les comportements intelligents émergent d'esprits situés, localisés dans un environnement, et incarnés, via l'expérience sensitive apprise. Dans cette logique, le corps influence l'esprit autant que l'esprit influence le corps, constituant un système dynamique.

Les applications pratiques de la cognition incarnée pour l'instanciation de système intelligents proviennent principalement de la robotique, une contribution majeure étant les travaux de Rodney Brooks (voir par exemple [4]). Brooks a combattu l'approche de construction de l'intel-

ligence *top-down* basée sur le raisonnement automatique, essentiellement parce que les ressources utilisées pour construire une représentation symbolique du monde, la traiter et planifier les actions d'un robot (1) sont très coûteuses en terme de ressources et (2) ne permettent pas de prendre en compte aisément les environnements changeants.

Observant que l'intelligence n'est pas la pensée (les algorithmes à base de fourmi en étant un exemple SMA bien connu), Brooks a proposé d'utiliser une approche sub-symbolique et partiellement réactive. L'intelligence émerge alors de l'interaction et des boucles de rétroaction entre l'environnement dynamique et le robot, alors même que la plupart des tâches ne sont pas traitées symboliquement [21]. Dans la même idée, les recherches actuelles en robotique s'appuie également sur les matériaux de fabrication de leurs corps, de façon à simplifier le module de contrôle. Les progrès récents en marche bipède proviennent ainsi d'une meilleur conception des membres, par l'utilisation de propriétés physiques telles que l'élasticité et l'absorption de chocs [22].

Il pourrait être contre-argumenté que puisque la communauté multi-agent s'occupe principalement d'agents purement logiciels, la cognition incarnée n'est pas pertinente pour notre domaine. Bien que la transposition directe de l'incarnation des agents ne semble pas une approche pertinente, deux arguments sont en faveur de notre intérêt : premièrement, un grand nombre de nos approches sont inspirées par les systèmes biologiques. Ainsi, fonder nos modèles sur des travaux de philosophie et de sciences cognitives ne signifie pas que ceux-ci doivent être reproduits exactement, mais ils peuvent être une source de concepts utiles, tels que les affordances [12], la cognition étendue [24] et l'énaction [33], lesquels peuvent être adaptés aux systèmes logiciels en environnements dynamiques. Deuxièmement, les agents autonomes sont désormais face à la même difficulté que les robots en terme de complexité de conception. Traçant le parallèle, nous pouvons utiliser le concept de corps logiciels pour déléguer une partie de la complexité des agents à une entité modulaire et distincte, simplifiant ainsi la conception des agents (esprits) dédiés au raisonnement symbolique et réduisant leur charge de calcul.

Il est alors nécessaire de distinguer deux approches, la cognition incarnée *forte* et la cognition incarnée *faible*, de façon similaire à l'intelligence artificielle forte et faible. L'argument pour une cognition incarnée forte est que puisque l'intelligence humaine est profondément incarnée, la production d'une intelligence artificielle "réelle" (ou forte) nécessite d'utiliser également une approche incarnée. L'argument pour une cognition incarnée faible est qu'inspirer nos travaux de la façon dont les humains et animaux produisent leur comportement intelligent en minimisant leur charge cognitive ouvre des voies pour concevoir de meilleurs systèmes - en termes d'adaptativité, de qualité et de robustesse. De plus, l'approche incarnée est naturellement adaptée aux besoins de la simulation, de façon à reproduire et comprendre les comportements réels.

Dans cet article, j'adopte une approche faible de la cognition incarnée. De façon à en utiliser les concepts, il faut comprendre comment mettre en oeuvre une triade agent/corps/environnement et proposer une architecture compatible. En particulier, définir le rôle et les responsabilités du corps logiciel permettrait de diffuser son utilisation. Dans la section suivante, nous illustrons comment cette approche peut être utilisée pour concevoir une architecture de calcul des émotions.

3 Illustration : calcul des émotions

La simulation de systèmes biologiques toujours plus complexes, tels que les humains ou les systèmes sociaux, est difficile notamment parce qu'ils traitent de phénomènes situés à plusieurs échelles.

Ainsi, les émotions évoluent en fonction de trois influences [7] : les événements ponctuels, la dynamique temporelle, et la contagion émotionnelle. Traditionnellement, tous les processus sont intégrés dans l'architecture de l'agent, voir par exemple [14, 17]. Si l'évaluation de l'impact des événements est nécessairement traitée par le processus interne de l'agent, il est possible de décentraliser les autres mécanismes dans le corps logiciel de l'agent et dans l'environnement.

Bien qu'il n'y ait pas de consensus sur la façon dont les émotions sont gérées dans les systèmes biologiques, de nombreux modèles computationnels ont été proposés. Dans [30], nous avons fondé notre modélisation sur la thèse selon laquelle le calcul des émotions face à la perception d'un événement est le résultat d'un processus dual intuitif et cognitif [28]. Le premier

est semi-automatique et souvent inconscient. Il représente les modifications immédiates liées à un percept émotionnel, et concerne les émotions dites primaires (telles que la joie et l'amusement). Le second est une évaluation cognitive qui dérive de la cohérence entre les croyances, buts, percepts de l'agent et ses émotions courantes, et résulte en des émotions primaires et secondaires (telles que la honte ou la fierté).

À cette évaluation de l'impact émotionnel des événements perçus, s'ajoute une composante temporelle : d'une part, les émotions d'un individu tendent vers un niveau neutre (dépendant de l'individu) si aucun événement ne se produit ; d'autre part les variations d'émotions entre deux instants successifs sont limitées.

Enfin, la contagion émotionnelle est nécessaire à l'émergence de comportements collectifs cohérents. Hatfield *et al.* [15] ont montré que la contagion émotionnelle se situe à un niveau de conscience significativement plus bas que l'empathie, grâce à des processus automatiques, *i.e.* non contrôlés. Ces processus automatiques sont à la fois perceptifs et actifs, notamment via les comportements d'imitation, qui font partie des normes sociales du dialogue.

Ces mécanismes sont typiquement candidats à une "externalisation" hors de l'agent, puisqu'ils ne sont pas contrôlés par l'esprit dans les systèmes biologiques. De façon à proposer une architecture adéquate pour la contagion émotionnelle, un système multi-agent inspiré de la cognition incarnée va reposer sur deux concepts : un environnement actif et une séparation corps/esprit. Comme nous l'avons mentionné dans l'introduction, l'environnement peut être en charge d'accéder et de diffuser une partie des informations sur l'état des agents [37]. Dans le contexte de la modélisation des émotions, l'environnement peut récupérer les états émotionnels des agents et prendre en charge le calcul du résultat de la contagion émotionnelle à la place des agents.

Dans la même logique, nous pouvons considérer que l'agent est composé de deux parties : son esprit et son corps [23]. L'esprit contient le processus de décision de l'agent et est autonome, tandis que le corps est influencé par l'esprit, mais contrôlé par l'environnement. Cela correspond au fonctionnement humain : bien que l'esprit puisse prendre n'importe quelle décision arbitrairement, les limites à la réalisation de ces décisions sont imposées à la fois par les capacités du corps et par les règles de l'environnement.

En pratique, cette proposition implique que les états du corps de l'agent sont observables et que leur accessibilité est gérée par l'environnement, même pour l'agent lui-même. Pour le calcul des émotions (figure 1), le résultat des perceptions d'événements (1) est la responsabilité de l'esprit, les dynamiques temporelles (2) sont gérées par le corps et la contagion émotionnelle (3) par l'environnement. C'est alors la fusion de ces trois influences qui permet de déterminer l'évolution de l'état de l'agent. Plus de détails sur cette architecture peuvent être trouvés dans [30].

Utilisation dans des processus cognitifs de haut niveau

Dans cet exemple de calcul des émotions, une forme basique de cognition incarnée est mise en oeuvre, puisqu'il est considéré que le corps a seulement deux propriétés : des dynamiques internes (déléguées par l'esprit) et des règles d'accessibilité (déléguées par l'environnement). De cette façon, nous avons montré dans [30] que cela permet (1) de proposer un modèle modulaire avec des responsabilité distinctes pour chaque composant et (2) de diminuer le temps de calcul global.

Cependant, le paradigme de la cognition incarnée [38] argue que l'ensemble du processus cognitif émerge de l'interaction entre esprit, corps et environnement. Deux exemples de processus incarnés de plus haut niveau sont la réalisation de tâche et la planification.

Dans [1], Ballard *et al.* étudient les stratégies utilisées par des humains pour reproduire des motifs de blocs colorés sous pression temporelle. Au lieu de mémoriser le motif, les sujets se réfèrent de façon répétée (à travers leur perception, détectée à l'aide d'un oculomètre) aux blocs situés dans le modèle. Cette action de perception est réalisée de façon stratégique pour obtenir des informations partielles à la volée, par exemple en vérifiant d'abord la couleur du bloc, puis ensuite sa localisation précise. De cette façon, les humains utilisent une stratégie de mémoire minimale, en utilisant l'environnement comme son meilleur modèle.

Dans [32], les auteurs utilisent un graphe d'information sensorielle pour encoder les informations de navigation d'un robot. Ce graphe est construit au fur et à mesure de l'exploration de la zone. De façon à gérer les espaces encore inexplorés, ils fournissent au robot de fausses informations sensorielles qui viennent compléter

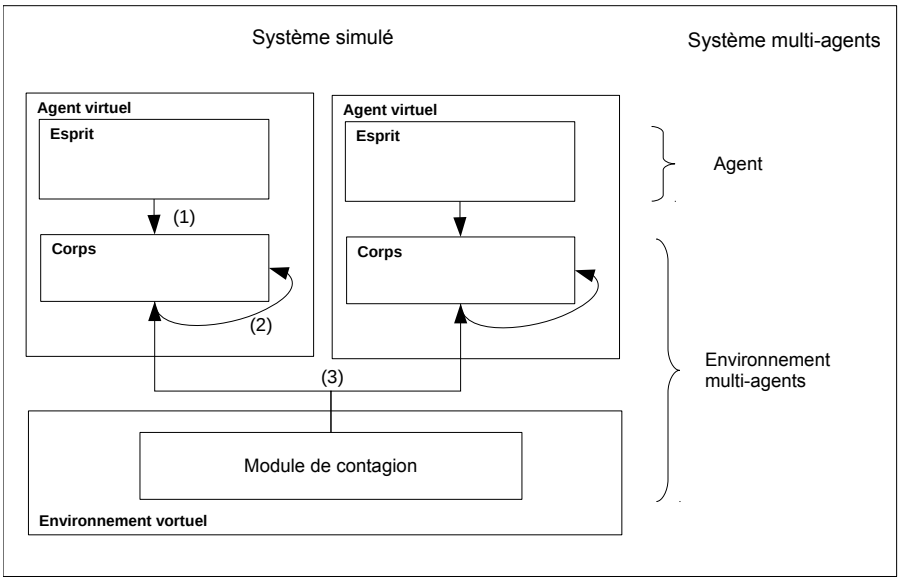


FIGURE 1 – Esprit, corps et environnement dans une architecture pour le calcul des émotions.

le graphe des endroits connus, au lieu d'utiliser une carte allocentrique. Le processus de planification est alors géré par auto-simulation au sein du graphe d'informations sensorielles, sans utiliser de traitement symbolique.

Au delà des aspects physiques

Investiguer le concept de corps implique d'également investiguer les différents types d'environnement dans lesquels l'agent interagit avec le monde. La métaphore physique est évidente, ce qui explique pourquoi c'est en premier dans le domaine de la simulation que le concept est apparu, par exemple dans [2, 30].

Cependant, l'environnement des agents n'est pas seulement physique, réel ou simulé. L'agent est également présent dans un ou plusieurs environnements sociaux, relationnels, et organisationnels. Dans chacune de ces dimensions, l'agent peut interagir avec les autres. Ceci permet d'envisager différentes utilisation du corps en fonction du type d'environnement dans lequel l'agent est plongé, voire l'utilisation de plusieurs corps, chacun étant rattaché à un environnement particulier. Dans les travaux de Soulié [31], chaque agent possède une instance dans chaque environnement auquel il participe.

Dans chacun de ces espaces, le corps de l'agent est utilisé comme une interface avec l'environnement, fournissant observabilité (ce qu'il montre), moyens d'actions (comment il influence le monde extérieur), interprétation (comment les capteurs fournissent l'information), dynamiques (évolutions temporelles) et réactions (traitement des stimuli de bas niveau). Par exemple, dans les systèmes cyber-physiques, il est possible d'utiliser un même esprit agissant dans des univers virtuels et physiques par le biais de corps différents. De la même façon, si on considère le corps comme le moyen d'exposer des attitudes vis à vis des autres, la notion de profil public dans les réseaux sociaux peut être considérée comme l'observabilité du corps d'un agent, ou le rôle de l'agent dans une société.

4 Agents incarnés - principes

Dans cette section, je propose une première définition des responsabilités du corps, et son impact en terme d'architecture du système multi-agent. La principale difficulté provient de la place particulière du corps, dont les états et capacités dépendent à la fois de l'esprit et de l'environnement, le plaçant en interface entre les deux. Cette proposition dérive des travaux de Platon *et al.* [23] sur les *softbodies*, des modèles ELMS [19] et MIC* [13], l'approche multi-environnementale de Soulié [31] et des modèles influence/réaction tels que [18]. Dans ces travaux, les effets des actions des agents sont calculés par l'environnement de façon à vérifier à l'exécution l'intégrité de l'environnement et à modéliser l'incertitude des résultats des actions.

Deux autres principes clés ont guidé cette pro-

position : d'une part, l'incertitude des résultats des actions nécessite des boucles de rétroaction pour que l'agent puisse contrôler et si nécessaire modifier ses futures sélections d'actions ; d'autre part le corps possède son propre ensemble de règles. Dans le modèle influence/réaction, les influences représentent le niveau microscopique, tandis que la réaction représente le niveau macroscopique. Cependant, des règles à la fois individuelles et collectives doivent être appliquées aux influences des agents, et les règles individuelles peuvent varier entre les agents, par exemple pour exprimer des capacités différentes. Ainsi, encapsuler les règles locales dans des entités locales, les corps, permet de modulariser la conception des dynamiques de l'environnement.

4.1 Responsabilités du corps logiciel

Le corps logiciel de l'agent encapsule les responsabilités suivantes :

1. **Accès aux ressources et observation de l'environnement** : le corps médie l'accès aux ressources dans l'environnement, et le processus de perception. Il fournit les moyens de définir grâce à des capteurs la perception de l'agent via des processus à la fois dirigés par l'agent lui-même (perception active) et par l'environnement (*awareness*). Il encapsule également les capacités d'action de l'agent, via ses influences.

2. **Accès et observation de l'agent** :
   — envers l'esprit, le corps permet l'introspection, autrement dit l'observation de ses états et processus par l'esprit ;
   — envers l'environnement, le corps fournit l'observabilité de l'état public de l'agent, et une interface pour prendre en compte les influences externes

3. **Dynamiques et règles propres** : le corps régule et contrôle l'ensemble des interfaces de l'esprit via un ensemble de règles – d'action, de perception et d'introspection. Il possède également un ensemble de dynamiques indépendantes ou en réaction à des influences de l'environnement ou de l'esprit.

4.2 Point de vue SMA (vue d'ensemble de l'architecture)

Intégrer l'approche incarnée nécessite de déplacer le focus de l'architecture vers une approche holistique environnement / corps / esprit. La figure 2 montre les différentes interactions entre ces trois éléments. Les agents incarnés sont premièrement situés dans un environnement, qui contient les ressources du système, les règles environnementales et les dynamiques du monde. Le corps médie l'ensemble des interactions entre agent et environnement. De plus, c'est une entité elle-même dynamique, dans le sens où elle encapsule des processus automatiques liés à l'agent, et une entité de régulation qui contient ses propres règles.

De cette façon si l'environnement et le corps sont des composants actifs, seul l'agent, qui correspond à l'esprit, est autonome, c'est à dire proactif envers ses buts.

Dans l'objectif d'améliorer la modularité, et puisque dans l'approche incarnée l'esprit n'a pas le contrôle total de son corps, la présence de rétroactions permet à l'agent de s'adapter à son corps. L'esprit est donc également impacté dans la mesure où il doit apprendre à utiliser son corps en fonction des capacités et dynamiques internes de celui-ci.

Les règles de l'environnement et celles du corps doivent être compatibles. Dans l'approche actuelle, les règles de l'environnement encapsulent à la fois des règles liées à l'individu (capacités) et des règles environnementales (telles que des règles physiques ou des normes), voir par exemple les règles de perception de [29]. Dans cette nouvelle approche, les deux types de règles sont séparées puisqu'elles sont sémantiquement et fonctionnellement distinctes.

5 Discussion et perspectives

Si cet article se focalise sur la notion de corps et, par là même, sur la décomposition architecturale du système multi-agent, c'est parce qu'il s'agit de l'élément le moins couramment utilisé et le moins normé dans la littérature, alors qu'il s'agit d'un concept primordial pour la cognition incarnée. Cependant, c'est bien du système agent - corps - environnement qu'émerge le processus cognitif global, composé des dynamiques propres de chacun de ses éléments et de leurs interactions.

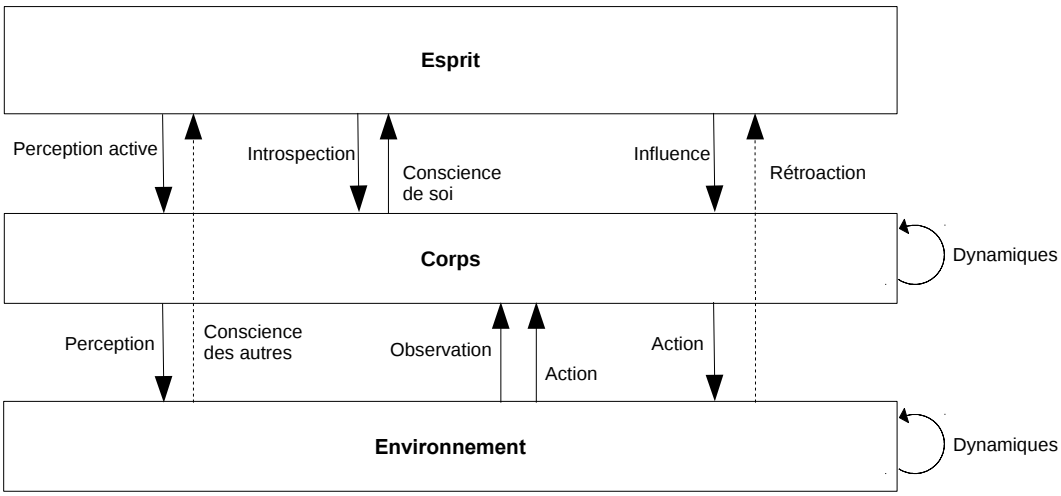


FIGURE 2 – Interactions esprit - corps - environnement.

Statut du corps de l'agent

Introduire la notion de corps pour les agents et la conception de systèmes multi-agents pose la question de leur statut. Bien que l'entité virtuelle précédemment nommée agent contienne conceptuellement à la fois un corps et un esprit, par exemple dans le cas de la simulation, la conception logicielle pratique de ce concept peut se matérialiser de trois façons : le corps peut être autonome, contrôlé par l'agent ou contrôlé par l'environnement.

Bien que toutes ces solutions soient meilleures en termes de modularité et de complexité que d'intégrer tous les processus dans les agents, une approche mixte qui corresponde aux principes conceptuels de la cognition incarnée telle qu'exposée dans les sections précédentes est la plus cohérente : le corps n'est pas autonome, mais influencé par l'esprit et régulé par l'environnement.

Les corps des agents peuvent être contrôlés par l'environnement, dont les services sont généralement matérialisés dans la plateforme multi-agent. Certains auteurs proposent cependant de créer des corps logiciels qui ne font pas partie de l'environnement, par exemple dans [34] pour gérer l'animation d'agents conversationnels. L'idée d'utiliser l'esprit pour générer les décisions de haut niveau et déléguer l'implémentation de ces décisions au corps est effectivement au coeur de l'approche incarnée computationnelle.

Problèmes opérationnels

Directement liées au statut du corps sont les problématiques de mise en oeuvre de l'approche. Notons premièrement que l'environnement et les corps des agents sont des composants fonctionnels. Ainsi, une centralisation fonctionnelle du contrôle du corps par l'environnement ne signifie pas nécessairement que l'environnement lui-même est centralisé.

La communauté des agents virtuels a monopolisé l'usage du terme d'agent incarné. Dans ce domaine, cela signifie que l'agent a une représentation visuelle qui est utilisé en tant qu'interface avec l'utilisateur. Cependant, cette représentation visuelle n'est pas nécessairement modélisée en tant que composant distinct de l'agent. Ainsi, il faut distinguer deux types d'incarnation : représentative ou conceptuelle. L'agent peut alors ne pas avoir d'incarnation, n'en avoir qu'une ou les deux. Par exemple, comme nous l'avons mentionné précédemment, certaines recherches proposent l'utilisation d'un corps logiciel [34] pour contrôler l'animation d'agents conversationnels et ainsi décharger une partie des processus décisionnels de l'agent. Dans ce cas, bien que l'utilisation du corps soit principalement utilisée pour décharger la partie motrice et non sensorielle de la cognition, les deux types d'incarnation, conceptuelle et représentative, sont utilisées.

Il n'existe à notre connaissance pas de modèle générique de corps pour les agents logiciels. Cependant, un certain nombre de sources d'inspi-

ration peut être trouvée, notamment dans la littérature sur les environnements multi-agents.

Platon et al. [23] ont introduit le concept d'*oversensing* : les agents ont des corps logiciels, possédant leurs états publics, qui sont médiés à la fois en terme d'accessibilité et d'observabilité par l'environnement. L'information sur les modifications de ces états est diffusée automatiquement par l'environnement aux autres agents, de façon à gérer la notion de conscience des autres. Cependant, dans ce cas, le modèle est conçu pour l'observabilité et la perception et non l'action. Soulié [31] propose des délégations d'actions et de perceptions dans un cadre multi-environnement, ainsi que la capacité de traduire les percepts de bas-niveau en concepts utilisables par le système conatif. Le modèle ELMS [19] propose de considérer le corps de l'agent comme encapsulant les aspects physiques de l'agent : ce modèle ajoute des capacités d'action et des contraintes de perception au rôle d'observabilité externe.

Les artifacts [27] peuvent être une façon d'implémenter des agents incarnée, en programment le corps et les responsabilités des agents en tant que réactions et dynamiques. Dans ce cas, les corps seraient des artifacts particuliers, puisque nécessairement liés à un agent. MIC* [13] définit les objets d'interaction comme des moyens d'interaction avec les autres agents et l'environnement tout en préservant l'autonomie de l'agent. Les institutions électroniques [8] conjuguent souvent règles sociales et contrôle local. Ces lois locales (voir par exemple [39]) peuvent être assimilées à la fonction de régulation des corps logiciels, et mis en oeuvre par leur biais dans une plateforme multi-agent.

Ces travaux fournissent des éléments de réflexion pour la modélisation d'agents situés et incarnés interagissant avec leur environnement. Cependant, ils ne considèrent pas l'interface entre esprit et agent avec la même acuité. Comme nous l'avons mentionné auparavant, le corps joue également un rôle dans le processus cognitif de l'agent, en fournissant des rétroactions, de l'introspection, et des capacités de conscience de soi. De plus, tout comme dans les systèmes biologiques, le corps peut encapsuler des capacité de traduction, c'est à dire des moyens de fonder les symboles manipulés par les processus de haut niveau sur des expériences sensorimotrices à partir de l'expérience des agents. Pour modéliser ce processus, des travaux tels que [6, 31] proposent d'inclure une couche entre la "réalité" et l'esprit qui contient le modèle conceptuel du monde, filtrant ainsi les percepts en une information sémantique grâce à une ontologie.

Remarquons que le corps et l'environnement doivent être séparés : ils fournissent des services distincts, principalement différenciés par le niveau auquel ils sont appliqués. Les responsabilités du corps sont dédiées à l'esprit (agent) auquel il est rattaché, tandis que l'environnement gère les interactions sociales et ressources externes.

Autonomie de l'agent

Concernant l'autonomie des agents, plusieurs travaux [18, 2, 30] ont montré que séparer l'esprit et le corps des agents permet d'assurer que l'agent n'est pas vulnérable à l'environnement et aux autres agents. En considérant l'autonomie comme l'intégrité interne de l'agent [13], *i.e.* "*une contrainte de programmation qui définit l'agent comme un système délimité comprenant une structure et des dynamiques internes qui ne sont ni contrôlables ni observables directement par une entité externe*", le corps est un concept approprié pour assurer l'autonomie de l'esprit. Il encapsule les mécanismes d'action et de perception, créant ainsi un espace tampon observable et accessible sans modifier l'état des agents eux-mêmes.

Changement de conception

La cognition incarnée est une approche holistique de l'intelligence. Ainsi, un changement de conception est nécessaire pour son intégration dans nos pratiques. Inspiré de l'expérience de la robotique, les objectifs principaux sont (1) de réduire la dépendance à l'approche symbolique à ce pour quoi elle est vraiment utile, par exemple la cognition 'hors-ligne' (telle que l'utilisation de la mémoire ou la planification de long-terme déconnectée des mécanismes sensorimoteurs), en se fondant plutôt sur des solutions adaptatives rapides liées au caractère situé du processus cognitif, (2) d'améliorer la modularité et de réduire la complexité de conception en séparant conceptuellement les responsabilités de chaque partie de la cognition, et (3) de trouver de nouvelles inspirations dans les systèmes biologiques.

Ce dernier point est un défi important. La psychologie du développement de Piaget (exposée par exemple dans [9]) souligne l'importance des

automatisme acquis dans le comportement intelligent des adultes. Les circuits sensorimoteurs, profondément ancrés dans les processus cognitifs matures, permettent de décharger l'esprit d'une partie de ses tâches. Dans les systèmes multi-agents, ceci implique que les boucles de rétroactions devraient être utilisées par l'agent pour acquérir une maîtrise de son corps, *i.e.* des motifs de bonne utilisation de ses capacités dans l'exécution de ses tâches. Une proposition à ce sujet est celle de Ribeiro *et al.* [25]. Les auteurs proposent d'utiliser des modules pour émuler la composition de processus entre capteurs et actionneurs, sans *a priori* sur le niveau de cognition concerné (esprit ou corps).

L'objectif de cet article n'est pas d'opposer l'intelligence artificielle symbolique et l'intelligence artificielle incarnée. L'intelligence artificielle symbolique a prouvé son utilité pour la conception de nombreux systèmes intelligents [11]. La difficulté principale dans la prise en compte effective de l'approche située et incarnée n'est pas de concevoir des systèmes réactifs efficaces (tels que les algorithmes à base de fourmis ou les véhicules automatiques), ni de concevoir des intelligence artificielles puissantes (tels que Deep Blue [5]). C'est de concevoir des solutions flexibles à des problèmes de haut niveau, adaptatifs aux changements de l'environnement (nécessitant donc des processus permettant la conscience de ces changements et s'appuyant sur la perception plutôt que la mémorisation et le calcul hors-ligne) pour produire des solutions cognitives non-triviales. Il s'agit également de pouvoir concevoir un seul esprit pour plusieurs modalités de mise en oeuvre (corps), ce qui implique des esprits adaptatifs et apprenants.

L'utilisation de plusieurs corps pour un même esprit permet également de mettre en oeuvre une hétérogénéité qui n'est pas directement modélisée dans le processus de raisonnement [31, 10]. Sur ce principe, les approches évolutionnaires pourraient être mises à profit de deux façons : sélectionner les meilleurs corps pour la réalisation d'une tâche, ou calibrer les systèmes multi-agents.

Finalement, formaliser l'approche fondée sur les agents incarnés, dans laquelle les relations entre esprit, corps et environnement(s) sont formalisées permettrait de faciliter la modélisation des processus humains (ou inspirés des humains). En particulier, le choix de retirer du module de contrôle (*i.e.* retirer de l'agent) une partie des calculs de bas-niveau, tels que la contagion émotionnelle ou les aspects physiques / interactionnels réactifs, permet de simplifier l'architecture des agents. Des recherches approfondies sont alors nécessaires pour mieux comprendre la façon dont les concepteurs de systèmes multi-agents peuvent appliquer ce principe à tous les cas dans lesquels les agents sont situés et interagissent avec leur environnement, quel qu'il soit, et proposer une méthodologie de conception adéquate.

Références

[1] D. H. Ballard, M. M. Hayhoe, P. K. Pook, and R. P. Rao. Deictic codes for the embodiment of cognition. *Behavioral and Brain Sciences*, 20(04) :723–742, 1997.

[2] F. Behe, S. Galland, N. Gaud, C. Nicolle, and A. Koukam. An ontology-based metamodel for multiagent-based simulations. *International Journal on Simulation Modelling, Practice, and Theory*, 40 :64–85, Jan. 2014.

[3] O. Boissier, R. H. Bordini, J. F. Hübner, A. Ricci, and A. Santi. Multi-agent oriented programming with jacamo. *Science of Computer Programming*, 2011.

[4] R. A. Brooks. Intelligence without representation. *Artificial intelligence*, 47(1) :139–159, 1991.

[5] M. Campbell, A. J. Hoane Jr, and F.-h. Hsu. Deep blue. *Artificial intelligence*, 134(1) :57–83, 2002.

[6] P. H.-M. Chang, K.-T. Chen, Y.-H. Chien, E. Kao, and V.-W. Soo. From reality to mind : A cognitive middle layer of environment concepts for believable agents. In *Environments for Multi-Agent Systems*, pages 57–73. Springer, 2005.

[7] A. Czaplicka, A. Chmiel, and J. A. Holyst. Emotional agents at the square lattice. *ACTA PHYSICA POLONICA A*, 117 :688–694, 2010.

[8] M. Esteva, J. Rodriguez-Aguilar, B. Rosell, and J. Arcos. Ameli : An agent-based middleware for electronic institutions. In *Proceedings of the 3rd International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2004)*, pages 236–243. ACM Press, 2004.

[9] J. H. Flavell. *The developmental psychology of Jean Piaget.* D Van Nostrand, 1963.

[10] S. Galland, N. Gaud, J. Demange, and A. Koukam. Environment model for multiagent-based simulation of 3D urban systems. In *the 7th European Workshop on Multiagent Systems (EUMAS09)*, Ayia Napa, Cyprus, dec 2009. Paper 36.

[11] J.-G. Ganascia. Epistemology of ai revisited in the light of the philosophy of information. *Knowledge, Technology & Policy*, 23(1-2) :57–73, 2010.

[12] J. J. Gibson. *The Ecological Approach to Visual Perception*. Lawrence Erlbaum Associates, 1979.

[13] A. Gouaïch, F. Michel, and Y. Guiraud. Mic* : a deployment environment for autonomous agents. In D. Weyns, H. V. D. Parunak, and F. Michel, editors, *Proceedings of Environment for Multi-Agent Systems, Workshop held at the third AAMAS conference*, volume 3374 of *LNAI*, pages 109–126. Springer Verlag, 2005.

[14] J. Gratch and S. Marsella. A domain-independent framework for modeling emotion. *Cognitive Systems Research*, 5(4) :269 – 306, 2004.

[15] E. Hatfield, J. Cacioppo, and R. Rapson. *Emotional contagion*. Cambridge Univ Pr, 1994.

[16] J. Haugeland. *Artificial intelligence : The very idea*. The MIT Press, 1989.

[17] M. Lhommet, D. Lourdeaux, and J.-P. BarthÃ¨s. Never alone in the crowd : A microscopic crowd model based on emotional contagion. In *Web Intelligence and Intelligent Agent Technology (WI-IAT), 2011 IEEE/WIC/ACM International Conference on*, volume 2, pages 89 –92, aug. 2011.

[18] F. Michel. Le modèle irm4s, de l'utilisation des notions d'influence et de réaction pour la simulation de systèmes multi-agents. *Revue d'Intelligence Artificielle*, 21(5-6) :757–779, 2007.

[19] F. Y. Okuyama, R. H. Bordini, and A. C. da Rocha Costa. Elms : An environment description language for multi-agent simulation. In *Environments for Multi-Agent Systems*, pages 91–108. Springer, 2005.

[20] H. Parunak and D. Weyns, editors. *Journal of Autonomous Agents and Multi-Agent Systems, Special issue on environments for multi-agent systems*, volume 14(1). Kluwer Academic Publishers-Plenum Publishers, 2007.

[21] R. Pfeifer and J. Bongard. *How the body shapes the way we think : a new view of intelligence*. MIT press, 2007.

[22] R. Pfeifer and A. Pitti. *La révolution de l'intelligence du corps*. Éditions Xanadu, Paris, 2012.

[23] E. Platon, N. Sabouret, and S. Honiden. Tag interactions in multiagent systems : Environment support. In *Proceedings of Environment for Multi-Agent Systems, Workshop held at the Fifth Joint Conference in Autonomous Agents and Multi-Agent Systems*, volume 4389 of *Lecture Notes in Artificial Intelligence*, pages 106–123. Springer Verlag, 2007.

[24] D. Pritchard. Cognitive ability and the extended cognition thesis. *Synthese*, 175(1) :133–151, 2010.

[25] T. Ribeiro, M. Vala, and A. Paiva. Censys : A model for distributed embodied cognition. In *Intelligent Virtual Agents*, pages 58–67. Springer, 2013.

[26] A. Ricci, A. Omicini, M. Viroli, L. Gardelli, and E. Oliva. Cognitive stigmergy : Towards a framework based on agents and artifacts. In D. Weyns, H. V. D. Parunak, and F. Michel, editors, *Proceedings of Environment for Multi-Agent Systems, Workshop held at the fifth AAMAS conference*, volume 4389 of *LNAI*, pages 124–140. Springer Verlag, 2007.

[27] A. Ricci, M. Piunti, M. Viroli, and A. Omicini. Environment programming in cartago. In *Multi-Agent Programming :*, pages 259–288. Springer, 2009.

[28] F. Rosis, C. Castelfranchi, P. Goldie, and V. Carofiglio. Cognitive evaluations and intuitive appraisals : Can emotion models handle them both ? *Emotion-Oriented Systems*, pages 459–481, 2011.

[29] J. Saunier, F. Balbo, and S. Pinson. A formal model of communication and context awareness in multiagent systems. *Journal of Logic, Language and Information*, 23(2) :219–247, 2014.

[30] J. Saunier and H. Jones. Mixed agent/social dynamics for emotion computation. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, pages 645–652. International Foundation for Autonomous Agents and Multiagent Systems, 2014.

[31] J.-C. Soulié. *Vers une approche multi-environnements pour les agents*. PhD thesis, Université de la Réunion, 2001.

[32] L. A. Stein. Imagination and situated cognition. *Journal of Experimental & Theoretical Artificial Intelligence*, 6(4) :393–407, 1994.

[33] J. Stewart, O. Gapenne, and E. A. Di Paolo. *Enaction : toward a new paradigm for cognitive science*. The MIT Press, 2010.

[34] M. Thiebaux, S. Marsella, A. Marshall, and M. Kallmann. Smartbody : Behavior realization for embodied conversational agents. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 1*, pages 151–158, 2008.

[35] F. J. Varela, E. Rosch, and E. Thompson. *The embodied mind : Cognitive science and human experience*. MIT press, 1992.

[36] D. Weyns, N. Boucké, and T. Holvoet. Gradient field-based task assignment in an agv transportation system. In *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, pages 842–849. ACM, 2006.

[37] D. Weyns, A. Omicini, and J. Odell. Environment as a first-class abstraction in multi-agent systems. *Autonomous Agents and Multi-Agent Systems*, 14(1) :5–30, Feb. 2007. Special Issue on Environments for Multi-agent Systems.

[38] M. Wilson. Six views of embodied cognition. *Psychonomic bulletin & review*, 9(4) :625–636, 2002.

[39] W. Zheng, C. Serban, and N. Minsky. Establishing global properties of multi-agent systems via local laws. In D. Weyns, H. V. D. Parunak, and F. Michel, editors, *Proceedings of Environment for Multi-Agent Systems, Workshop held at the fifth AAMAS conference*, volume 4389 of *LNAI*, pages 170–183. Springer Verlag, 2007.

Gestionnaire multiagent de contexte pour les applications d'intelligence ambiante

A. Sorici[†,⋆] sorici@emse.fr
G. Picard[†] picard@emse.fr
O. Boissier[†] boissier@emse.fr
A. Florea[⋆] adina.florea@cs.pub.ro

[†]Laboratoire Hubert Curien UMR CNRS 5516
Institut Henri Fayol, MINES Saint-Etienne, France

[⋆]University Politehnica of Bucharest
Department of Computer Science, Romania

Résumé

L'intelligence ambiante (AmI) connaît actuellement un développement croissant d'applications. Afin de mettre en place flexibilité d'installation et généricité dans le déploiement et l'approvisionnement de solutions de gestion de contexte, nous proposons CONSERT, un integiciel de gestion du contexte (CMM), basé sur les techniques et les principes du Web sémantique et des systèmes multiagents. Dans cet article, nous nous appliquons à montrer comment l'architecture multiagent de ce CMM offre la souplesse nécessaire pour déployer différents types de schémas de provisionnement de contexte pour répondre à différentes applications d'AmI. Nous présentons l'utilisation de notre solution avec un scénario issu du domaine de la gestion d'une université "intelligente".

Mots-clés : *Intelligence ambiante, gestion du contexte, systèmes multiagents, web sémantique, flexibilité du déploiement*

Abstract

Ambient Intelligence (AmI) is experiencing an increasing development of applications. As to install flexibility and genericity in the deployment and provisioning of context management solutions, we propose CONSERT, a CMM, based on techniques and principles from the Semantic Web and Multiagent Systems domains. In this paper we focus on showing how the multiagent architecture of this CMM provides the necessary flexibility to deploy different kinds of context provisioning patterns to address different AmI applications. We showcase the usage of our solution with a scenario from the domain of smart university life management.

Keywords: *Ambient Intelligence, Context Management, Multiagent Systems, Semantic Web, Deployment flexibility*

1 Introduction

Aujourd'hui, l'industrie commence à se saisir de scénarios et d'idées issues du domaine de l'intelligence ambiante (AmI) : surveillance de foyers, assistance à la vie autonome, villes intelligentes, etc. Couplée avec les visions *Internet-of-Things* et *Sensing-as-a-Service* [1], l'espace des applications possibles s'élargit rapidement. Une réelle emphase est mise sur la flexibilité du déploiement et l'adaptation en cours de fonctionnement de la gestion du contexte, pour rendre ces applications capables d'exécuter les bonnes actions au bon moment. C'est pourquoi ces dernières années, les intergiciels de gestion du contexte (ou CMM) sont devenus essentiels dans le développement de telles applications ubiquitaires et sensibles au contexte.

Si les développements continuent dans cette direction, les CMM vont devoir fournir des solutions de provisionnement de contexte dynamiques et multi-dimensionnelles. Alors que les solutions proposant des méthodes pour la gestion du contexte (e.g. mécanismes d'acquisition et de raisonnement, architectures de calcul) existent, à notre connaissance, aucune ne prend en charge la spécification et le contrôle d'un déploiement flexible, en cours de fonctionnement, d'une architecture de provisionnement du contexte. L'objectif de nos travaux est de créer un CMM flexible et configurable pour répondre à ces limitations.

Notre approche consiste à utiliser les techniques et principes du Web sémantique et des systèmes multiagents pour définir un CMM, appelé CONSERT. Les ontologies et les traitements sémantiques sont utilisés pour définir un modèle de contexte extensible et expressif, doté de méta-propriétés (e.g. source, validité temporelle, paramètres de qualité), sur lequel nous pouvons définir des méthodes de raisonnement et de provisionnement de contexte [2].

Afin de fournir la flexibilité nécessaire, nous embarquons ensuite le mécanisme de raisonnement obtenu au sein d'une architecture qui *agentifie*

le processus de provisionnement du contexte en différents agents qui coopèrent les uns avec les autres pour gérer de manière flexible et pour fournir le contexte aux applications s'appuyant sur le CMM CONSERT. La flexibilité du provisionnement de contexte tire profit de la structuration des informations de contexte suivant des dimensions et des domaines prenant en charge à la fois le déploiement (e.g. centralisé/décentralisé, mobile/statique, permanent/temporaire) et le provisionnement de contexte (e.g. acquisition, dissémination). Dans cet article nous nous focalisons sur l'architecture multiagent et les mécanismes que nous utilisons pour prendre en charge et déployer un provisionnement de contexte flexible.

Dans la section 2, nous motivons notre proposition au travers d'un scénario et de l'analyse de la littérature qui soulignent les besoins qui ne sont pas (complètement ou partiellement) pris en charge dans les solutions existantes. La section 3 présente brièvement les fondements du CMM CONSERT en termes de représentation, de raisonnement et d'architecture. Les sections 4 et 5 décrivent ensuite notre solution pour permettre un déploiement et un provisionnement de contexte flexibles. Nous illustrons l'utilisation de notre approche et discutons de ses avantages et limitations dans la section 6, puis concluons et dressons quelques perspectives.

2 Motivations

Les motivations de nos travaux proviennent de l'évolution actuelle des applications d'AmI qui mettent en évidence la nécessité, pour les activités d'une même application, de mettre en place un provisionnement dynamique d'informations de contexte relatives à plusieurs lieux et situations. Nous illustrons et discutons ceci plus en détail et analysons les limites des travaux existants au regard des exigences identifiées.

2.1 Un cas d'utilisation d'AmI

Considérons un scénario issu d'une application (*smart university*) que nous sommes en train de développer afin de prendre en charge la vie académique des membres d'une université.

A 13h55, Alice, étudiante à l'UPB, est dans un tram, en chemin pour son cours d'informatique, prévu de commencer à 14h00. Le smartphone d'Alice reçoit une notification automatique du système de transport intelligent du tram l'informant que l'arrêt UPB sera atteint dans 7min. Prenant en compte que l'arrêt de tram est à 3min de marche de l'UPB, son smartphone prévient le service de gestion des activités pédagogiques à l'UPB qu'elle sera en retard de 5min pour son cours. Bob, le professeur d'informatique est inscrit à ce service. Il reçoit la notification d'Alice, et décide de retarder son cours de 5min pour l'attendre. Après le cours, Alice rencontre Cécile et Dan pour travailler sur leur projet d'AmI. En arrivant au laboratoire d'AmI, ils s'assoient autour d'un bureau libre et partagent leurs idées. A partir de capteurs de détection de postures et de niveau de bruit, le laboratoire d'AmI déduit que les trois personnes sont en pleine réunion ad-hoc*, et notifie leurs smartphones, qui mettent à jour automatiquement leur calendrier et se mettent en mode silencieux.*

Ce scénario met en lumière des propriétés importantes d'une application sensible au contexte qui impactent la façon suivant laquelle le provisionnement de contexte doit être pris en charge. Comme nous pouvons le voir dans l'application gérant l'emploi du temps et les activités universitaires d'Alice, les informations de contexte utilisées peuvent être partitionnées et structurées suivant plusieurs *domaines* logiques comme les lieux (le tram, le laboratoire d'AmI), les activités (cours d'informatique, réunion *ad-hoc*) ou les organisations (en tant qu'étudiant à l'UPB). De plus, une modélisation expressive du contexte et des capacités de raisonnement sont nécessaires pour faire face à la diversité et la richesse de ces informations.

Le scénario souligne également le besoin de mécanismes de provisionnement de contexte flexibles et adaptables. En effet, les domaines *ne doivent pas tous être provisionnés au même moment* (e.g. réunion *ad-hoc* dans le laboratoire, souscription aux mises à jour provenant du tram uniquement lorsqu'on est à l'intérieur). Ils peuvent être *dépendants* (e.g. les informations du tram influent sur le cours) ou *indépendants* (e.g. les informations du tram et celles de la réunion) les uns des autres. De plus, *différents degrés de complexité de raisonnement sur le contexte* sont requis (e.g. simple calcul de retard, détection de réunion *ad-hoc* dans le laboratoire).

2.2 Limitations des CMM actuels

Maintenant, analysons et positionnons certains CMM proposés ces dernières années. Nous nous tournons uniquement vers des propositions offrant une approche indépendante du domaine et flexible de la gestion du contexte.

Comme nous pouvons le voir dans PACE [6], CROCO [8] ou CoCA [9], il existe des solutions offrant une représentation expressive et des capacités de raisonnement. C'est le cas de PACE, basé sur le modèle CML (Context Modeling Language), ou pour CROCO ou CoCA avec leurs intergiciels de gestion du contexte ba-

sés sur des ontologies. CoCA offre des capacités de représentation riche du contexte, telles que le contenu, les méta-propriétés, la représentation des contraintes de dépendances, ou les relations temporelles. Dans ce paysage, Guo et al. [10] et les auteurs d'ACAI [11] proposent des infrastructures de gestion du contexte, où les applications peuvent considérer différents domaines de contexte comme des unités d'administration (e.g. maison, bureau, extérieur) qui peuvent contenir des contextes différents pour une même entité.

Concernant le provisionnement de contexte, PACE ne considère pas explicitement des unités pour l'acquisition et la dissémination de contexte. Grâce à l'utilisation d'Elvin [1] pour fournir les mécanismes de découverte de producteurs, de routage de mises-à-jour et de requêtes/souscriptions de consommateurs, il peut prendre en charge la mobilité et le cycle de vie dynamique des productions/consommations d'informations de contexte. Cependant, il ne prend pas en charge la configuration ou la gestion, en cours de fonctionnement, de ces dynamiques de provisionnement, comme requis dans notre scénario. Quant à CROCO, son architecture et ses exemples d'applications (gestion documentaire, co-navigation adaptative) [8] suggèrent une gestion centralisée sans prise en charge de la configuration du provisionnement de contexte. Dans CoCA [9] le provisionnement dynamique est partiellement offert par l'utilisation de méthodes heuristiques pour charger uniquement dans un raisonneur, le sous-ensemble du modèle de contexte le plus pertinent actuellement. Cependant, il nécessite encore l'existence d'un unique modèle de contexte. CoCA utilise un système P2P pour fournir la découverte ou les producteurs, et pour aider les consommateurs à trouver des services de gestion du contexte, mais l'intergiciel ne fournit pas explicitement d'unités pour consommer ou requêter. La gestion de ces aspects est laissée à l'entière responsabilité de l'application.

Alors qu'ACAI ou les travaux décrits dans [10] pourraient être utilisés pour mettre en œuvre la partie gestion de notre scénario (e.g. les gestionnaires du tram, du cours d'informatique et du laboratoire), ils sont trop rigides pour prendre en charge le développement d'applications à échelle plus réduite (e.g. raisonnement local sur le smartphone d'Alice). De plus, aucun support pour la configuration lors de la conception de l'infrastructure n'est proposé dans ces travaux. Le développement d'applications doit se conformer à une définition uniquement spatiale des domaines de contexte, excluant par là même toute autre dimension structurante comme les activités ou les relations.

COPAL [7] est le seul à adopter une conception d'intergiciel à base de composants et à proposer un langage spécifique au domaine (DSL) pour spécifier de manière déclarative toutes les dimensions de la gestion du contexte (e.g. définitions des types de contexte, spécifications des traitements de contexte, définitions des requêtes et des écouteurs). Le résultat d'une spécification utilisant COPAL-DSL est un *bundle* OSGi [2] qui, bien que les auteurs ne le mentionnent pas, peut permettre à l'application de gérer le cycle de vie d'un déploiement de COPAL en cours de fonctionnement (comme requis dans notre scénario par la succession d'interactions diverses). Cependant, COPAL ne permet que des déploiements centralisés.

Il est important de noter qu'aucun des travaux analysés ne fournit de moyen de structurer ou d'empaqueter les différentes unités de provisionnement de contexte qui ont besoin d'être utilisées uniquement dans certaines situations. Dans notre scénario, les écouteurs et raisonneurs sur le smartphone d'Alice qui exploitent le contexte du laboratoire d'AmI sont uniquement utilisés lorsque l'information de contexte précisant le lieu où se trouve Alice est observée. Les CMM actuels ne fournissent ni méthode de conception ni environnements d'exécution pour de tels mécanismes.

Tournons-nous maintenant vers la description générale du CMM CONSERT et voyons comment il aborde cette gestion dynamique de multiples modèles de contexte.

3 Survol de CONSERT

CONSERT est un CMM qui offre (i) un support pour la modélisation expressive et le raisonnement sur le contexte, et (ii) des options de déploiement flexible et des mécanismes adaptables de provisionnement de contexte. Nous fournissons ici une vue synthétique des propriétés principales de CONSERT, des points de vue de la représentation (l'*ontologie*), du raisonnement (le *moteur*) et de l'architecture (les *unités de gestion*).

3.1 Ontologie CONSERT

L'ontologie CONSERT définit un métamodèle du contexte [2], et des règles pour l'inférence de contexte et la spécification de contraintes (basées sur SPARQL et l'API SPIN [3]). Cette ontologie est structurée en trois modules –*core*, *annotation*

1. http://avis.sourceforge.net

2. http://www.osgi.org
3. http://spinrdf.org/

et *constraint*– qui définissent le méta-vocabulaire pour la représentation du contexte. Les développeurs peuvent les utiliser pour construire le *modèle de contexte* de leurs applications.

Le module *core*[4] définit le vocabulaire utilisé pour exprimer le contenu d'une information de contexte. Il est exprimé à partir d'une brique de base (*ContextAssertion*) décrivant la situation des entités en le connectant aux éléments (*ContextEntity*) qui jouent un rôle dans cette situation (e.g. une personne, un lieu ou un objet) et en les qualifiant grâce à des propriétés statiques (*EntityDescription*) (e.g. une relation d'inclusion spatiale entre deux lieux). Les *ContextAssertions* peuvent être n-aires et sont enrichies avec une spécification de leur méthode d'acquisition en cours de fonctionnement (captée, profilée ou dérivée). Par exemple, `locatedIn(alice, ami-lab)` est une *ContextAssertion* spécifiant la position d'Alice (où `Person(Alice)` et `UniversitySpace(ami-lab)` sont des *ContextEntities*), alors que `includedIn(ami-lab, cs-building)` est une *EntityDescription* fournissant la description de la relation d'inclusion entre deux *ContextEntities* de type `UniversitySpace`.

Le module *annotation*[5] définit le vocabulaire pour les métapropriétés (annotations) d'une *ContextAssertion* (e.g. source, validité temporelle, qualité). Il distingue les annotations *basiques* (e.g. source) et *structurées* (e.g. validité temporelle, incertitude).

Le module *constraint*[6] présente le vocabulaire pour définir les contraintes d'intégrité, d'unicité et de valeur sur les *ContextAssertions*. Les classes et les propriétés expriment une information complète sur la violation déclenchée, le type de la *ContextAssertion* affectée et les instances spécifiques d'information de contexte conflictuelles.

Des ontologies de plus haut niveau, comme SOUPA [3], peuvent être couplées à ce métamodèle pour fournir la base des *ContextEntities* d'un modèle de contexte d'une application. Notons cependant que l'ontologie de plus haut niveau peut être changée à volonté, en fonction des besoins de modélisation de l'application.

3.2 Moteur de raisonnement CONSERT

Dans CONSERT [2], un modèle de contexte développé avec l'ontologie est exploité en cours de fonctionnement par le moteur CONSERT. Ce composant logiciel est en charge des mises-à-jour de contexte, d'inférences de haut-niveau, de vérifications de contraintes et de cohérence,

4. http://purl.org/net/consert-core-ont
5. http://purl.org/net/consert-annotation-ont
6. http://purl.org/net/consert-constraint-ont

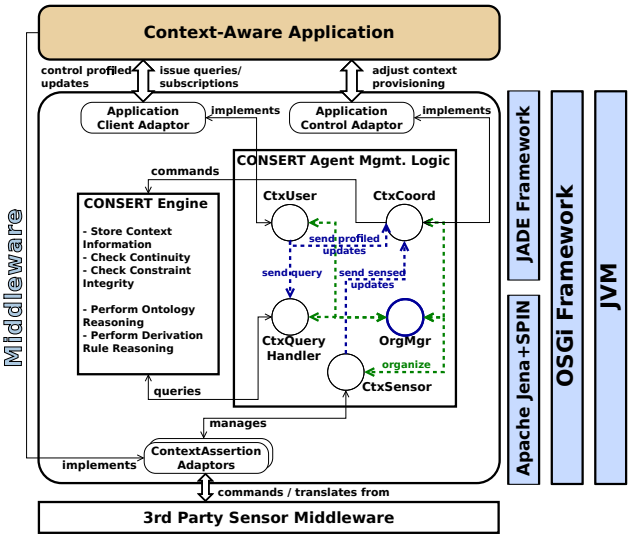


FIGURE 1 – L'intergiciel CONSERT : architecture multiagent et interactions

ainsi que de la prise en charge asynchrone des requêtes. Le mécanisme d'inférence du moteur utilise le traitement sémantique d'événements, en employant une approche par déduction basée sur des règles utilisant des requêtes `SPARQL CONSTRUCT` couplées à du raisonnement ontologique.

Le cycle de contrôle du moteur est gouverné par un comportement transactionnel lors duquel une série de vérifications est effectuée. Par exemple, lors de l'insertion d'une nouvelle instance de *ContextAssertion*, il vérifie la continuité (pour voir si la mise-à-jour est une partie d'une situation précédemment détectée), la violation de contraintes et le déclenchement d'inférences (pour voir si une mise-à-jour déclenche l'exécution d'une *DerivationRule*).

Le moteur est implanté comme un composant logiciel exposant et nécessitant des interfaces de services. De cette manière, le niveau applicatif peut personnaliser la procédure précédente. Par exemple, en cas de violation de contrainte, un *service de résolution de contraintes* (qui peut être personnalisé par les développeurs de l'application) décide quelles instances de *ContextAssertion* doivent être conservées dans la base de connaissances avant d'engager la transaction. Pour des raisons de limitation de place, nous ne détaillerons pas ces services.

3.3 Unité de gestion CONSERT (CMU)

CONSERT est architecturé comme un système multiagent où les agents encapsulent et contrôlent différentes fonctionnalités impliquées dans le cycle de vie du provisionnement de contexte [4] (voir Fig. 1).

Comme nous le verrons plus loin, ceci permet un contrôle plus flexible. Un ensemble de cinq types d'agents est défini. Ils sont regroupés en ce que nous appelons une unité de gestion CONSERT, ou *CONSERT Management Unit* (CMU). Pour une application donnée, de multiples CMU peuvent être déployées et organisées de différentes façons pour gérer et fournir les informations de contexte aux applications.

Un **`CtxSensor`** est responsable de la gestion des interactions avec les capteurs et avec l'agent `CtxCoord` de sa propre CMU, pour prendre en charge les commandes de provisionnement (e.g. démarrer/arrêter d'envoyer des mises-à-jour, changer la fréquence de mise-à-jour).

Le **`CtxCoord`** est responsable de la gestion du cycle de vie principal de la CMU. Il encapsule un moteur CONSERT pour contrôler le raisonnement et l'intégrité. Grâce à un `Application Control Adaptor`, le niveau applicatif peut commander cet agent pour fixer ou modifier les paramètres qui contrôlent le cycle de provisionnement de la CMU.

Le **`CtxQueryHandler`** est responsable de la dissémination des informations de contexte. Par défaut, il utilise une base de connaissances locale pour répondre aux requêtes. En cas de configurations plus complexes, impliquant plusieurs agents, il peut participer à un protocole de *fédération* (voir section suivante).

Un **`CtxUser`** est responsable de l'interfaçage avec l'application. Il expose une interface de service `Application Client Adaptor` qui permet de lancer des requêtes et des souscriptions, d'envoyer des informations de contexte profilées (e.g. agir comme un senseur) ou des descriptions d'entités statiques.

Le **`OrgMgr`** est responsable du contrôle du déploiement d'une CMU et de l'intergiciel en coopération avec d'autres `OrgMgr`. Il lance et contrôle les états (démarré/arrêté/désinstallé) des agents d'autres types. Il agit comme un agent pages jaunes, gère les aspects mobilité et maintient une vue globale de la hiérarchie des *Context Domains* dans le cas de déploiements décentralisés (voir section suivante).

Du point de vue de l'implantation, ces agents s'exécutent sur JADE [7], utilisant le cycle de raisonnement offert par cet intergiciel ainsi que la conformité FIPAdes protocoles d'interaction et les options de distribution (e.g. conteneurs JADE, protocoles de transfert de messages). Une approche par composants de service basée sur OSGi [8] est utilisée pour mettre en œuvre les comportements et les adapteurs des agents, afin d'augmenter la flexibilité de la gestion, en profitant des mécanismes de contrôle du cycle des services offerts par OSGi.

Cette section a présenté les éléments de bases qui soutiennent les fonctionnalités d'une instance de CONSERT. Dans la suite, nous expliquons comment déployer une ou plusieurs instances de CMU qui peuvent être configurées et gérées en fonctions des besoins applicatifs.

4 Déploiement du CONSERT

Les options de déploiement de CONSERT visent à répondre aux exigences soulevées par notre scénario illustratif, afin de réduire les efforts de développement et d'adaptation. Par exemple, en ce qui concerne la gestion des activités universitaires d'Alice, l'application gérant ses interactions dans l'université fait face à plusieurs dimensions et types d'informations de contexte. Nous distinguons deux types de *nœuds de calcul* sur lesquels peuvent être déployées les CMU : *fixes* (e.g. gestion des activités d'enseignement dans l'université, contrôle des capteurs du laboratoire AmI) et *mobiles* (e.g. les smartphones des membres de l'université). Le raisonnement sur le contexte peut prendre place sur un nœud fixe (e.g. déduire qu'une réunion ad-hoc prend place dans le laboratoire) ou mobile (e.g. calculer le retard estimé d'Alice).

D'après l'analyse de la section 2, une des limitations des intergiciels existants est l'incapacité à fournir aux développeurs des moyens d'organiser et de configurer l'ensemble des modules requis pour la gestion des informations de contexte utilisées dans une situation donnée. Dans cette section, nous introduisons tout d'abord les notions utilisées pour concevoir et spécifier le déploiement d'une application sensible au contexte reposant sur CONSERT. Ensuite, nous détaillons le vocabulaire de configuration que nous avons défini et comment il est manipulé par les agents `OrgMgr` pour déployer dynamiquement un CMM pour une application donnée.

4.1 Styles de déploiement

Un de nos objectifs est d'être capable d'utiliser CONSERT pour développer des applications à différentes échelles. Afin de définir différents styles de déploiement, nous enrichissons l'ontologie CONSERT avec deux concepts : *ContextDimension* et *ContextDomain*.

Une *ContextDimension* est une *ContextAssertion* binaire. Son sujet est une *ContextEntity* qui peut être généralement vue comme un *consommateur* d'information de contexte (e.g. un utilisateur).

7. http://jade.tilab.com
8. http://www.osgi.org/Technology/WhatIsOSGi

Les objets de cette relation sont des instances de *ContextEntity* qui définissent les *valeurs* de la dimension, représentant des *ContextDomains* (voir ci-après). Une dimension appartient à une des cinq catégories suivantes [5] : *individualité*, *espace*, *temps*, *activité* et *relations*. La plupart des applications de moyenne ou large échelle structurent leur processus de provisionnement de contexte suivant une de ces dimensions.

Un *ContextDomain* définit une partition logique du modèle de contexte de l'application, le long d'une *ContextDimension* choisie. Pour chaque domaine, l'application peut définir un processus spécifique de provisionnement de contexte (acquisition, coordination, dissémination). Par défaut, les *ContextDomains* issus d'une *ContextDimension* forment un réseau complet à plat. Ils peuvent également être organisés en une hiérarchie arborescente si les *ContextEntities* donnant ses valeurs à une *ContextDimension* sont caractérisées par des *EntityDescriptions* d'inclusion (comme l'exemple `includedIn(ami-lab, cs-building)` présenté plus haut).

Dans notre scénario, le smartphone d'Alice souscrit aux mises-à-jour de la vitesse du tram dans le *ContextDomain* `tram` et aux notifications de réunions ad-hoc dans le *ContextDomain* `ami-lab`. Ces domaines appartiennent à différentes *ContextDimensions* spatiales (`locatedIn(Person, PublicTransport)` et `locatedIn(Person, UniversitySpace)`, respectivement) et ont différents modèles de contextes et exigences de gestion. De manière similaire, les informations échangées entre Alice et le système de l'université qui gère le *ContextDomain* `cs-lecture` (cours d'informatique) appartient à la dimension `engagedIn(Person, TeachingActivity)`.

Pour ces deux concepts (*ContextDimension* et *ContextDomain*), nous pouvons définir différents styles de déploiement. Ici, nous allons nous concentrer sur deux d'entre eux : *centralisé-local* et *décentralisé-hiérarchique*.

Un schéma *centralisé-local* représente une organisation dans laquelle il y a un seul modèle de contexte d'application et une seule CMU. Les capteurs physiques et virtuels (i.e. les producteurs de contexte) sont gérés par les `CtxSensors` de cette CMU et l'application (i.e. le consommateur de contexte) interagit avec le `CtxUser` de la CMU. Un tel schéma peut être utilisé pour des applications fournissant une gestion intelligente de document sur un équipement comme dans [8] ou également pour des plates-formes "tout-en-un" de maison intelligente [7].

Un déploiement *décentralisé-hiérarchique* est configuré suivant une ou plusieurs *ContextDimensions* et les hiérarchies de *ContextDomain* qu'elles peuvent former. Un tel style de déploiement vise des applications à plus large échelle, avec des modèles de contexte distribués et de multiples CMU (organisées en hiérarchie, si nécessaire) qui comprennent à la fois des nœuds mobiles et fixes. Les capteurs sont gérés par les agents `CtxSensors` d'une seule CMU (pour préserver la localité des informations captées), mais les consommateurs de cette information peuvent appartenir à des CMU différentes (i.e. requêtes et souscriptions peuvent être envoyées entre CMU). Des exemples de ce style de déploiement se retrouvent dans les applications de ville intelligente, par exemple.

Comme nous le discuterons dans la section 6, dans notre scénario de référence, nous pouvons rencontrer les deux types de déploiement. Par exemple, les informations personnelles d'Alice (e.g. son agenda, sa vitesse de marche) sont gérées et évaluées par une application déployées dans une CMU installée sur son smartphone. D'un autre côté, la gestion des informations de contexte dans le laboratoire AmI et les autres salles de l'université est prise en charge par un déploiement décentralisé des CMU. De plus, les informations de contexte rassemblées par les CMU gérant le laboratoire AmI sont *consommées* par des CMU sur des nœuds mobiles (e.g. les smartphones des personnes se réunissant de manière ad-hoc) qui entrent dans une pièce ou qui la consultent à distance (voir section 5).

4.2 Spécifications de déploiement

Nous pouvons définir maintenant comment le déploiement de CONSERT est spécifié pour configurer les agents des CMU et affecter les CMU à des *ContextDomains*. Cette spécification utilise les technologies du Web sémantique et repose sur un vocabulaire défini dans une ontologie [9] (Fig. 2) qui traite des configurations de la plate-forme (`platform-config.ttl`), des agents (`agent-config.ttl`) et du modèle de domaine de contexte (`agent-config.ttl`). Ces fichiers de configuration, avec les fichiers OWL des ontologies définissant le modèle de contexte de l'application, sont encapsulés dans des *bundles* OSGi. Chaque *bundle* définit une CMU qui peut être installée, démarrée, arrêtée ou désinstallée en cours d'exécution de l'application. Une CMU se voit affectée la responsabilité de la gestion d'informations de contexte pour une paire *ContextDimension-ContextDomain*. Ceci signifie que pour chaque *ContextDimension* de l'application et pour chaque *ContextDomain* le long de cette dimension, il y aura un *bundle* OSGi conte-

9. http://purl.org/net/consert-deployment-ont

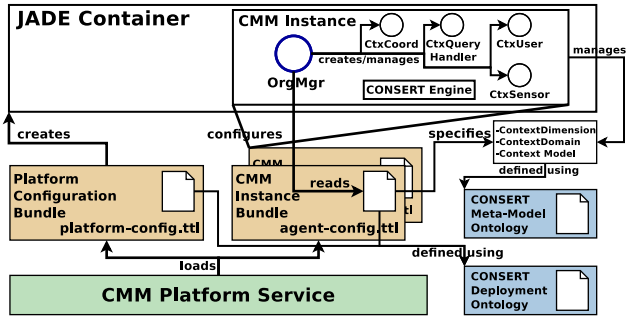


FIGURE 2 – Modèle de déploiement de l'intergiciel CONSERT

nant un fichier `agent-config.ttl` et des fichiers OWL définissant le modèle de contexte de ce *ContextDomain*.

Configuration de plate-forme. La spécification de la plate-forme (`platform-config.ttl`) fournit les informations techniques pour configurer un conteneur JADE pour la machine physique qui hébergera les agents s'exécutant dans les CMU. Par exemple, dans notre scénario, l'application sur le smartphone d'Alice est engagée dans plusieurs *ContextDomains* (e.g. le tram, le laboratoire AmI). Ainsi, de multiples CMU doivent s'exécuter sur le même nœud de calcul (son smartphone) pour aider l'application à interagir avec ces domaines.

Configurations des agents et du modèle de domaine du contexte. Le fichier de configuration `agent-config.ttl` fournit des indications quant aux types et au nombre d'agents CMM qui doivent être déployés dans le cadre d'une CMU associée à un *ContextDomain*. La configuration du modèle de contexte utilisé dans ce domaine est également spécifiée comme un ensemble de documents RDF. Ils contiennent les définitions des *ContextAssertions*, annotations, contraintes et règles d'inférence modélisées grâce à l'ontologie CONSERT.

Notons que dans un style *centralisé-local*, avec un seul modèle de contexte, tous les agents de la CMU résident dans le même conteneur. Normalement une seule instance de `CtxCoord`, de `CtxQueryHandler` et de `CtxUser` et une ou plusieurs instances de `CtxSensor` sont déployées. Dans un déploiement *décentralisé-hiérarchique* cependant, le type des agents composant une CMU déployée sur une machine dépendra du rôle de cette CMU : gestionnaire ou client du modèle de contexte associé à un *ContextDomain*. Dans notre scénario, par exemple, la CMU s'exécutant sur une machine gérant le contexte du laboratoire AmI contiendra une instance de `CtxCoord` et de `CtxQueryHandler`. Cependant, sur le smartphone d'Alice, une CMU associée au même *ContextDomain* laboratoire AmI contiendra une instance de `CtxUser`. Enfin, les capteurs requis pour détecter les réunions ad-hoc peuvent être pris en charge par les agents `CtxSensor` déployés dans la CMU de la machine de gestion de la salle, ou dans une CMU déployée sur un nœud dédié à chaque capteur, mais affectée au même *ContextDomain*.

Exécution du déploiement. Comme nous l'avons mentionné dans la section 3.3, l'agent `OrgMgr` est un médiateur entre le niveau applicatif et le CMU qu'il gère. En utilisant le service `CMMPlatformService`, l'application peut requérir l'exécution d'un CMU pour un *ContextDomain*. Etant le premier agent créé, l'`OrgMgr` lit les fichiers de configuration d'agents et déploie les agents configurés suivant la séquence : (i) `CtxCoord`, (ii) `CtxQueryHandler`, (iii) `CtxSensor`, puis (iv) `CtxUser`. L'`OrgMgr` maintient également une vue des *ContextDomains* existants en accord avec le style de déploiement spécifié. En fonction du type de CMU qu'il doit gérer, l'`OrgMgr` peut jouer le rôle de *root*, *node*, *central* ou *mobile*. Lorsqu'affecté au rôle de *root* (resp. *node*), un `OrgMgr` supervise le CMU qui gère et coordonne le provisionnement du domaine de plus haut niveau (resp. niveau intermédiaire) d'une hiérarchie de *ContextDomains*. Un rôle *central* est affecté à un `OrgMgr` dans un style de déploiement *centralisé-local*, alors que le rôle *mobile* correspond à un `OrgMgr` qui supervise les agents d'une CMU s'exécutant sur un nœud de calcul mobile qui est sujet à des changements de *ContextDomains*. Les rôles informent les `OrgMgr` des configurations spécifiques qui leur sont affectées. Un `OrgMgr` *root* se connectera à tous les autres `OrgMgrs` roots définis pour les *ContextDomains* de la même *ContextDimension*. Les `OrgMgrs` jouant le rôle de *node* sont conscients qu'ils font partie d'une hiérarchie et se connecteront ainsi à un parent et éventuellement enregistreront plusieurs agents `OrgMgrs` enfants. Un agent `OrgMgr` *central* sait qu'il est employé uniquement dans le cadre d'un équipement local, alors que dans le cas *mobile* l'agent réalise que son parent sera défini en cours de fonctionnement.

Comme ils maintiennent une vue de la hiérarchie, les agents `OrgMgrs` jouant les rôles de *root* et de *node* sont responsables d'informer les nœuds mobiles lorsqu'ils entrent ou sortent du *ContextDomain* qu'ils représentent. Il est également à leur charge d'aider les `CtxQueryHandlers` à router leur requêtes basées sur un domaine qui nécessitent une *fédération* (appel à plusieurs CMU pour obtenir le résultat). L'ontologie CONSERT définit des modèles pour la détection de nœuds

mobiles. Ils permettent au développeur de spécifier des requêtes SPARQL que l'agent `OrgMgr` utilise pour observer quand une *ContextAssertion* qui représente la *ContextDimension* du domaine qu'il gère a été insérée ou dérivée. Cependant, en raison d'une limitation de place, nous ne détaillerons pas les protocoles mis en œuvre pour la détection de nœuds mobiles ou pour le routage de requêtes fédérées.

5 Provisionnement de contexte

Une fois déployés par leur `OrgMgr`, les agents d'une CMU fournissent les informations de contexte du *ContextDomain* sous leur responsabilité, et relaient/accèdent à des informations de contexte provenant d'autres *ContextDomains*.

5.1 Provisionnement intra-CMU

Au sein d'une CMU, le processus de provisionnement relatif à un *ContextDomain* consiste en deux séquences d'interaction principales : (*i*) la *captation* qui concerne les mises-à-jour envoyées par les agents `CtxSensor` au `CtxCoord` d'une CMU, (*ii*) la *requête* qui correspond aux requêtes et souscriptions que les agents `CtxUsers` font au `CtxQueryHandler` d'une CMU.

Ces deux chaînes sont régies par des protocoles d'interaction dédiés. Pour la captation, un `CtxSensor` s'enregistre auprès d'un `CtxCoord` et utilise le protocole *FIPA Propose* pour publier ses capacités de mises-à-jour de *ContextAssertion*. Les mises-à-jour de capteurs sont envoyées via des messages *FIPA Inform*. Les interactions supportées par ce protocole sont adaptées et contrôlées par les agents impliqués par les politiques de captation et de coordination définies dans [12]. Dans le protocole d'interaction pour les *requêtes*, un agent `CtxUser` demande à un `CtxQueryHandler` une information via les protocoles *FIPA Query* ou *FIPA Subscribe*. Si les *ContextAssertions* demandées sont actives (suivant la politique de coordination du provisionnement), elles sont envoyées, sinon, le protocole continue avec des interactions entre les agents `CtxQueryHandler`, `CtxCoord` et `CtxSensor` de la CMU pour accéder à l'information requise (détails dans [12]).

Le provisionnement de contexte pris en charge par ces séquences d'interaction est principalement contrôlé par l'agent `CtxCoord` qui assure que le contexte échangé dans son *ContextDomain* reste toujours cohérent. Pour ce faire, il utilise les services de détection et de résolution de contraintes du moteur CONSERT. Il utilise également des statistiques en temps réel fournies par le moteur et les politiques de coordination du provisionnement [12] pour assurer *comment* et *quelles* informations de contexte doivent être provisionnées, suivant les besoins actuels de l'application. Ces décisions concernent l'activation/désactivation ou changement de fréquence des mises-à-jour, ou des changements dans la fonctionnalité du moteur CONSERT (e.g. activer/désactiver une règle de dérivation qui altère le service d'ordonnancement des inférences).

5.2 Provisionnement inter-CMU

Dans le style de déploiement décentralisé, en cas de multiples *ContextDomains*, des requêtes et des diffusions d'information visant un domaine particulier peuvent être réalisées. De telles interactions sont prises en charge par l'agent `CtxUser` qui, grâce à son interface `ApplicationClient Adaptor`, peut effectuer les types suivants de requêtes inter-domaines : à *domaine exact* et à *portée de domaine* (cette dernière pouvant être une simple requête ou une diffusion).

Les requêtes de type *domaine exact* sont envoyées par un `CtxUser` à un *ContextDomain* spécifique. Dans notre scénario, par exemple, considérons Dan, un ami d'Alice, qui est dans un autre laboratoire du même bâtiment. Dan, souhaitant savoir si Alice est disponible, envoie une requête au *ContextDomain* `ami-lab`. L'agent `CtxQueryHandler` de la CMU en charge de ce domaine répond. Comme précisé dans la section 4, les agents `CtxQueryHandler` et `OrgMgr` collaborent pour router les requêtes basées sur un domaine entre CMU. Pour plus de détails, les interactions du protocole de routage et une analyse du nombre du messages échangés peuvent être consultées dans [13].

Les requêtes de type *portée de domaine* peuvent être utilisées dans des déploiements décentralisés où figurent des hiérarchies de *ContextDomains*. Ce type de requêtes ne vise pas un unique domaine, mais définit plutôt les bornes supérieures et inférieures sur le type de l'objet *ContextEntity* définissant les valeurs des *ContextDomains*. Par exemple, dans notre exemple, si Dan ne sait pas où Alice est, mais suspecte qu'elle est dans le bâtiment, il peut envoyer une requête de domaine à tous les *ContextDomains* de type `LabRoom`, où `LabRoom` est un sous-type de l'entité `UniversitySpace` qui est l'objet de la dimension `locatedIn(Person, UniversitySpace)`. Ici, les limites de type inférieure et supérieure coïncident : `LabRoom`.

Le dernier type de message est la *diffusion à portée de domaine* où le `CtxUser` joue le rôle de fournisseur de contexte en envoyant des données le long de la hiérarchie de domaines d'une dimension. Les spécifications de portée de do-

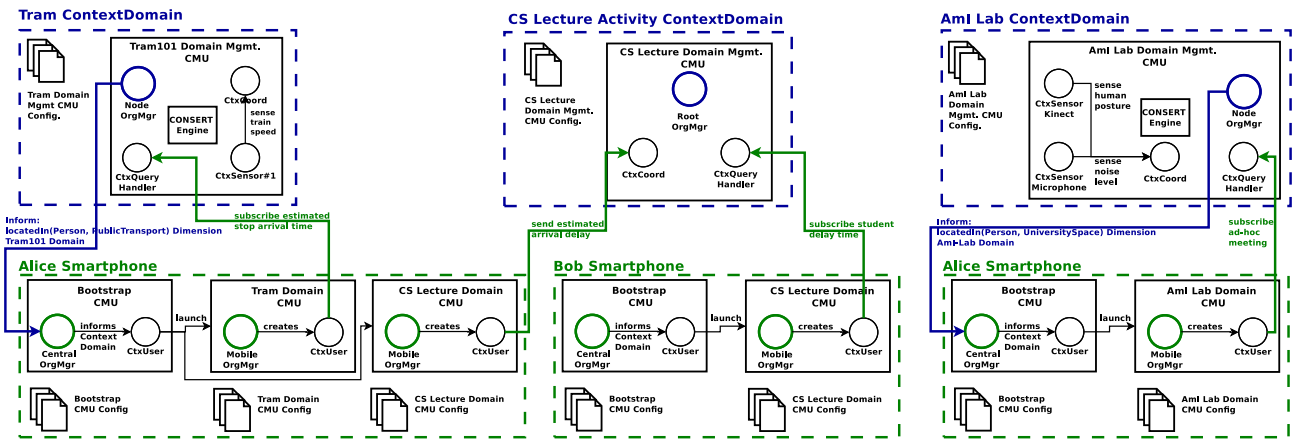


FIGURE 3 – Les *ContextDomains*, le déploiement des CMU et les connexions entre agents pour le scénario de référence.

maine sont les mêmes que pour les requêtes, mais l'interaction de routage survient entre `OrgMgr` et `CtxCoord` (comme les agents `CtxCoord` sont les récepteurs de contexte dans une CMU).

Remarquons que les types de messages présentés ci-dessus permettent à une application sensible au contexte d'échanger des données entre *ContextDomains* du moment qu'ils font partie de la même *ContextDimension*. Un agent `CtxUser` peut demander des informations de contexte d'un domaine distant et les utiliser dans le sien, ou même les diffuser le long de la hiérarchie de domaines, et ainsi agir comme un relai. De cette façon, la cohérence et les options d'adaptation du provisionnement de contexte décrites plus tôt dans cette section peuvent être appliquées aux informations de contexte provenant d'autre domaines de la même dimension.

6 Utilisation du CMM CONSERT

CONSERT a été évalué en termes de performances en temps d'exécution du moteur dans un rapport détaillé [14] et en termes de capacité de contrôle et d'adaptation du provisionnement de contexte, grâce à une simulation du laboratoire AmI dans les situations de notre scénario, en utilisant la plate-forme iCasa [12].

Ici, nous nous intéressons aux facilités de développement d'applications offertes par CONSERT, étant données ses options de configuration de déploiement flexible et de provisionnement. Nous revisitons le scénario de référence et expliquons comment l'application dans son intégralité peut être ainsi implantée.

Comme souligné dans la section 2, la gestion des activités d'Alice manipule des informations de contexte provenant de multiples domaines. La figure 3 représente l'application en termes de *ContextDimensions* et de *ContextDomains*. Elle montre la composition des nœuds fixes et mobiles des CMU en charge des modèles de contexte affectés à chaque *ContextDomain*. Sur le smartphone d'Alice, la CMU d'amorçage (*bootstrap*) gère les informations privées d'Alice, mais agit également comme réceptrice pour des notifications d'entrée/sortie de *ContextDomain*. Ainsi, en utilisant l'agent `CtxUser` de cette CMU, le niveau applicatif est informé lorsqu'Alice entre dans le tram et dans le laboratoire, ou quand il est temps de prendre en compte ses activités pédagogiques.

Lorsqu'Alice entre dans le tram, une notification est envoyée à l'application pour l'informer qu'il peut activer la CMU responsable de la dimension `locatedIn(Person, PublicTransport)` *ContextDimension* et demander la connexion de l'agent `CtxUser` de cette CMU à l'agent `CtxQueryHandler` s'exécutant sur la CMU du *ContextDomain* `Tram101`. Au même moment, comme Alice doit assister à son cours d'informatique, l'application est notifiée d'options similaires pour la dimension `engagedIn(Person, CourseActivity)` et la CMU du domaine `cs-lecture`. Le `CtxUser` de la CMU du tram du smartphone d'Alice découvre le temps estimé d'arrivée à la station. Le niveau applicatif transfert cette information à l'instance d'amorçage, où la vitesse de marche d'Alice est exploitée pour calculer son retard estimé. Ensuite, le niveau applicatif peut utiliser cette information et le `CtxUser` du cours d'informatique pour informer de ce retard le `CtxCoord` sur le nœud gérant le domaine du cours d'informatique à l'université. Entre temps, l'application s'exécutant sur le smartphone de Bob aura suivi des étapes identiques pour souscrire aux notifications de retard d'étudiants.

Lorsqu'Alice descend du tram, le CMU d'amorçage est notifié qu'elle a quitté le domaine `Tram101`. L'application peut ainsi automatique-

ment arrêter le CMU responsable des informations de contexte de ce domaine.

Dans le laboratoire AmI, le nœud gérant le *ContextDomain* contient des agents *CtxCoord* et *CtxQueryHandler*, ainsi que deux *CtxSensors* qui envoient des mises-à-jour à propos des postures de corps détectées (caméra Kinect) et des niveaux sonores (microphone) près de chaque bureau du laboratoire. Lorsqu'Alice entre dans le laboratoire, le CMU d'amorçage est informé qu'elle est entrée dans le domaine `ami-lab` de la dimension `locatedIn(Person, UniversitySpace)`. Le `CtxUser` de la CMU lancée par l'application sur son smartphone pour le domaine `ami-lab` se connectera au `CtxQueryHandler` sur le nœud de gestion et souscrira aux notifications de de mise en place de réunion ad-hoc, pour savoir quand Alice est occupée.

Les interactions précédentes montrent les bénéfices de développer des applications sensibles au contexte avec le CMM CONSERT. Créer des instances de *bundles* pour chaque domaine de contexte d'une application rend le développement de modèles de contexte spécifiques plus simple, aide à encapsuler la logique de provisionnement associée, et fournit un support au contrôle du cycle de vie d'une CMU en cours d'exécution.

7 Conclusions et perspectives

Nous avons présenté les défis soulevés par l'ingénierie d'applications sensibles au contexte au travers d'un scénario d'usage. Ces défis soulignent le besoin de développer des CMM capables de prendre charge la dynamique, des domaines multiples, et des épisodes d'interaction contextuelle indépendants et interdépendants. Concevoir un CMM qui répond aux besoins de déploiement et de contrôle de l'infrastructure de gestion et de calcul dans de tels scénarios implique des exigences de modularités et de configurabilité des unités de provisionnement de contexte. L'intergiciel CONSERT que nous avons présenté répond à ces défis de développement par la mise en place d'une architecture multiagent et d'options de déploiement flexible de ses unités de gestion du contexte. L'implantation par composants logiciels de CONSERT ajoute en flexibilité en permettant un contrôle du cycle d'exécution de ces unités de gestion, en cours de fonctionnement.

Nous envisageons de tester CONSERT par une implantation réelle de l'application *smart university* et également d'améliorer le comportement de provisionnement des agents en ajoutant la capacité d'établir des accords de niveau de contexte (analogues aux accords de niveau de service, ou SLA), pour augmenter le niveau de contrôle et d'adaptation du provisionnement.

Remerciements

Ce travail a été financé par le *Sectoral Operational Programme Human Resources Development* 2007-2013 du Ministère roumain des fonds européens par l'accord de financement POSDRU/159/1.5/S/134398.

Références

[1] C. Perera, A. Zaslavsky, P. Christen, D. Georgakopoulos. *Sensing as a service model for smart cities supported by internet of things.* Transactions on Emerging Telecommunications Technologies, 25(1), pp. 81-93, 2014.

[2] A. Sorici, G. Picard, O. Boissier, A. Zimmermann, A. Florea. *CONSERT : Applying Semantic Web Technologies to Context Modeling in Ambient Intelligence.* Computers and Electrical Engineering - An International Journal, (in press), 2015.

[3] H. Chen, T. Finin, A. Joshi. *The SOUPA ontology for pervasive computing.* Ontologies for agents : Theory and experiences. Birkhäuser Basel, pp. 233-258, 2005.

[4] C. Perera, A. Zaslavsky, P. Christen, D. Georgakopoulos. *Context aware computing for the internet of things : A survey.* IEEE Communications Surveys & Tutorials, 2013.

[5] A. Zimmermann, A. Lorenz, R. Oppermann. *An operational definition of context.* Modeling and using context, Springer, pp. 558-571, 2007.

[6] K. Henricksen, J. Indulska, T. McFadden, S. Balasubramaniam. *Middleware for distributed context-aware systems*. In On the Move to Meaningful Internet Systems 2005 : CoopIS, DOA, and ODBASE, Springer, pp. 846-863, 2005.

[7] F. Li, S. Sehic, S. Dustdar. *Copal : An adaptive approach to context provisioning.* Wireless and Mobile Computing, Networking and Communications (WiMob), pp. 286-293, IEEE, 2010.

[8] S. Pietschmann, A. Mitschick, R. Winkler, K. Meißner. *Croco : Ontology-based, cross-application context management.* Third International Workshop on Semantic Media Adaptation and Personalization, SMAP'08, pp. 88-93, IEEE, 2008.

[9] D. Ejigu, M. Scuturici, L. Brunie. *Hybrid Approach to Collaborative Context-Aware Service Platform for Pervasive Computing.* Journal of computers 3(1), 2008.

[10] B. Guo, L. Sun, D. Zhang. *The architecture design of a cross-domain context management system.* Pervasive Computing and Communications Workshops (PERCOM Workshops), pp. 499-504, IEEE, 2010.

[11] M. Khedr, A. Karmouch. *ACAI : Agent-Based Context-Aware Infrastructure for Spontaneous Applications.* Journal of Network and Computer Applications 28(1), pp. 19-44, 2005.

[12] A. Sorici, G. Picard, O.Boissier, A. Florea. *Policy-based Adaptation of Context Provisioning in AmI.* 6th International Symposium on Ambient Intelligence, 2015. Article submitted.

[13] A. Sorici. *CONSERT Middleware Domain-Based Query Routing.* Technical Report, UPB, 2015. Available at http://bit.ly/1dnP4av.

[14] A. Sorici. *CONSERT Engine Performance Analysis.* Technical Report, UPB, 2015. Available at http://bit.ly/1PRcB1x.

Patterns multi-niveaux pour les SMA

A. Maudet[a] adrien.maudet@ign.fr
G. Touya[a] guillaume.touya@ign.fr
C. Duchêne[a] cecile.duchene@ign.fr
S. Picault[b] sebastien.picault@univ-lille1.fr

[a]Université Paris Est/IGN, COGIT, Saint-Mandé, France

[b]Équipe SMAC, CRIStAL, Université Lille 1, Villeneuve d'Ascq, France

Résumé

Depuis quelques années, les travaux sur les SMA multi-niveaux ont pris une importance croissante. Devant la diversité des modèles proposés, nous pensons qu'il est utile d'identifier des situations récurrentes et de les caractériser d'une manière suffisamment abstraite pour pouvoir comparer de manière formelle les modèles existants et faciliter la conception de nouveaux modèles. Dans ce but, nous proposons une première liste de patterns SMA multi-niveaux. Ces patterns sont issus d'un travail d'unification de modèles SMA multi-niveaux dédiés à la résolution d'un problème spatialisé (la généralisation cartographique). La structure et la dynamique de chaque pattern sont décrites formellement et accompagnées d'exemples issus d'une part du contexte de la généralisation cartographique, d'autre part d'autres contextes applicatifs multi-agents, en simulation notamment. Nous discutons également la possibilité de réutiliser et composer ces patterns.

Mots-clés : *SMA multi-niveaux, patterns, agents situés, conception de SMA, généralisation cartographique, simulation*

Abstract

Multi-level MAS have become a growing research topic in the last years. Due to the diversity of proposed models, we claim that recurrent situations have to be identified and characterized in an abstract way, in order to allow formal comparisons between existing models and to facilitate the design of new models. Therefore, we propose a first list of multi-level MAS patterns, coming from an attempt to unify several multi-level MAS models dedicated to a spatialized problem solving (cartographic generalization). The structure and dynamic of each pattern are formally described, and illustrated through examples drawn from the context of cartographic generalization and other MAS applications (especially in simulation). We also discuss the possibility to re-use and compose those patterns.

Keywords: *Multi-level MAS, patterns, situated agents, MAS design, cartographic generalisation, simulation*

1 Introduction

La recherche dans le cadre des systèmes multi-agents multi-niveaux connaît de récentes avancées. Ainsi, nombre de cas d'application impliquent la prise en compte de plusieurs niveaux, par exemple de simulation d'évolution de zones urbaines [7] ou encore des mouvements de foules [30]. Les aspects multi-niveaux communs à certains modèles ont été comparés dans [14]. Extraire de ces cas d'application concrets des situations génériques permet de mettre en commun des connaissances éprouvées, et d'enrichir la connaissance dans le domaine des systèmes multi-niveaux.

Plusieurs travaux ont mené à la formalisation des niveaux entre eux. Ces formalisations portent principalement sur l'aspect structurel des modèles (les liens entre les différents niveaux), ou sur des aspects spécifiques au passage d'un niveau à un autre. Dans [7], les phénomènes émergents sont réifiés en proposant l'introduction d'un élément d'interposition qui permet de contrôler les actions de l'agent créé à partir d'un phénomène émergent sur l'environnement de simulation. Dans [5], une formalisation est proposée pour les problèmes impliquant l'utilisation parallèle de deux modélisations à deux niveaux différents en proposant de définir des fonctions d'émergence et d'immergence permettant de maintenir la cohérence entre les deux niveaux. Dans [4], une formalisation de l'identification de groupes est proposée avec pour objectif de permettre une visualisation de ces derniers lors de différentes simulations. Cette formalisation est indépendante de la méthode de *clustering* choisie pour identifier un groupe. Dans [30], un méta-modèle est proposé pour exprimer la notion de capture et de libération d'agent par un agent de niveau supérieur. Lorsqu'il est capturé, un agent peut voir sa

position dans l'espace gérée par l'agent qui l'a capturé. Il récupère la gestion de son positionnement dans l'espace au moment de sa libération.

Toutes ces approches ont pour particularité de mettre l'accent sur le passage d'un niveau à un autre, que ce soit dans le cadre de la construction d'objet d'un niveau supérieur à partir d'agents d'un niveau inférieur, ou pour la transmission d'information du résultat d'interactions du niveau supérieur aux agents du niveau inférieur. Lors de l'étude de la modélisation de SMA pour un cas de résolution d'un problème spatialisé contraint, la généralisation cartographique, nous avons identifié que d'autres mécanismes plus précis se répétaient. L'objectif de cet article est de proposer une modélisation de ces situations sous forme de *patterns* génériques, en s'appuyant sur les cas concrets de la généralisation cartographique, et d'étudier dans quelle mesure ces *patterns* permettent de traiter des situations similaires rencontrées dans d'autres cadres applicatifs.

Le plan de cet article est le suivant : la partie 2 explicite nos motivations pour l'élaboration de *patterns*, et le cadre de la généralisation cartographique dans lequel ils ont été identifiés ; dans la partie 3, nous présentons le cadre théorique que nous utilisons dans la partie 4 pour la définition de ces patterns ; enfin, dans la partie 5 nous discutons nos propositions avant de conclure dans la partie 6.

2 Démarche et contexte applicatif

2.1 Motivation pour l'identification de patterns multi-niveaux

Lors de la résolution de problèmes multi-niveaux dans le domaine de la généralisation (présentée dans la section suivante), nous avons rencontré des motifs structurels ou comportementaux récurrents. Nous supposons que ces motifs sont susceptibles d'être utilisés pour l'analyse et la modélisation d'autres situations dans d'autres domaines, voire pour l'implémentation de solutions. Pour cela, nous essayons de décrire ces situations récurrentes d'une façon abstraite (en les caractérisant *a minima* et indépendamment du domaine d'où elles sont issues), d'identifier des exemples où elles apparaissent hors du domaine d'origine, et de montrer les liens qu'elles entretiennent les unes avec les autres.

À travers la notion de *pattern*, nous entendons

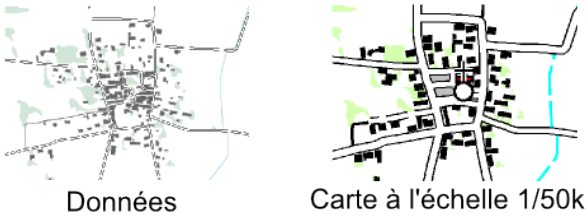


FIGURE 1 – Données initiales et carte généralisée.

construire une représentation abstraite minimale décrivant une situation rencontrée de façon récurrente, en nous affranchissant le plus possible des choix relatifs au cadre de modélisation multi-niveau, de même que les *Design Patterns* du génie logiciel [12] cherchent à s'affranchir du langage objet utilisé. Des *patterns* pour les SMA ont par ailleurs déjà été proposés, notamment des *patterns* d'analyse agent [6] et pour la conception des environnements [22]. Dans cet article, nous cherchons principalement à construire des abstractions destinées à simplifier l'analyse de situations récurrentes pouvant être rencontrées dans divers systèmes multi-niveaux.

2.2 La généralisation cartographique, un problème spatialisé

Les cartes représentent l'information géographique d'une zone donnée de manière d'autant plus simplifiée que l'échelle de la carte est petite (figure 1). Le procédé de simplification, appelé généralisation cartographique, est soumis au respect de contraintes de lisibilité, d'adéquation de la représentation avec le niveau d'abstraction souhaité et de cohérence avec la réalité. La volonté d'automatiser le processus de création de cartes à partir de bases de données géographiques, où la forme des objets est stockée sous forme vectorielle (points, polylignes, polygones), a conduit à la création d'algorithmes permettant d'effectuer cette simplification objet par objet. Néanmoins, les choix des algorithmes, tout comme leur paramétrage, sont autant influencés par l'objet sur lequel ils s'appliquent que par les autres objets en relation (*e.g.* bâtiment à proximité d'un autre, route parallèle à un alignement de bâtiments). Ce constat a motivé l'utilisation de modèles multi-agents pour la généralisation automatisée de cartes.

Le principe de ces modèles multi-agents repose sur la modélisation des objets (*e.g.* bâtiment, tronçon de route, îlot urbain ou pâté de maison)

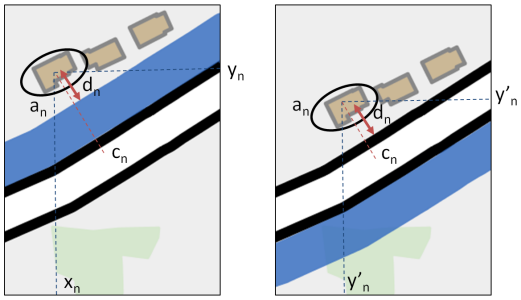


FIGURE 2 – Extrait de carte avec agents bâtiments avec coordonnées cartésiennes et coordonnée en abscisse curviligne du projeté du centroïde sur la route à proximité et distance au projeté. Si le symbole de la route change (par exemple, suite au changement de coté de l'itinéraire de randonnée bleue porté par cette route), alors la position du bâtiment reste la même dans l'environnement de la route, mais change dans celui de la carte.

sous forme d'agents qui cherchent à se généraliser de façon à satisfaire leurs contraintes. Ces contraintes reflètent des critères géométriques de lisibilité et peuvent porter sur un seul objet (e.g. contrainte de taille minimale d'un bâtiment) ou sur plusieurs (e.g. contrainte de non-superposition entre un bâtiment et une route). La construction de l'espace géographique conduit à considérer son organisation selon différents niveaux (e.g. un bâtiment appartient à un îlot urbain, mais peut aussi appartenir à un ou plusieurs alignements de bâtiments ; des points d'intérêts particuliers, comme un refuge ou une table d'orientation, peuvent être situés sur un itinéraire). Plusieurs modèles multi-agents ont été proposés [2, 8, 11, 18, 27, 32], chacun ayant une approche différente des interactions entre niveaux. Nous étudions l'unification de ces modèles [24] en nous appuyant sur le paradigme multi-niveau PADAWAN [21].

2.3 Motifs récurrents en généralisation

Le travail d'unification de modèles en cours nous a conduit à identifier dans un premier temps des motifs récurrents propres à la généralisation automatisée. Nous sommes confrontés à un problème spatialisé impliquant des définitions variées de ce qui peut constituer un groupe d'agents et une multiplicité des aspects multi-niveaux, tant dans la construction des différents niveaux que dans la nature de ces derniers.

En terme de construction de groupes, nous observons plusieurs cas :

- la création d'agents représentant des entités cohérentes d'un point de vue cartographique, et sur lesquelles l'application d'algorithmes particuliers peut s'avérer efficace (e.g. en ville, élimination de bâtiments décidée au niveau de l'îlot urbain pour garder les plus représentatifs la relation meso/composant décrite par [2, 26]) ;
- la division d'un agent, menant à la création d'agents du même type, pour permettre la généralisation de ses sous-parties de manière individuelle, tout en lui laissant un contrôle sur la situation ;
- la création d'agents-points composant la géométrie d'un objet, agents-points pouvant interagir avec d'autres objets de la carte [11] ;
- l'émergence de cas problématiques dont la non-résolution prolongée, voire l'aggravation, peut amener à préférer une résolution à un niveau plus global [9] ;
- des objets dont la position relative à un autre est importante, ce qui nous amène à considérer l'ensemble des ces objets comme appartenant à un même espace ayant un référentiel propre.

Nous observons aussi des types d'environnements différents :

- l'environnement de base, cartésien 2D, où la position des agents est exprimée selon leur coordonnées (x, y) ;
- des environnements relationnels, où les agents identifient leur voisinage à partir d'un graphe de relations, établi *a priori* à partir de l'environnement cartésien, en mesurant la proximité des agents entre eux [8] ;
- des environnements en abscisse curviligne et distance au projeté permettant de positionner un objet particulier vis-à-vis d'un objet linéaire (figure 2) ;
- des environnements décrits de façon à tenir compte de la forme même de cet environnement et de points saillants (e.g. intersections ou virages d'une route) pouvant servir de points de repère [17].

De la diversité de ces situations, nous pensons pouvoir extraire des situations suffisamment génériques. Avant de décrire les *patterns* que nous proposons, nous exposons le cadre formel dans lequel nous les définissons.

3 Cadre formel

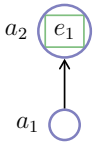
L'approche par *patterns* n'est intéressante que si elle permet au minimum une description uni-

voque et opérationnelle des situations rencontrées, ce qui suppose de disposer d'un cadre formel suffisamment abstrait pour être appliqué à divers contextes. C'est en général UML qui assure cette fonction dans le cas des *patterns* de conception en programmation par objets. Ici, nous proposons d'utiliser une formalisation particulière des relations entre agents (et environnements), que nous illustrons graphiquement. Nous nous appuyons pour cela sur les concepts introduits dans le modèle PADAWAN [21], dont nous reprenons certaines notations. Notre objectif étant toutefois de nous abstraire des choix de modélisation et d'implémentation de l'architecture multi-niveau, nous n'avons conservé de PADAWAN que la part la plus abstraite, susceptible de donner lieu à des réalisations différentes selon les contextes de déploiement visés.

En particulier, la décomposition d'un système complexe multi-niveau dans PADAWAN s'appuient sur deux sortes d'éléments : les agents ($\mathcal{A}$) bien sûr mais également les environnements ($\mathcal{E}$). Ces derniers peuvent désigner indifféremment des « groupes sociaux » ou des portions de l'espace physique, et peuvent eux-mêmes être décrits par des *patterns* [22]. Nous précisons, par souci de s'abstraire de la diversité des implémentations, que nous désignons par environnement un espace doté d'une métrique, dans lequel les agents peuvent se percevoir et interagir. Deux relations fondamentales sont définies entre les agents et les environnements. D'une part, **la situation** d'un agent a dans un environnement e (notée $\mathbf{a} \lhd \mathbf{e}$) exprime le fait que l'agent a peut percevoir, être perçu, agir ou subir des actions dans e. Un agent peut être situé dans un nombre quelconque d'environnements. D'autre part, **l'encapsulation** d'un environnement e par un agent a (notée $\mathbf{a} \sim \mathbf{e}$) signifie que l'agent a « contient » e : il est notamment capable de percevoir tous les agents qui y sont situés et en est **l'hôte**.

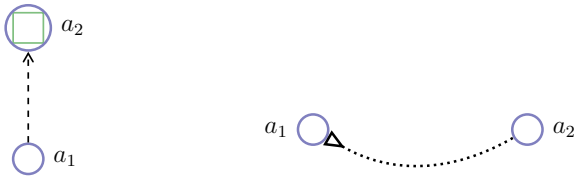
À partir de ces deux relations nous définissons la relation **d'hébergement** qui permet de s'affranchir du choix fait dans PADAWAN de multiplier les environnements. Un agent a_1 est **hébergé** par un agent a_2, (ou a_2 est **l'hôte** de a_1), noté $a_1 \sqsubset a_2$ si et seulement si : $\exists e \in \mathcal{E} \mid a_2 \sim e \wedge a_1 \lhd e$. Les relations d'hébergement (avec la situation et l'encapsulation sous-jacentes) sont représentées par un graphe (fig. 3).

Une autre hypothèse de travail liée au multi-niveau en général est que les comportements d'un agent dépendent de sa situation, autrement dit un agent n'agira pas de la même façon selon



$a_1 \sqsubset a_2$ (a_1 est hébergé par a_2)
i.e. $a_1 \lhd e_1$ (a_1 est situé dans e_1)
et $a_2 \sim e_1$ (a_2 encapsule e_1)

FIGURE 3 – Représentation graphique de la relation d'hébergement



(a) Hébergement optionnel de a_1 par a_2

(b) Dépendance de a_1 vis-à-vis de a_2

FIGURE 4 – Notation des liens d'hébergement optionnel et de dépendance entre agents.

qu'il est hébergé par a_1 ou par a_2. Dans PADAWAN, cela se traduit par le fait que chaque environnement contient une *matrice d'interaction* au sens de l'approche IODA [20] qui définit qui peut faire quoi avec qui. Ici, il suffit de supposer plus généralement que chaque hôte doit spécifier les comportements des agents qu'il héberge. Par ailleurs, ces comportements peuvent s'appuyer sur un certain nombre de primitives, certaines étant directement destinées à assurer des transformations cohérentes des relations d'hébergement. Par exemple dans la suite nous utiliserons **put(a,b)** pour indiquer que l'agent a est désormais hébergé par b ainsi que **remove(a,b)** pour supprimer la relation d'hébergement de a par b ; **del(a)** pour désigner la destruction d'un agent ; et enfin **merge(a,b)** pour noter la fusion de deux agents. Ces primitives, qui ont une définition univoque dans PADAWAN, ont évidemment vocation a être implémentées de façon adéquate dans chaque modèle multi-niveau où l'on souhaite utiliser les patterns que nous avons identifiés.

Outre ces relations, nous notons dans la suite comme indiqué sur la figure 4 deux cas utiles pour la description des patterns : d'une part, un lien d'hébergement *optionnel*, qui indique que l'agent concerné peut ou non être hébergé par l'autre sans affecter la définition du pattern ; d'autre part, un lien de *dépendance*, noté aussi $depends(a_1, a_2)$, qui exprime le fait que l'existence de l'agent a_1 est conditionnée à celle de l'agent a_2 (autrement dit, la suppression de a_2 entraîne par défaut celle de a_1).

4 *Patterns* proposés

L'identification de *patterns* suppose de délimiter des situations univoques, caractérisées par une certaine structuration et d'éventuels effets sur le comportement. Pour chacun des *patterns* proposés, nous décrivons la situation de notre cas d'application qui en est à l'origine, une formalisation et d'autres situations où il peut s'appliquer.

4.1 *Pattern* Agrégation

Origine : Lors de la généralisation automatisée, nous avons besoin, en plus des objets géographiques de base, d'objets qui en sont déduits. Par exemple les villes, qui sont construites par enrichissement des données initiales, par agrégation spatiale de bâtiments proches les uns des autres. Ces agents villes sont appelés à interagir avec les agents qu'ils hébergent : bâtiments, routes, etc ...

Une autre situation est l'identification de groupes constitués d'agents ne réussissant pas à satisfaire leurs contraintes. Dans [9], deux méthodes sont proposées pour identifier les conflits résiduels et créer des entités intermédiaires ayant en charge la résolution de ces conflits.

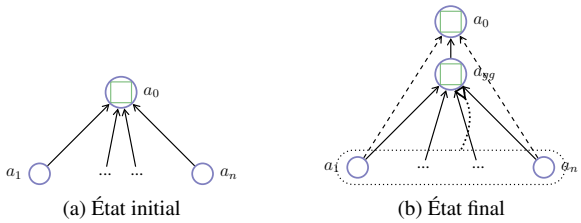
De tels cas nous amènent à considérer le fait qu'un agent construit à partir d'autres agents, peut inclure plusieurs agents différents de ceux qui ont permis sa construction. Nous proposons donc de formaliser cet aspect en introduisant deux fonctions agissant sur un ensemble d'agents :

- une fonction **build :** $\mathcal{A}_x \mapsto a_{gg}$ qui construit un agent agrégé a_{gg} à partir d'un ensemble d'agents $\mathcal{A}_x$;
- une fonction **populate :** $\mathcal{A}_x \mapsto \mathcal{A}_{agg}$ qui sélectionne les agents qui seront hébergés par l'agent ainsi créé.

Formalisation : `compose` **(**$\mathcal{A}_x$**, build, populate) :**

- Prérequis
  - Tous les agents de $\mathcal{A}_x$ sont hébergés pas un même agent.
- Caractérisation :
  $a_{gg} \leftarrow build(populate(\mathcal{A}_x))$

Cas d'application : cette situation évoque des cas fréquemment rencontrés lors de la simulation à savoir la réification de phénomènes émergents et la création de *clusters*. Ainsi la fonction **build** peut être construite à partir de la fonction d'émergence proposée par [5] ou de la fonction de *clusterisation* proposée par [4].



(a) État initial (b) État final

FIGURE 5 – Notation graphique du pattern *Agrégation*.

Plus généralement, ce pattern permet de réifier par un agent toute structure émergente, tout groupe dans lesquels des agents estiment pouvoir appartenir. Il est à noter que c'est autour de cette problématique de création de niveaux d'abstraction supérieurs que se sont développés les premiers travaux sur la simulation multi-niveau, à travers le projet RIVAGE [29] en particulier. Depuis, tout un pan de recherche porte sur la façon d'écrire des fonctions **build** et **populate** de façon générique.

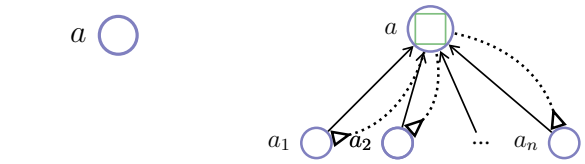
4.2 *Pattern* Décomposition

Origine : Dans le modèle GAEL [11], les objets géographiques considérés comme déformables (*e.g.* courbes de niveaux) ont leur géométrie décomposée en agents points. Dans notre adaptation au sein d'un modèle commun, ces points sont formalisés comme des agents, et sont situés dans un environnement appelé environnement déformable, tout comme les agents points. Ces derniers interagissent de manière à permettre des déformations locales des objets, tout en gardant au mieux leur forme.

De même, dans le cadre de la généralisation du réseau routier [2], un tronçon de route peut être divisé en sous-tronçons qui seront plus faciles à généraliser individuellement. Néanmoins, afin de maintenir la cohérence de l'ensemble, il est nécessaire que les agents issus de cette subdivision reste liés ensemble. L'agent initial encapsule alors les agents créés, et coordonne leurs actions.

De ces situations, nous constatons que, d'une manière symétrique à l'agrégation, nous pouvons proposer une fonction **build :** $a_x \mapsto \mathcal{A}_x$, pour construire les agents du niveau inférieur. Notons que dans ce cas il n'est pas nécessaire de proposer une fonction **populate** séparée.

Formalisation : **`decompose`** (a, **build**) **:**
- — Prérequis : Aucun
- — Caractérisation :
  $\mathcal{A}_x \leftarrow build(a)$
  pour tout $a_i \in \mathcal{A}_x : put(a_i, a)$



(a) État initial (b) État final

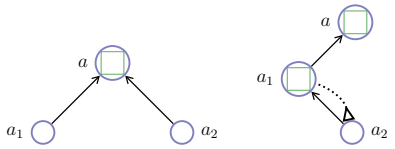
FIGURE 6 – Notation graphique du pattern *Décomposition*.

Cas d'application : les cas de décomposition s'effectuent lorsqu'une entité de niveau supérieur doit laisser la place à une entité de niveau moindre. Cela se produit par exemple dans [30], lorsqu'un agent est libéré, ou dans [5], lorsqu'une fonction d'immergence est appelée.

4.3 *Pattern* Hiérarchisation

Origine : Le cas de la hiérarchisation intervient lorsqu'on veut établir une relation hiérarchique entre deux agents. En généralisation, nous pouvons être amenés à définir ce genre de situation lorsqu'un objet semble clairement situé sur un autre objet (e.g. un point d'intérêt comme une table d'orientation indiquée sur le parcours d'un itinéraire décrit sur la carte). Il est important que, suite à la modification de l'objet support, la relation soit préservée, afin que l'information reste cohérente.

Formalisation : **`hierarchise`** (a_1, a_2) **:**
- — Prérequis :
  - — a_1 et a_2 sont hébergés pas un même agent.
- — Caractérisation :
  put(a_2,a_1)



(a) État initial (b) État final

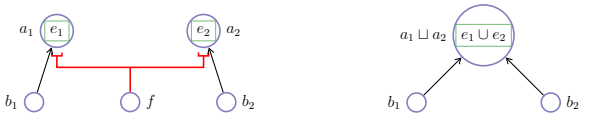
FIGURE 7 – Notation graphique du pattern *Hiérarchisation*.

Cas d'application : la hiérarchisation s'applique souvent dans les cas les plus simples où un agent entre dans un autre agent qui se déplace dans le même environnement, comme par exemple un individu qui entre dans un véhicule.

4.4 *Pattern* Frontière

Origine : Dans le cadre de la généralisation, deux îlots urbains adjacents séparés par une route doivent être fusionnés lorsque cette route séparatrice disparaît pour une raison tierce, comme la volonté de diminuer la densité du réseau routier dans une ville. Le résultat de cette fusion, un nouvel îlot, doit héberger l'ensemble des bâtiments qui étaient hébergés dans les îlots initiaux. Dans [1], cette situation a pu être intégrée par l'introduction de la notion de frontière. Nous parlons de frontière lorsque deux environnements sont séparés par un objet appartenant aux deux environnements. Le rôle de la frontière est d'interdire le passage d'agents d'un environnement à l'autre.

`frontiere` **(f, { a_1, a_2 }) :**
- — Prérequis
  - — $f \sqsubset a_1$ et $f \sqsubset a_2$
  - — a_1 et a_2 sont des agents « compatibles » (on peut les « fusionner »).
- — Caractérisation : **`del`** **(f)** $\rightarrow$ **`merge`** **(a_1, a_2)**



(a) f est un agent frontière (b) Après suppression de f

FIGURE 8 – Notation et résultat de la suppression d'un agent frontière.

Cas d'application : l'agent frontière peut être utilisé dans les situations où des environnements sont séparés par un élément dont le dynamisme (qui peut, comme dans notre exemple, se limiter à sa disparition) justifie sa modélisation sous forme d'agent.

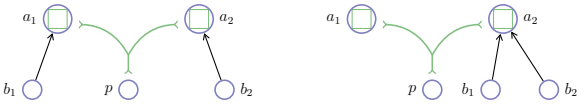
Nous pouvons nous interroger ici sur le fait que, dans les exemples proposés, la frontière est hermétique. Or, certaines situations impliquent une certaine porosité, comme la modélisation de la paroi d'une cellule, ou la modélisation d'un sas. Pour tenir compte de ce genre de situation, nous proposons un autre *pattern* spécifique : porte.

4.5 *Pattern* Porte

Origine : Nous ne sommes pas confronté à cette situation dans la cadre de la généralisation,

mais la caractérisation du pattern frontière nous amène à identifier un *pattern* porte.

`porte` (p, { a_1, a_2 }) :
- Prérequis :
- $p \sqsubset a_1$ et $p \sqsubset a_2$
- Caractérisation : Le pattern porte permet de définir une primitive **cross(a,p)** qui permet à un agent a situé dans a_1 (resp. a_2) de demander à l'agent porte p de le déplacer vers a_2 (resp. a_1). Cette primitive peut être assujettie à des conditions propres à la porte et aux agents hôtes concernés. La suppression d'une porte n'a pas d'effets sur les hôtes.



(a) p est un agent porte (b) b_1 a traversé la porte p

FIGURE 9 – Notation et traversée d'un agent porte.

Cas d'application : un exemple d'application du modèle PADAWAN [21], la pompe sodium-potassium, peut être modélisé selon ce *pattern* porte. La simulation d'un système impliquant un sas est un autre exemple.

4.6 *Pattern* Embarquement

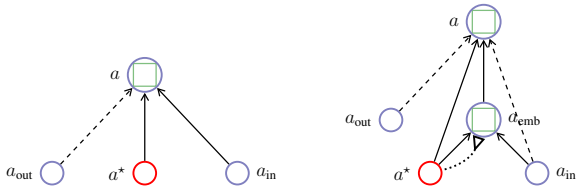
Origine : En généralisation, lorsque nous modifions la géométrie ou le symbole d'un objet, il est souhaité que les relations que cet objet possède par ailleurs avec d'autres objets soient préservées. Par exemple dans [23], une impasse peut glisser le long de la route la supportant afin de laisser de la place aux bâtiments. Si cela se produit les bâtiments dans son voisinage doivent être déplacés afin de garder leur position relative par rapport à l'impasse. De même (figure 2), les objets à proximité d'une route doivent garder une position relative identique lorsque la route change de symbole, par exemple lorsque le symbole d'un itinéraire change de position.

Pour résoudre cette situation dans le cadre de PADAWAN, nous nous appuyons sur la notion d'« environnement embarqué ». Lorsqu'un agent possède, dans son voisinage, un ensemble d'objets dont la position relative par rapport à lui est importante, un environnement embarqué est créé, ainsi qu'un agent encapsulant cet environnement. Cet environnement embarque à la fois l'agent à l'origine de sa création, et les agents de son voisinage.

L'environnement embarqué définit son propre référentiel, qui s'appuie sur l'agent à l'origine de sa création, et dans lequel les autres agents ont des coordonnées relatives. Ainsi, si l'agent à l'origine de l'embarquement se voit modifié, la relation entre le référentiel de l'environnement embarqué et le référentiel de l'environnement dans lequel il se situait initialement change aussi. Par conséquent, comme les agents doivent préserver leurs positions relatives dans l'environnement embarqué, ils devront modifier leur position dans l'environnement où ils se situaient auparavant.

Dans le cadre de nos exemples, un référentiel simple consiste à utiliser des coordonnées composées de l'abscisse curviligne du projeté du centroïde de l'agent embarqué, et de sa distance à l'objet linéaire (route) à l'origine de l'embarquement. Lors de modifications de l'objet linéaire, le référentiel local est modifié par rapport au référentiel cartésien de la carte, ce qui implique une modification de la position des objets de l'environnement embarqué dans le référentiel de la carte.

`embarque` ($a^\star$, $\mathcal{A}_x$) :
- Prérequis : $a^\star$ identifié comme jouant un rôle clef pour le référencement spatial dans son voisinage $a^\star$, $\mathcal{A}_x$.
- Caractérisation :
  - Création de l'environnement embarqué e_{emb} et de l'agent l'hébergeant : c'est un cas particulier d'agrégation, où l'agent à l'origine de l'embarquement a une relation de dépendance avec l'agent agrégat (encapsulant l'environnement embarqué) créé.
  - modification de $a^\star \rightarrow$ mise à jour de la position des agents $a \in \mathcal{A}_x$.
  - modification de la position dans l'environnement initial d'un agent $a \in \mathcal{A}_x \rightarrow$ mise à jour des coordonnées de a.



(a) État initial (b) État initial

FIGURE 10 – Notation graphique du *pattern Environnement embarqué*.

Cas d'application : Plusieurs situations de simulation peuvent être modélisées selon la no-

tion d'agent embarqué. Par exemple, une simulation de déplacement d'astres peut modéliser leur champs de gravitation. Dans une telle modélisation, la Terre définirait un environnement embarqué dans lequel se situerait la Lune. Le déplacement de la Terre autour du Soleil impliquerait une modification de la position de la Lune dans le référentiel héliocentrique.

4.7 *Patterns* Meso

Les interactions hiérarchiques, c'est-à-dire impliquant un hôte et ses hébergés, sont particulières, dans la mesure où elles peuvent impliquer des prises de décisions fortes d'un niveau supérieur pour les agents hébergés.

Origines : Dans le modèle AGENT, un *meso* désigne un agent qui est composé d'autres agents (ces dernier pouvant être soit *micro*, c'est-à-dire des agents existant initialement, soit *meso* eux aussi). Les *mesos* ont plusieurs rôles vis-à-vis des autres agents. Plusieurs de ces rôles sont décrits dans [26] : le **Législateur**, le **Contrôleur** et le **Coordinateur**. Le **Coordinateur** décrit une situation particulière qui fait l'objet d'un *pattern* Ordonnanceur décrit à part.

Le **Législateur** agit directement sur les connaissances des agents qu'il héberge, ainsi que leurs motivations. Dans le cadre de la généralisation, cela se traduit par le fait que le *meso* peut modifier directement les contraintes des ses composants, et ainsi influer leur comportement. Le **Contrôleur** assure que les actions effectuées par les agents vont dans le sens de ses intérêts (dans le cadre de la généralisation, dans le sens de la satisfaction de ses contraintes, mais aussi de celle de ses composants). Il peut décider d'annuler une action effectuée par un de ses agents hébergés. Dans [18], les agents hébergés sont amenés à interagir eux-mêmes pour se généraliser, mais peuvent parfois s'en remettre à un agent hôte pour décider d'une action qu'ils ne peuvent résoudre eux même. Nous proposons la dénomination d'**Arbitre** pour ce cas de figure. Dans [25], les agents bâtiments appartenant à un alignement de bâtiments peuvent décider de s'en remettre à l'agent alignement pour effectuer une action qui sera mieux effectuée par ce dernier. Nous proposons ici le terme de **Délégataire** pour décrire ce rôle de *meso*.

Cas d'application : Dans le cadre d'une simulation avec un groupe hiérarchisé, comme un orchestre où le chef choisit l'ordre d'activation des membres du groupe, nous pouvons modéliser des situations impliquant certains des rôles du meso.

4.8 *Pattern* Ordonnanceur

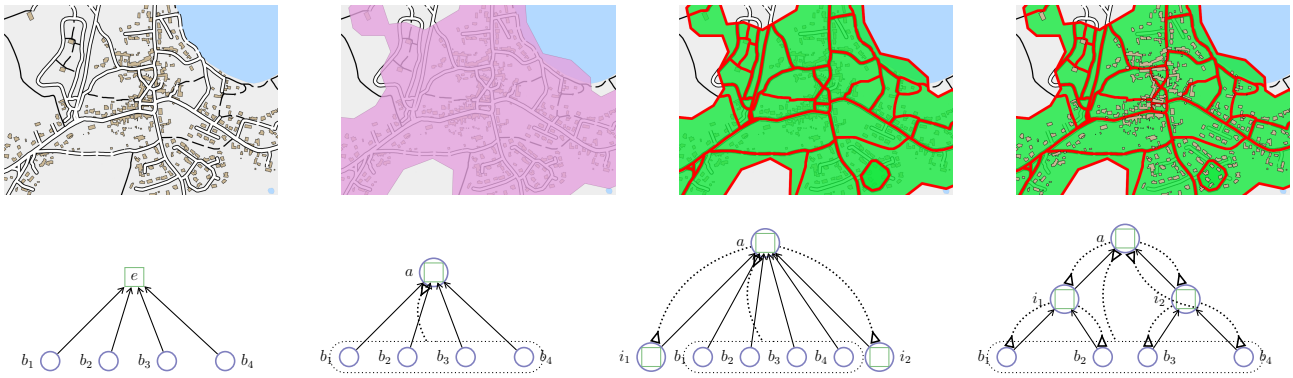
Origine : Dans le modèle AGENT, les agents meso ont la possibilité d'activer leurs composants, afin qu'ils se généralisent eux-même. Cette activation s'effectue lors du cycle de vie de l'agent meso. Ce rôle d'activation appelé **Coordinateur** dans [26], permet d'identifier le meilleurs moment pour la généralisation individuelle des bâtiments. Les connaissances qu'ont les îlots de l'intégralité de ces composants en font les agents les plus appropriés identifier l'ordre d'activation le plus opportun.

Cas d'application : Dans [13], une approche pour la simulation propose de donner aux holons un rôle équivalent à celui de Coordinateur.

5 Discussion

Les patterns proposés en section 4 sont des premières propositions pour l'identification de patterns. Ces *patterns* permettent l'analyse et la comparaison de situation récurrentes. Leur intérêt réside aussi dans la combinaison de ces *patterns*, et dans leur utilisation lors de la conception de modèle. Par exemple, plusieurs situations particulières propres à la généralisation cartographique peuvent être reproduites à partir d'une combinaison de ces *patterns*. Ainsi, la création d'îlots urbains [3], et leur relation avec les bâtiments qu'ils hébergent, peut s'exprimer à l'aide des trois *patterns* de création de niveau utilisés séquentiellement, comme montré dans la figure 11. Ensuite, à partir des agents îlots ainsi obtenus, des relations hiérarchiques spécifiques sont appliquées : ce sont les relations de Contrôleur et de Législateur décrites lors de la définition du pattern meso. Ainsi les *patterns* de création de niveau peuvent s'accompagner d'autres *patterns* spécifiques. L'utilisation conjointe de *patterns* est aussi pertinente en dehors de la généralisation. Par exemple, la modélisation d'une membrane cellulaire nécessite l'utilisation conjointe des *patterns* frontière et porte.

Outre qu'ils peuvent être utilisés conjointement, nous constatons aussi que les *patterns* proposés peuvent présenter des liens. Ainsi le *pattern* Agrégation et le *pattern* Embarquement présentent des similitudes. Néanmoins, les spécifi-



État initial : les bâtiments b_i sont situés dans un même environnement

Étape 1 : **agrégation** des bâtiments b_i en une ville a

Étape 2 : **décomposition** de la ville a en îlots i_k

Étape 3 : **hiérarchisation** des bâtiments au sein de leurs îlots respectifs

FIGURE 11 – Explicitation de la construction d'îlots urbains proposée par [3] à l'aide de *patterns*.

cités propres à l'agent embarqué, et la définition de comportement de maintien de la position relative qui motive sa création justifient sa définition de manière indépendante.

En termes d'exhaustivité, les *patterns* proposés sont issus des recherches effectués dans le cadre de la généralisation s'appuyant sur des modèles orientés agents. Une des contraintes de la généralisation étant de préserver relativement la position des objets sur la carte, le procédé n'implique pas une grande mobilité des agents. Il est donc très probable que les *patterns* pouvant être extraits des SMA pour la généralisation cartographique ne constituent pas une liste exhaustive, et que de nombreux *patterns* multi-niveaux intéressants restent à être identifié.

En revanche, en termes de généricité nous avons cherché à nous affranchir autant que possible du modèle d'origine dans lequel les *patterns* proposés ont été mis en œuvre, afin de ne retenir que les caractéristiques les plus générale d'une décomposition multi-niveau.

6 Conclusion et perspectives

Nous avons proposé un début de formalisation de *patterns* multi-niveaux pour des SMA. Ces *patterns*, identifiés à partir de cas particulier rencontrés lors de la mise en commun de modèles pour la généralisation cartographique, portent sur trois aspects spécifiques de relations multi-niveaux : la création de niveaux différents, des aspects structurels de la relation entre niveaux et les relations spécifiques d'un agent avec un agent d'un niveau supérieur dans lequel il est hébergé. Ces *patterns* d'analyse ont, à moyen terme, vocation à être étendus pour servir également à la conception de systèmes multi-niveaux. La liste des *patterns* proposés n'est pas exhaustives et gagnerait à être enrichie de nouvelles situations, à identifier à partir d'autres SMA multi-niveaux.

Références

[1] A. Atrash. Adaptation of PADAWAN multi-agent model for the solution of a spatial problem : the cartographic generalization. Rapport de stage, Univ. Paris Sud, 2011.

[2] M. Barrault, N. Regnauld, C. Duchêne, K. Haire, C. Baeijs, Y. Demazeau, P. Hardy, W. Mackaness, A. Ruas, et R. Weibel. Integrating multi-agent, object-oriented, and algorithmic techniques for improved automated map generalization. In *20th International Cartographic Conference (ICC'01)*, vol. 3, p. 2110—2116. International Cartographic Association, 2001.

[3] A. Boffet. Analyse multi-niveaux des espaces urbains. *Revue Internationale de Géomatique*, 12(2) :215–260, 2002.

[4] P. Caillou et J. Gil-Quijano. Description automatique de dynamiques de groupes dans des simulations à base d'agents. In P. Chevaillier et B. Mermet, éds., *Actes des 20e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2012)*, p. 23–32. Cépaduès, 2012.

[5] B. Camus, J. Siebert, C. Bourjot, et V. Chevrier. Modélisation multi-niveaux dans AA4MM. In P. Chevaillier et B. Mermet, éds., *Actes des 20e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2012)*, p. 43–52. Cépaduès, 2012.

[6] V. Couturier, D. Telisson, et M.-P. Huget. Patterns d'analyse pour l'ingénierie des systèmes multi-agents. In M. Occello et L. Rejeb, éds., *Actes des 18e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2010)*, p. 55–64. Cépaduès, 2010.

[7] D. David, D. Payet, et R. Courdier. Réification de zones urbaines émergents dans un modèle simulant l'évolution de la population à La Réunion. In

E. Adam et J.-P. Sansonnet, éds., *Actes des 19e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2011)*, p. 63–72. Cépaduès, 2011.

[8] C. Duchêne, A. Ruas, et C. Cambier. The CARTACOM model : transforming cartographic features into communicating agents for cartographic generalisation. *International Journal of Geographical Information Science*, 26(9) :1533–1562, 2012.

[9] C. Duchêne et G. Touya. Emergence de zones conflits dans deux modèles de généralisation cartographique multi-agents. In M. Occello et L. Rejeb, éds., *Actes des 18e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2010)*, p. 33–42. Cépaduès, 2010.

[10] J. Ferber, F. Michel, et J. Báez. AGRE : Integrating environments with organizations. *LNCS*, vol. 3374, p. 48–56. Springer, 2005.

[11] J. Gaffuri, C. Duchêne, et A. Ruas. Object-field relationships modelling in an agent-based generalisation model. In *Workshop on generalisation and multiple representation*, 2008.

[12] E. Gamma, R. Helm, R. Johnson, et J. Vlissides. *Design Patterns, Elements of Reusable Object-Oriented Software*. Addison Wesley, 1994.

[13] N. Gaud, S. Galland, et A. Koukam. Towards a multilevel simulation approach based on holonic multiagent systems. In *Computer Modeling and Simulation. UKSIM 2008. Tenth International Conference on*, p. 180–185. IEEE, 2008.

[14] J. Gil-Quijano, G. Hutzler, et T. Louail. De la cellule biologique à la cellule urbaine : retour sur trois expériences de modélisation multi-échelles à base d'agents. In Z. Guessoum et S. Hassas, éds., *Actes des 17e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2009)*, p. 187–198. Cépaduès, 2009.

[15] N. Hamani, J.-P. Jamont, M. Occello, et M. Koudil. Ant-MWAC : Une approche conjointe multi-agent et colonie de fourmis pour gérer les communications dans les réseaux de capteurs sans fil. In E. Adam et J.-P. Sansonnet, éds., *Actes des 19e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2011)*, p. 85–94. Cépaduès, 2011.

[16] T.-T.-H. Hoang, M. Occello, et J.-P. Jamont. Un modèle multi-agent générique récursif pour simplifier la supervision de systèmes décentralisés multi-niveaux. In E. Adam et J.-P. Sansonnet, éds., *Actes des 19e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2011)*, p. 41–50. Cépaduès, 2011.

[17] K. Jaara, C. Duchêne, et A. Ruas. Preservation and modification of relations between thematic and topographic data throughout thematic data migration process. In M. Buchroithner et al., éds., *Sel. Contrib. to the 26th Int. Conf. of the ICA*, Lecture Notes in Geoinformation and Cartography, p. 103–118. Springer, 2013.

[18] N. Jabeur, B. Boulekrouche, et B. Moulin. Using multiagent systems to improve real-time map generation. In *Advances in Artificial Intelligence*, p. 37–48. Springer, 2006.

[19] Y. Kubera, P. Mathieu, et S. Picault. Everything can be agent ! In W. van der Hoek et al., éds., *9th Int. Joint Conf. on Auton. Agents and Multi-Agent Systems (AAMAS)*, p. 1547–1548, 2010.

[20] Y. Kubera, P. Mathieu, et S. Picault. IODA : An interaction-oriented approach for multi-agent based simulations. *J. Auton. Agents and Multi-Agent Systems (JAAMAS)*, 23(3) :303–343, 2011.

[21] P. Mathieu et S. Picault. An interaction-oriented model for multi-scale simulation. In T. Walsh, éd., *Proc. of the 22nd Int. Joint Conf. on Artificial Intelligence (IJCAI'2011)*, p. 332–337. AAAI, 2011.

[22] P. Mathieu, S. Picault, et Y. Secq. Les environnements : en avoir ou pas ? Formalisation du concept et patterns d'implémentation. In R. Courdier et J.-P. Jamont, éds., *Actes des 22e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2014)*, p. 55–54. Cépaduès, 2014.

[23] A. Maudet, G. Touya, C. Duchêne, et S. Picault. Improving multi-level interactions modelling in a multi-agent generalisation model : first experiments. In D. Burghardt, éd., *Proc. of the 16th ICA Workshop on Generalisation and Map Production*, 2013.

[24] A. Maudet, G. Touya, C. Duchêne, et S. Picault. Representation of interactions in a multi-level multi-agent model for cartography constraint solving. In Yves Demazeau et al., éds., *Proc. of the 12th Int. Conf. on Practical Applications of Agents and Multi-Agent Systems (PAAMS'2014)*, no. 8473 in LNCS, p. 183–194. Springer, 2014.

[25] J. Renard. Introduction de structures réactionnelles à activation ascendante dans une organisation hiérarchique descendante d'agents - application à la généralisation des alignements urbains. In P. Chevaillier et B. Mermet, éds., *Actes des 20e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2012)*, p. 139–149. Cépaduès, 2012.

[26] A. Ruas. The role of meso objects for generalisation. In *9th Int. Symposium on Spatial Data Handling (SDH'00)*, Beijing (China), 2000.

[27] M. N. Sabo, Y. Bédard, B. Moulin, et E. Bernier. Toward self-generalizing objects and on-the-fly map generalization. p. 155–173, 2008.

[28] L. Sanders, D. Pumain, H. Mathian, F. Guérin-Pace, et S. Bura. SIMPOP : a multi-agents system for the study of urbanism. *Environment and Planning B*, 24 :287–305, 1997.

[29] D. Servat, E. Perrier, J.-P. Treuil, et A. Drogoul. When agents emerge from agents : Introducing multi-scale viewpoints in multi-agent simulations. *LNCS*, vol. 1534, p. 183–198, Paris, France, 1998. Springer.

[30] D. A. Vo, A. Drogoul, et J.-D. Zucker. An operational meta-model for handling multiple scales in agent-based simulations. In *Proc. of the IEEE Int. Conf. on Computing & Communication Technologies, Research, Innovation, and Vision for the Future (RIVF)*. IEEE Press, 2012.

[31] D. Weyns, H.V.D. Parunak, F. Michel, T. Holvoet, et J. Ferber. Environments for multiagent systems. State-of-the-Art and research challenges. *LNCS*, vol. 3374, p. 1–47. Springer, 2005.

[32] X. Zhang et E. Guilbert. A multi-agent system approach for feature-driven generalization of isobathymetric line. In *Advances in Cartography and GIScience. Volume 1*, p. 477–495. Springer, 2011.

Combiner les expertises au sein d'une simulation multi-agent multi-niveau

T. Huraux[a,b] thomas.huraux@edf.fr
N. Sabouret[c] nicolas.sabouret@limsi.fr
Y. Haradji[a] yvon.haradji@edf.fr

[a] EDF R&D,
Département ICAME, France

[b] LIP6, CNRS UMR 7606,
Université Pierre et Marie Curie, France

[c] LIMSI-CNRS, UPR 3251,
Université Paris-Sud, France

Résumé

Cet article aborde la question de la simulation multi-agent mettant en jeu plusieurs disciplines et expertises. Nous montrons que cela requiert de combiner la simulation multi-expert avec la modélisation multi-niveau. Pour ce faire, nous proposons le modèle SIMLAB [1] basé sur une représentation unifiée des concepts par des agents pouvant s'influencer les uns les autres dans différents axes et différents niveaux d'abstraction. Nous illustrons le potentiel de cette approche sur un exemple typique de système complexe multi-expert : la réduction de la consommation énergétique qui combine des expertises sur l'activité humaine, l'efficacité énergétique, la thermodynamique, etc.

Mots-clés : *Modélisation Multi-Niveau, Systèmes Multi-Agents, Simulation*

Abstract

This paper tackles the question of multi-agent simulation for multiple domain experts. We show that this requires to combine multi-domain simulation with multi-level modeling. To this purpose, we propose the SIMLAB agent model which is based on an unified representation of concepts as agents that can influence each other within different axes and different levels of abstraction. We illustrate this approach on a typical example of multi-expert complex system : reduction of household consumption, which requires to combine expertise on human activity, energy efficiency, thermodynamics, etc.

Keywords: *Multi-level Modeling, Multiagent Systems, Simulation*

1. SIMLAB Is Multi-Level Agent Based

1 Introduction

La simulation de systèmes complexes comme la prédiction de l'activité humaine en relation avec les réseaux sociaux [1], le transport [7] ou l'économie [19] requiert généralement d'assembler différentes expertises provenant d'un large éventail de domaines, allant des sciences humaines et sociales à l'étude de phénomènes physiques. Le modélisateur doit alors faire face à un grand nombre de variables interdépendantes, dont la dynamique est difficile à étudier par des modèles classiques (e.g. modèles équationnels).

La plupart des approches existantes considèrent la vue d'un seul expert principal sur le système pour la modélisation des agents, tandis que les autres domaines sont représentés comme des éléments de l'environnement [14]. Dans ce cas, étudier les phénomènes à travers plusieurs disciplines reste une question difficile, car le modèle est centré sur les agents d'un domaine particulier.

C'est pourquoi il apparait nécessaire de proposer des modèles de simulations multi-agents capables à la fois de rester proches des concepts manipulés par les différents experts (pour faciliter le processus de validation dans leurs domaines respectifs), et de combiner les différents niveaux d'abstraction de ces concepts, de manière à ce que chaque expert puisse comprendre les dynamiques des éléments liés à son domaine.

Dans cet article, nous considérons le cas spécifique de l'activité humaine en relation avec la consommation électrique résidentielle. Il s'agit là d'un exemple typique de système multi-expert. En effet, il touche différentes expertises telles que l'activité humaine dans l'habitat, la consommation des appareils, l'efficacité énergétique du bâtiment, le confort thermique,

l'impact de groupes sociaux sur le comportement individuel, *etc*. Notre premier défi dans ce contexte est donc de lier consommation et activité humaine dans un modèle qui reste accessible aux experts de l'énergie et des sciences humaines, de sorte que le résultat de la simulation puisse être validé à la fois du point de vue de la consommation et du point de vue de l'activité humaine. En ce sens, notre approche se distingue des modèles tels que [33] dans lesquelles chaque partie est vue comme une « boite noire ».

Dans cet article, nous présentons un SMA permettant d'envisager la modélisation et la simulation multi-expert de manière à ce que chaque expert puisse comprendre la dynamique des éléments liée à son propre domaine. Nous présentons dans la section 2 les travaux sur lesquels se fonde notre approche, combinant simulation de la consommation énergétique, SMA multi-experts et SMA multi-niveaux. Nous discutons ensuite les objectifs d'un SMA multi-expert pour le secteur de l'énergie et nous présentons notre modèle SIMLAB (section 3). Nous montrons dans la section 4 comment ce modèle peut être utilisé pour l'étude des comportements humains liés à la consommation. Nous concluons en section 5 et présentons les principales perspectives.

2 Travaux connexes

Bien que plusieurs travaux aient été consacrés à simuler la consommation électrique (les smart-grids [18], la simulation du marché de l'énergie [35], les modèles de bâtiments basse consommation [9], etc), peu de recherches se sont intéressés à l'activité humaine qui est pourtant un facteur important de la consommation [28]. Dans cette section, nous positionnons notre travail parmi les études énergétiques tenant compte de l'activité humaine, et nous discutons de notre positionnement sur la simulation multi-expert et la simulation multi-niveau.

2.1 L'humain dans les travaux énergétiques

La présence de l'utilisateur peut-être prise en compte dans plusieurs contextes : pour l'amélioration des interfaces de commande et de contrôle [27], pour le contrôle automatique des appareils selon la présence des utilisateurs [34], la simulation de l'activité des utilisateurs pour optimiser la performance énergétique [17, 6]. Nos travaux se situent dans cette troisième catégorie et plus précisément dans la simulation de l'activité dans le secteur résidentiel, qui est relativement peu étudié [10].

Le modèle probabiliste proposé par [21], basé sur des processus de Markov représentant des emplois du temps, propose un modèle SMA simple d'accès pour des experts non-spécialistes. Malheureusement, il ne prend en compte qu'une seule dimension (l'activité), en ignorant les questions de thermique du bâtiment, le confort individuel ou la consommation des appareils. Au contraire, [2] se concentre sur le confort thermique (le comportement des occupants est simulé en réponse à une forte chaleur estivale) mais les dimensions d'activité ne sont pas validées.

Dans le contexte résidentiel, un travail original est proposé par Kashif et al. [14] sur la conception d'un modèle détaillé du comportement humain pour la gestion énergétique. En se basant sur l'analyse de journaux d'activité, ils proposent un modèle de comportement des habitants à base d'agents BDI prenant en compte aussi la thermique du bâtiment et la consommation des appareils. Mais comme l'a souligné [11], un tel modèle est difficile à mettre en œuvre avec des experts du domaine (non-informaticiens) car il exige des cadres précis de modélisation. Ainsi, il constitue un modèle centré sur l'expertise de l'informaticien représentant la cognition des agents et les contraintes du système : il n'est pas accessibles aux experts du domaine et il devient difficile de valider chaque contribution.

En fait, la conception d'une plate-forme de simulation permettant d'étudier l'activité humaine et la consommation qu'elle induit doit, d'une part être en mesure de rassembler des expertises, et d'autre part, de combiner une description microscopique de l'activité humaine (*i.e.* les individus, les tâches, ...) avec l'utilisation de connaissances macroscopiques d'autres experts du domaine (*i.e.* les groupes sociaux, les habitudes, ...). C'est ce que propose le modèle SIMLAB présenté ici.

2.2 Du multi-expert au multi-niveau

Nous pouvons distinguer deux grandes catégories d'approches pour la simulation multi-expert. D'une part, la co-simulation [33] dans laquelle chaque expert gère ses propres sous-systèmes dont il connaît le fonctionnement et l'utilisation, les autres simulateurs sont utilisés comme des bibliothèques à la manière de

boîtes noires. D'autre part la simulation multi-niveau qui consiste à utiliser des représentations à la fois microscopiques et macroscopiques dans une même simulation, comme AA4MM [5], dans laquelle chaque niveau correspond à un modèle hétérogène. C'est dans cette deuxième catégorie que se placent nos travaux, mais notre objectif est de permettre aux experts de considérer différents niveaux au sein d'un même modèle.

Plusieurs travaux proposent de combiner différents niveaux au sein d'un même modèle. Par exemple SWARM [20] où les niveaux macro prennent le contrôle du niveau micro, [23, 22] dans la simulation de foule où les entités macro sont considérées comme des agrégations pour accélérer les simulations. Dans ces modèles, un seul niveau est activé à la fois : les niveaux micro et macro ne coexistent pas à un moment donné de la simulation.

Au contraire, les approches comme RIVAGE [29] ou les travaux de Tranouez [31] font coexister plusieurs niveaux en interaction forte et se concentrent sur l'étude des phénomènes multi-niveaux. Malheureusement, les modèles proposés sont fortement dépendants du domaine et ne peuvent pas être utilisés dans un autre cadre. Au contraire, l'approche SIMLAB, bien que conçue dans le cadre de la simulation de la consommation résidentielle, a pour objectif de proposer un méta-modèle qui peut être mis en œuvre dans un autre contexte applicatif.

Parmi les approches plus génériques, le modèle PADAWAN [24] permet la coexistence d'agents et d'environnements correspondant à différentes échelles spatiales et temporelles grâce à l'approche IODA [16]. Les auteurs proposent de représenter tous les concepts dans le système par des agents qui peuvent changer de niveau (*i.e.* d'environnement) en fonction de leur activité. Dans notre travail, nous ne nous sommes pas intéressés à avoir différentes échelles de temps ou des changements de niveau, mais nous voulons prendre en compte différents niveaux de modélisation du système et combiner des modèles de différents domaines. Nous partageons le point de vue que toutes les entités du système sont des agents pour fournir une homogénéité des concepts manipulés par le modélisateur [15] lorsqu'il combine les expertises. Dans cet article, les agents sont une métaphore, un paradigme nous permettant d'incarner et animer la connaissance des experts dans la simulation. De ce fait, nous nous distinguons des modèles organisationnels tels que AGR [8] et RIO [12] dont les agents cherchent à réaliser des buts en fonction de leurs rôles.

3 Le modèle SIMLAB

Le modèle que nous proposons ci-après est basé sur les deux observations suivantes.

Tout d'abord, certains éléments d'expertise, issus de différentes sources, peuvent être considérés comme appartenant à un même domaine de connaissance. Par exemple, l'expert de la consommation d'un réfrigérateur manipule des éléments liés à la température, le débit d'air et la consommation d'énergie qui sont aussi des éléments clés pour l'expert du rendement thermique des bâtiments. Chacun de ces éléments est caractérisé par un ensemble de propriétés et il n'est pas rare que ces propriétés soient partagées par d'autres domaines d'expertise. Ces propriétés communes définissent ce qui est typique d'un certain domaine de la connaissance, qui peut être considérée comme un axe du problème étudié.

Deuxièmement, différents niveaux de représentation apparaissent dans chacun de ces axes. En effet, la tâche de modélisation est toujours menée avec une vue spécifique au domaine cible. Cela conduit chaque expert à se positionner à un certain niveau de granularité (par exemple, l'individu, le ménage, la ville) pour la simplification et l'observation du système. Le défi d'un SMA multi-expert est de combiner la vue multi-axe avec une approche multi-niveau.

L'originalité du modèle SIMLAB repose sur l'idée que les agents peuvent avoir des *propriétés communes* caractéristiques d'un certain domaine et former ce que nous appelons un *axe de modélisation*. Les agents d'un même axe s'organisent en différents *niveaux* d'abstraction qui vont interagir les uns avec les autres de manière à rester accessible aux experts. Enfin, nous distinguons les *interactions* des agents (*i.e.* les échanges d'informations ou demandes) et les *influences* intra-axes, qui captent les dynamiques *multi-niveaux* et assurent la cohérence du système.

3.1 Agents et propriétés

Nous notons *Agt* l'ensemble des agents qui seront désignés en utilisant les lettres x, y, z... De façon classique, un agent est associé à un et un seul *type* caractérisé par un ensemble de

propriétés qui définissent les notions caractéristiques du concept modélisé. Par exemple, l'âge d'un individu, la puissance d'un appareil, *etc* sont représentés par des propriétés d'agents. En s'inspirant de la programmation orienté objet, chaque agent a des valeurs instanciées pour chacune des propriétés associées à son type.

Dans la suite, nous noterons ω un type d'agent, Ω l'ensemble des types, x *is* ω le fait que x est un agent de type ω, $\omega.p$ la propriété p des agents de type ω et $x.p$ la propriété p de l'agent x de type ω. Nous notons $\mathcal{P}$ l'ensemble de toutes les propriétés et $\mathcal{P}(\omega)$ l'ensemble des propriétés du type ω.

3.2 Axes de modélisation

Certaines questions transverses regroupent un ensemble d'objets d'étude communs à plusieurs experts. Ainsi, la consommation électrique est étudiée au niveau des appareils, des logements, des villes... Il est alors possible d'identifier des *propriétés communes* aux différentes expertises. Les propriétés propres à un type sont appelées *propriétés spécifiques*.

En partant de cette observation cruciale, nous introduisons la notion d'*axe de modélisation* qui regroupe les points de vue de différentes disciplines autour d'un ensemble de propriétés communes. Ce sont ces propriétés communes qui définissent ce qui est caractéristique de l'axe de modélisation.

Chaque type d'agent apporte les propriétés spécifiques d'un aspect du système étudié. Un type appartient à un et un seul axe de modélisation, noté χ. En ce sens, l'ensemble des axes de modélisation représente une partition de l'ensemble des types. Un axe est caractérisé par un ensemble de propriétés communes des types de cet axe, noté $\mathcal{P}(\chi)$. Ainsi, $\mathcal{P}(\chi) = \bigcap_{\omega \in \chi} \mathcal{P}(\omega)$

Remarque. Les propriétés portent sur des concepts et se détachent du vocabulaire des experts. Ainsi deux experts peuvent utiliser des termes différents pour une même propriété.

3.3 Relation inter-niveau

La notion de propriétés communes n'est pas suffisante pour caractériser un axe de modélisation. En effet, bien qu'ils partagent certaines propriétés, les agents d'un même axe peuvent avoir des niveaux d'abstraction différents. Chaque expert adopte un but de modélisation spécifique au départ et réalise des simplifications en produisant son modèle, la première étant de se placer à un certain niveau de granularité pour observer le système étudié. Autrement dit, l'expert se positionne à un certain niveau d'abstraction pour la représentation des propriétés spécifiques à son domaine d'expertise. Les agents qui capturent ces propriétés spécifiques sont donc situés à des niveaux d'abstraction différents au sein de l'axe de modélisation et il est nécessaire, pour la structuration du modèle, de définir les relations entre les entités des différents niveaux.

Dans notre modèle, nous définissons la relation inter-niveau $\sqsubset$ entre les types d'agent. $\omega_1 \sqsubset \omega_2$ signifie que les agents du type ω_2 sont des représentations de concepts qui agrègent des agents du type ω_1. Par exemple, en considérant la biologie comme un axe de modélisation, nous avons : *cellule* $\sqsubset$ *tissue* $\sqsubset$ *organe* $\sqsubset$ .... Cette relation de hiérarchie est non-transitive et acyclique : elle ne relie les agents que d'un niveau à l'autre. Toutefois, un type d'agent peut être relié à plusieurs types d'agents de niveau supérieur et plusieurs types d'agents de niveau inférieur. Enfin, la relation $\sqsubset$ ne donne pas d'information sur l'arité des instances d'agents. Un agent peut avoir plusieurs super-agents d'un même type (par exemple, un individu peut appartenir à plusieurs groupes sociaux) et il aura généralement plusieurs sous-agents (un groupe est composé de plusieurs individus).

Pour conclure cette définition, un axe de modélisation χ peut être décrit comme un sous-ensemble connexe d'Ω pour la relation $\sqsubset$ et un ensemble de propriétés communes $\mathcal{P}(\chi)$.

Remarque. Il est important de noter que cette organisation est conceptuelle : dans notre modèle, les agents de tous les niveaux co-existent à l'exécution et s'échangent des informations au sein de l'axe de modélisation et entre les axes.

3.4 Influences

La coexistence dans le modèle de différents niveaux n'est pas uniquement un choix de modélisation : elle s'appuie sur la définition de liens entre certaines de leurs propriétés, c'est-à-dire d'influences entre des agents issus de domaines d'expertise et de niveaux de représentation différents. Ainsi, deux types d'agents sont reliés par la relation $\sqsubset$ bénéficient d'influences entre leurs propriétés. Le terme *influence* est utilisé ici pour son sens aussi large qu'*interaction*. Nous nous démarquons des travaux de [8] où

les influences permettent de modéliser les actions simultanées sur l'environnement. Les influences présentées ici permettent aux experts de tester des hypothèses de lien entre les disciplines. Même si certaines sont triviales comme le fait que la consommation d'un logement soit la somme des consommations des différentes pièces qui le composent. D'autres en revanche, relèvent de la prospective, comme par exemple l'influence du type de famille sur la gestion du chauffage par les individus.

Nous notons $\mathcal{F}$ l'ensemble des influences. Une influence $\delta \in \mathcal{F}$ est caractérisée par un triplet $\langle p_{src}, p, infl \rangle$ avec $p_{src} \in \mathcal{P}$ la propriété "influencante", $p \in \mathcal{P}$ la propriété influencée et $infl$ la fonction d'influence correspondante telle que $infl : D \times D_{src} \rightarrow D$ où D et D_{src} les domaines de définition de p et p_{src}.

Les fonctions d'influence sont systématiquement déclenchées lorsque la propriété p est modifiée (par une action, une interaction ou encore une autre influence). En cas de cycle, les influences sont appelées une seule fois. Par ailleurs, deux types d'agents d'un même niveau ne peuvent pas être reliés par une fonction d'influence : cela constituerait une violation du principe d'encapsulation des SMA[13]. Les fonctions d'influence ne définissent pas des interactions au sens SMA du terme : il n'y a pas d'intention dans la communication. Il s'agit juste de mettre à jour la propriété d'un parent ou d'un enfant pour maintenir la cohérence dans le système. Par exemple, la consommation du logement doit être ajustée lorsque la consommation d'un appareil change.

Remarque. L'introduction des influences nous permet de définir des propriétés communes spéciales avec une influence qui se propage comme une fonction récursive dont le calcul se fait de l'agent vers les sous-agents. Ce mécanisme permet au modélisateur d'établir une forme de cohérence automatique des comportements entre les différents niveaux (*e.g.* par exemple établir un lien entre les préférences d'un individu et ceux d'un groupe auquel il appartient).

3.5 Dynamique des agents

La représentation des actions et interactions présentée ci-après n'est pas inédite. Néanmoins, le modèle SIMLAB ne saurait être compris sans en donner une description succincte. Nous utiliserons l'opérateur ":=" pour définir le changement d'état d'une propriété : $x.p := v$ signifie qu'une valeur donnée v *est allouée* à la propriété p de l'agent x.

Actions. Nous notons $\mathcal{A}(\omega)$ l'ensemble des actions pour les agents du type ω. Une action $a \in \mathcal{A}(\omega)$ est un couple (Op, pre) avec : Op l'ensemble des allocations caractérisant l'action et pre une fonction booléenne de $\mathcal{P}(\omega)$ définissant la précondition de l'action. Les allocations Op et la fonction booléenne pre sont évaluées dans le contexte de l'agent qui effectue l'action, c'est-à-dire que chaque propriété p apparaissant dans l'une de ses expressions est remplacée par $x.p$ où x est l'agent de type ω qui effectue l'action. Les actions ne peuvent pas utiliser de propriétés d'un autre agent, conformément au principe d'encapsulation.

Nos agents s'exécutent de manière non-déterministe synchrone : à chaque pas de simulation, tous les agents effectuent toutes les actions dont les conditions pre sont satisfaites, dans un ordre aléatoire. Notre modèle n'offre pas de mécanisme de gestion des modifications concurrentes sur une propriété (soit directement par allocation ou par interaction) : le modélisateur doit gérer ce problème quand il met en œuvre les agents (typiquement, il doit s'assurer que les préconditions forment une partition de l'espace d'état).

Interactions. Inspirées par IODA[16], les interactions sont définies comme extérieures aux agents. Nous notons $\mathcal{I}(\omega)$ l'ensemble des modèles d'interaction des agents de type ω. Par exemple, un agent de type *individu* peut interagir avec un agent *tâche* pour la commencer. Un modèle d'interaction $mi \in \mathcal{I}(\omega)$ est un n-uplet $\langle T, perf, Op, pre(T), c \rangle$ avec T un ensemble décrivant les agents paramètres du modèle d'interaction, sous la forme $\{x \; is \; \omega_x\}$; $perf$ le performatif caractérisant cette interaction (le message qui sera reçu par les agents destinataires) ; Op la modification associée ; pre la précondition pour l'agent source (qui peut dépendre des variables définies dans T) ; et c la condition à évaluer chez le ou les agents cibles pour que l'interaction soit réussie. La condition garantit l'autonomie des agents en leur permettant de refuser l'interaction.

Par exemple , nous définissons le modèle d'interaction pour l'activation d'un appareil électrique par une tâche :

```
T ← {t is Tache, a is Appareil}
perf ← activer
```

```
Op ← {a.state := ON}
pre(T) ← t.necessite(a)
c ← a.disponible
```

A partir de ces modèles, nous définissons, au niveau de chaque agent en exécution, des interactions en instanciant les paramètres du modèle d'interaction pour définir une interaction possible. Elle sera effectuée si et seulement si la précondition, qui dépend de ces paramètres, est vérifiée.

4 Application dans le contexte énergétique

Le modèle SIMLAB a été implémenté en Java et intégré au sein de la plate-forme de simulation SMACH [2] [3]. Cette plate-forme permettant de simuler l'activité d'un foyer en rapport avec sa consommation électrique ne permettait pas de combiner différentes expertises. L'architecture générale a été modifiée pour l'organisation en axes et la gestion des influences. De nouveaux agents ont été ajoutés aux agents *individus*, *tâches* et *appareil* historiquement présents dans la plateforme. Par manque de place, nous ne présenterons pas ici le détail des agents dans le formalisme de la section 3 mais nous décrivons les principales propriétés des agents dans le modèle étendu, afin d'illustrer comment notre contribution (le modèle SIMLAB) est mise en œuvre dans le cadre de la simulation de la consommation résidentielle.

Comme le montre la figure 1, le modèle SMACH basé sur SIMLAB utilise trois axes de modélisation, correspondant aux trois principaux domaines d'études que nous avons identifiés : la population (des individus aux groupes sociaux), l'activité (de la tâche au style de vie) et l'environnement de consommation (de l'appareil électrique au logement). Nous présentons ci-après les caractéristiques de ces trois axes et les agents qui les composent.

4.1 Axe des populations

Cet axe vise à représenter les expertises en sciences humaines (de la psychologie à la sociologie). Un agent appartenant à cet axe de modélisation possède un ensemble d'activités possibles et les préférences associées. Cet axe est également caractérisé par une fonction de *priorité* sur les activités de l'agent, qui fait référence

2. Simulations Multi-Agents des Comportements Humains - `http://www.youtube.com/watch?v=DViBg3-crxM`

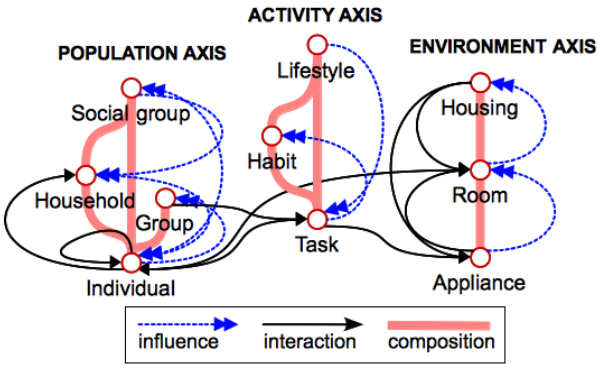


FIGURE 1 – Aperçu des 3 axes.

à la motivation humaine manipulée par des experts en sciences humaines. Celle-ci joue un rôle clé dans la sélection des activités. Elle est définie comme une propriété commune dont la valeur est influencée par la priorité de ses sous-agents.

Individu. Les individus représentent les membres d'un foyer et sont caractérisés, entre autres, par un confort thermique calculé en utilisant l'équation de Fanger [32]. Les individus interagissent avec les tâches pour les réaliser en fonction de leurs priorités, conformément à la modélisation du comportement humain utilisée par les experts ergonomes [11]. En plus de l'état interne de l'individu, la priorité peut être influencée par des facteurs externes tels que le niveau du prix de l'énergie ou les invitations des autres individus, représentées par des interactions. Les individus possèdent un ensemble ad-hoc d'actes de langage (*i.e.* demander à un autre ce qu'il fait, encourager quelqu'un à accomplir une activité, ...).

Groupe. Les agents de type « groupe » permettent l'étude de l'activité collective au sens de [4]. Nous considérons un groupe comme une entité temporaire du système : les groupes se font et se défont au fur et à mesure de l'activité. Il peut s'agir par exemple de personnes qui passent une soirée ensemble ou d'une mère qui aide son enfant avec ses devoirs. Un groupe est principalement caractérisé par ses membres qui agissent régulièrement ensemble (*i.e.* même pièce et même période). Les membres d'un groupe peuvent ne pas exercer la même tâche.

Foyer. En intégrant les expertises statistiques de l'enquête sur les ménages de l'INSEE [3], un ménage est défini par sa structure familiale,

3. Institut national de la statistique et des études économiques

sa sensibilité au confort (éco-orienté, moyen, axée sur le confort) et son revenu. Il dispose d'un confort thermique et agit sur le gestionnaire énergétique pour réguler les températures de consigne dans les pièces en fonction de ses propriétés. Le ménage met à jour son confort en fonction des influences de ses sous-agents (individus), obtenu suivant le mécanisme d'influence (voir 3.4).

Les propriétés du foyer influencent les propriétés des individus et modifient donc indirectement leur activité. Ainsi, un ménage sensible à la consommation est modélisé par une influence qui réduit la valeur de la priorité lorsqu'un appareil consommant beaucoup d'énergie (*e.g.* un four ou un lave-linge) est nécessaire pour effectuer la tâche correspondante.

Groupe Social. Un groupe social est composé de foyers et d'individus. Il est caractérisé par un ensemble de logiques d'action qui agissent comme une sorte de prisme de décision pour les sous-agents. Ces logiques d'action sont issus de la théorie sociale [30] et nous permettent de tester plusieurs hypothèses interdisciplinaires à l'aide du mécanisme des influences (*e.g.* comment une logique d'action donnée peut influencer les actions des individus). Dans nos travaux, nous ne considérons que quatre logiques d'action basées sur les travaux des sociologues EDF : l'importance du confort, la recherche d'économies, la responsabilité écologique et la gestion rationnelle du logement. Les logiques d'action influencent la priorité des sous-agents.

4.2 Axe de l'activité

En s'inspirant des principes de l'action et de la cognition située en ergonomie [26], l'axe de l'activité représente toutes les activités humaines à différentes granularités. Une activité est contrainte par un ensemble d'activités préconditions (*e.g.* faire la lessive est pré-condition de repasser) et un environnement nécessaire (*e.g.* repasser nécessite un fer à repasser). Il dispose également d'un état d'activation (non réalisée, réalisée, en cours). Une activité est impossible tant que ses préconditions ne sont pas satisfaites : l'environnement n'est pas disponible ou une activité pré-conditionnelle est non réalisée. Lorsqu'une activité liée à un environnement se réalise, elle interagit avec celui-ci pour l'activer.

Tâche. Nous définissons la tâche comme une action générique dans la maison. Elle peut être faite individuellement ou collectivement et peut avoir un rythme correspondant à une certaine régularité dans sa réalisation modifiant la probabilité qu'un individu la réalise (*e.g.* le dîner a lieu tous les jours entre 19h et 21h et dure environ 1h) . Une tâche peut avoir des tâches pré-conditionnelles (*e.g.* le dîner doit être préparé). Un individu effectue une tâche donnée si et seulement s'il a l'information que toutes les tâches pré-conditionnelles sont réalisées. Enfin, une tâche peut interagir avec un appareil pour modifier son état d'activation (*e.g.* faire la lessive active la machine à laver) et donc entraîner une consommation d'énergie.

Habitude. Une habitude correspond à un ensemble de tâches. Il représente des motifs récurrents dans l'activité humaine (*e.g.* la routine du soir en semaine). Une habitude est caractérisée par un rythme résultant de l'influence des rythmes de ses sous-agents « tâches ». Dans le cadre d'études énergétiques, il est intéressant de donner aux motifs de tâches une certaine visibilité : cela permet aux experts de mettre ces habitudes en relation avec la consommation électrique du foyer (par exemple quel est l'impact des activités de loisirs sur la consommation au niveau des individus, des foyers et du groupe social). Ils permettent au modélisateur de détecter de trop grande régularité dans les comportements produits (ce qui pourrait correspondre à des erreurs de modélisation ou des paramètres de simulation qui n'ont pas été correctement configurés).

Mode de vie. Inspirés des travaux en sociologie de l'énergie [30], les modes de vie sont les agents qui incarnent un aspect de la connaissance des experts en sociologie à la manière des groupes sociaux dans l'axe de la population. Basé sur des concepts sociaux, un mode de vie est également caractérisé par un ensemble de pratiques (*i.e.* peut être vu comme des marqueurs des caractéristiques sociologiques) qui influent sur la durée des tâches (*e.g.* ceux qui passent beaucoup de temps dans la préparation des repas) et la fréquence des rythmes (*e.g.* faire plus souvent des activités de loisirs) basée sur des hypothèses d'étude de l'énergie.

4.3 Axe de l'environnement de consommation

Cet axe représente les agents liés à la consommation énergétique (de l'appareil au logement). Un agent de cet axe se caractérise par une puis-

sance électrique, une fonction de consommation et un état d'activation (arrêt, veille ou allumé).

Appareil. Un appareil met à jour sa consommation électrique et peut percevoir la température ambiante. Les profils de consommation électrique des appareils proviennent de la BDD REMODECE [4] mesurée par des experts de l'énergie en situations réelles. Tous les appareils influencent la température de la pièce en fonction de leur propriété de rayonnement, caractéristique du fonctionnement des appareils électriques.

Pièce. On définit une pièce comme un super-agent d'appareils caractérisé par un nombre de personnes présentes, une température courante et une température cible pour les appareils de chauffage. Une pièce met à jour sa température en utilisant un modèle thermodynamique et peut percevoir le nombre de personnes présentes. Cet agent a pour principal objectif d'intégrer l'expertise en thermique du bâtiment [25] et permet d'étudier son impact au niveau des habitants, contrairement aux scénarios normatifs habituellement utilisés dans les études sur le confort thermique et l'énergie du bâtiment comme [17].

Logement. Les agents logement sont composés de pièces et visent à intégrer les expertises sur l'efficacité énergétique et l'enveloppe thermique et à tester l'efficacité de différentes stratégies de chauffe dans le logement. Un logement est caractérisé par un indicateur de présence des individus. Il peut, en fonction de son gestionnaire énergétique, interagir avec les pièces pour modifier les températures de consigne. La consommation d'électricité, définie comme une propriété calculée récursivement, dépend des influences des ses sous-agents (pièces).

4.4 Mise en œuvre

Grâce aux différents agents présentés ci-dessus, nous voyons la diversité des expertises nécessaires pour représenter le problème difficile de la simulation des comportements humains, même limitée au contexte résidentiel. Pourtant, la structure du problème en plusieurs axes de modélisation et la représentation unifiée des concepts en agents permettent d'avoir un modèle de simulation accessible aux experts. D'une part, ils restent proche de leur domaine en termes de description des concepts et de niveaux d'abstraction, et d'autre part, ils peuvent travailler sur la dynamique interdisciplinaires grâce au mécanisme des influences et les interactions.

4. REMODECE : base de données européenne sur la consommation résidentielle - http ://remodece.isr.uc.pt

FIGURE 2 – Extrait d'un diagramme d'activité illustrant l'adaptation de l'activité des individus sous l'influence de leur foyer.

La figure 2 illustre un résultat de simulation dans la plate-forme SMACH. Le diagramme d'activité permet de visualiser l'impact de la sensibilité aux prix de l'énergie du foyer sur la réalisation des tâches par les individus. À la manière d'un agenda, l'axe vertical représente l'heure du jour alors que l'axe horizontal renseigne sur la date du jour : une colonne par jour et une couleur par tâche. Par souci de clarté, seules les tâches consommatrices sont affichées. Dans cette simulation les périodes fléchées (de 18h à 20h les jours avec une pastille bleue) indiquent des prix plus chers et nous pouvons ces jours là constater une réorganisation importante de l'activité des individus, sous l'influence du foyer. Les tâches consommatrices sont reportées plus tard dans la soirée. Cette mise en œuvre concrète d'une influence multi-niveau, ici l'influence du foyer sur l'individu, est un exemple typique d'hypothèse que les experts énergéticiens souhaitent pouvoir étudier en simulation.

5 Conclusion

Dans cet article, nous avons présenté le modèle SIMLAB pour la simulation multi-expert de systèmes complexes. Il s'articule autour de la notion d'axe de modélisation qui permet de capturer les propriétés communes aux différents domaines d'expertises. Les agents situés à différents niveaux d'abstraction, peuvent s'influencer au sein d'un axe et interagir avec les agents d'autres axes. Nous avons décrit la mise en

œuvre à l'aide de SIMLAB d'un modèle agent dans le contexte de la consommation électrique résidentielle.

Ce modèle est implémenté dans la plateforme SMACH. Nous travaillons actuellement sur sa validation en se basant sur la réalisation de simulations participatives sur un échantillon de foyers en Bretagne. L'objectif est double : (1) valider les éléments micro du modèle (*i.e.* individu-tâche-appareil) en confrontant au regard des clients EDF leur activité simulée et en comparant consommation simulée et réelle. (2) valider le modèle multi-niveau multi-expert SIMLAB en montrant l'apport des différents niveaux, axes et influences sur la capacité de description et d'analyse du système réel. L'idée générale consiste à ajouter des expertises les unes après les autres sous la forme d'agents de différents niveaux et de démontrer que les résultats tendent à se rapprocher des mesures sur le système réel.

Par la suite, il sera intéressant d'utiliser notre modèle dans d'autres contextes que l'activité humaine et la consommation énergétique pour tester l'adaptation de notre approche à d'autres contextes multi-experts.

Références

[1] A. Alaali, M. A. Purvis, and B. T. R. Savarimuthu. Vector opinion dynamics : An extended model for consensus in social networks. In *Web Intelligence and Intelligent Agent Technology, 2008. WI-IAT'08. IEEE/WIC/ACM International Conference on*, volume 3, page 394–397, 2008.

[2] A. Alfakara and B. Croxford. Using agent-based modelling to simulate occupants' behaviours in response to summer overheating. In *Proceedings of the Symposium on Simulation for Architecture & Urban Design*, page 13. Society for Computer Simulation International, 2014.

[3] E. Amouroux, T. Huraux, F. Sempé, N. Sabouret, and Y. Haradji. Smach : Simuler l'activité humaine pour limiter les pics de consommation électrique. In *Journées Francophones sur les Systèmes Multi-Agents (JFSMA)*, pages 51–60. Cepadues Editions, 2013.

[4] J. Bourbousson, C. Seve, and T. McGarry. Space–time coordination dynamics in basketball : Part 1. intra-and inter-couplings among player dyads. *Journal of Sports Sciences*, 28(3) :339–347, 2010.

[5] B. Camus, C. Bourjot, and V. Chevrier. Multi-level modeling as a society of interacting models. In *Proceedings of the Agent-Directed Simulation Symposium*, page 3. Society for Computer Simulation International, 2013.

[6] D. Claridge, B. Abushakra, J. Haberl, and A. Sreshthaputra. Electricity diversity profiles for energy simulation of office buildings (rp-1093). *ASHRAE Transactions*, 110(1) :365–365, 2004.

[7] A. Doniec, R. Mandiau, S. Piechowiak, and S. Espié. A behavioral multi-agent model for road traffic simulation. *Engineering Applications of Artificial Intelligence*, 21(8) :1443–1454, 2008.

[8] J. Ferber and J.-P. Müller. Influences and reaction : a model of situated multiagent systems. In *Proceedings of Second International Conference on Multi-Agent Systems (ICMAS-96)*, pages 72–79, 1996.

[9] R. Z. Freire, G. H. Oliveira, and N. Mendes. Predictive controllers for thermal comfort optimization and energy savings. *Energy and Buildings*, 40(7) :1353 – 1365, 2008.

[10] A. Grandjean. *Introduction de non linéarités et non stationnarités dans les modèles de représentation de la demande électrique résidentielle*. PhD thesis, Thèse de doctorat, Mines Paristech, 2013.

[11] Y. Haradji, G. Poizat, and F. Sempé. *Human Activity and Social Simulation*, pages 416–425. CRC Press, 2012.

[12] V. Hilaire, A. Koukam, P. Gruer, and J.-P. Müller. Formal specification and prototyping of multi-agent systems. In *Engineering Societies in the Agents World*, pages 114–127. Springer, 2000.

[13] N. R. Jennings. An agent-based approach for building complex software systems. *Communications of the ACM*, 44(4) :35–41, 2001.

[14] A. Kashif, S. Ploix, J. Dugdale, and X. H. B. Le. Simulating the dynamics of occupant behaviour for power management in residential buildings. *Energy and Buildings (online pre-print)*, 2012.

[15] Y. Kubera, P. Mathieu, and S. Picault. Everything can be agent ! In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems : volume 1-Volume 1*, pages 1547–

1548. International Foundation for Autonomous Agents and Multiagent Systems, 2010.

[16] Y. Kubera, P. Mathieu, and S. Picault. Ioda : an interaction-oriented approach for multi-agent based simulations. *Autonomous Agents and Multi-Agent Systems*, 23(3) :303–343, 2011.

[17] Y. S. Lee, Y. K. Yi, and A. Malkawi. Simulating Human Behaviour and its Impact on Energy Uses. In *Proc. of the 12th Conference of International Building Performance Simulation Association (IBPSA)*, pages 1049–1056, 2011.

[18] A. S. Massoud and B. Wollenberg. Toward a smart grid : power delivery for the 21st century. *Power and Energy Magazine, IEEE*, 3(5) :34–41, 2005.

[19] P. Mathieu and O. Brandouy. A generic architecture for realistic simulations of complex financial dynamics. In *Advances in Practical Applications of Agents and Multiagent Systems*, pages 185–197. Springer Berlin Heidelberg, 2010.

[20] N. Minar, R. Burkhart, C. Langton, and M. Askenazi. The swarm simulation system : a toolkit for building multi-agent simulations. *GEMAS Studies in Social Analysis*, Working Paper 96-06-042, 1996.

[21] M. Muratori, M. C. Roberts, R. Sioshansi, V. Marano, and G. Rizzoni. A highly resolved modeling technique to simulate residential power demand. *Applied Energy*, 107 :465–473, 2013.

[22] L. Navarro, F. Flacher, and V. Corruble. Dynamic level of detail for large scale agent-based urban simulations. *Proc. of 10th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2011)*, pages 701–708, 2011.

[23] T. N. A. Nguyen, J.-D. Zucker, N. H. Du, A. Drogoul, and D.-A. Vo. An hybrid equation-based and agent-based modeling of crowd evacuation on road network. *International Conference on Complex Systems*, 2011.

[24] S. Picault, P. Mathieu, et al. An interaction-oriented model for multi-scale simulation. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, volume 22, page 332, 2011.

[25] G. Plessis, É. Amouroux, and Y. Haradji. Coupling occupant behaviour with a building energy model-a fmi application. In *Proceedings of the 10th International ModelicaConference*, 2014.

[26] M. Relieu, P. Salembier, and J. Theureau. Introduction au numéro spécial activité et action/cognition située. *Activités*, 1(2) :3–10, 2004.

[27] A. Rogers, S. Maleki, S. Ghosh, and J. Nicholas R. Adaptive home heating control through gaussian process prediction and mathematical programming. In *ATES 2011*, pages 71–78, May 2011.

[28] C. Seligman, J. M. Darley, and L. J. Becker. Behavioral approaches to residential energy conservation. *Energy and buildings*, 1(3) :325–337, 1978.

[29] D. Servat, E. Perrier, J.-P. Treuil, and A. Drogoul. When agents emerge from agents : Introducing multi-scale viewpoints in multi-agent simulations. *LNCS*, 1534 :183–198, 1998.

[30] M.-P. Thomas. Energy consumption in residential and transport sectors : an analysis based on a lifestyle approach. In *4th ECLEER Seminar (European Centre and Laboratories for Energy Efficiency Research), EDF R&D*, 2011.

[31] P. Tranouez, C. Bertelle, and D. Olivier. Changing levels of description in a fluid flow simulation. In *Emergent Properties in Natural and Artificial Dynamical Systems*, pages 87–99. Springer, 2006.

[32] J. van Hoof. Forty years of Fanger's model of thermal comfort : comfort for all ? *Indoor Air*, 18(3) :182–201, 2008.

[33] M. Wetter. Modelica-based modelling and simulation to support research and development in building energy and control systems. *Journal of Building Performance Simulation*, 2(2) :143–161, 2009.

[34] R. Yang and L. Wang. Development of multi-agent system for building energy and comfort management based on occupant behaviors. *Energy and Buildings*, 56 :1–7, 2013.

[35] Z. Zhou, W. K. V. Chan, and J. H. Chow. Agent-based simulation of electricity markets : a survey of tools. *Artificial Intelligence Review*, 28(4) :305–342, 2007.

Auto-organisation d'agents embarqués pour l'apprentissage par démonstration : principes et expérimentations

N. Verstaevel[a, b] verstaev@irit.fr
C. Régis[b] regis@irit.fr
M.P. Gleizes[b] gleizes@irit.fr
F. Robert[a] fabrice.robert@sogeti.com

[a]Sogeti High Tech,
3 Chemin de Laporte, Toulouse, France

[b]IRIT, Équipe SMAC,
Université Paul Sabatier, Toulouse, France

Résumé

L'adaptation des systèmes ambiants aux besoins spécifiques des utilisateurs est une tâche complexe. Pour rendre l'interaction humain-système aussi naturelle que possible pendant l'adaptation, nous proposons une approche basée sur l'apprentissage par démonstration. Cet apprentissage requiert des techniques d'apprentissage adaptatifs. Nous présentons Alex, un système multi-agent capable d'apprendre dynamiquement un comportement à partir de démonstrations réalisées par un tuteur. Les résultats d'expérimentations réalisées à la fois sur un robot réel et virtuel mettent en avant des propriétés intéressantes de notre technologie pour des applications ambiantes.

Mots-clés *: Robotique collective, Intelligence ambiante, Apprentissage, Auto-organisation*

Abstract

The adaptation of an ambient system to the specific needs of its users is a challenging task. Because human-system interaction has to be as natural as possible, we propose an approach based on Learning from Demonstration (LfD). However, using LfD in ambient systems needs adaptivity of the learning technic. We present Alex, a multi-agent system able to dynamically learn and reuse contexts from demonstration made by a tutor. Results of experiments performed on both a real and virtual robot show interesting properties of our technology for ambient applications.

Keywords*: Collective robotic, Ambient intelligence, Learning, Self-organization*

1 Introduction

Autrefois restreintes à une science du contrôle en environnements contraints, les recherches en robotique s'intéressent aujourd'hui à l'intégration de systèmes intelligents ou cyber-physiques autonomes dans des environnements où les tâches sont multiples, complexes et évolutives (Kaminka, 2012). La robotique de service diffère de sa version industrielle parce qu'elle fournit des services à des utilisateurs plutôt que de réaliser des tâches répétitives adaptées au processus de fabrication. C'est pourquoi les scientifiques s'intéressent à l'utilisation de dispositifs robotiques dans des applications ambiantes (Chung, 2014). Ces systèmes ambiants se caractérisent par leur dynamique et leur complexité. Un grand nombre de dispositifs (capteurs et effecteurs) hétérogènes peuvent apparaître et disparaître pour interagir de manière opportuniste. (Russel et al, 1995) décrivent l'environnement de ces dispositifs comme :

- **Inaccessible** : chaque entité n'a qu'une observation partielle de l'environnement.

- **Continu** : le nombre d'observations et

d'actions dans le monde réel n'est pas discret.

- **Non-déterministe** : les conséquences d'une action réalisée dans le monde réel ne peuvent pas être déterminées à l'avance.

- **Dynamique** : les actions du système, l'activité des utilisateurs, l'apparition et la disparition de dispositifs peuvent changer l'environnement.

Le caractère *distribué* des systèmes ambiants ne permet pas l'utilisation d'un processus de contrôle centralisé. À l'inverse, chaque dispositif robotique doit être doté de son propre contrôleur. La conception *ad hoc* d'un contrôleur pour un dispositif robotique dans un système ambiant est une tâche complexe qui nécessite une forte expertise et implique des coûts exponentiels de développement et de maintenance. En effet, elle nécessite de lister *a priori* tous les cas d'utilisation du dispositif robotique ainsi que ses différentes interactions. La tâche est d'autant plus complexe si l'on prend en compte le caractère spécifique, multiple et changeant des besoins des utilisateurs. Doter ces dispositifs de la capacité d'apprendre à s'adapter aux besoins de ses utilisateurs est une tâche particulièrement difficile (Hagras, 2004) qui présente un intérêt certain dans la réduction des coûts de développement et de maintenance. Dans cet article, nous proposons une approche nommée « *eXtreme Sensitive robotic*» (Verstaevel, 2015). L'*XS Robotic* considère chaque fonctionnalité du robot comme une entité autonome. Un robot est donc un agrégat de plusieurs *XS Function*, globalement une pour chaque capteur et effecteur. Une XS Function est soit sensitive (capteur température, ultrason,…) soit active (un moteur, une LED,…). Chaque *XS Function* est conçue comme *non finale*, c'est-à-dire que la fonctionnalité doit pouvoir s'adapter. En exploitant ses capacités d'auto-observation, le dispositif robotique observe des variations dans son environnement et déclenche les mécanismes d'adaptation adéquats. Avec cette approche, la production d'un comportement complexe par un ensemble de dispositifs émerge de l'interaction entre les différentes *XS Function* et l'environnement. Il n'y a donc pas de différence de conception entre un robot composé de plusieurs capteurs et effecteurs, une application multi-robot composée de plusieurs robots en interaction, ou un système ambiant. Tous ces systèmes sont composés de fonctionnalités distribuées au sein de différents dispositifs robotiques qui doivent s'adapter pour satisfaire leurs utilisateurs.

Pour être naturelle, cette adaptation doit reposer sur un procédé qui ne requiert pas d'expertise particulière pour l'utilisateur. De plus, elle nécessite à la fois d'être **générique**, pour être applicable à tout type de dispositif et pour tout utilisateur, ainsi que de respecter la propriété d'**ouverture** pour gérer l'apparition et la disparition de dispositifs. Le respect de la généricité et de l'ouverture requiert l'utilisation de techniques d'apprentissage *agnostiques* (Kearns, 1994) qui ne font que peu d'hypothèses sur leur environnement. Pour respecter ces propriétés, nous proposons l'utilisation de l'apprentissage par démonstration, un paradigme permettant l'apprentissage dynamique de nouveaux comportements.

2 Apprentissage par démonstration

L'apprentissage par démonstration, aussi appelé « apprentissage par imitation », est un paradigme principalement étudié en robotique qui permet à un système d'apprendre dynamiquement de nouveaux comportements (Argall, 2009). L'idée principale est qu'un contrôleur adéquat peut être appris de l'observation de l'activité d'une entité externe (humaine ou virtuelle) nommée *tuteur*. Le tuteur interagit avec le système au travers d'un processus de *démonstrations*, qui consiste en la réalisation d'une succession d'actions dans des situations particulières. Le système doit alors apprendre une fonction corrélant l'observation de son environnement avec les actions réalisées par le tuteur. L'avantage principal de cette technique est qu'elle ne nécessite aucune programmation explicite et aucune expertise sur le système de la part du tuteur. Des travaux

récents proposent une étude du domaine de l'apprentissage par démonstration (Argall, 2009) (Billard, 2008) et (Billing, 2010) fournit une formalisation complète. L'apprentissage par démonstration est un problème d'imitation où une entité doit produire un comportement similaire à une autre entité. Un *tuteur* évoluant dans un monde Ω peut réaliser un ensemble d'actions *A*. Le tuteur suit une politique $\Pi : \Omega \rightarrow A$ qui associe à chaque situation du monde une action. Il est supposé que cette politique est optimale pour satisfaire l'utilisateur. L'apprenant (aussi nommé *imitateur*) possède un ensemble d'observations *O* (nommé *espace d'observations*) sur le monde Ω et suit une politique $\Pi' : O \rightarrow A$ telle *que* $\Pi' \equiv \Pi$. L'apprenant doit donc trouver des corrélations entre la réalisation d'une action par son tuteur et des variations dans l'observation qu'il fait du monde. La prochaine section présente Alex, un système multi-agent pour l'apprentissage par démonstration conçu à partir de la vision *XS Robotic*.

L'application de ce paradigme aux systèmes ambiants repose sur la richesse des interactions avec leurs utilisateurs. L'utilisateur peut être vu comme tuteur du système et chacune de ses actions sur un dispositif comme une démonstration du comportement désiré. En effet, si l'utilisateur nécessite d'intervenir sur une fonctionnalité, c'est que le service fourni par cette fonctionnalité ne le satisfait plus. Le dispositif robotique contrôlant la fonctionnalité concernée peut alors exploiter cette information pour s'adapter. Le rôle d'un dispositif robotique est alors de corréler l'activité de l'utilisateur à l'observation qu'il fait de son environnement.

3 Alex: Adaptive Learner by EXperiments

L'utilisation du paradigme multi-agent (Ferber, 1999) dans l'XS Robotic apparaît naturelle pour sa capacité à faire face à la complexité et à apprendre en interaction avec son environnement (Panait, 2005). (Kaminka, 2012) montre l'apport des systèmes multi-agents pour la robotique. Plusieurs approches proposent dès le début des années 2000 l'utilisation de systèmes multi-agents pour permettre l'organisation d'un collectif de robots (Collinot, 1999) (Parker, 2000) (Picard, 2005) et encore de nos jours ce paradigme est utilisé pour des tâches complexes telles que l'exploration collective (Souliman, 2014) ou l'intervention en situation de crise (Lacouture, 2012). Ces approches considèrent souvent le robot comme un simple agent. Dans cet article, nous soutenons que la distribution du contrôle au sein de chaque fonctionnalité du robot permet une plus grande adaptation de celui-ci. Un robot devient alors un système multi-agent, composé d'un ensemble de fonctionnalités autonomes. Nous présentons Alex, un système multi-agent pour l'apprentissage par démonstration conçu à partir de la vision *XS Robotic*.

Alex est un système multi-agent construit pour apprendre à contrôler une fonctionnalité à partir de démonstrations réalisées par un tuteur. Sa conception est basée sur l'approche des systèmes multi-agents adaptatifs (AMAS) (Gleizes, 2012) qui adresse la problématique des systèmes complexes par une approche « bottom-up » où le concept de coopération agit comme moteur de l'auto-organisation. Le théorème de l'adéquation fonctionnelle (Capera, 2003) stipule que « pour tout système fonctionnellement adéquat, il existe au moins un système à milieu interne coopératif qui réalise la même fonction dans le même environnement ». Le rôle d'un AMAS est alors d'automatiquement détecter et réparer des situations de non-coopération en s'auto-organisant pour atteindre un état fonctionnellement adéquat. La conception d'Alex suit la méthodologie ADELFE2.0 (Bonjean, 2014) qui guide le concepteur d'un système adaptatif.

3.1 Context-Learning

La partie 2 illustre l'impossibilité de lister *a priori* toutes les situations qu'un robot peut rencontrer. Cette incapacité à spécifier

complètement le comportement du dispositif rend obligatoire son adaptation aux différents contextes d'utilisations (Makkonen, 2009). Le terme « contexte » n'a pas de définition communément admise, aussi fait-il référence dans cet article à toute information externe à l'activité d'une entité qui affecte son activité. Cet ensemble d'informations décrit l'environnement tel que le perçoit l'entité (Guivarch, 2014). Un système capable d'exploiter des informations de son contexte est alors nommé « context-aware ». Alex est conçu pour interagir continuellement avec son environnement et apprendre dynamiquement ses différents contextes. Ainsi, il construit une politique Π qui associe à chaque contexte l'action adéquate à réaliser. Pour ce faire, il dispose de capacités *d'auto-observation* pour construire dynamiquement des corrélations entre la réalisation d'une action et l'effet de cette action sur l'environnement. Une instance d'Alex est composée d'un ensemble non fini d'agents contextes. Chaque agent contexte est porteur d'une corrélation associant à un état de l'environnement une action. Cet ensemble, vide au départ, est dynamiquement peuplé au fur et à mesure qu'Alex détecte de nouvelles situations. Les agents contextes s'auto-organisent (voire disparaissent) pour effectuer un contrôle sur une fonctionnalité.

Les agents contextes sont décrits en décomposant leur comportement en deux types. Le comportement *nominal* de l'agent correspond à son comportement lorsque le dispositif robotique se trouve dans un état fonctionnellement adéquat qui ne nécessite pas d'adaptation. Le comportement *coopératif* est une subsomption du comportement nominal qui s'exécute lorsqu'une situation non coopérative est détectée. Il conduit alors à l'auto-organisation de l'entité en interaction avec les autres entités du système afin de retourner dans un état fonctionnellement adéquat.

3.2.1 Comportement nominal

Un ***agent contexte*** associe une description du contexte noté V (voir section 3.2.2) à une action unique dont il ne connait pas la sémantique. Ces actions peuvent être interprétées par un observateur humain comme « Avance », « Tourne à droite » ou « vitesse à 42% ». Un agent contexte reçoit un ensemble O de signaux de son environnement qu'il utilise pour caractériser le contexte courant du système. Quand l'observation qu'il fait de l'environnement correspond à sa description V, l'agent contexte déclenche son action $a \in A$ ou décide de ne rien faire (θ). Le comportement nominal N d'un agent contexte se résume à la fonction $N : O \times V \rightarrow a||\theta$. Le comportement Π du dispositif est donc résultant de l'interaction entre les agents contextes.

3.2.2 Description du contexte

Pour déclencher son action, l'agent contexte a besoin de comparer la situation courante avec sa représentation V. Cette représentation du contexte est réalisée grâce à des plages de valeurs (nommées ***plages de validité***). Un agent contexte reçoit un ensemble d'observations O de son environnement. Chaque $o \in O$ est une valeur continue incluse dans un espace de définition $o \in [o_{min}, o_{max}]$. Par exemple, une observation émanant d'un capteur de température $temp \in [-20,40]$ où -20 et 40 sont les températures minimales et maximales du capteur. Ces valeurs *min* et *max* ne sont pas connues *a priori* par Alex.

Une plage de validité v est associée à chaque observation telle que v_o correspond à la plage de validité associée à l'observation o. L'ensemble des plages de validité V forme le ***domaine de validité***. Une plage de validité est composée de deux valeurs v_{min} et v_{max} telle que $[v_{min}, v_{max}] \subseteq [o_{min}, o_{max}]$. Le rôle d'une plage de validité est d'adapter dynamiquement la valeur des deux bornes v_{min} et v_{max}. Une plage de validité est ***valide*** si et seulement si $o \in [v_{min}, v_{max}]$, c'est-à-dire si la valeur courante de l'observation est comprise dans les bornes de la plage de validité. Le domaine de validité sera *valide* si toutes ses plages de validité sont *valides*. Ainsi, un agent contexte dont le

domaine de validité est *valide* déclenche son action, considérant que le contexte courant du système correspond à sa représentation. La gestion des plages de validité se fait grâce à des trackers adaptatifs nommés Adaptive Value Trackers (AVT) (Lemouzy, 2011). Cet outil permet de trouver automatiquement une valeur à partir de feedbacks successifs. Il est à noter qu'avec cette représentation, aucune sémantique n'est associée au signal et que seuls des tests d'inclusion ensembliste sont réalisés.

3.2.3 Comportement coopératif

À chaque cycle, le dispositif robotique ne peut réaliser qu'une unique action dans un seul état à la fois. Les agents contextes *valides* doivent alors coopérer pour déterminer si leur action est la plus appropriée dans le contexte courant. Pour être dans un état fonctionnellement adéquat, deux agents contextes *valides* ne devraient pas proposer d'actions antagonistes (par exemple, deux contextes proposant chacun d'allumer et d'éteindre une lumière). Cependant, en ne conservant que le comportement nominal, des situations de ***non coopération*** doivent être résolues pour amener le système dans un état fonctionnellement adéquat. On distingue trois situations de non-coopération. La première (a) se produit lorsqu'au moins deux agents contextes proposent de réaliser une action (*concurrence*). La seconde (b) se produit si l'action proposée par un agent contexte est en contradiction avec l'action réalisée par le tuteur (*conflit*). La dernière (c) se produit lorsqu'aucun agent contexte ne propose d'action (*inutilité*). Pour résoudre ces situations, les agents contextes suivent un comportement coopératif.

3.2.3.1 Confiance

Un agent contexte détermine un niveau de *confiance* qui représente la pertinence de la réalisation de son action. Cette confiance permet aux agents contextes de coopérer. Si un agent contexte se trouvant *valide* veut réaliser une action alors que sa confiance est inférieure à celle d'un autre agent contexte *valide*, il doit modifier son domaine de validité en ajustant les bornes de ses plages de validité afin d'exclure la situation courante. Ainsi, la prochaine fois que la même situation se reproduira, cet agent contexte ne proposera pas son action. La confiance *c* d'un agent contexte est régie par la fonction suivante :

$$c_{t+1} = c_t * (1 - \alpha) + F_t * \alpha.$$

c_{t+1} est la confiance de l'agent contexte à l'instant t+1. $F_t \in 0|1$ est la valeur de feedback et $\alpha \in [0,1]$ est un paramètre fixé qui modélise l'importance du feeback. Chaque fois qu'un agent contexte a raison de proposer son action, $F_t = 1$. À l'inverse, chaque fois que cet agent fait une mauvaise proposition, $F_t = 0$. Dans notre implémentation, α est empiriquement fixé à 0,8 et la valeur de confiance est initialisée à 0,5. La confiance des agents contextes permet de lever l'ambiguïté soulevée par la situation de non coopération (a). Plus l'agent contexte fait de bonnes propositions, plus sa confiance est élevée. À l'inverse, plus l'agent a fait de mauvaises propositions, plus sa confiance est faible. La valeur minimale de confiance est fixée à 0,1, ce qui signifie que tout contexte avec une confiance inférieure à 0,1 s'autodétruira.

3.2.3.2 Adéquation avec le tuteur

Lorsque le tuteur agit sur un dispositif, son action est perçue par les agents contextes. Si un agent contexte est *valide* dans la situation courante, mais propose une action différente de celle de l'utilisateur, il va alors ajuster les bornes de ses plages de validité afin d'exclure la situation courante en ajustant les bornes v_{min} et v_{max}. Si l'agent contexte perçoit une donnée de l'environnement qui n'est pas encore incluse dans ses plages de validité (suite par exemple à l'ajout d'un capteur dans l'environnement), il ajoute une nouvelle plage de valeur associée à cette observation. À l'inverse, si l'agent contexte propose une action en adéquation avec le tuteur, il renforce sa confiance. Dans tous les cas, quand le tuteur

agit, son action subsume celle des agents contextes. Ces mécanismes permettent la résolution de la situation de coopération (b). Le processus d'adaptation des bornes des plages de validité peut conduire à une situation où $v_{max} < v_{min}$. Si une telle situation se produit, la plage de valeur concernée est plus valide. L'agent contexte s'autodétruit.

3.2.3.3 Anticipation et création d'agent contexte

L'état fonctionnellement adéquat du dispositif robotique nécessite qu'il existe un unique agent contexte dont l'action est activée par cycle de décision. Cependant, le dispositif peut se retrouver dans des situations entièrement nouvelles où aucun agent contexte n'est *valide* et par conséquent être dans une situation d'incompétence. Deux mécanismes peuvent résoudre cette situation : étendre les bornes de validité de l'agent contexte existant ou créer un nouvel agent contexte pour représenter cette nouvelle situation.

Pour anticiper une situation d'incompétence, le concept de ***validable*** est introduit. Une plage de validité v_o est validable si et seulement si $o \notin [v_{min}, v_{max}]$ et $o \in [v_{min} - \Delta_{min}, v_{max} + \Delta_{max}]$. Δ_{min} et Δ_{max} sont deux valeurs (une pour chaque borne) gérées dynamiquement par les AVTs pour contrôler l'évolution de chaque borne. Δ peut être interprété comme le prochain incrément de la borne. Avec le concept de *validable*, les agents contextes peuvent proposer leur action dans des situations suffisamment leur plage de validité. Si l'action proposée par l'agent est la plus adéquate, il adapte alors ses bornes pour inclure la situation courante. À l'inverse, si la proposition n'était pas pertinente, les AVT diminueront leur Δ pour ne pas se proposer la prochaine fois.

Lorsqu'aucun agent contexte n'est *valide* ou *validable*, le dispositif robotique est face à une situation d'incompétence, il ne peut décider de l'action à réaliser. L'agent contexte *valide* à l'étape précédente (mais qui n'est plus *valide* dans la situation courante) peut alors créer un nouvel agent contexte destiné à le remplacer en lui associant l'action effectuée par le tuteur. L'agent contexte ainsi créé initialise ses plages de validité pour représenter la situation courante.

Si le tuteur n'a pas effectué d'action, deux mécanismes peuvent se produire, selon l'autonomie désirée par le tuteur. Avec une approche *collaborative*, le dispositif peut interroger le tuteur sur l'action à réaliser. Avec une approche *autonome*, le dispositif maintient la dernière action qu'il a su réaliser, considérant l'inaction du tuteur comme une demande de maintien de l'action courante. Ce mode peut être choisi à l'initialisation du dispositif et être changé dynamiquement.

Ces deux mécanismes, le concept de validabilité et la création d'agents contexte, permettent de résoudre la situation de non coopération (c). Dans la suite de cet article, nous proposons d'illustrer le comportement d'Alex sur une application concrète.

4 Expérimentations

Un robot muni de deux roues sans aucun capteur est immergé dans une arène de 2mx2m composée de deux artefacts bleu et vert. Une caméra située 2m au-dessus du centre de l'arène observe la scène et peut analyser les pixels pour capturer la position des artefacts bleu et vert par rapport à la position du robot (Figure 1). La caméra peut produire quatre observations sur la scène correspondant à la distance, et l'angle pour chaque artefact. Pour montrer la capacité à gérer l'apparition et la disparition de signaux, la camera envoie les

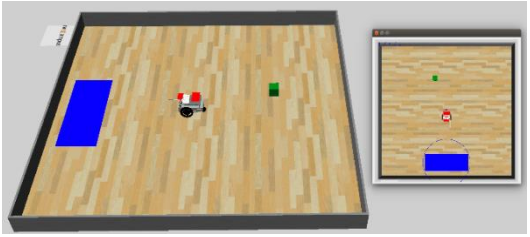


Figure 1- L'expérimentation sous Webots™.

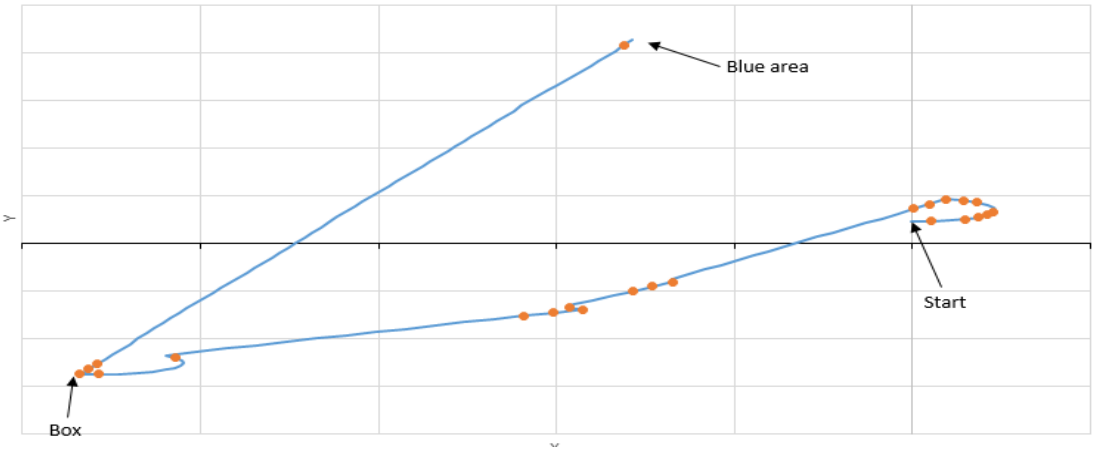


Figure 2 – Trajectoire du robot et zones de création d'agents contextes.

informations de positions au robot uniquement si les artefacts se situent face au robot. À chaque étape, la caméra peut alors produire zéro, deux ou quatre observations.

Le tuteur humain peut contrôler le robot au travers d'une manette. Chaque joystick contrôle la vitesse d'une roue (le joystick gauche contrôle la roue gauche et réciproquement). Cette valeur, comprise entre [-100,100], correspond au pourcentage de la vitesse maximale à appliquer ainsi qu'au sens de rotation. Le but de l'expérimentation est de montrer la capacité d'Alex d'apprendre des comportements complexes à partir de l'observation des actions du tuteur. Pour ce faire, le tuteur peut montrer un ensemble de comportements dans l'arène qu'Alex doit imiter. L'expérience est réalisée à la fois sur un robot virtuel et sur un robot réel.

En accord avec la vision *XS Robotic*, une instance d'Alex est associée à chaque roue leur permettant d'agir de manière autonome. Il n'existe aucune communication explicite entre les deux roues. Le rôle d'une instance d'Alex est alors d'apprendre à contrôler une roue en corrélant l'activité du tuteur aux perceptions qu'il reçoit de la caméra. À chaque étape de décision, Alex peut recevoir les observations de la caméra ainsi que l'action réalisée par le tuteur. Un cycle de décision intervient tous les 250 ms. La même implémentation d'Alex est utilisée à la fois en simulation et dans l'application réelle. OpenCV est utilisé sur les deux applications pour effectuer le traitement d'images. L'expérience est simulée sous Webots™. L'expérience réelle a été réalisée grâce à un robot Boe-Bot basé sur une plateforme Arduino et l'utilisation d'un protocole de communication Xbee. Une des activités que le tuteur réalise dans l'arène est une tâche de collecte d'artefacts. Le robot est muni de deux branches solides permettant la capture de petits objets. Ces branches sont fixes et ne comportent aucun capteur. Le but de l'activité est alors de capturer les artefacts verts et de les transporter vers l'artefact bleu. À chaque fois qu'un artefact est déposé dans la zone bleue, il est soit, dans la simulation, déplacé aléatoirement dans l'arène, soit déplacé par le tuteur dans l'application réelle. Les résultats présentés ont été obtenus à partir des simulations et constatés sur l'expérimentation réelle. L'expérimentation se propose d'illustrer le processus d'adaptation durant les phases de démonstrations pour l'apprentissage et les phases de fonctionnement en autonomie du robot.

4.1 L'activité du tuteur permet la création d'agent contextes

La figure 2 montre la trajectoire suivie par le robot pendant une phase de démonstration du comportement de collecte. Dans cette démonstration, le tuteur prend le contrôle du robot et le conduit afin de capturer l'artefact vert puis le ramener vers l'artefact bleu. Chaque point correspond à la création d'un

agent contexte par l'une des deux instances d'Alex. En observant la figure, on constate que les agents contextes ne sont créés qu'au cours de changements de direction commandés par le tuteur. À l'inverse, quand le tuteur maintient sa direction, aucun agent contexte n'est créé. Alex observe chaque action du tuteur et détermine si celle-ci appartient à un contexte existant ou à un nouveau contexte. La création de nouveaux agents contextes résulte donc de l'observation de l'activité du tuteur.

4.2 L'activité du tuteur permet l'adaptation des agents contextes

La figure 3 montre la structure d'un agent contexte particulier à sa création (3.a) et à la fin d'une phase démonstration (3.b). Chaque ligne correspond à la structure d'une plage de validité associée à une observation. Les plages les plus claires correspondent à la zone *valide*, les plus foncées à la zone *validable*. Le curseur blanc représente la valeur courante du signal. À sa création (3.a), l'agent contexte initialise ses plages de validité pour représenter la situation courante et est associé à l'action courante du tuteur. Tant que l'action proposée par cet agent contexte est en adéquation avec celle du tuteur, l'agent contexte adapte ses plages de validité. Les plages de validité à la fin de la démonstration (3.b) sont la résultante d'un processus d'auto-organisation entre les agents contextes. En observant plus précisément l'évolution des plages de validité, on se rend compte que les plages étiquetées *BlueA* et *BlueD* sont plus larges que celles étiquetées *GreenA* et *GreenD*. Cela signifie que ce contexte particulier est bien plus sensible aux variations des données liées à l'artefact vert qu'aux données liées à l'artefact bleu. Ce contexte particulier est valide lorsque l'artefact vert est proche de l'avant du robot et que le robot est face à la zone bleue. Il est donc impliqué dans la phase de l'activité où le robot ramène l'artefact capturé dans la zone bleue.

4. 3 De la démonstration du tuteur naît l'autonomie

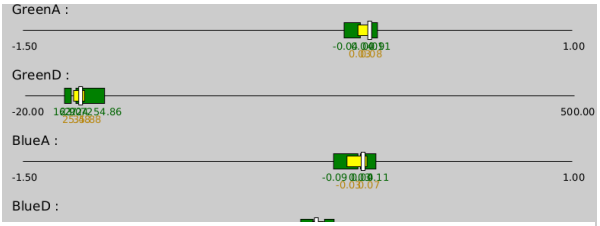


Figure 3.a – Domaine de validité d'un agent contexte à sa création

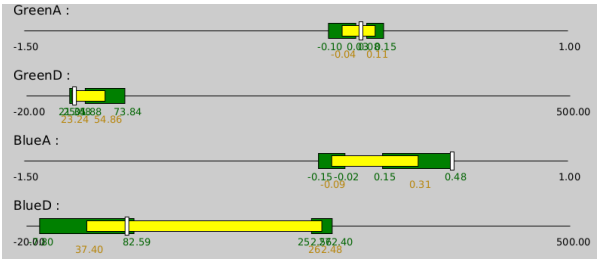


Figure 3.b – Domaine de validité du même agent contexte à la fin de la démonstration

Pour observer la capacité d'Alex à imiter les performances du tuteur, celui-ci réalise une démonstration de cinq minutes au cours de laquelle un certain nombre d'artefacts (nommés *boîtes*) sont collectés. Le nombre de boîtes collectées par le tuteur sert alors de métrique pour observer la capacité d'Alex à proposer des performances semblables à celle du tuteur. Au bout de cinq minutes, les dispositifs agissent de manière autonome, c'est-à-dire que le tuteur n'intervient plus sur leur fonctionnalité. Le nombre de boîtes collectées est calculé toutes les cinq minutes. La figure 4 montre le résultat obtenu au cours d'une de ces expérimentations. Au pire cas, les dispositifs autonomes réalisent la tâche aussi bien que le tuteur : *12* boîtes sont collectées. En moyenne, le système parvient à collecter *14* boîtes. Deux facteurs influencent ce résultat. Le premier vient du caractère aléatoire du déplacement de la boîte, la distance à parcourir pour collecter une boîte variant. L'autre facteur réside dans le fait que le tuteur suit un processus décisionnel plus lent que le système et peut aussi se contredire. Ce phénomène est observable sur la figure 2. À la moitié du chemin entre la position de départ et l'artefact vert, on peut observer un changement de trajectoire. Ce changement est une erreur de

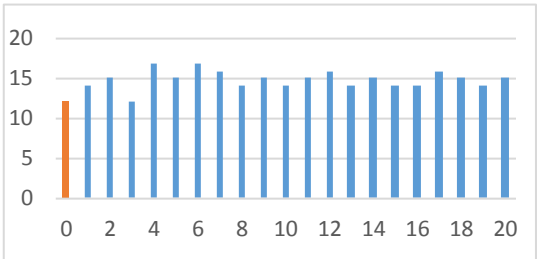


Figure 4 – Nombre de boîtes collectées toutes les 5 minutes. Le pas 0 correspond au score de référence.

télé-opération du tuteur. Les agents contextes créés pour représenter cette situation possèdent une valeur de confiance faible car leur action n'est jamais reproduite par le tuteur. Ainsi, le comportement appris par Alex ne retenant que les propositions d'action des agents les plus confiants, les agents contextes issus de ces erreurs ne seront jamais sélectionnés. Ceci permet à Alex de réaliser la tâche de manière plus efficace.

4. 4 Synthèse

Cette expérimentation permet d'illustrer le mécanisme d'auto-organisation des agents contextes au niveau d'un dispositif robotique qui conduit à l'observation d'un comportement complexe au niveau du robot. Le processus d'auto-organisation est la résultante de l'interaction entre les instances d'Alex et le tuteur. Les expérimentations réalisées à la fois en simulation et sur un robot réel ont permis de mettre en avant des propriétés qui différencient Alex des approches traditionnelles:

Généricité : le processus d'apprentissage est indépendant de la tâche à réaliser et ne nécessite pas de sémantique sur les signaux.

Ouverture : de nouveaux signaux peuvent être dynamiquement intégrés dans le processus d'apprentissage.

Distribution : l'auto-observation permet à chaque dispositif d'être autonome dans son apprentissage tout en restant collaboratif.

Temps réel : l'auto-organisation est réalisée sans interrompre l'activité du système en collaboration avec le tuteur.

5 Conclusion

Le travail présenté traite de l'apprentissage par démonstration en milieu ambiant. Il présente Alex, un système multi-agent conçu pour apprendre un comportement à partir des démonstrations réalisées par un tuteur. Les expériences réalisées montrent la capacité d'Alex à apprendre de l'interaction avec son tuteur. Cela signifie concrètement que la simple démonstration d'une activité permet à chaque système multi-agent associé à chacun des effecteurs du système de comprendre de manière autonome quelles sont les données pertinentes pour imiter collectivement le comportement de son tuteur sans aucune forme de contrôle centralisé.

L'expérimentation montre que deux instances d'Alex (ici deux roues) peuvent coopérer sans communication directe. Le fait que chacune des instances d'Alex corrèle son activité avec les variations de son environnement suffit pour que les deux SMA Alex agissent de manière coopérative. Cet article présente une preuve de concept illustrant qu'un système multi-agent capable d'apprendre à partir de l'auto-observation de son contexte (*context-learning*) est une réponse adéquate à la complexité inhérente aux systèmes ambiants. Cette approche présente un intérêt particulier pour des applications dont la spécification ne peut être complètement définie *a priori*.

Forts de ces résultats, les travaux en cours portent sur l'application de cette technologie dans un contexte industriel. Dans cette perspective et en partenariat avec plusieurs industriels, nous utilisons Alex en robotique collaborative pour le contrôle de bras robotisés dans un contexte industriel, mais aussi pour des applications grand-public.

Enfin, nous désirons à la fois formaliser notre approche pour l'inclure dans une théorie de l'apprentissage multi-agent contextuelle et comparer les performances de notre approche avec d'autres techniques d'apprentissage à la

pointe de l'état de l'art.

Remerciements

Les auteurs souhaitent remercier Sogeti High Tech et l'ANRT pour le soutien apporté à ces travaux.

Références

Argall, Brenna D., Chernova, Sonia, Veloso, Manuela, et al. A survey of robot learning from demonstration. *Robotics and autonomous systems*, vol. 57, no 5, p. 469-483, 2009.

Billard, Aude, et al. "Robot programming by demonstration." *Springer handbook of robotics*, p. 1371-1394, 2008.

Billing, Erik A. et Hellström, Thomas. A formalism for learning from demonstration. *Paladyn*, vol. 1, no 1, p. 1-13, 2010.

Bonjean, Noelie, et al. "ADELFE 2.0." *Handbook on Agent-Oriented Design Processes*. Springer Berlin Heidelberg, p. 19-63, 2014.

Capera, Davy, et al. "The AMAS theory for complex problem solving based on self-organizing cooperative agents." *Proceedings of Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises* IEEE, 2003.

Collinot, A., Drogoul, A., & Benhamou, P. Agent oriented design of a soccer robot team. *In Proceedings of the Second International Conference on Multi-Agent Systems*, pp. 41-47, 1999.

Chung, Joohyun. Ambient robotics for meaningful life of the elderly. *Advanced Construction and Building Technology for Society*, 2014.

Ferber, J. *Multi-agent systems: an introduction to distributed artificial intelligence*, Vol. 1, 1999.

Gleizes, Marie-Pierre. Self-adaptive complex systems. *Multi-Agent Systems*. Springer Berlin Heidelberg, p 114-128, 2012.

Guivarch, Valérian, et al. Self-Adaptation of a Learnt Behaviour by Detecting and by Managing User's Implicit Contradictions. *Proceedings of the 2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, vol. 3. IEEE Computer Society, 2014.

Hagras, H., Callaghan, V., Colley, M., Clarke, G., Pounds-Cornish, A., & Duman, H. Creating an ambient-intelligence environment using embedded agents. *Intelligent Systems*, IEEE, vol. 19, no 6, p.12-20, 2004.

Kaminka, G. A. Autonomous agents research in robotics: A report from the trenches. *AAAI Spring Symposium Series*, 2012.

Kearns, M. J., Schapire, R. E., & Sellie, L. M. Toward efficient agnostic learning. *Machine Learning*, vol. 17, no 2-3, p. 115-141, 1994.

Lacouture, Jérôme, et al. Rosace: Agent-based systems for dynamic task allocation in crisis management. *Advances on Practical Applications of Agents and Multi-Agent Systems*. Springer Berlin Heidelberg. p. 255-259, 2012.

Lemouzy, S., Camps, V., & Glize, P. Principles and properties of a mas learning algorithm: A comparison with standard learning algorithms applied to implicit feedback assessment. *Web Intelligence and Intelligent Agent Technology*. IEEE, vol. 2, p. 228-235, 2011.

Makkonen Jarmo, Avdouevski Ivan, Kerminen Riitta and Visa Ari. Context Awareness. *Human-Computer Interaction*, Inaki Maurtua (Ed.), ISBN: 978-953-307-022-3, InTech, DOI: 10.5772/7743, 2009.

Panait, L., & Luke, S. Cooperative multi-agent learning: The state of the art. *Autonomous Agents and Multi-Agent Systems*, vol. 11, no 3, 387-434, 2005.

Parker, Lynne E. Current state of the art in distributed autonomous mobile robotics. *Distributed Autonomous Robotic Systems 4*. Springer Japan, p. 3-12, 2000.

Picard, Gauthier, and Marie Pierre Gleizes. Cooperative self-organization to design robust and adaptive collectives. *ICINCO*. 2005.

Russell, S., Norvig, P., & Intelligence, A. A modern approach. *Artificial Intelligence*. Prentice-Hall, Egnlewood Cliffs, vol. 25, 1995

Souliman, A., Joukhadar, A., Alturbeh, H., & Whidborne, J. F. Intelligent Collision Avoidance for Multi Agent Mobile Robots. *In Intelligent Systems for Science and Information*. *Springer International Publishing*, p. 297-315, 2014

Verstaevel, N., Régis C., Guivarch, V., Gleizes, M.P., Robert, F. Extreme Sensitive Robotic: A Context-Aware Ubiquitous Learning. *Proceedings of the 2015 International Conference on Agents and Artificial Intelligence*, vol. 1, p. 242-248, 2015.

Une méthode incrémentale de conception dirigée par les tests pour la simulation multi-agent

Lancelot SIX[a,c]
lancelot.six@quiet-oceans.com

Julien SAUNIER[b]
julien.saunier@insa-rouen.fr

Zahia GUESSOUM[c]
zahia.guessoum@lip6.fr

Sio-Song IENG[a]
sio-song.ieng@ifsttar.fr

[a] IFSTTAR — 14-20 Boulevard Newton — Cité Descartes, Champs sur Marne — F-77447 Marne la Vallée Cedex 2

[b] INSA Rouen —Avenue de l'Université —76801 Saint-Étienne-du-Rouvray Cedex

[c] Lip6 — 4 place Jussieu — 75005 Paris

Résumé

L'approche multi-agent est par nature adaptée à une conception incrémentale des modèles. La modularité de l'approche permet la conception progressive des éléments du système cible, par l'ajout de nouvelles entités, de nouveaux modes d'interaction et d'organisation, ou l'inclusion de nouveaux comportements. Cependant, les méthodes de conception de logiciels usuelles ne sont en général pas applicables dans leur ensemble au développement des systèmes de simulation à base d'agents, la principale difficulté étant l'émergence de comportements collectifs. Dans cet article, nous proposons une méthode de conception dirigée par les tests adaptée aux systèmes multi-agents, inspirée à la fois des modèles en spirale et de la conception dirigée par les tests. L'ajout d'une fonction est spécifiée au niveau du système, tandis que l'analyse et la conception se fonde sur la vérification des propriétés individuelles. Nous illustrons cette méthode avec un cycle de conception d'un modèle de poids-lourd dans une simulation de trafic.

Mots-clés : *Conception incrémentale, Simulation de trafic, Modélisation*

Abstract

Multi-agent systems are well suited to incrementally develop models. Their modularity allow to develop the system by gradually adding new interaction or organization models, new entities, or new behaviors. However, usual software engineering methods cannot be fully used to develop multi-agent systems due to their lack in considering un-anticipated emergent phenomena. In this article, we propose a test-driven inspired incremental development method tailored for agent based simulation models. We illustrate the use of this method to improve a heavy vehicle model to be used in traffic simulations.

Keywords: *Incremental development, Traffic simulation, Modeling*

1 Introduction

La simulation multi-agent fournit un outil puissant pour la reproduction et l'étude de comportements humains sociaux. Elle a été appliquée avec succès aux systèmes complexes tels que l'économie, les réseaux sociaux ou le transport. Dans ces différentes applications, les agents possèdent des comportements complexes et difficiles à spécifier, puisque ceux-ci sont à la fois réactifs et cognitifs. De plus, les modèles à base d'agents sont généralement conçus pour étudier des propriétés macroscopiques émergentes, qui sont le résultat des interactions entre agents.

Dans cette perspective, créer un modèle d'agent est une tâche non triviale, car les sorties du modèle (les phénomènes émergents) ne sont pas directement spécifiées dans le modèle initial [6]. De plus, quand un modèle produit un phénomène émergent adéquat, il n'est souvent pas évident de savoir laquelle des hypothèses de modélisation en est responsable [11]. La simulation de trafic illustre ces difficultés. Le principal objectif du modélisateur se situe sur la qualité au niveau macroscopique, de façon à comprendre et prédire les flux et les optimiser. Depuis la fin des années 50, de nombreux modèles ont été proposés et validés de cette façon [3, 14, 15].

De façon à faciliter la conception de modèles à base d'agents, nous proposons de nous inspirer d'une approche à base de tests. Nous appliquons cette méthode à la conception d'un modèle de véhicules lourds, lesquels sont pour l'instant rarement pris en compte dans les simulations de trafic, malgré un impact fort sur

les flux de véhicules [4]. Le modèle IDM [25] est une illustration de ce problème : les camions sont seulement considérés comme des véhicules longs capables de faibles accélérations et décélérations. Cependant, d'autres facteurs différencient les poids-lourds des véhicules légers telle qu'une dynamique changeante en fonction notamment de la charge et de la déclivité, ce qui a un impact sur les stratégies de conduite (et notamment d'anticipation).

L'approche agent est particulièrement adaptée à une conception incrémentale des modèles. Un cycle de développement incrémental permet de bâtir la solution logicielle par étapes, en ajoutant une ou des fonctionnalités (incréments) à chacune de ces étapes. Une partie des modèles multi-agents tire avantage de cette propriété et sont construits sur la base d'un modèle existant, en se concentrant sur le développement d'un aspect qui intéresse le modélisateur. Un exemple est le modèle HDM [17], qui est une sur-couche sur IDM [25] se concentrant sur les aspects stratégiques de la conduite. Les bases offertes par le modèle d'origine (capacité de suivi de véhicule) ne sont pas remises en cause et le modèle est enrichi.

Cependant, les méthodes de conception de solutions logicielles usuelles ne sont en général pas applicables dans leur ensemble au développement des systèmes de simulation à base d'agents de par leur nature, puisque l'émergence des phénomènes macroscopiques ne peut être directement spécifiée. Dans la section suivante, nous précisons les motivations de notre approche de conception, puis décrivons des méthodes de conception usuelles. En section 3, nous proposons notre méthode de conception dirigée par les tests, et en illustrons une itération dans la section 4. Enfin, nous concluons et proposons quelques perspectives en section 5.

2 Motivation et état de l'art

Le premier et principal point de difficulté de l'utilisation de méthodes de génie logiciel pour la conception de systèmes multi-agents est lié au principe d'émergence qui est recherché par de tels systèmes. Cette approche postule que la modélisation des comportements individuels des composants du système est suffisante pour faire émerger le comportement du tout. Or, les processus de développement usuels sont dirigés par les buts (spécification du système dans son ensemble). Le développement incrémental est fait pour ajouter une fonctionnalité attendue au logiciel. Ces buts correspondent généralement à un besoin applicatif, par exemple produire un simulateur permettant de reproduire une ou des propriétés d'un objet d'étude et répondre à une question sur cet objet d'étude. Il y a donc une incompatibilité entre une approche où le développement est orienté par le résultat et un développement où le résultat n'est pas nécessairement anticipé par le modélisateur.

Par ailleurs, Galan et al. [11] soulignent que l'interaction de nombreux modèles - qu'ils concernent les agents, l'environnement, les communications ou encore la plateforme elle-même - entraîne un niveau de complexité accru. En effet, afin d'aboutir à une implémentation fonctionnelle de chacun de ces composants, de nombreux choix doivent être pris *a priori*. Par exemple, une discrétisation de l'environnement spatial peut être nécessaire à la réalisation de la plateforme, et ce choix impactera nécessairement l'implémentation des agents évoluant dans ce monde, ainsi que les résultats de simulations. Cette difficulté d'exprimer l'ensemble des décisions nécessaires à la réalisation et éventuellement à la reproduction d'un système à base d'agents est la principale motivation des travaux de Chevrier et Fatès [5]. Ils proposent une méthode de formalisation de systèmes à base d'agents visant à ne pas permettre la présence de choix implicites, qui empêcheraient l'appropriation et l'analyse du modèle par une tierce partie (utilisateur du modèle, relecteur apportant un regard critique sur la proposition…).

Enfin, le processus de modélisation implique différent acteurs. Trois acteurs jouent des rôles importants selon [8] : le *thématicien* est l'acteur apportant une expertise sur le système modélisé et définissant les besoins applicatifs. Ceux-ci sont généralement exprimés de manière non formelle et peuvent donc contenir des ambiguïtés et des imprécisions.Le *modélisateur* est l'acteur qui, à partir du modèle non formel fourni par l'expert thématique, produit une version formelle du modèle, à l'aide d'un langage de modélisation. Enfin, l'*implémenteur* est l'acteur qui produit une version exécutable du modèle produit par le modélisateur. L'interaction entre les différents acteurs, chacun disposant de méthodes et connaissances propres, peut introduire un grand nombre d'approximations. Le mode d'interaction entre ces différents acteurs n'est par ailleurs pas guidé par les méthodes de développement dont nous avons connaissance.

Dans l'objectif d'aboutir à un modèle de dé-

veloppement incrémental [1] tenant compte des spécificités ainsi que des difficultés intrinsèquement liées aux systèmes de simulation à base d'agents, nous proposons une méthode dirigée par la vérification. Nous présentons donc dans un premier temps deux approches de développement répandues dont nous nous inspirons qui sont 1) le modèle en spirale et 2) le développement dirigé par les tests.

Le développement en spirale. La première approche que nous abordons est le développement en spirale proposé par [2]. Alors que dans la plupart des méthodes antérieures le développement était centré sur le code fourni, cette approche est focalisée sur le risque. Pour chaque itération, l'évaluation des changements envisagés, de leurs alternatives, et des risques qui leurs sont associés est plus importante que l'amélioration elle-même.

Pour ce modèle de développement, chaque itération se décompose en quatre phases (ou cadrants) : *(1) Description des objectifs*- la première étape consiste à évaluer les objectifs de l'itération et à s'interroger sur leur pertinence. De manière similaire à ce qui est proposé par les méthodes agiles, il s'agit de pouvoir s'adapter à des changements pouvant survenir dans les besoins. *(2) Analyse des risques*- il s'agit du point central du cycle en spirale. Chaque itération comporte une phase d'évaluation des risques associés aux objectifs (risque de dépassement de délais, risque de blocage technique, *etc.*). L'objectif est de ne continuer l'itération que si les risques sont suffisamment contrôlés. Il s'agit donc d'un point de décision important permettant de garder la maîtrise des développements effectués soit en préférant une alternative moins risquée, soit en abandonnant l'itération courante. *(3) Développement*- il s'agit de l'activité usuelle de production du logiciel ainsi que des différentes phases de tests. *(4) Préparation de la phase suivante*- enfin, la dernière étape de l'itération consiste à planifier la suite des itérations.

Contrairement aux cycles de développement classiques qui sont centrés soit sur la délivrance de documentations (comme c'est le cas pour le développement en cascade) soit sur la délivrance de codes (comme c'est le cas pour les méthodes agiles), le développement en spirale se concentre sur l'analyse des risques et de la pertinence des travaux à effectuer. Ainsi, ce modèle propose différents points de décision permettant d'orienter ou réorienter les évolutions à venir selon leur pertinence.

Développement dirigé par les tests. L'autre famille de méthodes dont nous nous inspirons est le développement dirigé par les tests, particulièrement utilisé dans les méthodes agiles [23]. Les méthodes agiles sont des méthodes de développement incrémentales. Elles se caractérisent par des cycles très courts ayant pour objectif de pouvoir réagir très rapidement à un changement du besoin et de minimiser le temps nécessaire à fournir une version exploitable du logiciel. Le développement d'une solution est décomposé en un ensemble de fonctionnalités, chacune d'elles faisant l'objet d'une itération. L'ordre des itérations dépend de la priorité accordée à chaque fonctionnalité ainsi qu'au coût de développement associé.

Une partie de ces méthodes (dont *«eXtreme Programming»* (XP)[1], pour ne citer qu'un exemple) se fonde sur un développement dirigé par les tests. Il s'agit d'écrire les tests auxquels sera soumis le code produit avant même que le code ne soit lui-même écrit. Une fois les tests écrits, l'objectif est de produire une implémentation qui permette de satisfaire l'ensemble des tests. Cette approche va à l'opposé de ce qui est généralement pratiqué, à savoir écrire un code et ensuite la série de tests permettant de s'assurer que le code est conforme aux spécifications.

L'écriture *a priori* des tests a plusieurs avantages. Tout d'abord, elle est une façon d'expliciter et de mettre à l'épreuve la spécification du module. Cela permet de se concentrer sur l'interface du code à produire. Cette première étape permet donc de détecter au plus tôt certaines erreurs de conception qui aboutiraient à une interface incomplète ou difficilement utilisable. De plus, une fois les tests écrits, l'objectif est de produire une implémentation telle que l'ensemble des tests soient satisfaits. Ceci incite donc à la concision et évite de produire des fonctionnalités non nécessaires.

L'ensemble des tests produits pour une itération peuvent être utilisés pour effectuer des tests de non régression qui seront utilisés dans deux cas : lors d'un éventuel remaniement du code (*refactoring*) afin d'en faciliter la maintenance dans le futur, ou lors du développement de nouvelles

1. L'approche incrémentale est à distinguer des approches itératives. Dans une approche incrémentale, chaque cycle de développement a pour objet l'ajout ou la modification d'une fonctionnalité alors que dans un processus itératif, une itération correspond au déroulement d'un cycle de développement comprenant analyse, conception, implantation et tests, sans pour autant nécessairement modifier le comportement du logiciel produit.

fonctionnalités afin de vérifier que les modifications apportées ne dégradent pas l'existant.

3 Amélioration de modèles d'agents dirigée par les tests : TIM4MAS

En s'inspirant des deux méthodes précédentes, nous présentons dans cette section notre modèle de développement de systèmes à base d'agents qui adapte ces deux modèles à une utilisation dans le cadre de systèmes multi-agents : *Test driven Incremental development Method for Multi Agent Systems* (TIM4MAS). Cette méthode est adaptée à l'amélioration de modèles existants, il est préférable de se référer aux méthodes de conception multi-agent classiques (voir section 5) pour une conception *ex-nihilo* de systèmes.

Le processus de développement dirigé par la vérification repose sur cinq phases :

- L'identification et la caractérisation d'une propriété du comportement qui doit être reproduite. Cette première phase consiste à identifier un aspect du comportement des agents à modéliser.
- L'évaluation de l'impact attendu de l'aspect du comportement sur le déroulement de la simulation.
- La spécification des propriétés de cette facette à l'aide d'un ensemble de tests.
- L'implémentation. Cette phase consiste à proposer une implémentation d'un modèle (ou un ensemble de modifications d'un modèle existant) telle qu'elle permette de reproduire les propriétés attendues (au niveau microscopique).
- La planification des phases suivantes, en fonction des résultats de validation (au niveau macroscopique).

La figure 1 montre une adaptation du cycle en spirale adapté au développement de modèles d'agents. Les trois premiers cadrants correspondent à la phase de spécification des besoins. Le rendu de cette première grande phase est la spécification des tests que l'implémentation devra par la suite être en mesure de passer. La phase suivante correspond au développement proprement dit. Enfin, la dernière correspond au point de planification de la suite du projet.

3.1 Identification d'une facette de comportement

La première étape du développement consiste à identifier une facette f du comportement

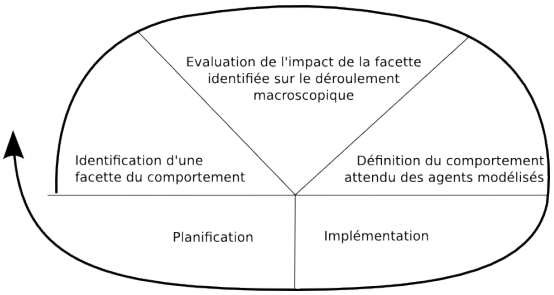


FIGURE 1 – Déroulement général d'une itération de développement.

Rôle	expert thématique	L'expert thématique identifie la facette sur laquelle l'itération sera centrée.
Entrées	$M_{SMA} = \{m_1, ...m_n\}$	modèle de SMA à l'itération courante, composé de d'un ensemble de modèles d'agents et d'un modèle d'environnement
Sorties	f	facette du comportement

d'agent(s) devant être améliorée. Une facette du comportement est un aspect de l'action d'un agent dans une situation donnée. Par exemple, la stratégie de décélération d'un véhicule à l'approche d'une zone de congestion ainsi que sa procédure de changement de voie sont des facettes du comportement d'un conducteur.

Une facette peut également être un mécanisme interactionnel entre plusieurs agents.

L'identification et la caractérisation de ces facettes est une tâche qui relève du domaine de l'expert thématique (selon les rôles décrits par [8]). Il est en charge d'identifier où il estime que des améliorations sont opportunes.

3.2 Évaluation de l'impact attendu

Une fois l'identification de la facette achevée, il faut évaluer si les modifications envisagées sont pertinentes du point de vue du système modélisé, et si elles peuvent potentiellement modifier son fonctionnement. Une modification n'ayant pas d'impact significatif sur le comportement du système ne représente pas un objectif prioritaire en terme de développement du modèle. De cette analyse résulte un point de décision pour savoir s'il faut développer le modèle ou plutôt identifier un autre aspect du comportement plus pertinent ou prioritaire du point de vue du système. L'analyse peut être menée grâce à la littérature, la connaissance experte du domaine, ou le développement d'outils spécifiques (par exemple

l'étude des phénomènes d'hystérésis dans [24]).

3.3 Spécification par les tests

Rôle	Expert thématique	L'export exprime, via un jeu de test, les propriétés du modèle qu'il souhaite obtenir.
Entrées	M_{SMA}, f	modèle de SMA, facette étudiée
Sorties	$\mathcal{T}_f$	jeu de tests

Une fois qu'une facette du comportement a été identifiée comme pertinente par l'expert, il convient de s'assurer que le modèle d'agent développé peut la reproduire de manière adéquate. La spécification — non-formelle — du comportement que doivent avoir les agents est également une tâche qui relève de l'expert thématique. Dans l'approche que nous proposons, plutôt que de décrire un modèle de comportement, l'expert définit les propriétés de celui-ci.

La description des propriétés que le comportement des agents doit respecter est assimilée à la description d'un ensemble de n tests $\mathcal{T}_f = \{t_1, t_2, ..., t_n\}$ liés à la facette f. Ces tests forment l'interface entre d'une part l'expert thématique qui exprime ce qu'il attend du modèle à produire et d'autre part le modélisateur et l'implémenteur qui doivent fournir une implémentation fonctionnelle répondant au besoin et définie au niveau microscopique. Il est pertinent de fournir plusieurs tests dépendant des différents paramètres à prendre en compte comme par exemple la puissance du véhicule considéré. De manière analogue à des tests logiciels «classiques» qui doivent couvrir un ensemble large de cas d'utilisation d'un programme, les propriétés spécifiées doivent être aussi exhaustives que possible afin de pouvoir être assimilées à une spécification complète.

3.4 Modélisation / Implémentation

Une fois les propriétés du modèle décrites par l'expert thématique, la tâche du modélisateur ainsi que de l'implémenteur est de produire une version exécutable d'un modèle ayant les propriétés désirées. De cette façon, la problématique d'ajout d'approximations qui arrive en passant d'une description originale d'un modèle faite par l'expert à son équivalent exécutable par un ordinateur est évitée. Le modèle n'étant jamais décrit initialement, il ne peut être question d'approximation. Si les propriétés sont reproduites, le modèle correspond à l'attente. En revanche, si les choix de modélisation faits par les intervenants mènent à un modèle exécutable ne remplissant pas les propriétés désirées alors l'implémentation est incorrecte [2]. On cherche donc lors de l'implémentation à s'assurer que notre modèle exécutable m_v vérifie $m_v \in \mathcal{M}_{valides} = \{m | \forall t \in \mathcal{T}_f, t(m)\}$.

Rôle	Modélisateur + Implémenteur	Le modélisateur propose, avec le concours de l'implémenteur, un modèle opérationnel répondant aux exigences exprimées par le thématicien.
Entrées	$M_{SMA}, \mathcal{T}_f$	modèle SMA, jeu de tests
Sorties	m_v	modèle amélioré

Dans le procédé que nous présentons ici, l'accent est donc mis sur le lien qui existe entre les propriétés du modèle idéal et les propriétés de l'implémentation qui en est faite. Il s'agit d'identifier l'adéquation entre un modèle abstrait et sa mise en œuvre, ce qui correspond à un processus de vérification [22]. Notons que ce processus est à distinguer d'un processus de validation dont l'objet est de s'assurer que le système complet permet de répondre avec suffisamment de précision à une question que l'on se pose sur le système modélisé [18]. Étant donné la nature des modèles à base d'agents, le processus de spécification et de tests décrit ici est appliqué à l'échelle d'un agent, et donc individuel (microscopique). Cependant, et notamment pour l'ajout de facettes interactionnelles, des tests sur des propriétés d'échelles mésoscopiques ou macroscopiques peuvent être intégrés à $\mathcal{T}_f$.

Cette phase n'est pas directement intéressée par la validité macroscopique du système produit, mais bien à la concordance entre les propriétés désirées pour les comportements individuels avec le comportement des entités effectivement mises en œuvre.

Cette approche est particulièrement inspirée du développement dirigé par les tests et cherche à en reproduire les propriétés. Ainsi, elle incite à la concision et à produire une version *a minima* permettant de répondre aux exigences exprimées. Un modèle exhibant plus de propriétés que celles attendues se montrerait être une perte de temps puisque soit les propriétés développées

2. L'utilisation ici du terme «incorrecte» ne correspond pas à sa définition usuelle selon laquelle une implémentation est dite incorrecte si elle ne correspond pas au modèle abstrait dont elle doit être la traduction. Il s'agit ici de décrire une implémentation ne présentant pas les propriétés attendues.

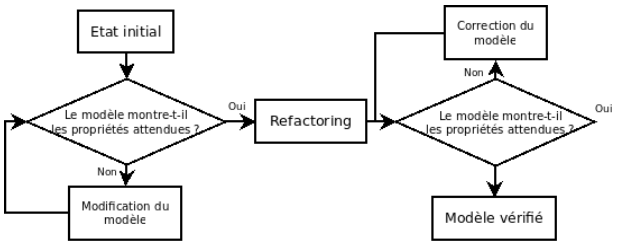


FIGURE 2 – Processus de conception et développement orienté par les tests incluant une phase de *refactoring*.

ne correspondent pas à un besoin exprimé par l'expert thématique, soit ces propriétés sont intéressantes, auquel cas elles devraient faire l'objet d'une itération à part entière. De même, le processus est adapté à l'adoption d'une phase de «*refactoring*» permettant d'accroître la maintenabilité du programme produit comme le présente la figure 2.

3.5 Planification

Rôle	Expert thématique	L'expert thématique valide (aux vues de ses objectifs de simulation) le système, et initie au besoin une nouvelle itération.
Entrées	$M_{SMA}, \mathcal{O}$	modèle SMA, objectifs de simulation
Sorties	d	décision

De manière analogue au modèle en spirale, notre modèle se termine par une phase de planification. Son objectif est entre autre de déterminer si le système multi-agent M_{SMA} est valide au regard de l'objectif $\mathcal{O}$ ou si des itérations supplémentaires sont nécessaires.

Cette phase est l'occasion de confronter le modèle développé à des données réelles. Ceci est généralement lié au processus de validation, mais la confrontation aux données réelles peut également être le processus par lequel l'expert identifie de nouveaux points sur lesquels le comportement des agents est insuffisant.

Dans le cas où les objectifs macroscopiques n'ont pas été atteints, malgré une cohérence à l'ensemble des tests définis dans la phase de spécification, cela signifie soit que l'amélioration était insuffisante, auquel cas il est nécessaire d'ajouter de nouvelles facettes, soit que l'hypothèse selon laquelle l'introduction de cette facette permet d'atteindre l'objectif de simulation est erronée. Dans ce cas, l'itération suivante sera réalisée sur la base du modèle précédent et non du nouveau modèle.

Ce dernier cadrant sert donc 1) à déterminer s'il est nécessaire ou non de poursuivre le développement du modèle plus avant et 2) à profiter de la confrontation du modèle à des données réelles afin d'identifier des pistes d'évolution. Cette phase relève de la responsabilité de l'expert thématique, dont le rôle peut être rapproché de celui du «*product owner*» des méthodes agiles, dans le sens où il exprime le besoin et valide la solution en fin de développement.

4 Illustration d'un cycle de conception

Dans cette partie, nous illustrons la méthode proposée à l'aide de l'application suivante : proposer un modèle permettant de prendre en compte les spécificités des véhicules lourds dans les simulations de trafic routier réalisées à l'aide de systèmes à base d'agents. Afin de servir de base à l'implémentation de nos travaux, nous avons choisi d'utiliser la plate-forme de simulation ARCHISIM. Cette plate-forme dédiée à la simulation microscopique du trafic routier a été le support de nombreux travaux de recherches tels que ceux illustrés dans [10]. ARCHISIM permet de faire interagir différents acteurs indépendants sur une même infrastructure routière. Sur cette base, nous souhaitons identifier et concevoir un modèle de poids lourds améliorant la plateforme de façon incrémentale.

4.1 Identification d'une facette du comportement

Afin d'étudier l'apparition des phénomènes émergents que sont les vagues de sur-congestion, l'expert thématique identifie une facette de comportement qu'il juge intéressant. Il peut, par exemple, décider de s'intéresser aux accélérations des véhicules. Il doit alors déterminer s'il est pertinent d'investir dans la reproduction des capacités d'accélération et si cela aura un impact sur les simulations.

4.2 Évaluation de l'impact attendu

Si nous considérons l'expérience des modèles macroscopiques du premier ordre supposant des changements de vitesse instantanés, il semblerait que cet aspect ne soit pas important dans la propagation des ondes de sur-congestion. Cependant, les travaux de [9] mettent en avant

que, dans un trafic proche de la saturation, le temps qu'un véhicule met à rejoindre sa vitesse désirée suite à l'apparition d'une perturbation (par exemple suite à l'insertion à vitesse faible d'un véhicule sur l'infrastructure) influence fortement la formation de congestions. Nous pouvons en déduire qu'un modèle de comportement de véhicules lourds doit être en mesure d'exhiber des capacités d'accélération réalistes afin de pouvoir reproduire leur influence sur l'écoulement d'un trafic simulé.

4.3 Définition des tests

Nous cherchons alors à proposer un test permettant de vérifier la propriété d'accélération différenciée entre véhicules légers et lourds. Pour ce faire, nous mesurons en situation réelle le profil d'accélération sur un véhicule de type semi-remorque. Le véhicule utilisé pour cette expérience est un ensemble tracté par un IVECO Stralis d'une puissance de 440 chevaux. L'ensemble tracté a une masse de 39T. La donnée de vitesse du véhicule est extraite de son bus «CAN» [3] à l'aide d'outils de diagnostique spécialisés.

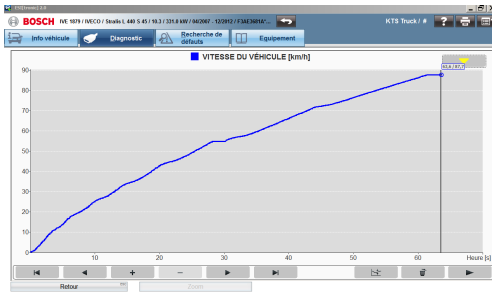


FIGURE 3 – Profil d'accélération d'un véhicule composé d'un semi-remorque / tracteur.

La figure 3 montre que le véhicule parvient à atteindre sa vitesse maximale ($88km.h^{-1}$, ce qui correspond à la vitesse à laquelle le véhicule est bridé) en environ 60 secondes.

4.4 Modifications du modèle

Le modèle de simulation ARCHISIM dispose de bases permettant la simulation du comportement de véhicules lourds. Nous évaluons la capacité de ce modèle à reproduire les capacités d'accélération souhaitées. Nous montrons dans cette partie 1) en quoi le modèle original est incapable de répondre aux exigences 2) comment nous proposons d'améliorer le modèle et 3) la vérification microscopique des comportements obtenus. Ces différentes étapes, réalisées par le modélisateur et avec l'aide de l'implémenteur, correspondent à la phase de développement.

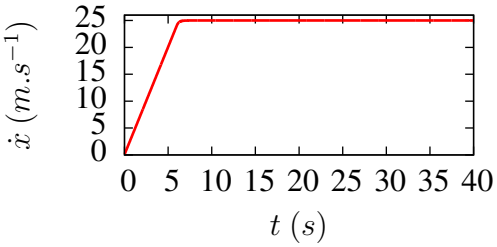


FIGURE 4 – Accélération d'un PL dans ARCHISIM

Évaluation du modèle existant. La figure 4 montre le profil en accélération d'un véhicule lourd passant d'une vitesse de $0km$ à $90km$ sur une route plate tel que simulé par ARCHISIM. Cette figure fait apparaître que le modèle ARCHISIM, dans sa version initiale, n'est pas en mesure de reproduire le comportement attendu. Il appartient donc au modélisateur de proposer un amélioration permettant de remédier à cela. Le résultat de cette modélisation sera appelée ARCHIPL.

Proposition de modélisation. Afin d'obtenir des profils d'accélération réalistes pour les véhicules ARCHIPL, nous proposons d'intégrer une version adaptée du modèle proposé par [21] au modèle ARCHISIM. Ce modèle dynamique est utilisé afin de déterminer quelle est l'accélération maximale qu'un véhicule peut adopter. Cette accélération maximale est fonction de plusieurs facteurs propres au véhicule (sa vitesse courante, sa puissance…) ainsi que de différentes caractéristiques de son environnement telles que l'inclinaison de la route, la vitesse du vent… Le modèle mis en œuvre simplifie la dynamique des véhicules en différents points de façon à obtenir un compromis entre fidélité du modèle et capacité de paramétrisation.

Afin de modéliser l'effet de la puissance tractrice moteur, un bilan simplifié des forces s'appliquant au véhicule est fait. Comme l'illustre la figure 5, les forces considérées dans ce modèle sont les suivantes : 1) La force motrice, 2) les forces aérodynamiques freinant l'avancement du véhicule, dues au vent relatif, 3) la force de gravité ralentissant le franchissement de pentes, essentiellement du fait de la masse du véhicule, et 4) la force de réaction du support.

3. CAN : Controller Area Network

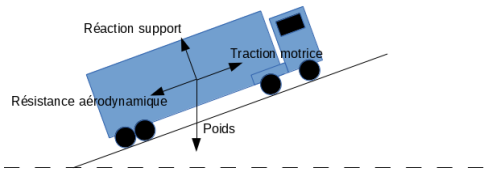


FIGURE 5 – Bilan des forces appliquées à un véhicule sur la route.

Le poids, force verticale, peut se décomposer en deux composantes telles que $\vec{F}_{poids} = \vec{P}_{perp} + \vec{P}_{para}$. $\vec{P}_{perp}$ est perpendiculaire au support et est l'opposée de la force de réaction du support. $\vec{P}_{para}$ est la composante du poids parallèle au support de valeur donnée par $m_a * g * sin(\alpha)$ où α est l'inclinaison du support et g l'accélération de la pesanteur.

La force de traction $\vec{F}_{mot}$ est générée par le moteur du véhicule. Nous supposons que le moteur est exploité au maximum de sa puissance, et négligeons les pertes mécaniques (internes, et liées au contact pneumatique / chaussée). L'intensité de la force motrice délivrée par le véhicule est donnée par

$$F_{mot} = \left\| \vec{F}_{mot} \right\| = P_a / \left(\dot{x}_m \left(t \right) \right) \qquad (1)$$

la force motrice et la vitesse du véhicule étant dirigées selon le même axe. Notons le cas particulier où le véhicule quitte l'arrêt. Dans ce cas, $\dot{x}_a = 0$, et l'équation 1 ne peut être calculée. La force de traction maximale possible est bornée par $F_{mot}^{max} = \mu * F_{essieu\_tracteur}^{z} = g * M_{essieu\_tracteur} * \mu$ [4].

Enfin, la force de résistance aérodynamique $\vec{F}_{aero}$ (qui s'exerce à l'opposé du mouvement du mobile, donc uniquement opposée à la force motrice) a une intensité de $\left\| \vec{F}_{aero} \right\| = q * \mathcal{S} * c_x$. $\mathcal{S}$ est la surface de référence du véhicule (surface exposée au vent), c_x le coefficient de traînée du véhicule, $q = \frac{1}{2} \rho \dot{x}_a^2$. ρ est la masse volumique de l'air et $\dot{x}_a$ la vitesse longitudinale du véhicule.

L'accélération maximale possible pour un véhicule a à un instant donné t est donc donnée, une fois le bilan des forces appliqué, par $\gamma_a^{max}(t) = \frac{1}{m_a} \left[\frac{P_a}{\dot{x}_a} - k * \dot{x}_a^2 - m * g * \sin\left(\alpha\right) \right]$.

4. cette valeur étant constante pour un véhicule donné, elle est fournie par un paramètre du modèle.

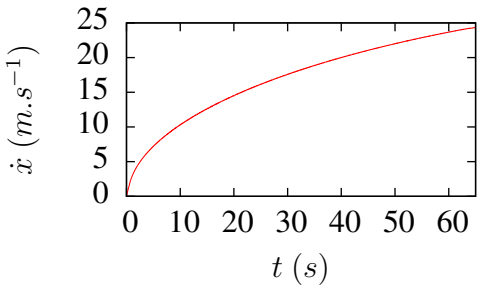


FIGURE 6 – Profil d'accélération d'un véhicule ArchiPL.

Évaluation des capacités d'accélération du modèle ARCHIPL. Nous cherchons maintenant à évaluer la pertinence des améliorations que nous avons apportées au modèle ARCHISIM dans le cadre du développement du modèle ARCHIPL. Pour ce faire, nous réutilisons le scénario que nous avons utilisé pour mettre en défaut le modèle ARCHISIM. Nous confrontons le résultat donné par le modèle mis en œuvre avec le profil d'accélération tel que défini dans le test.

Le véhicule simulé, un véhicule de type «PL3» correspondant à un ensemble semi-remorque / tracteur dans la plate-forme ARCHISIM, parvient à atteindre la même vitesse que le véhicule réel en environ 65 secondes. Bien que les résultats expérimentaux et simulés montrent des différences durant les premières phases de l'accélération (les 15 premières secondes), l'amélioration est importante par rapport aux résultats présentés dans la figure 4, dans laquelle le PL atteint cette vitesse en 7 secondes environ.

Ces résultats permettent d'achever le cadrant courant de développement du modèle ARCHIPL.

4.5 Planification

Ce cadrant (relevant de l'expert thématique) a pour objet d'une part la validation macroscopique du système de simulation une fois le modèle individuel modifié intégré et d'autre part le choix d'initier ou non une nouvelle itération (et donc de s'intéresser à une nouvelle facette de comportement).

Les résultats de validation associés à cette tâche dépassent le cadre de cet article et ne sont donc pas abordés ici. L'itération concernant la modélisation des accélérations des véhicules lourds dans ARCHISIM est donc considérée achevée.

5 Discussion

Les méthodes de développement de système à base d'agents. De nombreux travaux se sont intéressés aux méthodes de réalisation de modèles multi-agents. Une revue de différentes méthodes ainsi que de nombreuses références peuvent être trouvées dans [12]. Ainsi, une grande quantité de méthodes est disponible dans la littérature afin de guider le développement de systèmes à base d'agents. Nous pensons notamment aux méthodes Gaia [26], MaSE [7], INGENIAS [20], ou Prometheus [19] pour n'en citer que quelques-unes. Ces méthodes proposent des outils et démarches applicables dans le cadre de la capture des besoins, de l'analyse, de la conception et éventuellement de l'implantation. Gaia propose ainsi un cadre conceptuel permettant l'analyse et la conception d'un système à base d'agents centré sur le concept d'organisation. Cette méthode propose de décrire le système en terme d'agents, environnement, rôles, responsabilités, autorisations, règles et organisations. MaSE propose d'analyser le système par ses buts, et de définir les rôles servant à atteindre ces buts. Ingenias ré-utilise des concepts et outils de RUP [16] en y ajoutant des concepts spécifiques aux systèmes multi-agents tels que l'agent, le rôle, la tâche et l'organisation.

Ces méthodes sont centrées sur le développement initial de modèles à base d'agents, mais non sur leur évolution. Elles fournissent un cadre permettant de comprendre et de spécifier une organisation en décrivant ses membres, leurs rôles, leurs buts, leurs interactions ou leurs moyens. Dans le cas général comprendre ces relations n'est pas une tâche aisée. Cependant, en ce qui concerne la conception d'un système de trafic à base d'agents, cette analyse est presque triviale. Les buts de chacun des agents sont indifférenciés, et ce, qu'ils soient poids lourds ou véhicules légers : atteindre sa destination en minimisant son temps de trajet. Les plans d'actions élaborés par les différents conducteurs sont similaires. Les contraintes propres à chaque agent font cependant que leur exécution peut différer légèrement et entraîner des perturbations au niveau des populations de véhicules. Nos travaux se focalisent surtout sur l'identification et la reproduction de ces variations, dont les conséquences macroscopiques ne sont pas nécessairement bien anticipées. De plus, la littérature existante du trafic nous fournit une base suffisante de modèles théoriques et opérationnels fonctionnels. Une partie des choix de conception sont donc réutilisables. Notre travail cherche à raffiner des modèles de comportements de manière à faire ressortir les variations entre les comportements des individus s'ils impactent l'écoulement du trafic. Les choix de conception déjà effectués ne sont *a priori* pas remis en cause, sauf s'ils montrent leurs limites.

Une question d'échelles. Lors de la réalisation d'un système de simulation à base d'agents, l'objet d'étude — c'est-à-dire l'objet dont le modélisateur cherche à comprendre et reproduire les propriétés — est d'ordre macroscopique. Les attentes et exigences en terme de résultats se portent sur les propriétés émergentes. De ce fait, l'activité de validation — activité par laquelle on cherche à s'assurer que le modèle produit permet, par analogie, d'étudier un objet réel avec suffisamment de précision — est intrinsèquement liée à l'ordre macroscopique. L'activité de développement des modèles d'agents utilisés est, elle, réalisée à l'échelle microscopique. La simulation à base d'agents a donc cela de particulier que les objets qu'elle cherche à étudier ne sont jamais modélisés directement, et que les modèles produits ne sont pas étudiés directement. Les modèles d'agents n'étant pas étudiés directement, la notion de validation ne s'applique pas à eux.

Le processus de développement que nous décrivons cherche à maintenir une distinction entre le travail sur les modèles microscopiques et la construction macroscopique. Les phases de travail sur les entités microscopiques ne sont ainsi pas liées à une activité de validation. Cette dernière ne peut avoir lieu que dans la dernière étape du cycle incrémental, et est nécessaire à l'arrêt des itérations.

6 Conclusion

L'utilisation de procédures de développements assurant la qualité et l'exactitude des codes produits est particulièrement critique. Comme le souligne [13], la qualité des résultats scientifiques obtenus à l'aide de codes informatiques produits en dehors de cadres de développement rigoureux est fortement affectée.

Dans cet article, nous avons présenté une méthode de développement incrémental de modèles d'agents, TIM4MAS, inspirée de deux approches classiques de la littérature : le modèle en spirale et le développement dirigé par les tests. Dans l'approche proposée, le pilotage du développement est confié à l'expert thématique. Celui-ci définit les propriétés attendues du

modèle d'agent plutôt que son fonctionnement. Tout modèle exécutable remplissant les attentes peut ainsi être considéré comme vérifiant les besoins de l'évolution courante, et la question de la validité du modèle peut être posée. Cette approche permet de limiter les risques d'apparition d'approximations handicapantes dans le processus d'implémentation d'un modèle, et permet de se concentrer sur les aspects ayant la plus forte influence sur l'apparition des phénomènes émergents recherchés.

Nous avons illustré l'utilisation de cette approche pour la conception d'un modèle de poids-lourd sur la base d'un modèle de véhicule classique. Plus généralement, l'aspect incrémental de notre méthode est adapté aux processus de recherche en simulation à base d'agents, en permettant à chaque cycle d'introduire de nouveaux éléments à un modèle existant, puis d'en vérifier la non-régression. La mise à disposition de bibliothèques de spécification permettrait alors de mettre en œuvre des bancs d'essai communs.

Références

[1] K. Beck. Embracing change with extreme programming. *Computer*, 32(10) :70–77, 1999.

[2] B. W. Boehm. A spiral model of software development and enhancement. *Computer*, 21(5) :61–72, 1988.

[3] M. Brackstone and M. McDonald. Car-following : a historical review. *Transportation Research Part F : Traffic Psychology and Behaviour*, 2(4) :181–196, 1999.

[4] S. Chanut. *Modélisation dynamique macroscopique de l'écoulement d'un trafic routier hétérogène poids lourds et véhicules légers*. PhD thesis, Institut National des Sciences Appliquées de Lyon, 2005.

[5] V. Chevrier and N. Fatès. An example of a multi-agent system described as a discrete dynamical system. In *European Workshop on Multi-agent Systems (EuMAS)*, volume 1, page 17, 2010.

[6] N. David, J. Sichman, and H. Coelho. Towards an emergence-driven software process for agent-based simulation. *Multi-Agent-Based Simulation II*, pages 49–78, 2003.

[7] S. A. DeLoach, M. F. Wood, and C. H. Sparkman. Multiagent systems engineering. *International Journal of Software Engineering and Knowledge Engineering*, 11(03) :231–258, 2001.

[8] A. Drogoul, D. Vanbergue, and T. Meurisse. Multi-agent based simulation : Where are the agents ? In *Multi-agent-based simulation II*, pages 1–15. Springer, 2003.

[9] A. Duret. *Hétérogénéités du trafic autoroutier. Identification, qualification, modélisation et impact sur l'écoulement*. PhD thesis, Ecole nationale des travaux publics de l'état, 2010.

[10] S. Espié. *Simulation comportementale et réalité virtuelle : vers une simulation globale du système de trafic*. PhD thesis, UPMC, December 2004. Habilitation thesis.

[11] J. Galán, L. Izquierdo, S. Izquierdo, J. Santos, R. Del Olmo, A. López-Paredes, and B. Edmonds. Errors and artefacts in agent-based modelling. *Journal of Artificial Societies and Social Simulation*, 12(1) :1, 2009.

[12] J. J. Gómez-Sanz, M.-P. Gervais, and G. Weiss. A survey on agent-oriented oriented software engineering research. In *Methodologies and Software Engineering for Agent Systems*, pages 33–62. Springer, 2004.

[13] E. C. Hayden. Mozilla plan seeks to debug scientific code. *Nature*, 501(7468) :472–472, 2013.

[14] D. Helbing. Traffic and related self-driven many-particle systems. *Reviews of modern physics*, 73 :1067, 2001.

[15] S. Hoogendoorn and P. Bovy. State-of-the-art of vehicular traffic flow modelling. *Proceedings of the Institution of Mechanical Engineers, Part I : Journal of Systems and Control Engineering*, 215(4) :283–303, 2001.

[16] P. Kruchten. *The rational unified process : an introduction*. Addison-Wesley Professional, 2004.

[17] Y. Luo and L. Bölöni. Modeling the conscious behavior of drivers for multi-lane highway driving. *Workshop on Agents in Traffic and Transportation*, pages 93–103, 2012.

[18] M. Minsky. Matter, mind and models. 1965.

[19] L. Padgham and M. Winikoff. Prometheus : A methodology for developing intelligent agents. In *Agent-Oriented Software Engineering III*, pages 174–185. Springer, 2003.

[20] J. Pavón, J. J. Gómez-Sanz, and R. Fuentes. The ingenias methodology and tools. *Agent-oriented methodologies*, 9 :236–276, 2005.

[21] H. Rakha, I. Lucic, S. Demarchi, J. Setti, and M. Aerde. Vehicle dynamics model for predicting maximum truck acceleration levels. *Journal of transportation engineering*, 127(5) :418–425, 2001.

[22] R. Sargent. Verification, validation, and accreditation : verification, validation, and accreditation of simulation models. In *Proceedings of the 32nd conference on Winter simulation*, pages 50–59. Society for Computer Simulation International, 2000.

[23] M. Shannon, G. Miller, and R. Prewitt Jr. *Software Testing Techniques : Finding the Defects that Matter*. Cengage Learning, 2004.

[24] L. Six, S. Ieng, J. Saunier, and Z. Guessoum. Les boucles d'hystérésis comme outil d'analyse des comportements de conducteurs. *Journés Francophones sur les Systèmes Multi-Agents*, 2012.

[25] M. Treiber, A. Hennecke, and D. Helbing. Microscopic simulation of congested traffic. In *Traffic and Granular Flow*, volume 99, pages 365–376, 2000.

[26] F. Zambonelli, N. R. Jennings, and M. Wooldridge. Organisational abstractions for the analysis and design of multi-agent systems. In *Agent-Oriented Software Engineering*, pages 235–251. Springer, 2001.

Etude de la propagation d'une perturbation dans un réseau d'interaction formé par un système multi-agent

H. Rabai[a] haifa.rabai@univ-lehavre.fr
R. Charrier[a] rodolphe.charrier@univ-lehavre.fr
C. Bertelle[a] cyrille.bertelle@univ-lehavre.fr

[a]Université du Havre, France

Résumé

Nous étudions la propagation d'une perturbation dans un réseau d'interaction formé par un système multi-agent. Ce réseau est représenté par un graphe où les agents sont les noeuds et les interactions sont les arcs dirigés. La perturbation considérée ici est celle de l'état interne de l'agent qui est réduit à une variable réelle dépendant du temps. Cet état peut être "stable" ou "perturbé" selon la nature dynamique de cette variable. Nous supposons que le réseau d'interaction possède initialement un agent perturbé qui est une série temporelle chaotique. Notre objectif consiste à détecter l'ensemble des agents qui deviennent perturbés suite à l'interaction dans le réseau. Nous cherchons également à identifier la source de la perturbation dans le graphe à partir des séries temporelles des agents. Nous proposons alors un algorithme permettant de remonter à la source depuis un agent quelconque impacté par la perturbation. Nous présentons dans cet article les résultats des simulations effectuées pour tester cet algorithme.

Mots-clés : *Réseau d'interaction, agents chaotiques, Coupled Map Network, entropie de transfert*

Abstract

We study the spread of a disturbance in an interaction network formed by a multi-agent system. This network is represented by a graph where the nodes are the agents and interactions are oriented edges. The disturbance considered here is related to the internal state of the agent and is reduced to a real time-dependent variable. This state can be "stable" or "disturbed". We assume that the interaction network initially contains only one disturbed agent that has chaotic time series. We aim to detect the agents that become disturbed due to the interaction in the network. We also seek to identify the source of the disturbance in the graph based on the agents time series. Therefore, we propose an algorithm to search the source starting from any impacted node. We present here the results of our simulations to test this algorithm.

Keywords: *Interaction network, chaotic agents, Coupled Map Network, transfer entropy*

1 Introduction

De nombreux problèmes de la diffusion d'information, de rumeurs, de maladies ou de virus qui ont pour environnement l'internet, les réseaux sociaux réels ou virtuels ou encore les réseaux de communications, sont modélisés par des graphes dans lesquels les noeuds représentent les agents et les arcs représentent l'interaction existant entre ces agents. D'autre part, les noeuds de ces graphes possèdent des états discrets ou continus qui peuvent être modifiés par le processus de diffusion existant au sein du graphe.

Parmi les modèles à base d'agents à états discrets, on peut citer certains modèles épidémiologiques où les noeuds peuvent avoir notamment l'un des états discrets suivants : "Susceptible", "Infected" ou "Recovered" [2]. Dong et al. [5] présentent un système multi-agent pour la modélisation des infections où les agents peuvent être soit susceptibles soit infectés. Le modèle permet de faire des prédictions utiles sur les probabilités de contracter des infections au sein d'une communauté.

De même, dans l'étude de la propagation des rumeurs, on retrouve généralement trois états qui sont : "Spreader", "Stiffler" et "Ignorant" [3]. Tang et al. [14] proposent un système multi-agent où les agents possèdent l'un de ces états pour étudier la propagation des rumeurs dans les réseaux sociaux.

Concernant les modèles à base d'agents à états continus, on peut citer en particulier le modèle de Deffuant et al. [4] car il est assez proche du notre dans sa conception. Dans ce modèle, les agents sont caractérisés par des variables d'opinion continues ayant des valeurs entre -1 et 1. L'opinion d'un agent est générée à partir d'une fonction linéaire et change en fonction de sa dis-

tance par rapport à l'opinion d'un autre agent en interaction avec lui. Ainsi, dans ce modèle l'état d'une opinion x d'un agent évolue dans le temps en fonction de l'opinion y d'un autre agent :

$$x^{t+1} = x^t + \mu(y^t - x^t) \quad (1)$$

où μ est le paramètre de convergence.
Dans la même catégorie de modèle, on retrouve le modèle de Fast et al. [6] qui étudient la propagation de l'anxiété dans un système multi-agent suite à la propagation d'une maladie. L'état des agents est ici une variable générée initialement aléatoirement entre 0 et 1, il change d'une part, en fonction de l'interaction avec les voisins et d'autre part, à la réception d'une excitation venant des médias à cause de la propagation d'une maladie.

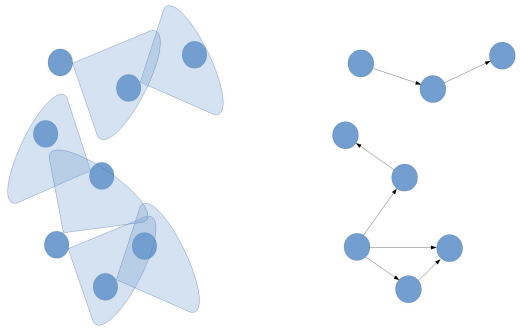
Notre travail se rapproche des modélisations d'agents à états continus citées ci-dessus. Nous nous intéressons à l'analyse de la propagation d'une perturbation dans un réseau d'interaction formé par un système multi-agent : les noeuds sont les agents et les arcs représentent l'interaction entre deux agents. L'interaction ici est directement liée à la perception de l'agent : si un agent perçoit un autre agent, on considère qu'il est en interaction avec lui et cette interaction est représentée dans le graphe par un arc. La perturbation considérée ici est celle de l'état interne de l'agent qui est réduit à une variable réelle dépendant du temps. L'état interne de l'agent peut être considérée comme une information (rumeur,...) mais à ce stade de modélisation, il demeure une grandeur abstraite. A l'instar du modèle de Deffuant, l'état d'un agent est une grandeur continue x dépendant du temps. Dans notre modèle, cet état peut être "stable" ou "perturbé" selon la nature de son évolution dynamique. Toutefois, cette évolution est gouvernée par une fonction non linéaire contrairement à la fonction linéaire de Deffuant (cf. section 2).
La situation que nous étudions est la suivante : nous supposons que le réseau d'interaction de notre système-agent possède initialement un seul agent perturbé. Notre objectif consiste alors dans un premier temps à détecter l'ensemble des agents qui deviennent perturbés suite à l'interaction dans le réseau sur un temps suffisamment long. Puis dans un second temps, il s'agit d'identifier la source de la propagation de la perturbation dans le graphe.
Pour réaliser notre objectif final, nous allons procéder en deux phases : une première phase correspondant au problème direct qui va calculer l'évolution dynamique du réseau d'une façon contrôlée au moyen d'un modèle de calcul appelée réseau d'itérations couplées ("Coupled Map Network"). Puis dans une deuxième phase correspondant au problème inverse, nous cherchons à analyser le réseau uniquement à partir de la dynamique des états internes des agents générés dans la première phase. Dans cette seconde phase, nous cherchons à retrouver l'ensemble des noeuds impactés par la perturbation ainsi que la source de cette propagation.
Cette recherche est rendue possible par l'utilisation des mesures basées sur l'entropie de Shannon. Concernant l'identification de la source de la perturbation dans le réseau, nous utilisons l'entropie de transfert [13] et nous proposons un algorithme de recherche qui se base sur cette mesure et permet de remonter à la source à partir d'un agent quelconque de l'ensemble des agents impactés par la propagation. Notre algorithme de recherche dépend de deux paramètres et nous présentons dans cet article les simulations effectuées pour tester cet algorithme et étudier l'influence de ces paramètres sur le taux de réussite de la détection de la source.
Dans cet article, nous montrons dans une première section comment on construit le réseau d'interaction à partir du système multi-agent qui est la base de l'étude. Dans une section suivante, nous présentons le modèle de calcul du réseau d'itérations couplées (CMN) [8]. Puis les sections suivantes concerneront le problème inverse et les simulations effectuées, notamment, nous proposons un algorithme de détection de la source de la perturbation. Pour conclure, nous discutons des résultats et présentons les perspectives de ce modèle à des problématiques plus concrètes.

2 Réseau et modèle d'interaction

2.1 Construction du réseau d'interaction

Le système multi-agent que nous étudions est formé par un nombre n d'agents immobiles qui sont situés dans un environnement spatial (que nous assimilons à un espace à deux dimensions carré, par conséquent, limité). Chaque agent possède une position dans l'environnement et une perception limitée sur l'environnement et sur les autres agents. Cette perception limitée est modélisée par un cône d'angle égal à 120° inspiré du cône de la perception visuelle humaine et d'un rayon proportionnel à la taille de l'environnement (environ 10%). Ces valeurs ne sont qu'indicatives à ce niveau du modèle car elles ne servent qu'à générer le réseau d'interaction

qui est la base de départ de notre modèle. Ainsi,



(a) Système multi-agent (b) Réseau d'interaction

FIGURE 1 – Réseau de perception assimilé au réseau d'interaction potentielle des agents.

cette configuration du système multi-agent permet de construire le graphe d'interaction comme suit : comme on l'a indiqué dans l'introduction, nous définissons l'interaction comme un lien de perception entre les agents, par conséquent, la construction du graphe se déroule comme suit : Si un agent k perçoit un autre agent l et donc, est en interaction avec lui (en réalité l'agent k est de cette manière influencé par l'agent l), nous créons un arc dirigé du noeud X_k vers le noeud X_l (cf. figure 1). Dans toute la suite de cet article, nous désignerons noeud k et agent k par la même variable X_k. Au final, nous obtenons ainsi un réseau d'interaction dynamique et orienté :

– L'ensemble des agents / noeuds $V = \{X_1, X_2, ..., X_n\}$ est associé aux variables d'état mesurables $X^t = (x_1^t, x_2^t, ..., x_n^t)$.
– L'ensemble des arcs $E = \{E_1, E_2, ..., E_m\}$ définit les interactions entre les noeuds, où m est le nombre d'arcs existants entre les noeuds.

Les états des agents évoluent en fonction des interactions d'une part et en fonction d'une application non linéaire. L'ensemble constitue le modèle de calcul de réseau d'itérations couplées que nous explicitons dans la section suivante.

2.2 Réseau d'itérations couplées

Comme il a été dit dans l'introduction, nous proposons de modéliser la dynamique du réseau d'interaction par un réseau d'itérations couplées connu dans la littérature comme un modèle d'oscillateurs non linéaires couplés (CMN [8]) à temps discret.

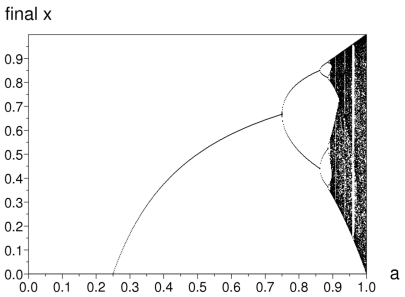


FIGURE 2 – Diagramme de bifurcation de l'application logistique.

Au préalable, nous avons à expliquer plus précisément l'utilisation d'une application non linéaire pour générer l'évolution temporelle des états des agents individuels puis préciser la notion d'interaction dans le système multi-agent. Tout d'abord, nous définissons un état stable pour un agent comme un état dont les valeurs sont constantes dans le temps (point fixe) ou périodiques. D'autre part, nous définissons un état perturbé comme une variable temporelle prenant des valeurs aléatoires au cours du temps. Dans le cadre de notre modèle de calcul d'itérations couplées, ces deux caractéristiques distinctes vont pouvoir être générées par une seule et même application déterministe paramétrée monodimensionnelle et non linéaire.

L'état x_k d'un agent / noeud quelconque X_k généré par cette application non linéaire est donné pour tout instant t par l'équation suivante :

$$x_k^{t+1} = f(x_k^t, a_k) = 4\, a_k\, x_k^t (1 - x_k^t) \qquad (2)$$

où a_k est appelé paramètre de contrôle de cette application. Cette application est connue sous le nom d'application logistique. Elle est non linéaire puisque générée par un polynôme de degré 2 en x. D'autre part, les grandeurs a et x prennent leur valeur dans $[0, 1]$.
Cette application non linéaire a de nombreuses propriétés dynamiques en fonction de son paramètre de contrôle a : les séries temporelles générées sont résumées dans le diagramme de bifurcation de la figure 2. Ce diagramme s'interprète en partant de la gauche ($a = 0$) vers la droite (jusqu'à $a = 1$) avec en ordonnée, les différentes valeurs obtenues pour x après une centaine d'itérations (ou plus) de la suite (2) :

– Pour a de 0 à 0,25, on a une valeur finale de x nulle.
– Pour a de 0,25 a 0,75, on a une valeur finale de x constante mais non nulle.

– Pour a de 0,75 a 0,89, on observe des dédoublements successifs de la courbe initiale indiquant des cycles périodiques de périodes de plus en plus grandes.
– Au delà, on peut observer un mélange de cycles périodiques et non périodiques (correspondant aux bandes sombres).
– Pour $a = 1$, on a un seul trait noir couvrant tout l'intervalle $[0, 1]$ pour lequel la série obtenue bien que déterministe, est pseudo-aléatoire. Ce comportement dynamique est appelé chaos déterministe.

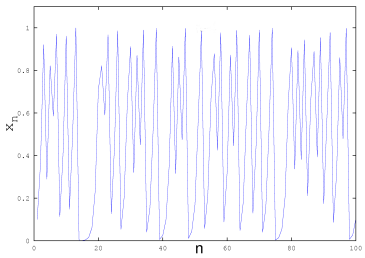
Le choix de l'application logistique est justifiée, entre autres, par la grande variété de comportements dynamiques qu'on peut générer et qui dépendent de la valeur de son unique paramètre a. En effet, un état stable sera généré pour des valeurs de a inférieures à 0,89. D'autre part, un état perturbé correspondra à une valeur de $a = 1$ générant une série chaotique. Dans la suite du document, nous appellerons l'agent dont l'état est chaotique un agent perturbé ou un agent chaotique.

Explicitons maintenant comment notre modèle de calcul d'itérations couplées prend en compte l'interaction entre les agents. Comme son nom l'indique, il réalise cette interaction par un couplage mathématique entre les états des agents. La force de cette interaction est quantifiée par un coefficient de couplage ϵ qui prend ses valeurs entre 0 et 1 et qui a une valeur uniforme pour tous les agents dans les simulations présentées. Mais rien n'interdit que ce coefficient puisse être spécifique à chaque agent dans une évolution de notre modèle. Ce couplage mathématique s'exprime au travers de l'équation générale d'évolution de l'état x_k de l'agent X_k suivante :

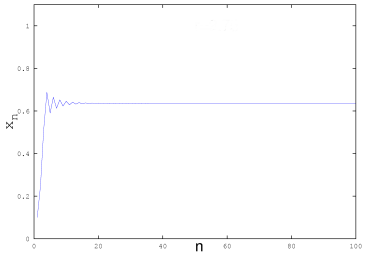
$$x_k^{t+1} = (1-\epsilon) f(x_k^t, a_k) + \frac{\epsilon}{N_l} \sum_{l=1}^{N_l} f(x_l^t, a_l) \quad (3)$$

où ϵ est le coefficient de couplage. N_l représente l'ensemble des agents perçus par l'agent X_k. l sont les indices des agents avec lesquels X_k est en interaction. Le calcul des états des agents se fait d'une façon synchrone sur l'ensemble du réseau.

Ce modèle est connu dans la littérature pour générer une synchronisation des états des noeuds (c'est à dire que tous les noeuds ont la même série temporelle au bout d'un certain temps) lorsque le graphe est complet et l'interaction est suffisamment forte (y compris pour des séries temporelles chaotiques). Dans le cas qui nous intéresse, les synchronisations ne sont que partielles et devraient s'affaiblir avec la taille du réseau.



(a) Etat perturbé. La série temporelle est générée avec $a = 1$.



(b) Etat stable. La série temporelle est générée avec $a = 0.68$.

FIGURE 3 – Exemples d'états.

Ce modèle permet dans cette première phase de générer toutes les séries temporelles des noeuds du graphe et par construction, nous pouvons déduire quel est le sous graphe d'agents impactés par la propagation (c'est à dire le couplage ici) sous l'influence d'un seul agent chaotique. Nous avons à présent tous les éléments pour aborder tout le problème inverse c'est à dire l'étude et la recherche du sous graphe de ces agents impactés à partir des seules séries temporelles obtenues du réseau d'itérations couplées présenté précédemment. Il est en effet difficile de déterminer la nature chaotique d'une série de donnée expérimentale ce qui nécessite des outils de mesures adaptées à ce type de série. La section suivante présente le traitement du problème inverse.

3 Analyse du problème inverse

3.1 Détection du groupe d'agents perturbés

Afin de caractériser les propriétés dynamiques des séries temporelles des agents, nous avons besoin d'une mesure qui peut détecter la nature chaotique d'un signal. Par conséquent, nous proposons d'utiliser l'entropie de Shannon qu'on calcule à partir des diagrammes de proche-retour [9]. Des travaux récents ont montré que l'entropie de Shannon est une mesure efficace pour détecter des séries temporelles chaotiques [10]. Nous allons utiliser cette mesure

pour identifier l'ensemble des agents perturbés. Un diagramme de proche-retour est obtenu d'une matrice $T \times T$ $R(i,j)$ construite comme suit. Soit x_k la série temporelle d'un agent donné X_k. Le principe consiste à comparer cette série temporelle à chaque pas de temps i à elle-même avec un délai j. T correspond à la taille de l'échantillon pris d'une série temporelle.

$$R_{X_k}(i,j) = \theta(\delta - \left|x_k^i - x_k^{i+j}\right|) \quad (4)$$

où $\theta(x_i)$ est la fonction de Heaviside : si $\delta - \left|x_k^i - x_k^{i+j}\right| > 0$ alors $\theta = 1$ sinon $\theta = 0$.
Si la différence $x_k^i - x_k^{i+j}$ est inférieure à un certain seuil δ, x_k^i et x_k^{i+j} sont dits récurrents et sont représentés par 1 dans la matrice. Sinon, ils sont considérés comme non récurrents et 0 est placé dans (i, j). Un exemple de diagramme de proche-retour est illustré sur la figure 4. Les segments noirs de ce diagramme correspondent aux points récurrents et inversement, les segments de couleur blanche sont les points non récurrents.
Le seuil δ permet de discrétiser les données et détermine la structure des diagrammes de proche-retour. Il est calculé comme suit :

$$\delta = (\left|min_{k=1..n}(x_k^t) - max_{k=1..n}(x_k^t)\right|)\alpha \quad (5)$$

où min et max sont respectivement les valeurs minimales et maximales calculées sur toutes les valeurs des séries temporelles des agents. Différentes règles [12] ont été proposées pour la définition du seuil δ dont la plus utilisée consiste à choisir un paramètre α égal à 10%. Les valeurs des séries temporelles de nos agents étant comprises entre 0 et 1, nous définissons par conséquent un seuil global pour tous les agents. δ aura donc les valeurs suivantes : 0,1, 0,01 et 0,001.
Des mesures ont été proposées pour interpréter les représentations graphiques des diagrammes de proche-retour en analyses statistiques. Parmi ces mesures, on cite le taux de récurrence et l'entropie de Shannon. Le taux de récurrence est défini comme la somme des points récurrents dans les diagrammes de proche-retour divisée par le nombre total d'éléments dans la matrice :

$$RR_{X_k} = \frac{1}{T^2}\sum_{i=1}^{T}\sum_{j=1}^{T} R_{X_k}(i,j) \quad (6)$$

L'entropie de Shannon est définie par :

$$S = -\sum_{g=1}^{H} P_g \log(P_g) \quad (7)$$

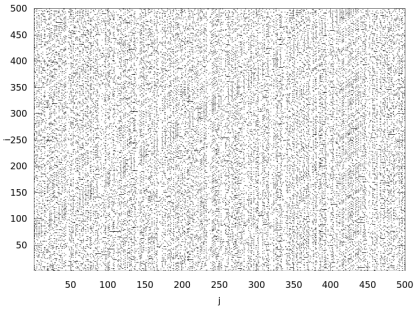


FIGURE 4 – Diagramme de proche-retour.

où P_g est la probabilité d'observer un segment non récurrent de longueur g, qui est le nombre de segments horizontaux de points non récurrents de longueur $g > 0$ divisé par le nombre total de segments de points non récurrents [11]. H est la séquence d'éléments récurrents la plus longue.
L'entropie est positive quand l'état de l'agent est perturbé, elle croît au fur et à mesure que la série temporelle est impactée par le chaos.

3.2 Identification de la source de la perturbation

L'entropie de Shannon permet d'identifier le groupe d'agents perturbés. Nous utilisons cet ensemble pour retrouver la source de la perturbation depuis un noeud perturbé quelconque. Rappelons que nous disposons uniquement des séries temporelles des agents perturbés et que nous ignorons les interactions entre eux. Par conséquent, afin de retrouver la source, on doit pouvoir calculer à partir des séries temporelles l'information chaotique transmise par les agents. Pour y parvenir, nous proposons d'utiliser l'entropie de transfert (TE) qui est une mesure de la théorie de l'information développée par Thomas Schreiber [13]. Elle permet d'identifier la direction de couplage entre deux systèmes dynamiques.
Soient X_l et X_k deux noeuds, l'information transmise de X_k vers X_l est donnée par :

$$\mathrm{TE}_{X_k \to X_l} = \sum_{i=1}^{T-1}\sum_{j=1}^{T} p(x_l^{i+1}, x_l^i, x_k^j) \log(\frac{p(x_l^{i+1}|x_l^i, x_k^j)}{p(x_l^{i+1}|x_l^i)}) \quad (8)$$

où $p(,)$ et $p(|)$ sont respectivement les probabilités conjointe et conditionnelle. Il s'agit

de rechercher toutes les combinaisons possibles de $(x_l^{i+1}, x_l^i, x_k^j)$ dans les deux diagrammes de proche-retour relatifs aux noeuds X_k et X_l.
Pour connaître les directions de couplage entre deux systèmes, nous avons besoin de calculer à la fois $\text{TE}_{X_l \to X_k}$ et $\text{TE}_{X_k \to X_l}$. Si $\text{TE}_{X_l \to X_k} > \text{TE}_{X_k \to X_l}$, alors X_l est le noeud qui a transmis une partie de son état à X_k.
Souvent, l'entropie de transfert est calculée sur des données discrétisées. Nous proposons ici d'utiliser les diagrammes de proche-retour présentés précédemment (cf. équation (4)) pour calculer cette mesure.
Ci-dessous, nous donnons les principales étapes du processus de la recherche de la source :

1. Choisir aléatoirement un noeud X_l de l'ensemble des noeuds chaotiques N_c.
2. Rechercher l'entropie de transfert la plus élevée entre ce noeud et tous les autres noeuds chaotiques.
3. L'entropie de transfert maximale nous donne le noeud X_k à traiter. Ce noeud est considéré comme celui qui influence le plus X_l.
   Refaire l'étape 2 en partant du noeud X_k
4. La boucle précédente se termine lorsque nous ne trouvons pas un noeud qui influence X_l. Par conséquent, ce noeud est considéré comme la source X_s de la perturbation.

L'algorithme de détection de la source de la perturbation dépend de deux paramètres qui sont respectivement, ϵ le coefficient de couplage assimilé à une "force d'interaction" entre les agents et δ le seuil de construction des diagrammes de proche-retour. Nous allons étudier l'influence de ces paramètres sur le taux de réussite de notre algorithme.

3.3 Simulations

Notre objectif consiste à détecter le groupe d'agents perturbés et à identifier la source de la perturbation dans le réseau d'interaction. Pour ce faire, nous avons simulé 10 graphes aléatoires construits à partir de systèmes multi-agents différents. Ces graphes ont des structures différentes et sont formés par 50 noeuds / agents ayant des indices de 0 à 49. Chaque graphe possède initialement un agent perturbé / chaotique qui est susceptible d'impacter d'autres agents en interaction avec lui d'une manière directe ou indirecte. Il est à noter que l'agent chaotique n'est jamais influencé par les autres agents avec qui il interagit.

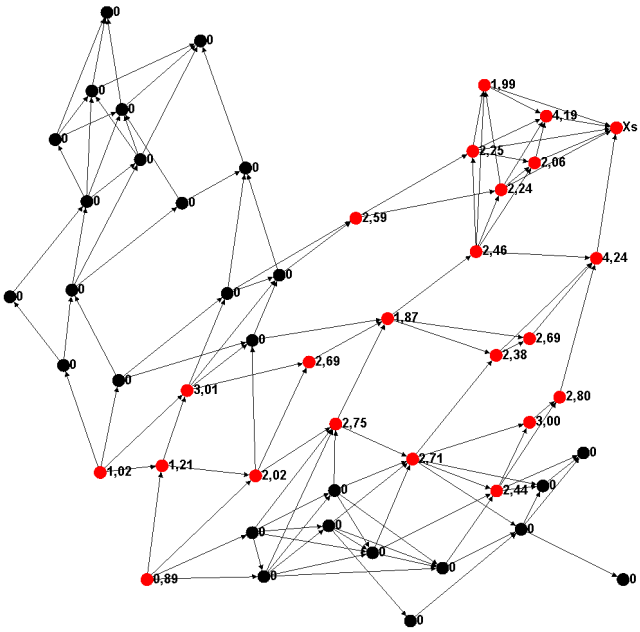


FIGURE 5 – Valeurs de l'entropie de Shannon. Les noeuds impactés par la perturbation sont colorés en rouge. Le noeud X_s sur la figure est la source de la propagation.

D'abord, nous avons utilisé le CMN pour générer les séries temporelles des noeuds. Dans ce modèle, nous connaissons les états des noeuds, notamment la source de la perturbation, ainsi que la structure des graphes.
Dans le problème inverse, nous cherchons à construire le CMN à partir uniquement des séries temporelles des noeuds. Dans les réseaux d'interaction, nous ne connaissons ni l'état des noeuds ni les interactions entre eux.
Pour tester l'algorithme de détection, nous allons étudier l'impact des paramètres ϵ et δ.
La force d'interaction ϵ est supposée influencer l'étendue de la propagation et le degré de la perturbation des états. Par ailleurs, un couplage fort conduit à un phénomène de synchronisation.
Pour confirmer ces hypothèses, nous avons varié la force de couplage ϵ dans nos simulations de 0,05 à 1.
Le seuil δ, quant à lui, détermine la structure des diagrammes de proche-retour. Plus le seuil est grand, plus les matrices contiennent des points récurrents. Il est donc intéressant d'élargir le seuil pour voir son impact sur la détection de la source. Par conséquent, δ aura les valeurs suivantes : 0,1, 0,01 et 0,001.
Pour la recherche de la source, nous avons choisi aléatoirement 5 agents de départ.
La figure 5 montre un exemple de graphe. Nous utiliserons cet exemple pour appliquer nos me-

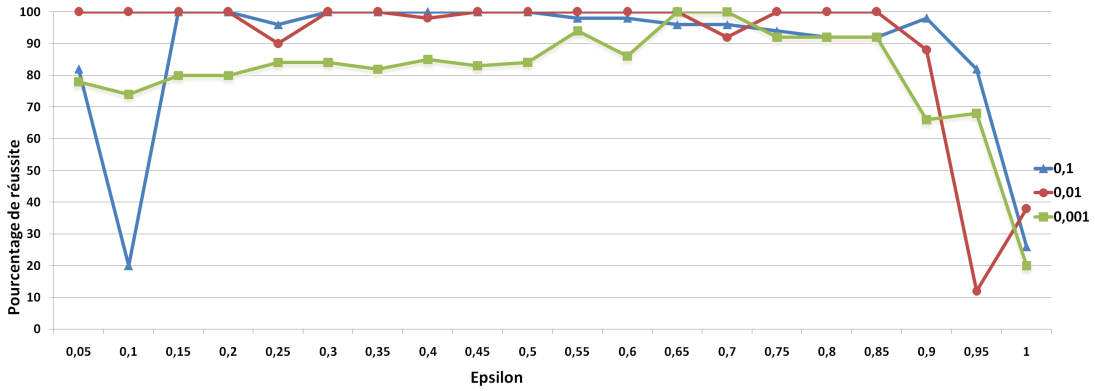


FIGURE 6 – Pourcentages de réussite de l'algorithme en fonction du seuil δ et pour toutes les valeurs d'epsilon.

sures. Puis, nous présenterons les résultats des simulations sur les 10 graphes pour en déduire les meilleures valeurs des paramètres de notre algorithme.

3.4 Résultats

Pour détecter l'ensemble des noeuds chaotiques, nous avons construit les diagrammes de proche-retour relatifs à chaque agent en utilisant différents seuils δ. Les séries temporelles étudiées sont obtenues après 1000 pas de de simulation. Ensuite, nous avons calculé l'entropie de Shannon (cf. équation (7)). Les agents ayant une entropie de Shannon positive sont considérés comme perturbés. Il est à noter que le nombre d'agents impactés croît quand on augmente la force de couplage ϵ.
Nous présentons sur la figure 5 les valeurs de l'entropie pour la configuration suivante : $\delta = 0,01$ et $\epsilon = 0,75$. Les noeuds chaotiques sont colorés en rouge. Ce résultat correspond à la configuration du CMN.
Ayant identifié le sous-groupe d'agents chaotiques, nous cherchons maintenant à retrouver la source de la propagation. Pour y parvenir, nous avons d'abord calculé l'entropie de transfert à partir des diagrammes de proche-retour entre chaque couple d'agents perturbés / chaotiques (cf. équations (8) et (4)).
Il y a 2 configurations de couplage entre deux noeuds :

– Si $\text{TE}_{X_l \to X_k}$ et $\text{TE}_{X_k \to X_l}$ sont des valeurs faibles, alors, X_l et X_k ne sont pas en interaction. On considère qu'une valeur est faible, si elle est inférieure à 0,0001.
– Si $\text{TE}_{X_l \to X_k} > \text{TE}_{X_k \to X_l}$, alors, X_l influence X_k et est plus proche de la source du chaos.

Pour expliquer notre algorithme, nous présentons quelques résultats dans le tableau 1. Les valeurs de l'entropie dans ce tableau nous permettent de retrouver la source en partant du noeud 48.
La première ligne ainsi que la première colonne du tableau correspondent aux indices de quelques agents perturbés. Une cellule correspond à $\text{TE}_{X_k \to X_l}$.
D'abord nous calculons les entropies de transfert entre ce noeud et les autres noeuds perturbés. L'entropie de transfert maximale trouvée est celle entre le noeud 48 et le noeud 3 : $\text{TE}_{3 \to 48}$. Ensuite, nous comparons $\text{TE}_{3 \to 48}$ et $\text{TE}_{48 \to 3}$. Si 3 n'influence pas le noeud 48, nous le supprimons de notre recherche et choisissons une autre entropie maximale. Cette comparaison nous amène à conclure que le noeud 3 influence le noeud 48. Nous recommençons la procédure en partant cette fois-ci du noeud 3. Le calcul de l'entropie de transfert nous mène au noeud 15 qui nous mène au noeud 2. Ce dernier nous amène à considérer le noeud 41. Nous calculons l'entropie entre le noeud 41 et les autres noeuds identifiés comme perturbés. Il s'avère qu'il n'y a aucun noeud qui influence l'agent 41. Par conséquent, cet agent est considéré comme la source de la perturbation dans le réseau et est représenté par le label X_s sur la figure 5.

Impact des paramètres δ et ϵ sur la détection de la source. Pour les 10 graphes étudiés, nous avons repris les valeurs de δ présentées dans la section Simulations et nous avons varié la valeur du coefficient de couplage de 0,05 à 1. Nous rapportons les pourcentages de réussite de notre algorithme sur la figure 6.

	2	**3**	**4**	**12**	**15**	**29**	**35**	**41**	**48**
2		0,015	$2,72e^{-4}$	$9,33e^{-5}$	0,044	$6,95e^{-9}$	0,016	$5,32e^{-4}$	$8,28e^{-4}$
3	0,0012		$1,08e^{-4}$	0,0014	$7,30e^{-4}$	0,020	0,004	$8,40e^{-5}$	0,35
4	0,011	0,013		$7,97e^{-5}$	0,037	$5,05e^{-5}$	0,016	$8,65e^{-4}$	$4,78e^{-4}$
12	$1,47e^{-6}$	$1,42e^{-5}$	$2,54e^{-6}$		$5,69e^{-5}$	$8,97e^{-4}$	$2,54e^{-5}$	$1,18e^{-7}$	$3,35e^{-4}$
15	0,001	0,045	$1,04e^{-4}$	$4,39e^{-4}$		0,003	0,012	$9,61e^{-5}$	0,071
29	$1,39e^{-5}$	$1,18e^{-4}$	$1,45e^{-5}$	0,004	$5,84e^{-5}$		$4,45e^{-4}$	$2,08e^{-5}$	$1,40e^{-4}$
35	$2,58e^{-4}$	0,009	$4,29e^{-6}$	$4,40e^{-4}$	$9,27e^{-5}$	0,008		$6,36e^{-6}$	0,143
41	0,014	0,010	0,0035	$6,11e^{-5}$	0,023	$3,97e^{-6}$	0,013		$3,58e^{-4}$
48	$9,54e^{-6}$	$3,05e^{-4}$	$3,11e^{-6}$	0,0027	$1,02e^{-4}$	0,341	0,002	$4,18e^{-7}$	

TABLE 1 – Valeurs de l'entropie de transfert. La première ligne et la première colonne contiennent les indices des agents perturbés. L'entropie de transfert maximale calculée à chaque étape du processus de détection est colorée en rouge.

La figure 6 montre que le pourcentage de réussite diminue quand on élargit le seuil (δ = 0,001). Ce résultat s'explique par le fait que lorsqu'on agrandit le seuil (on diminue sa valeur), on fait apparaître plus de segments récurrents dans les matrices de proche-retour. Or, nous cherchons à détecter l'influence chaotique des noeuds ce qui correspond aux segments non récurrents. Ce résultat est confirmé dans le tableau 2 où on a les moyennes des pourcentages d'erreur E_1 de l'algorithme, en fonction du paramètre δ, calculés pour toutes les valeurs d'epsilon.

δ	0,1	0,01	0,001
E_1	12	9	18

TABLE 2 – Pourcentages d'erreur selon la valeur du seuil δ.

Concernant le paramètre ϵ, à partir d'un couplage fort, c'est à dire un coefficient de couplage supérieur à 0,85, l'algorithme n'arrive plus à détecter la source. Nous allons tenter d'expliquer cet échec en se fondant sur les taux de récurrence des noeuds identifiés par erreur comme sources du chaos.
La figure 7 montre l'évolution des taux de récurrence en fonction du paramètre ϵ de l'agent source 41 et d'un autre agent 4 qui est en interaction directe avec lui. Quand le couplage est fort, les taux de récurrence deviennent presque identiques. Cela s'explique par un phénomène de synchronisation qui permet à deux états initialement différents de devenir identiques.

En nous basant sur les résultats de ces études, nous pouvons conclure que la meilleure calibration des paramètres de notre algorithme correspond à un seuil δ égal à 0,01 et une force d'interaction ϵ inférieure à 0,9.

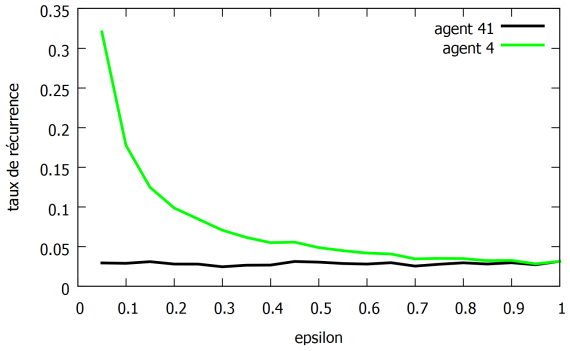


FIGURE 7 – Evolution des taux de récurrence des agents 4 et 41 en fonction de la force de couplage.

Impact de la distance sur la détection de la source. Les agents détectés par erreur comme noeuds sources sont majoritairement en interaction avec l'agent chaotique. Nous cherchons à connaître l'impact de la distance des noeuds par rapport à l'agent source sur le taux d'erreur E_2 de notre algorithme. Ce taux devient alors proportionnel à la distance et est calculé comme suit :

$$E_2 = \frac{1}{D}\sum_{k=1}^{M} d_k \qquad (9)$$

où d_k est la distance qui sépare un noeud X_k du noeud chaotique, X_k, étant un noeud identifié par erreur comme source. M représente l'ensemble des noeuds identifiés par erreur comme sources par l'algorithme. D est la distance maxi-

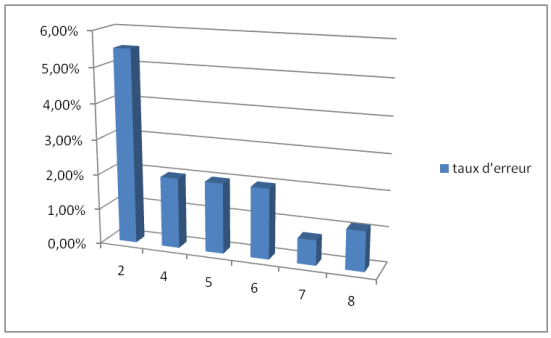


FIGURE 8 – Répartition du pourcentage d'erreur en fonction de D pour δ égal à 0,01.

male qui existe entre un noeud perturbé et l'agent source, elle est donnée par :

$$D = \max\{d(X_s, k) | k \in N_c\} \qquad (10)$$

où X_s est le noeud source et N_c représente l'ensemble des noeuds perturbés.

δ	0,1	0,01	0,001
E_2	3,1	2,1	5

TABLE 3 – Pourcentages d'erreur proportionnels à la distance au noeud source.

Le tableau 3 montre que la position des noeuds par rapport au noeud source impacte les résultats de notre algorithme.
Nous avons étudié l'impact de la distance maximale D sur la détection de la source dans le réseau d'interaction. Nous présentons sur la figure 8 les moyennes des pourcentages d'erreur en fonction de la distance D pour les 10 graphes étudiés. Ces pourcentages sont calculés pour un δ égal à 0,01 et pour toutes les valeurs d'epsilon. Quand l'étendue de la propagation de la perturbation est importante, on a moins d'erreur de détection.

3.5 Discussion

Le calcul de l'entropie de Shannon à partir des diagrammes de proche-retour a permis de détecter le groupe d'agents perturbés. Quand on augmente la force d'interaction, le nombre d'agents impactés par la perturbation augmente également.
En ce qui concerne la source de la perturbation, nous avons proposé un algorithme de détection qui dépend de deux paramètres : δ pour la construction des diagrammes de proche-retour et ϵ la force d'interaction.
Une première étude a montré que lorsqu'on élargit le seuil δ, le taux d'erreur relatif à la détection de la source augmente.
Nous avons également étudié l'impact du paramètre ϵ sur la détection de la source. Quand on a un couplage fort, on n'arrive plus à détecter la source. Ceci est dû au phénomène de synchronisation qui permet à deux états initialement différents de devenir identiques.
Enfin, nous avons étudié l'impact de la distance des noeuds identifiés par erreur comme sources de la perturbation. La position de ces noeuds influence la recherche de la source puisque les agents se trouvent majoritairement à proximité de l'agent chaotique. Aussi, plus l'étendue de la propagation est importante, moins on a d'erreurs de détection.
Toutes ces études nous ont permis de trouver la meilleure calibration des paramètres de notre algorithme pour réduire les erreurs de détection : il faut fixer la valeur de δ à 0,01 et assigner à ϵ une valeur inférieure à 0,9.

4 Intérêt de l'étude et perspectives

Nous envisageons d'appliquer notre modèle de propagation de perturbations au problème de la propagation du stress ou de l'anxiété dans les situations d'urgence.
Des psychiatres et neurologues se sont intéressés à l'anxiété et au stress en étudiant la dynamique des rythmes cardiaques et des fréquences respiratoires d'un groupe de patients souffrant de trouble de panique (panic disorder). Ils ont conclu que ces dynamiques étaient chaotiques dû à l'anxiété [1]. D'un autre côté, Fast et al. [6] considèrent l'anxiété comme une contagion émotionnelle qui se propage d'un individu à un autre au sein du réseau social.
En se basant sur ces études, nous pouvons assimiler l'anxiété aux données physiologiques chaotiques des personnes qu'on obtiendrait grâce aux capteurs sur ces derniers. L'état perturbé des agents correspondrait donc aux données physiologiques dont la dynamique est chaotique.
De même, la grandeur d'état de notre modèle pourrait correspondre au paramètre de nervosité du modèle des forces sociales de Helbing [7]. Ce modèle particulier prend en compte les situations d'urgence dans lesquelles la transition entre un comportement "rationnel" et un comportement "irrationnel" de panique est gérée par un seul paramètre appelé nervosité. Ce para-

mètre agit sur les décisions de déplacement du piéton.

5 Conclusion

Dans cet article, nous avons présenté notre travail de recherche qui consiste à étudier la propagation d'une perturbation dans un réseau d'interaction formé par un système multi-agent basé sur un modèle de propagation de type CMN.
L'entropie de Shannon, calculée sur les diagrammes de proche-retour, s'est avérée une mesure efficace pour la détection du groupe d'agents perturbés. Nous nous sommes aussi intéressés à la détection de l'agent source de la perturbation en proposant un algorithme qui utilise l'entropie de transfert calculée sur les diagrammes de proche-retour pour mesurer l'information chaotique transmise d'un agent vers un autre.
Les simulations effectuées nous ont permis de trouver les meilleures valeurs des paramètres de notre algorithme dans le but de minimiser le taux d'erreur de la détection de la source.
Une des perspectives de nos travaux est d'étudier l'effet de la position et du nombre des noeuds sources ainsi que des topologies sur la propagation de la perturbation dans les réseaux d'interaction.

Références

[1] Caldirola, D., Bellodi, L., Caumo, A., Migliarese, G., and Perna, G. Approximate entropy of respiratory patterns in panic disorder. *The American journal of psychiatry 161*, 1 (2004), 79–87.

[2] Christley, R. M., Pinchbeck, G. L., Bowers, R. G., Clancy, D., French, N. P., Bennett, R., and Turner, J. Infection in Social Networks : Using Network Analysis to Identify High-Risk Individuals. *Am J Epidemiol 162*, 10 (Sept. 2005), 1024–1031.

[3] Daly, E. M., and Haahr, M. Social network analysis for routing in disconnected delay-tolerant manets. In *Proceedings of the 8th ACM International Symposium on Mobile Ad Hoc Networking and Computing* (New York, NY, USA, 2007), MobiHoc '07, ACM, pp. 32–40.

[4] Deffuant, G., Neau, D., Amblard, F., and Weisbuch, G. Mixing beliefs among interacting agents. *Adv. Complex Syst. 3*, 1–4 (2000), 87–98.

[5] Dong, W., Heller, K. A., and Pentland, A. Modeling infection with multi-agent dynamics. In *Social Computing, Behavioral - Cultural Modeling and Prediction - 5th International Conference, USA* (2012), pp. 172–179.

[6] Fast, S. M., González, M. C., Wilson, J. M., and Markuzon, N. Modelling the propagation of social response during a disease outbreak. *Journal of The Royal Society Interface 12*, 104 (2015).

[7] Helbing, D., Farkas, I., Molnàr, P., and Vicsek, T. Simulation of pedestrian crowds in normal and evacuation situations. In *Pedestrian and Evacuation Dynamics* (Berlin, 2002), M. Schreckenberg and S. D. Sharma, Eds., Springer, pp. 21–58.

[8] Koiller, J., and Young, L.-S. Coupled map networks. *Nonlinearity 23*, 5 (2010), 1121.

[9] Mindlin, G. B., and Gilmore, R. Topological analysis and synthesis of chaotic time series. *Phys. D 58*, 1-4 (Sept. 1992), 229–242.

[10] Rabai, H., Charrier, R., and Bertelle, C. Close returns plots for detecting a chaotic source in an interaction network. In *ALIFE 14 : The Fourteenth Conference on the Synthesis and Simulation of Living Systems* (2014), vol. 14, pp. 726–733.

[11] Rabarimanantsoa, H., Achour, L., Letellier, C., Cuvelier, a., and Muir, J.-F. Recurrence plots and Shannon entropy for a dynamical analysis of asynchronisms in noninvasive mechanical ventilation. *Chaos (Woodbury, N.Y.) 17*, 1 (Mar. 2007), 013115.

[12] Schinkel, S., Dimigen, O., and Marwan, N. Selection of recurrence threshold for signal detection. *The European Physical Journal Special Topics 164*, 1 (2008), 45–53.

[13] Schreiber, T. Measuring information transfer. *Physical review letters 85*, 2 (2000), 461–464.

[14] Tang, M., Mao, X., Guessoum, Z., and Zhou, H. Rumor Diffusion in an Interests-Based Dynamic Social Network. *The scientific world journal 2013* (2013).

Approche Anytime pour l'ajustement de l'incrément dans les enchères multicritères automatisées

Imène Brigui-Chtioui[a] imene.brigui-chtioui@isg.fr
Philippe Caillou[b] caillou@lri.fr
Suzanne Pinson[c] Suzanne.pinson@dauphine.fr

[a]Institut Supérieur de Gestion, GRIISG,
147, Avenue Victor Hugo, 75116 Paris

[b]Université Paris Sud, LRI, INRIA, TAO Team,
Bat 660, Gif-Sur-Yvette

[c]Université Paris-Dauphine, Lamsade,
Place du maréchal de Lattre de Tassigny, 75775 Paris

Résumé

Dans cet article, nous proposons des stratégies muticritères pour la formulation de contre-propositions dans le cadre d'enchères anglaises multicritères inversées. Dans ce type d'enchères, un agent acheteur négocie avec plusieurs agents vendeurs un accord portant sur un produit unique. Le modèle de préférences de l'agent acheteur se fonde sur des points de référence qui représentent, d'une part, les valeurs souhaitées et d'autre part les valeurs minimales acceptables sur chaque critère. Afin d'assurer une évolution optimale du processus de négociation, les enchères anglaises font souvent appel à un incrément qui représente la surenchère minimale à respecter par toute proposition comparée à la meilleure proposition courante. Généralement. l'incrément est fixé avant le début des enchères et demeure fixe tout au long du processus. Notre objectif est de proposer un mécanisme d'ajustement de l'incrément au cours de la négociation : nous proposons un algorithme anytime *s'appuyant sur la méthode de lissage exponentiel qui adapte l'incrément au contexte de l'enchère, en considérant le nombre d'agents vendeurs encore en lice et le temps restant à chaque itération. Par ailleurs, nous présentons et démontrons plusieurs propriétés de l'algorithme proposé et nous validons l'approche proposée par des résultats expérimentaux.*

Mots-clés : *algorithme anytime, enchère multicritère, enchère inversée*

Abstract

In this paper we propose buyer counterproposal strategies for conducting automated Reverse auctions based on a multicriteria model. In this type of auction, a buyer agent (or auctioneer) negotiates with several seller agents (or bidders). To insure process evolution, automated auctions design often considers a bid increment that represents the minimal amount that a bidder must improve on the current best bid. Generally, the bid increment is fixed before the beginning of the auction and kept invariant during the process. This article aims at adjusting the bid increment as the auction process goes on. For this purpose, we propose buyer counterproposal strategies in order to implement anytime multicriteria auction insuring an acceptable solution at any given time. For this purpose, we refer to the auction context based on the number of remaining sellers or the remaining time at each process step. Finally, we provide results comparing fixed-increment strategy to our proposed strategies on the basis of a variation of different auction settings in order to evaluate the anytime properties relevance to our study context. We show that the proposed strategies provide better early results or better results than fixed strategies, depending on the fixed parameters.

Keywords : *Anytime algorithms, Multicriteria auctions, Counterproposal Strategies*

1 Introduction

Considérons une compagnie de construction aéronautique. Suite à une hausse de la demande, elle a besoin de dix réacteurs supplémentaires. Pour les obtenir dans les meilleures conditions (plus bas prix et meilleure qualité), elle peut organiser des enchères inversées. Il s'agit d'une situation classique d'enchères, bien étudiée dans la littérature. Considérons à présent que la compagnie a besoin de ces réacteurs dès que les avions ont fini d'être assemblés (pour les vendre le plus vite possible), mais qu'elle ne sait pas exactement de combien de temps elle dispose. Une solution pourrait être de réaliser des enchères rapides en temps limité. Mais, dans ce

cas, elle risquerait de ne pas obtenir le meilleur prix/qualité à cause de contraintes de temps (et donc du nombre limité de tours d'enchères). Une autre solution consiste à utiliser un mécanisme d'enchères *anytime* tel que la solution soit "satisfaisante" rapidement, mais continue a s'améliorer progressivement au cours du temps. C'est ce type d'enchères que nous considérons.

De façon plus générale, un mécanisme d'enchères respecte un protocole de négociation entre un participant ou commissaire-priseur qui dirige les enchères et des enchérisseurs qui sont en compétition pour remporter la transaction. Ce type de mécanisme a un très grand nombre d'applications possibles et a fait l'objet d'un grand nombre d'études, aussi bien au niveau théorique que pratique [10]. Il a l'avantage de proposer des protocoles détaillés, simples et avec des règles claires. Leur développement et l'analyse des propriétés du résultat final se situe à la frontière de l'aide à la décision, de l'intelligence artificielle et de la théorie des jeux computationelle [8]. De nombreux types d'enchères ont été étudiés dans la littérature [13, 14]. Nous nous intéressons ici aux enchères inversées pour lesquelles il y a un acheteur pour plusieurs vendeurs. L'étude de ce type d'enchères s'est particulièrement développé avec l'essor du commerce en ligne, afin de proposer des outils permettant d'assister un acheteur humain cherchant à mettre en concurrence plusieurs vendeurs potentiels. À notre connaissance, aucune stratégie *anytime* n'a toutefois été proposée et étudiée, telle que nous la proposons ici.

Lors d'une enchère anglaise inversée, à chaque étape, l'acheteur annonce une contrainte que chaque enchérisseur doit respecter pour rester dans la négociation. Les vendeurs soumettent alors leurs offres qui respectent cette contrainte ou abandonnent. À la fin de chaque itération, le meilleur vendeur est mis en attente pendant toute la durée de l'itération suivante (avec sa proposition comme offre minimale), et ainsi de suite jusqu'à ce que tous les vendeurs abandonnent (sauf celui qui a fait la dernière meilleure offre).

Les protocoles d'enchères inversées utilisent généralement un incrément minimum qui correspond au montant minimal qu'un vendeur doit proposer comme amélioration de la meilleure offre courante pour rester dans la négociation. La majorité des protocoles existants [1, 4, 6] utilisent soit un incrément fixe décidé à l'avance qui reste fixe durant toute la négociation [1, 10], soit laissent le vendeur complètement libre du montant qu'il souhaite ajouter (ce qui peut aboutir à un supplément infiniment petit) [5, 9]. Dans le premier cas, l'incrément minimum est fixe, et ne dépend donc ni de la meilleure offre courante, ni du temps restant. Sachant que l'on veut se situer ici dans le cadre d'une enchère à durée limitée et incertaine, un incrément faible risque de faire évoluer les enchères trop lentement au début. À l'inverse, un incrément élevé risque de faire rater à l'acheteur des propositions qui auraient été intéressantes au vu de ses préférences. Dans le second cas de l'incrément libre, le risque se situe dans le fait que les vendeurs, se trouvant libres d'enchérir d'un montant infiniment faible (ce qui est à la fois une solution intuitivement logique et une stratégie dominante), fassent augmenter de manière très importante le nombre d'itérations nécessaires pour éliminer tous les vendeurs sauf un. La solution obtenue dans ce cas en temps fini risque d'être très éloignée de la solution optimale.

Pour pallier ces limites, nous proposons d'utiliser un incrément minimum variable en utilisant un lissage exponentiel en fonction du temps et du nombre de participants restants en négociation. Ce mécanisme est particulièrement adapté pour le cas d'enchères *anytime* (c'est à dire d'enchères qui peuvent être interrompues à n'importe quel moment, la meilleure enchère courante remportant la vente). Pour étudier la qualité obtenue par un protocole *anytime* au cours de son déroulement, il existe des méthodes d'évaluation standardisées [21]. Dans le contexte particulier des enchères automatisées, il existe de nombreuses applications à ce type d'enchères *anytime* : en plus de notre exemple introductif, on peut imaginer la sélection d'un ordinateur sur une grille/cloud pour une tâche exécutable uniquement une fois que la précédente est terminée. Une enchère inversée *anytime* dans ce cas permettrait de mettre en concurrence des offres (capacités de calcul/contraintes) pour trouver progressivement la moins chère, tout en souhaitant commencer le calcul dès que la tâche courante est achevée.

Dans cet article, nous proposons une méthode *anytime* pour ajuster l'incrément minimum d'une enchère anglaise inversée tout au long du processus de négociation. Après avoir complété l'état de l'art en section 2, nous détaillerons et analyserons le protocoles en section 3 et la méthode de choix des contre-propositions en section 4. Une étude expérimentale est proposée en section 5 afin de comparer l'algorithme

au cas d'incrément fixe généralement utilisé.

2 Etat de l'art

Plusieurs travaux s'intéressent à la négociation multicritère [6, 9, 16, 11] et particulièrement aux enchères anglaises multicritères [4, 5]. Les enchères multicritères représentent une extension aux standards liés à la théorie des enchères [16]. Leur objectif est d'élargir la négociation à tous les critères qui caractérisent un produit et de ne pas la restreindre à un critère unique : le prix. Nombre d'approches sont proposées pour permettre ce type de négociation sur plusieurs critères. Une distinction peut être réalisée entre les approches qui n'utilisent pas de fonction de valorisation globale (ou fonction de scores) et celles qui s'appuient sur une fonction de scores explicite. Teich et. al [19] se situent par exemple dans la première catégorie et utilisent une méthode de "leap-frog" dans laquelle un enchérisseur est appelé à apporter une amélioration de la meilleure proposition courante sur au moins un critère tout en s'assurant de ne pas diminuer la qualité sur tous les autres critère. Dans la seconde catégorie, une fonction de valorisation permet d'apprécier l'utilité globale d'une proposition donnée [2]. Ainsi, tout au long du processus d'enchères, toutes les valeurs des différents critères peuvent évoluer. L'utilité est calculée en utilisant un modèle d'agrégation qui exprime l'évaluation globale d'une proposition en se référant aux valeurs de tous les critères qui la composent.

Dans les enchères inversées, le processus d'enchères est lancé lorsque l'agent acheteur envoie une demande concernant un produit désiré à tous les agents vendeurs intéressés. Les vendeurs évaluent la demande et envoient des propositions acceptables ou abandonnent les enchères. Dès que l'acheteur reçoit toutes les propositions, il sélectionne la meilleure, met le vendeur correspondant en attente et relance les autres vendeurs avec une nouvelle contre-proposition en se basant sur la meilleure proposition actuelle. Le processus évolue jusqu'à ce que tous les vendeurs abandonnent, à l'exception du meilleur de l'itération précédente [7]. La transmission de la fonction de valorisation de l'acheteur aux vendeurs peut également avoir un impact sur le résultat final (des expérimentations avec des étudiants ont ainsi montré qu'une telle transmission augmente la qualité du résultat pour le vendeur et pour l'acheteur [18]).

Dans un mécanisme itératif d'enchères, le processus est basé sur un nombre d'itérations donné. Le commissaire-priseur formule des contre-propositions aux enchérisseurs pour les aider à décider comment améliorer leurs propositions (une possibilité serait de proposer une nouvelle fonction de valorisation à chaque itération comme dans [3]). Dans le cas des enchères à base de prix, seul le prix peut être amélioré par le vendeur (une situation gagnant-perdant). Les enchères multicritères permettent d'améliorer l'utilité globale de la proposition en considérant tous les critères négociables (une alternative est le vote parallèle sur les différentes combinaisons de critères [17]). Dans cet article, nous nous plaçons dans le cadre des enchères anglaises multicritères inversées, et nous focalisons l'essentiel de notre contribution sur la formulation de la contre-proposition par l'acheteur.

3 Mécanisme d'enchères

Notre mécanisme d'enchères est fondé sur un protocole d'enchères anglaises inversées et un modèle d'enchères multicritères à points de référence.

3.1 Protocole d'enchères

Dans cette section, nous présentons le protocole d'enchères anglaises multicritères inversées. Le protocole de communication spécifie les actions valides de chaque agent dans un contexte donné. Il spécifie ainsi comment une négociation devrait être conduite et définit les règles qui régissent chaque séquence d'échanges de messages au cours des enchères [8].

Le graphe d'états de la Fig. 1 illustre le processus d'enchères. Au début du processus d'enchères, l'agent acheteur envoie un message $CallForPropose$ qui renferme le détail de ses préférences sur le produit en négociation. À chaque itération, les agents vendeurs restants envoient des propositions B^t que l'agent acheteur évalue. L'agent acheteur sélectionne parmi les propositions reçues la meilleure, met l'agent vendeur correspondant en attente, définit la valeur de l'itération suivante et la transmet aux vendeurs qui restent en compétition. Chaque agent vendeur a deux choix : faire une proposition qui respecte la contrainte imposée par l'agent acheteur ou abandonner. Notons que les agents vendeurs n'ont pas connaissance des propositions faites par les autres vendeurs en compétition.

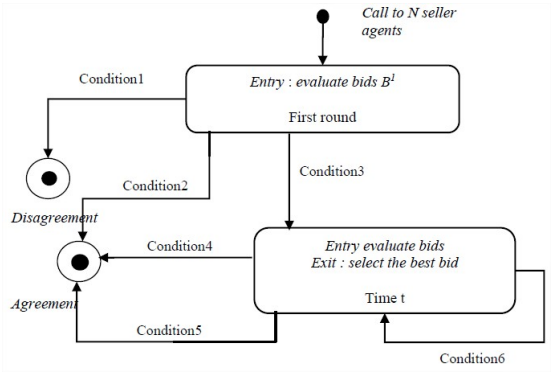


FIGURE 1 – Diagramme UML d'états-transitions modélisant le processus denchères anglaises inversées

Condition1 [$B^1 = \varnothing$] : Pas de propositions reçues à l'itération 1

Condition2 [$|B^1| = 1$]/ $winner := best^1$: Une seule proposition est reçue à l'itération 1, elle est considérée comme la proposition gagnante

Condition3 [$|B^1| > 1$] : Plusieurs propositions reçues à l'itération 1

Condition4 [$|B^t| = \varnothing$] / $winner := best^{t-1}$: Pas de propositions reçues à l'itération t, la proposition gagnante est la meilleure proposition à l'itération $t-1$

Condition5 [Elapsed time] : Temps consacré aux enchères écoulé

Condition6 [$|B^t| > 1$] /Call to ($|B^t| - 1$) agents vendeurs : Plus d'une proposition sont reçues à l'itération t.

3.2 Modèle d'enchères

Dans cette section, nous présentons le modèle d'enchères multicritères développé dans [4]. Nous commençons par proposer quelques notations préliminaires, puis nous présentons le modèle de préférence et le modèle d'agrégation.

- Soit p, le nombre d'attributs ;
- $D = D_1 \times ... \times D_p$, l'espace de décision où D_j désigne le domaine de valeurs de l'attribut j ($j = 1, ..., p$) ;
- $C = C_1 \times ... \times C_p$, l'espace de critères [1] ;
- v_j, la fonction de scores pour l'attribut j définie de D_j vers C_j ;
- $x = (x_1 \times ... \times x_p) \in D$ désigne une proposition, $b = (b_1 \times ... \times b_p) \in C$ où $b_j = v_j(x_j)$ désigne la proposition évaluée sur tous les critères $j = 1, ..., P$
- $\hat{t}$, le temps maximal consacré aux enchères

Modèle de préférence : le modèle de préférence est fondé sur deux points de référence :

- Le point d'aspiration, noté $a = (a_1 \times ... \times a_p)$ où $a_j = v_j(dv_j)$ désignent les niveaux d'aspiration et où $dv_j \in D_j$ désigne la valeur désirée par l'agent acheteur sur le critère j. Le point d'aspiration est gardé privé tout au long du processus d'enchères, l'agent acheteur ne le communique pas aux agents vendeurs en négociation.
- Le point d'exigence, noté $r = (r_1 \times ... \times r_p)$ où $r_j = v_j(mv_j)$ désignent les niveaux d'exigence, et où $mv_j \in D_j$ est la valeur minimale requise sur le critère j. Le point d'exigence peut être partiellement défini (uniquement sur quelques critères) et il est public à tous les agents vendeurs participants.

Modèle d'agrégation : le modèle d'agrégation détermine l'utilité associée à une proposition en considérant un point d'aspiration donné. Il est défini par une déviation par rapport au point d'aspiration. Cette déviation mesure la différence maximale entre les niveaux d'aspiration et les valeurs de la proposition d'un agent vendeur sur chaque critère. Le modèle calcule les différences entre les valeurs du point d'aspiration et les valeurs de la proposition sur chaque critère et garde comme résultat le plus grand écart observé. La fonction max est choisie pour s'assurer qu'un mauvais score sur un critère ne pourra pas être compensé par de bons scores sur d'autres critères. L'équation (1) présente l'utilité de la proposition b. Elle mesure le maximum des différences $(a_j - b_j)$ entre une proposition b et le point d'aspiration a sur chaque critère j.

$$U_a(b) = \max_{j=1,...,p} a_j - b_j \qquad (1)$$

La relation de préférence basée sur l'utilité présentée ci-dessus est donnée par l'équation (2).

$$b \succ_a b' \Leftrightarrow U_a(b) < U_a(b') \qquad (2)$$

Conformément à cette relation de préférence, l'agent acheteur cherche à minimiser la déviation au point d'aspiration. Une proposition b est préférée à une proposition b' si son utilité est inférieure à celle de la proposition b'. [2]

1. pour les calculs, nous allons considérer par la suite des espaces de critères identique $C_j = [0; 100]$

2. Cette mesure pourrait être appelée *désutilité* vu que l'acheteur cherche à la minimiser

4 Définition *anytime* de la contre-proposition

Dans le modèle d'enchères proposé dans [4], la contrainte à une itération $t+1$ est donnée par la relation suivante :

$$\forall t, U_a(b^{t+1}) \leq U_a(best^t) - \epsilon \quad (3)$$

$(best^t)$ désigne la meilleure proposition à l'itération t. L'incrément ϵ est absolu. Il demeure fixe tout au long du processus d'enchères et assure l'évolution des enchères. Il est appliqué à l'utilité de la meilleure proposition courante $(best^t)$. Cette procédure de définition de la contre-proposition vérifie les propriétés des enchères ascendantes automatisées qui ont été avancées dans [4].

Nous proposons un algorithme *anytime* pour la définition de la contre-proposition qui assure une évolution de l'incrément et qui satisfait la règle "beat-the-quote" introduite dans [20] qui impose la contrainte suivante :

$$\forall t, U_a(b^{t+1}) < U_a(best^t) \quad (4)$$

Notre approche se base sur la méthode du lissage exponentiel qui est une méthode classique utilisée en économie pour estimer le niveau de demande sur la base d'observations de la demande courante et des précédentes. Afin d'utiliser cette méthode, nous considérons deux observations à une itération donnée t : le nombre d'agents vendeurs encore en négociation $|B^t|$ et le temps restant $(\hat{t} - t)$. En se basant sur ces observations, nous formulons deux propositions pour l'ajustement de l'incrément ϵ^t.

DEFINITION 1 En appliquant la méthode du lissage exponentiel sur le nombre de vendeurs restants $|B^t|$, l'incrément ϵ^t est calculé comme suit :

$$\epsilon^t = (\alpha\epsilon^{t-1}) + ((1-\alpha)v(|B^t|)) \quad (5)$$

avec :

v : la fonction de normalisation,

ϵ^t : l'incrément à l'itération t avec $\epsilon^0 = 0$

$\alpha \in [0, 1]$: le facteur de lissage.

Les propositions de la période $t+1$ doivent respecter la contrainte :

$$\forall t, U_a(b^{t+1}) \leq U_a(best^t) - ((\alpha\epsilon^{t-1} + ((1-\alpha)v(|B^t|)) \quad (6)$$

DEFINITION 2. Nous proposons une nouvelle technique d'ajustement de l'incrément basée sur le temps restant comme suit :

$$\epsilon^t = (\alpha\epsilon^{t-1}) + ((1-\alpha)v(\hat{t} - t)) \quad (7)$$

Les propositions de la période $t+1$ doivent respecter la contrainte :

$$\forall t, U_a(b^{t+1}) \leq U_a(best^t) - ((\alpha\epsilon^{t-1} + ((1-\alpha)v(\hat{t} - t)) \quad (8)$$

Performance espérée : afin de permettre le contrôle des algorithmes *anytime*, l'amélioration de leur performance doit pouvoir être résumée quantitativement. La performance espérée ξ est définie par l'évaluation de l'accord final. Nous considérons comme dans [15] qu'un désaccord est le pire des résultats. Rappelons qu'un désaccord est constaté lorsque la condition $|B^1| = 0$ est vérifiée, ce qui signifie qu'aucun agent vendeur n'a pu répondre à la contrainte envoyée par l'agent acheteur à l'itération 1 (les niveaux d'exigence de l'agent acheteur étant trop hauts). Ainsi, nous établissons ce qui suit :

$$\xi(best) > \xi(Disagreement), \forall best \in B \quad (9)$$

Dans ce qui suit, nous présentons et démontrons deux propriétés satisfaites par l'algorithme proposé [21].

PROPRIETE 1. **Performance mesurable**. Cette propriété établit que la qualité d'un résultat peut être définie de manière précise. Nous proposons de mesurer cette performance en utilisant l'utilité de la proposition gagnante. Afin de maximiser la performance, nous minimisons l'utilité de la meilleure proposition.

PREUVE. Considérons deux accords donnés g' et g'', et $best'$ et $best''$ les propositions gagnantes correspondantes, nous établissons que :

$$\xi(g') > \xi(g'') \text{ ssi } U_a(best') < U_a(best'') \quad (10)$$

D'où, le mécanisme *anytime* proposé satisfait la propriété de performance mesurable.

PROPRIETE 2. **Monotonicité**. Cette propriété établit que la qualité d'un résultat est une fonction croissante du temps (ou nombre de vendeurs restants) et de l'incrément.

DEFINITION 3. [3]

$$\forall t, U_a(best^{t+1}) \leq U_a(best^t) - \epsilon^t \quad (11)$$

3. La preuve est la même en utilisant le nombre de vendeurs restants à la place du temps restant

Une condition suffisante pour la monotonicité est de démonter que : $\epsilon^t \geq 0$, donc $((\alpha\epsilon^{t-1} + ((1-\alpha)v(\hat{t}-t))) \geq 0$. Ceci est impliqué par $(\alpha\epsilon^{t-1} \geq 0)$ et $((1-\alpha)v(\hat{t}-t)) \geq 0$ ce qui peut être facilement déduit de $0 < \alpha < 1$ et $v(\hat{t}-t) \geq 0$. D'où, l'algorithme *anytime* proposé satisfait la monotonicité.

5 Etude expérimentale

Afin de comparer le comportement de notre modèle à une méthode utilisant un incrément fixe, nous présentons ici une étude expérimentale. Nous commençons par présenter les conditions expérimentales, les résultats d'une enchère standard, puis nous discutons l'impact des paramètres. Les simulations ont été réalisées à l'aide de la plateforme GAMA [12].

5.1 Conditions expérimentales

Pour chaque enchère simulée, nous considérons un agent acheteur et $NbBidders$ agents vendeurs, qui vont interagir durant au plus $\hat{t}$ itérations.

Agent acheteur

Nous comparons six comportements pour l'agent acheteur : quatre stratégies à incrément fixe et les deux stratégies à incrément évolutif présentées dans la section précédente.

- $FixU/N$, $Fix3U/N$, $Fix6U/N$, $Fix9U/N$: ϵ^t est fixe et dépend uniquement de l'écart entre point d'exigence et point d'aspiration de l'agent acheteur $Umax = U_a(r) - U_a(a) = U_a(r)$ et du nombre d'itérations maximum $\hat{t}$. La stratégie la plus simple consiste à considérer $\epsilon = Umax/\hat{t}$. Cette stratégie augmente linéairement l'offre minimum entre le point d'exigence et le point d'aspiration. Pour prendre en compte le fait que les offres des vendeurs ne se font pas toujours à la valeur minimum et que l'algorithme doit être *anytime*, donc trouver de bonnes solutions rapidement, nous considérons également des incréments fixes plus élevés :
  - $Fix3U/N : \epsilon = 3 * Umax/\hat{t}$
  - $Fix6U/N : \epsilon = 6 * Umax/\hat{t}$
  - $Fix9U/N : \epsilon = 9 * Umax/\hat{t}$

L'inconvénient d'utiliser des incréments élevés est que l'acheteur risque de manquer des offres potentiellement intéressantes entre les deux dernières itérations (manque de précision du résultat).

Les deux stratégies variables sont celles présentées précédemment :

- $VarN$ considère uniquement le nombre de vendeurs restants (voir Eq. 5).
- $VarT$ considère le nombre de tours maximums restants (voir Eq. 7).

$VarN$ et $VarT$ utilisent un paramètre α pour fixer leur vitesse d'adaptation.

Agent vendeur

Chaque agent vendeur s suit une simple stratégie rationnelle dans le cadre d'enchères anglaises : prendre la première proposition respectant la valeur minimum $CounterProp(t)$ tant que la valeur de celle-ci est supérieure à sa valeur privée $BidderMax(s)$.

Un nombre $Nbproposals(s)$ de propositions P_i possibles sont générées pour chaque agent vendeur. À chaque proposition est associée une utilité pour l'agent acheteur ($UB(i)$) et pour l'agent vendeur ($US(i)$). Si une proposition est dominée par une autre proposition (utilité inférieure à la fois pour l'agent vendeur et pour l'agent acheteur), une nouvelle proposition est générée (car elle ne sera jamais proposée de toutes façons). Cette modélisation nous permet de reproduire une situation mono ou multicritère (le prix étant un critère) sans rentrer dans le détail des propositions, ce qui nécessiterait une modélisation plus complexe (et donc plus de paramètres à analyser). Le prix étant considéré comme un critère, $Nbproposals(s)$ sera grand (pour prendre en compte les paramètres continus, comme le prix).

Les utilités sont choisies selon une loi uniforme :

$US(i) \sim U[0; Umax]$

$UB(i) \sim U[0; BidderMax(s)]$

Aucune proposition n'étant dominée, ces $NbProposals$ peuvent être ordonnées selon l'utilité de l'agent acheteur (ou de l'agent vendeur, ordre inverse). Cette ordre permet au vendeur de choisir simplement la première proposition répondant à la contre-proposition courante comme sa proposition suivante.

$BidderMax(s)$ est calibré en utilisant une loi uniforme :

$BidderMax(s) \sim U[0; UmaxBidders * Umax]$

$UmaxBidders$ permet de contrôler la distribution des utilités des vendeurs.

Pour pouvoir comparer les stratégies de l'acheteur, lors des simulations, les listes de propositions des vendeurs sont utilisées à l'identique pour les 6 stratégies possibles (de nouvelles listes étant générées pour chaque nouvelle série de simulation).

Variables observées

Notre objectif est de comparer les stratégies de l'agent acheteur. En plus des variables directement observables (tel que ϵ^t), nous allons donc mesurer à chaque tour t la performance relativement à la solution optimale $BestPotentialBid$ (la meilleure proposition acceptable par un vendeur). Dans les simulations et pour calculer la performance, nous utilisons une utilité inversée $Umax - U_a(bid)$ afin de considérer une fonction croissante. On obtient pour la performance :

$$Perf(t) = \frac{Umax - U_a(Best(t))}{Umax - U_a(BestPotentialBid)}$$

Paramètres globaux

Nous utilisons une valeur standard des paramètres globaux afin d'avoir une référence et de pouvoir analyser la sensibilité autour de cette situation :

- $NbBidder = 100$
- $\hat{t} = 100$
- $\alpha = 0.5$
- $UMax = 10$
- $UmaxBidders = 0.9$
- $NbProposals = 1000$

Les résultats présentés sont les moyennes de 20 simulations avec des paramètres identiques (sauf pour le résultat-exemple d'une enchère unique en Fig. 2 et Fig. 3).

5.2 Configuration standard

Les Fig. 2 et Fig. 3 donne un aperçu de résultats obtenus en utilisant la configuration standard définie précédemment (évolution des $Perf(t)$ et ϵ^t respectivement). Par exemple, supposons que la négociation soit arrêtée en $t = 5$ (l'enchère étant *anytime*, elle peut être arrêtée à n'importe quelle itération), la proposition choisie dans le cas de la stratégie $FixU/N$ aura une utilité pour l'acheteur ne représentant que 7% de la meilleure proposition ($Perf(5) = 7\%$). Dans le cas des autres stratégies, la valeur serait respectivement de 19% ($Fix3U/N$), 36% ($Fix6U/N$), 40% ($VarN$), 46% ($VarT$) et 53% ($Fix9U/N$).

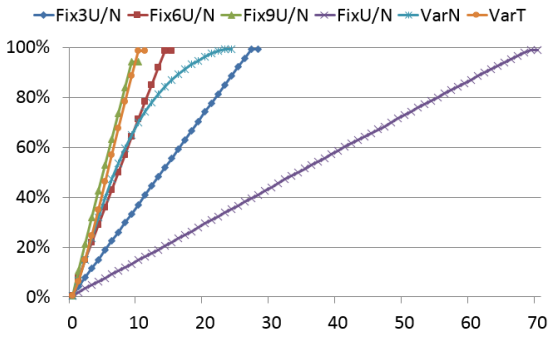


FIGURE 2 – Évolution de l'utilité de l'acheteur relative à la meilleure solution ($Perf(t)$) au cours d'une enchère, configuration standard, pour chaque stratégie acheteur possible.

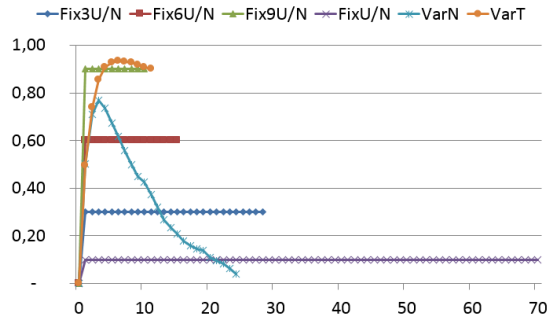


FIGURE 3 – Évolution de l'incrément minimum imposé par l'acheteur (ϵ^t) au cours d'une négociation, configuration standard, pour chaque stratégie acheteur possible.

Cette configuration standard nous permet de valider le bon fonctionnement de notre modèle et de la simulation. Comme on pouvait s'y attendre, les stratégies à incrément fixe engendrent une augmentation presque linéaire de l'utilité. Inversement, les stratégie à incrément variable commencent par augmenter rapidement l'incrément avant de le diminuer progressivement avec la sortie des vendeurs et le nombre d'itérations (Fig. 3). Cette évolution entraîne une évolution non-linéaire de l'utilité.

L'avantage des stratégies variables ($VarT$ et $VarN$) apparaît sur la Fig. 2 : une augmentation d'abord rapide de l'utilité de la meilleure offre (presque aussi rapide que $Fix9U/N$), puis une augmentation de plus en plus lente pour aboutir à une offre finale très élevée.

La qualité de l'offre finale (si l'enchère n'est pas interrompue avant qu'il ne reste qu'un seul vendeur) illustre l'inconvénient d'une stratégie fixe

trop rapide : $Fix9U/N$ est la seule stratégie a ne pas atteindre 100% si on laisse la négociation aller jusqu'à son terme. Même si c'est la plus rapide à se terminer (10 itérations), la valeur finale n'est que de 94%. Cela est dû au fait que l'incrément minimum est tellement élevé que la proposition minimum à la onzième itération devait être inacceptable pour tous les vendeurs, même s'il existait encore de la marge à l'itération précédente.

La Table 1 permet de détailler ces résultats en analysant le résultat final et le temps de négociation moyen sur 20 simulations. Les stratégies fixes obtiennent un résultat d'autant plus faible qu'elles sont rapides. $Fix9U/N$ est la plus rapide avec le plus mauvais résultat final (10 itérations et 93%), alors que $FixU/N$ arrive lentement (68 itérations) au meilleur résultat (98%).

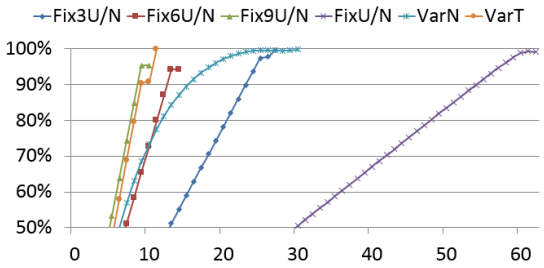


FIGURE 4 – Évolution de l'utilité de l'acheteur relative à la meilleure solution ($Perf(t)$), configuration standard sauf $NbSellers = 1000$, pour chaque stratégie d'acheteur.

TABLE 1 – Pour chaque stratégie, si l'enchère arrive à son terme, moyenne (et variance) de l'utilité relative de l'acheteur ($Perf(t)$) de l'offre finale, et nombre d'itérations moyen (et variance) pour y parvenir (pour 20 simulations).

Stratégie	U finale moy.	U finale var.	Tours moy.	Tours var.
FixU/N	98,2%	1,08%	68,3	1,19
Fix3U/N	97,3%	1,58%	27,1	0,46
Fix6U/N	97,1%	2,49%	14,6	0,49
Fix9U/N	93,4%	0,40%	10	-
VarN	98,3%	1,19%	25,6	3,97
VarT	96,2%	4,29%	10,5	0,5

Par rapport aux stratégies fixes, les stratégies variables montrent encore leur intérêt : $VarT$ permet d'arriver à un résultat presque aussi vite que $Fix9U/N$ (10,5 itérations en moyenne) mais avec un résultat bien meilleur (96,2%). $VarT$ permet d'arriver a un résultat aussi bon que $FixU/N$ (98,3%), mais beaucoup plus vite (25,6 itérations seulement de moyenne).

5.3 Analyse de sensibilité

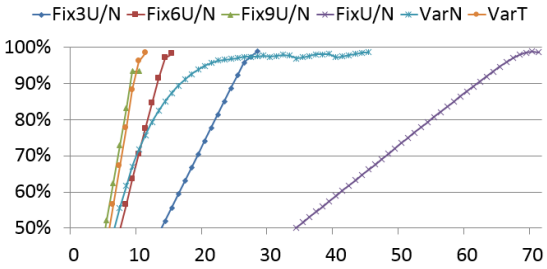


FIGURE 5 – Évolution de l'utilité de l'acheteur relative à la meilleure solution ($Perf(t)$), configuration standard sauf $NbProposals = 10000$, pour chaque stratégie d'acheteur.

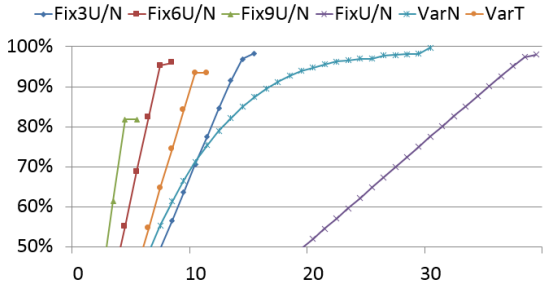


FIGURE 6 – Évolution de l'utilité de l'acheteur relative à la meilleure solution ($Perf(t)$), configuration standard sauf $\hat{t} = 50$, pour chaque stratégie d'acheteur.

Afin d'approfondir ces résultats, nous considérons les situations où le nombre de vendeurs est plus élevé (Fig. 4, $NbSellers$=1000) et le cas où chaque vendeur a plus de propositions disponibles (Fig. 5, $NbProposals$=10000). Dans les deux cas, nous améliorons la fluidité du marché et les choix possibles (en conservant les autres paramètres inchangés), ce qui conduit a une dynamique très similaire à la configuration standard (Fig. 2). $VarN$, en particulier, permet toujours d'obtenir de bons résultats rapidement puis de s'améliorer lentement, un comportement parfaitement adapté à des enchères *anytime* (alors que la variation du nombre de vendeurs total et du nombre d'offres auraient pu affecter son ralentissement, contrôlé par la proportion de vendeurs restants). $VarT$ confirme également les résultats précédents avec de bons résultats rapides, mais un résultat final moyen : la variable pilotant le ralentissement de l'incrément (le nombre d'itérations restant) reste en effet élevé du fait d'une fin trop rapide de l'enchère.

Avec un nombre d'itérations maximum plus faible (Fig. 6, $\hat{t} = 50$), on peut confirmer que le ralentissement est plus marqué pour $VarT$ (voir l'évolution d'ϵ^t Fig. 7), avec une offre finale obtenue en 11 itérations (contre 5 pour $Fig9U/N$ et 11 pour $Fig6U/N$). Ce résultat reste toutefois qualitativement faible comparé à $VarN$, voir $Fix6U/N$ et $Fix3U/N$ dans cette configuration.

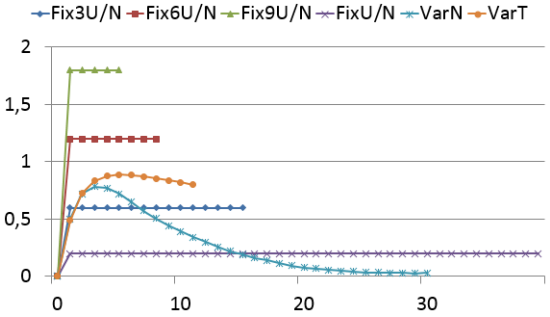


FIGURE 7 – Évolution de l'incrément minimum imposé par l'acheteur (ϵ^t), configuration standard sauf $\hat{t} = 50$, pour chaque stratégie d'acheteur.

Ces comportements de $VarT$ et $VarN$ sont contrôlés par leur paramètre α, fixé à 0,5. Pour visualiser son impact, nous avons étudié les résultats obtenus par $VarN$ pour différentes valeurs de α (Fig. 8) ainsi, que les ϵ^t correspondants (Fig. 9). L'analyse des ϵ^t (Fig. 9) permet en particulier de bien comprendre l'impact du paramètre : des valeurs élevées de α correspondent à un comportement plus prudent, avec une augmentation progressive de ϵ^t suivi d'une baisse également lente. Un tel comportement est adapté lorsque l'acheteur pense que la négociation ne sera pas interrompue trop rapidement et/ou qu'il a de grandes incertitudes sur la distributions des utilités des vendeurs. Au contraire, un α élevé conduit a un comportement beaucoup plus agressif, avec une augmentation très rapide au début, puis une chute également rapide après écrémage des vendeurs pour permettre un raffinement du prix de vente.

6 Conclusion

Dans cet article, nous avons présenté un modèle d'enchères multicritères anglaises inversées dans lequel la stratégie de formulation de la contre-proposition ne demeure pas fixe comme ce qu'on observe dans les modèles d'enchères classiques mais évolue tout au cours du processus de négociation. Nous avons observé qu'un

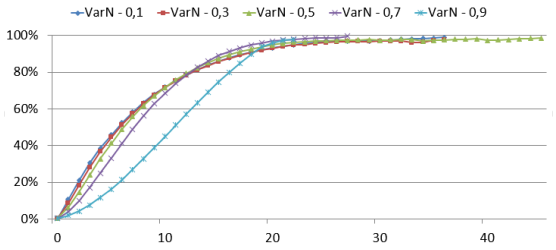


FIGURE 8 – Évolution de l'utilité d'un acheteur de stratégie $VarN$ relative à la meilleure solution ($Perf(t)$), pour différentes valeur de α.

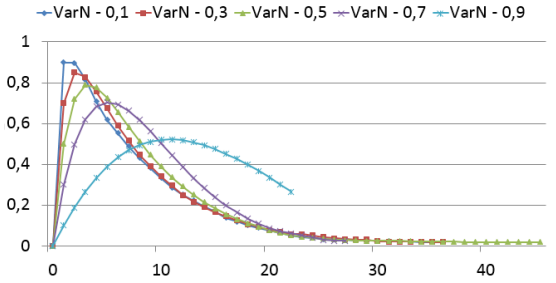


FIGURE 9 – Évolution de l'incrément minimum imposé par un acheteur de stratégie $VarN$ (ϵ^t) pour différentes valeur de α.

incrément variable défini sur la base du nombre d'agents vendeurs restants ou sur la base du temps restant accélère le processus d'enchères tout en assurant des résultats équivalents.

Afin de valider notre approche, nous avons conduit plusieurs expérimentations qui ont montré qu'un incrément variable basé sur le lissage exponentiel contribue à baisser la durée globale des enchères en comparaison à des enchères classiques à incrément fixe. De plus, cette approche permet de trouver une solution où la proposition gagnante est plus proche du point d'aspiration de l'agent acheteur. Nous envisageons, pour des recherche futures, plusieurs autres directions : (1) une exploration complète de l'espace des paramètres, (2) une analyse expérimentale d'autres méthodes de formulation de la contre-proposition, (3) la généralisation de nos résultats à d'autres types d'enchères aussi bien ascendantes que descendantes, (4) la prise en compte de différents comportements d'agents basés sur différents traits de personnalité des agents et différents styles cognitifs tels que l'optimisme, l'aversion au risque, la tolérance, etc.

Références

[1] Samir Aknine, Marie-Jo Bellosta, Sylvie Kornman, and Suzanne Pinson, 'A many-to-many negociation protocol for electronic commerce with risk commitment strategies', *The Fifth International Conference on Electronic Commerce Research (ICECR-5), Montréal*, 23–27, (2002).

[2] John Asker and Estelle Cantillon, 'Properties of scoring auctions', *The RAND Journal of Economics*, **39**(1), 69–85, (2008).

[3] Damian R Beil and Lawrence M Wein, 'An inverse-optimization-based auction mechanism to support a multiattribute rfq process', *Management Science*, **49**(11), 1529–1545, (2003).

[4] Marie-Jo Bellosta, Imène Brigui, Sylvie Kornman, and Daniel Vanderpooten, 'A multi-criteria model for electronic auctions', in *Proceedings of the 2004 ACM symposium on Applied computing*, pp. 759–765. ACM, (2004).

[5] Martin Bichler, 'An experimental analysis of multi-attribute auctions', *Decision Support Systems*, **29**(3), 249–268, (2000).

[6] Martin Bichler, 'A roadmap to auction-based negotiation protocols for electronic commerce', in *System Sciences, 2000. Proceedings of the 33rd Annual Hawaii International Conference on*, pp. 10–pp. IEEE, (2000).

[7] Martin Bichler and Jayant Kalagnanam, 'Configurable offers and winner determination in multi-attribute auctions', *European Journal of Operational Research*, **160**(2), 380–394, (2005).

[8] Martin Bichler and Jayant R Kalagnanam, 'Software frameworks for advanced procurement auction markets', *Communications of the ACM*, **49**(12), 104–108, (2006).

[9] Martin Bichler, Marion Kaukal, and Arie Segev, 'Multi-attribute auctions for electronic procurement', in *Proceedings of the first IBM IAC workshop on Internet based negotiation technologies*, pp. 18–19. Citeseer, (1999).

[10] Esther David, Rina Azoulay-Schwartz, and Sarit Kraus, 'Bidding in sealed-bid and english multi-attribute auctions', *Decision Support Systems*, **42**(2), 527–556, (2006).

[11] Eugénio de Oliveira, José Manuel Fonseca, and Adolfo Steiger-Garção, 'Multi-criteria negotiation on multi-agent systems', *CEEMAS'99*, 190, (1999).

[12] Arnaud Grignard, Patrick Taillandier, Benoit Gaudou, Duc An Vo, Nghi Quang Huynh, and Alexis Drogoul, 'Gama 1.6 : Advancing the art of complex agent-based modeling and simulation', in *PRIMA 2013 : Principles and Practice of Multi-Agent Systems*, 117–131, Springer, (2013).

[13] John H Kagel and Dan Levin, 'Auctions : a survey of experimental research, 1995–2008', *Kagel J, Roth A, editors*, **2**, (1990).

[14] Paul Klemperer, 'What really matters in auction design', *The Journal of Economic Perspectives*, **16**(1), 169–189, (2002).

[15] Sarit Kraus, Jonathan Wilkenfeld, and Gilad Zlotkin, 'Multiagent negotiation under time constraints', *Artificial intelligence*, **75**(2), 297–345, (1995).

[16] R Preston McAfee and John McMillan, 'Auctions and bidding', *Journal of economic literature*, **XXV**, 699–738, (1987).

[17] David C Parkes and Jayant Kalagnanam, 'Models for iterative multiattribute procurement auctions', *Management Science*, **51**(3), 435–451, (2005).

[18] Stefan Strecker, 'Information revelation in multiattribute english auctions : A laboratory study', *Decision Support Systems*, **49**(3), 272–280, (2010).

[19] Jeffrey Teich, Hannele Wallenius, and Jyrki Wallenius, 'Multiple-issue auction and market algorithms for the world wide web', *Decision Support Systems*, **26**(1), 49–66, (1999).

[20] Peter R Wurman, Michael P Wellman, and William E Walsh, 'Specifying rules for electronic auctions', *AI Magazine*, **23**(3), 15, (2002).

[21] Shlomo Zilberstein, 'Using anytime algorithms in intelligent systems', *AI magazine*, **17**(3), 73, (1996).

Vers une approche d'ingénierie multiagent à base de ligne de produits logiciels

Anarosa A.F. Brandao[a,b] anarosa.brandao@usp.br
Dounia Boufedji [b,d] dounia.boufedji@lip6.fr
Tewfik Ziadi[b] Tewfik.Ziadi@lip6.fr
Zahia Guessoum[b,c] zahia.guessoum@lip6.fr

[a] *Computing Engineering and Digital Systems Department*
University of Sao Paulo – Brazil

[b] *Sorbonne Université, UPMC Univ Paris 06,*
LIP6, F-75005, Paris, France

[c] *Université Reims Champagne Ardennes*
CReSTIC, F-51000, Reims, France

[d] *Université des Sciences et de la Technologie Houari Boumediene,*
RIIMA, Babezouar,Alger,Algérie

Résumé

Bien que plusieurs méthodes et outils aient été proposés pour l'ingénierie des SMA durant les deux dernières décennies, passer des modèles SMA au code reste une tâche difficile. La majorité de ces méthodes ne parvient pas à proposer une solution pour la réutilisation des implémentations existantes telles que les processus incrémentaux. Notre proposition répond à deux problèmes : combler le fossé entre la modélisation des SMA et l'implémentation d'une part, et fournir une approche incrémentale de développement de SMA en s'appuyant sur les lignes de produits logiciels d'autre part. Cette approche se base sur une description de la variabilité grâce à des modèles de caractéristiques et utilise un framework de ligne de produits logiciels pour la génération des différentes variantes de l'application.

Mots clefs*: SMA, lignes de produits, modèle des caractéristiques, dérivation des produits.*

Abstract

Although several methods and tools to support engineering MAS were proposed in the last decades, it is still a difficult task to go from MAS models to MAS code. Moreover, just a few MAS methods provide guidelines for that and such methods fail in proposing a solution to reuse MAS implementation, as in an incremental process. Our proposal intends to address both issues: filling in the gap between MAS modeling and implementation and providing guidance for incremental development of MAS using a Software Product Line (SPL) approach that goes beyond the variability description through feature models and proposes to generate different variants using existing SPL frameworks.

Keywords*: MAS-PL, product-line, feature model, product generation.*

1. Introduction

Le génie logiciel multiagent a engendré plusieurs méthodes et outils pour l'ingénierie des SMA durant ces deux dernières décennies [4][6][11][18]. Les méthodes proposées couvrent plusieurs étapes du cycle de développement telles que l'ingénierie des besoins, la conception, l'implémentation, etc. Par exemple, ASPECS [18] couvre toutes les étapes, elle est applicable aux SMA holoniques. Cependant, la majorité des méthodes multiagents ne couvre pas l'ensemble des étapes. Elles s'arrêtent souvent à la conception qui élabore des modèles multiagents. Néanmoins, le passage des modèles produits par les méthodes aux SMA opérationnels est un problème compliqué notamment pour les non spécialistes des SMA,

car les outils d'implémentation n'utilisent pas les mêmes modèles et les mêmes concepts que les méta-modèles des méthodes. Par ailleurs, le développement des SMA est souvent incrémental. Il nécessite ainsi, une double expertise à la fois en conception et en implémentation.

Nous pensons que l'écart entre les méthodes et les outils multiagents est dû à une caractéristique du développement des méthodes multiagents. Contrairement à l'approche objets où les méthodes et les langages de modélisation ont émergé de l'implémentation des systèmes à base d'objets, la majorité des méthodes multiagents a été développée sans s'appuyer sur une expertise en implémentation.

Pour combler l'écart entre la conception et l'implémentation et pour faciliter le développement incrémental, la communauté génie logiciel a introduit une nouvelle approche appelée Ligne De Produits logiciels (LdP). La LdP est un paradigme qui prône une vision de modélisation et de développement dans laquelle l'objectif n'est pas l'obtention d'un seul système logiciel à la fois, mais plutôt un ensemble de systèmes logiciels possédant des caractéristiques communes mais aussi qui diffèrent en certains points de variabilité. Ces logiciels peuvent être des variantes d'une famille d'applications similaires. La gestion de la variabilité est la première dimension clé dans l'ingénierie des LdPs. La deuxième dimension concerne la construction, appelée aussi dérivation des membres de la ligne d'un produit. Chaque logiciel variant est obtenu donc par dérivation en instanciant un ou plusieurs points de variabilité.

Différentes approches combinant les SMA et les LdPs proposent des extensions de méthodes SMA existantes afin d'intégrer la notion de variabilité dans les modèles SMA (voir par exemple [7][9][15][16][17]). Cependant, les solutions existantes ne se sont pas focalisées sur l'écart entre les méthodes et les outils d'implémentation multiagents et ne proposent pas des mécanismes permettant d'aller au delà de la description de la variabilité.

Dans cet article, nous proposons une approche pour combler l'écart entre les méthodes et les outils d'implémentation et faciliter le développement incrémental des variantes SMA. Tout d'abord, nous proposons des lignes directrices pour spécifier les modèles de caractéristiques (*features model*) dans le contexte des SMA. Ces caractéristiques pourraient être déduites des méta-modèles, des méthodes existantes ou des applications existantes (code).

Enfin, et pour réduire l'écart entre les modèles et le code, nous réutilisons des environnements de LdPs existants afin de générer des implémentations JADE pour plusieurs variantes de SMA. L'exemple de l'emploi du temps est utilisé pour illustrer notre approche.

Ce papier est organisé comme suit : la section 2 présente un état de l'art sur l'ingénierie des LdPs et discute les travaux existants. La section 3 présente l'exemple de l'emploi du temps. La section 4 décrit notre approche LdP-SMA. La section 5 conclut ce travail et présente quelques perspectives.

2. Contexte et état de l'art

Cette section introduit les LdPs et présente les approches combinant les LdPs et les SMA.

2.1 Lignes De Produits Logiciels (LdP)

Les LdPs sont une transposition des chaînes de production industrielle au monde logiciel. Pour réduire les coûts et le temps de développement, elles visent à produire une famille de systèmes au lieu d'un système unique. Clements et Northrop [8] définissent une LdP comme un ensemble de systèmes logiciels partageant un ensemble commun de fonctionnalités ou caractéristiques qui répondent à des besoins spécifiques d'une partie particulière du marché, ou mission et qui sont développés d'une manière prescrite à partir d'un noyau commun d'éléments.

L'ingénierie des LdPs s'axe sur la capture de points communs et de la variabilité entre plusieurs produits logiciels appartenant au même domaine [8]. Les points communs rassemblent des hypothèses qui sont vraies pour tous les membres de la LdP, tandis que la variabilité regroupe l'ensemble des hypothèses montrant comment les *produits*, membres de la *ligne* de *produits* diffèrent.

L'ingénierie du domaine consiste à développer et construire la LdP en utilisant l'analyse et l'implémentation du domaine.

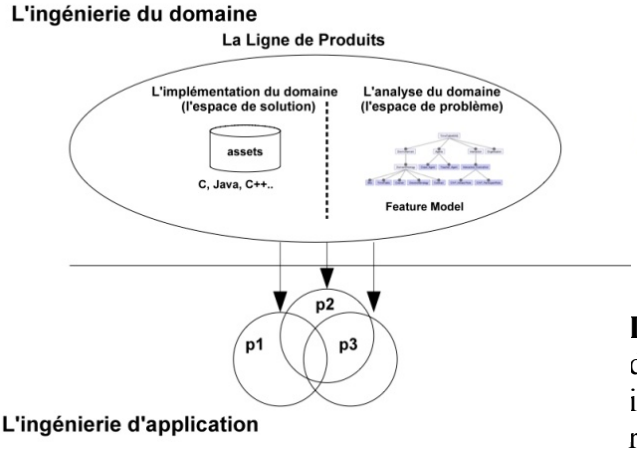


FIG. 1 – Ingénierie de LdP.

Lors de l'analyse du domaine, des points communs du domaine et la variabilité sont identifiés, les membres de la LdP sont ensuite spécifiés en utilisant des modèles de variabilité. Le modèle de caractéristiques (qui sera nommé FM[1] dans ce papier) est le formalisme principal pour la capture de points communs et de variabilité des LdPs [3][14]. Une caractéristique est définie comme un aspect important ou distinctif et apparent, la qualité ou attribut d'un système [14]. Un exemple illustratif d'un modèle de variabilité simple concernant une LdP simple d'une famille de voitures est illustré dans la figure 2. Les éléments communs de cette famille sont spécifiés en utilisant des caractéristiques obligatoires où la variabilité est décrite en utilisant les caractéristiques optionnelles, le *ou*, le *ou exclusif* ; ce qui est également complété par des contraintes.
La notion de *configuration* est utilisée pour représenter un produit d'une LdP. Elle consiste en une sélection de caractéristiques qui sont compatibles avec les contraintes du FM [2]. Dans l'exemple de voitures, une configuration peut contenir le choix des différentes options d'une voiture particulière.
Lors de l'implémentation du domaine, des éléments logiciels sont construits et associés à chaque caractéristique. Un élément logiciel est un artefact, qui est utilisé pour développer les produits logiciels. Les documents de spécification des besoins, les modèles, le code source, etc. sont des exemples de tels éléments.

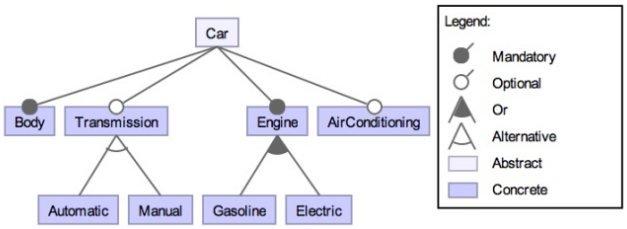


FIG. 2–Modèle de caractéristiques

L'ingénierie d'applications consiste à dériver des produits membres sur la base de la LdP issue de l'ingénierie du domaine. De nombreuses approches existantes sont proposées par la communauté LdP pour la mise en œuvre de la dérivation de produits. Dans cet article nous utilisons FeatureHouse qui offre des services de composition de caractéristiques [1]. Il est générique et peut prendre en charge de nombreux langages de programmation dont Java.

2.2 Approches combinant LdPs et SMA

Cette section décrit et analyse les approches existantes qui combinent les LdPs et les méthodes SMA. Cette combinaison a généré un nouveau paradigme : Les Lignes de Produits des SMA (LdP-SMA) dont le but est d'améliorer l'ingénierie des SMA en combinant les avantages des deux approches.
Dans [17], les auteurs comparent les deux approches et étudient les avantages et les défis de ce nouveau paradigme de l'ingénierie des SMA. Ils considèrent que les deux approches sont basées sur les mêmes concepts dans les premières phases de développement. Par exemple, les deux approches utilisent des modèles dans les phases d'analyse ; les LdPs utilisent des modèles de caractéristiques ; et les méthodes multiagents utilisent les modèles SMA. Ces modèles sont indépendants de la plateforme. Dans la phase d'implémentation, les LdPs s'appuient sur une architecture de base commune et des éléments réutilisables. Cependant, la phase d'implémentation n'est pas souvent prise en compte dans les méthodes AOSE. Plusieurs approches LdP-SMA ont ainsi

[1] **F**eature **M**odel

été proposées dont la majorité utilise Gaia et PASSI.
Dehlinger et Lutz ont introduit Gaia-LdP qui se concentre sur la documentation et la réutilisation des spécifications des besoins pour une LdP-SMA [9]. Leur proposition étend Gaia pour inclure les principes de réutilisation. Gaia-LdP propose deux contributions principales : (i) elle introduit des points de variation dans la conception et le développement des SMA ; (ii) elle propose un modèle de spécification des besoins.
Une autre approche basée sur Gaia, nommée MaCMAS, propose un FM pour documenter les points communs et les variabilités, ce qui permet la description d'une même caractéristique à différents niveaux d'abstraction permettant ainsi de spécifier et de tester les changements à chaque niveau d'abstraction. Cette approche a été proposée par Peña et al. [16] pour faire face à la conception et à la gestion des SMA en évolution en utilisant les LdP-SMA. MaCMAS vise la construction du noyau de l'architecture de base d'une approche LdP-SMA.
Nunes et al. proposent une approche qui s'avère de nos jours être la plus complète car elle recouvre toutes les étapes allant des besoins à l'implémentation [15]. Ils y ont adapté et repris des éléments de la méthode LdP PLUS. Cependant, cette approche utilise un modèle unique pour toute la LdP enrichi avec des annotations pour exprimer toutes les variabilités. Le résultat est en effet un modèle complexe qui est difficile à utiliser par les personnes qui ne sont pas impliquées dans son développement.
La LdP-SMA est une approche prometteuse pour l'ingénierie SMA. En fait, la plupart des développeurs des SMA utilisent une approche incrémentale pour construire des applications SMA et suivre implicitement une approche de ligne de produits. En outre, le paradigme LdP-SMA permet de formaliser la réutilisation dans le développement des SMA. Cependant, la plupart des solutions existantes introduisent des notations complexes qui sont difficiles à comprendre et à utiliser. Par ailleurs, elles ne sont pas adaptées au développement incrémental des SMA car elles sont basées sur des méthodes existantes qui ne considèrent pas la phase d'implémentation. Ainsi, notre approche est différente car nous proposons de partir de l'implémentation d'un SMA en se reposant ainsi sur une approche LdP extractive, où nous étudions les points de variation et définissons les caractéristiques.
Les LdP-SMA sont très proches des méthodes d'ingénierie à base de modèles. Cependant, ces dernières méthodes visent à développer un seul système alors que les LdP-SMA permettent de générer une famille de systèmes. Ils sont en effet mieux appropriés au développement incrémental.

3. Exemple

Pour illustrer notre approche, nous proposons d'utiliser le problème de l'emploi du temps. Dans cette section, nous commençons par présenter une variante simple de cet exemple, puis un ensemble de facteurs de variabilité est introduit.
Description : Le *benchmark* emploi du temps (appelé *TimeTable* ci-après) a été proposé par Bernon et al. [5] afin d'étudier certains problèmes dans le domaine des SMA. Il représente un problème d'allocation de ressources complexe pour lequel la recherche de solutions doit être collective.
Le diagramme de classes UML de la figure 3 résume l'implémentation de la variante simple de l'emploi du temps. Il comprend deux parties: 1) un ensemble de classes qui représentent l'ontologie du domaine et 2) un ensemble de classes représentant les agents et leurs comportements. Cette variante simple du problème de l'emploi du temps ne considère aucune contrainte concernant les ressources. Par exemple, elle considère que les enseignants et les groupes d'étudiants trouvent une solution pour les cours. Les classes sont ensuite affectées à ces cours avec tout le matériel nécessaire (Tableau noir, vidéo projecteur ...) et les contraintes (taille de la pièce ...).
Dans notre modèle, chaque utilisateur est doté d'un agent assistant dont le comportement est guidé par le protocole contrat Net (CNP) :

- L'agent Enseignant (*Teacher_Agent*) a pour objectif d'assurer sa charge d'enseignement (en créneaux) en fixant les créneaux et les cours qu'il doit assurer à toutes les classes (groupes d'étudiants).
- L'agent Classe (*Class_Agent*) représente un groupe d'étudiants auquel un enseignant sera assigné pour un cours spécifique.

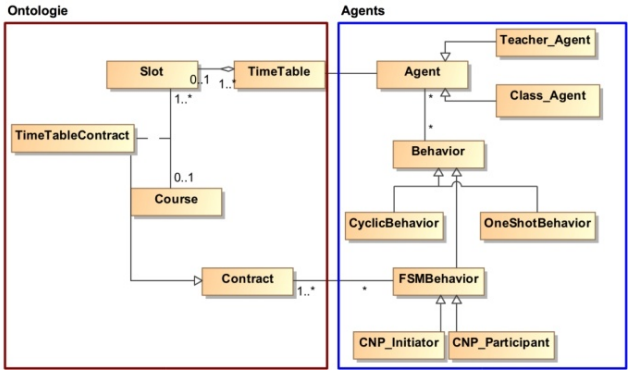


FIG. 3 – diagramme de classes UML pour l'exemple simple

Variabilité : La variante de l'emploi du temps décrite ci-dessus est simple. Toutefois, le problème de l'emploi du temps peut avoir plusieurs variantes de part l'inclusion ou l'exclusion de contraintes qui peut être très complexe [5]. En effet, si on considère le problème associé à ce système et qui consiste à trouver un emploi du temps cohérent en tenant compte des points suivants : (i) une liste de *m* enseignants, (ii) une liste de *n* classes (groupes d'élèves) et, si nécessaire, (iii) une liste des *p* salles. Compte tenu de ces nouvelles considérations, plusieurs variantes peuvent être déclinées. En effet, l'introduction de la variabilité concernant le nombre d'agents entraîne le changement du protocole d'interaction. En plus, la prise en charge des salles génère aussi d'autres variantes en fonction de la façon dont les ressources sont gérées.

Dans la section suivante, nous allons présenter notre approche pour utiliser les LdPs afin d'implémenter les SMA. Elle suit l'environnement général des LdPs présenté dans la section précédente. Nous commençons par montrer comment spécifier un FM. Nous montrons ensuite comment les différentes variantes du système de l'emploi du temps peuvent être obtenues en utilisant une approche de composition de LdP.

4. Id-MASPL[2]

Notre nouvelle approche LdP-SMA, nommée Id-MASPL, vise à traiter deux questions:1) le développement incrémental d'une famille de SMA du même domaine, et 2) la réduction de l'écart existant entre la modélisation et l'implémentation des SMA.

Nous proposons, en premier lieu, un canevas de FM des SMA. Nous montrons par la suite, comment raffiner ce FM pour prendre en considération les nouveaux facteurs de variabilité. Enfin, nous illustrons les questions d'implémentation concernant les artefacts logiciels et dérivation de variantes.

4.1 Vers un modèle pour les caractéristiques des SMA

Dans cette section, nous présentons une approche de LdP pour l'implémentation des SMA ou la réutilisation de solutions SMA existantes pour les raffiner ou les étendre.

Etant donnée une représentation abstraite d'un SMA, qui pourrait être le produit d'une méthode multiagent, notre idée est de construire un FM basé sur cette représentation abstraite et sur le paradigme Voyelles (Agent, Environnement, Interaction, Organisation) proposé par Yves Demazeau [10].

Notre FM est construit à partir du modèle SMA élaboré pour l'implémentation de la première variante de l'exemple. Il possède deux niveaux : un niveau abstrait, et un niveau spécifique au domaine. Au niveau abstrait nous considérons les principales abstractions qui font partie d'un SMA, selon le paradigme Voyelles, pour organiser les caractéristiques abstraites. Au niveau spécifique du domaine, en dessous de chaque abstraction correspondant à une voyelle, nous retrouvons les caractéristiques liées à l'Environnement, aux Agents, à l'Interaction et à l'Organisation.

Dans notre exemple, la description abstraite du système présentée dans le diagramme UML doit

[2] **I**ncremental **d**evelopment of **M**ulti-**A**gent **S**ystems with Software **P**roduct **L**ine

être mise en correspondance avec un FM composé des caractéristiques spécifiques du domaine

Caractéristiques de l'environnement : L'environnement des agents représente le contexte dans lequel les agents sont situés. Chacune des classes de l'ontologie du domaine (*Course*, *Slot* et *TimeTable*) est en effet associée à une caractéristique.
La classe *Contract* représente la stratégie de décision adoptée par les rôles d'interaction pour accepter et / ou rejeter les propositions reçues. Par conséquent, une caractéristique *Decision-Strategy* lui est associée.

Caractéristiques des Agents : La mise en correspondance est directe car les agents sont clairement identifiés dans le diagramme UML et chaque classe représentant un agent correspond à une caractéristique.
Vu que le problème que nous résolvons est la variante la plus simple mais en même temps la base, ces caractéristiques sont donc obligatoires.

Caractéristiques de l'Interaction : Les caractéristiques de l'interaction dépendent des types d'applications et par conséquent de la *plateforme*. Dans les SMA, il existe deux types d'interactions : L'interaction directe et l'interaction indirecte. Par exemple, JADE utilise l'interaction directe et fournit une librairie de protocoles d'interaction - chacun étant implémenté par deux rôles d'interactions. Dans le diagramme de classes, les classes *CNP-Initiator* et *CNP-Participant* représentent les deux rôles du protocole *Contract Net* (CNP). Par ailleurs, Netlogo utilise souvent l'interaction indirecte telle que les phéromones.

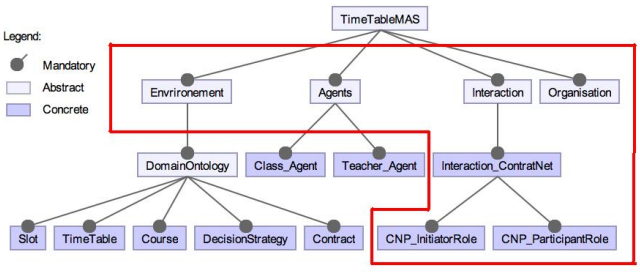


FIG. 4 – Modèle initial des caractéristiques pour le SMA de l'emploi du temps.

Puisque dans notre système l'utilisation de CNP est nécessaire, nous proposons d'associer la caractéristique *Interaction-ContractNet* au comportement *FSM_Behaviour*, avec ses sous-caractéristiques *CNP_InitiatorRole* et *CNP_ParticipantRole* pour chaque classe qui décrit les rôles d'interaction.

Caractéristiques de l'organisation : Compte tenu de la simplicité de la première version de notre exemple, il n'y a qu'un groupe d'agents (comprenant un agent Enseignant, *n* agents Classes) et deux rôles. Chaque agent joue un rôle. Par conséquent, nous n'avons pas nécessairement besoin d'un modèle explicite de l'organisation pour cette version.
Une capture d'écran du FM est donnée dans la figure 4, et elle représente une configuration du SMA-LdP pour l'exemple décrit dans la section 3. Les caractéristiques entourées en rouge sont indépendantes du domaine et représentent le FM. Le reste des caractéristiques est spécifique au domaine. Ce FM sera raffiné afin de montrer explicitement la variabilité associée à certaines variantes de notre exemple.

4.2 L'évolution des SMA par le raffinement des caractéristiques

Etant donnée la solution développée dans la section précédente, nous allons analyser et développer quelques extensions en améliorant le FM. Nous proposons d'augmenter la complexité du problème :

- en changeant la stratégie de décision pour les deux rôles. Dans notre premier FM, nous considérons que les différents agents sont homogènes. Ils utilisent la même stratégie pour sélectionner les propositions reçues. Les classes utilisent la même stratégie pour accepter ou rejeter les propositions, et tous les enseignants utilisent la même stratégie pour sélectionner un sous-ensemble de propositions à accepter ;
- en changeant le nombre d'agents Enseignants dans notre système (Version 1 : 1 Enseignant et n Classes ; Version 2 : m Enseignants et n Classes). Ainsi, les agents Classes peuvent recevoir plusieurs propositions pour le même cours. Ensuite, les agents Classes peuvent définir une stratégie pour sélectionner l'un d'entre eux à accepter. Cela signifie que nous devrions étendre le protocole d'interaction pour décrire cette nouvelle façon d'interagir ;

- en ajoutant des contraintes sur les ressources telles que la disponibilité des salles. Nous pouvons ainsi ajouter un agent pour gérer toutes les salles ou associer à chaque salle un gestionnaire ;
- en utilisant un autre protocole d'interaction.

Pour changer de protocole, nous avons juste besoin d'étendre le FM existant en incluant une caractéristique abstraite Interaction-Protocol qui est composée de plusieurs protocoles d'interaction, tels que les protocoles de FIPA *Contract-Net*, *Iterated-Contract-Net*, *English-Auction*, *Request-Interaction*, entre autres. Nous avons ensuite analysé les protocoles d'interaction et les comportements requis. Le but de cette analyse est de déterminer les comportements qui peuvent être génériques. Par exemple, l'initiateur CNP nécessite un contrat, un temps d'arrêt (*timeout*) et une stratégie d'évaluation. En outre, le participant exige seulement une stratégie pour construire des propositions. Jarraya et Guessoum [13] concluent que, pour plusieurs protocoles d'interaction, les comportements locaux (définis comme automates à états finis) sont génériques. Mais ils nécessitent certains paramètres d'entrée comme un contrat, un temps d'arrêt, et une ou plusieurs stratégies.

En effet, ils peuvent être adoptés et activés dynamiquement par des agents.

Nous proposons donc d'ajouter à notre FM un ensemble de caractéristiques représentant les rôles des protocoles d'interaction, et un ensemble de stratégies, qui enrichissent l'ontologie. En outre, et pour gérer les salles de classe et leur disponibilité, nous proposons d'ajouter le concept *Room* à l'ontologie et de créer une nouvelle classe représentant les gestionnaires (*Room_Agent*).

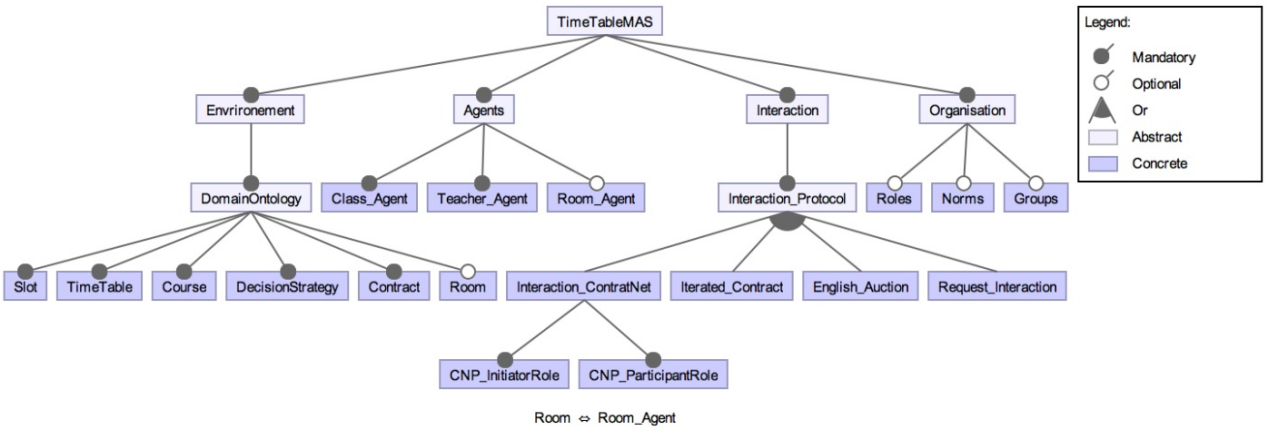


FIG. 5 – Le modèle des caractéristiques raffiné.

La Figure 5 illustre le nouveau FM qui raffine le modèle initial de la Figure 4. Il est à noter que les modèles de caractéristiques permettent également d'ajouter les dépendances entre les caractéristiques.

Nous ajoutons donc une contrainte spécifiant que la présence de la caractéristique liée à la ressource *Room* nécessite la présence de la caractéristique *Room-Agent* et vice versa.

Il est important de noter également que dans cet article nous ne considérons pas la caractéristique de l'organisation qui concerne cet exemple d'emploi du temps. En plus, les comportements des agents sont limités à des rôles interactifs et aucune norme n'est considérée. Toutefois, si l'on considère un problème plus complet pour gérer l'emploi du temps des collègues de l'Université, plusieurs aspects doivent être considérés. Par exemple, plusieurs normes doivent être prises en ligne de compte comme le nombre d'heures minimal ou maximal. Par ailleurs, un modèle d'organisation tel que AGR [11] devrait être considéré également. Nous avons donc ajouté trois caractéristiques optionnelles : les Rôles, les Normes et les Groupes.

4.3 Implémentation d'artefacts et dérivation de variantes

Comme souligné dans la section 2, la LdP est

définie par un FM mais aussi par des éléments logiciels (appelés *assets*). Ces derniers représentent les artefacts logiciels qui implémentent les différentes caractéristiques. De nombreuses approches ont été proposées pour mettre en œuvre les artefacts logiciels dans les LdPs [2]. Dans notre approche où l'objectif est de générer des applications JADE, nous nous sommes intéressés aux artefacts logiciels représentant des fragments de code JADE. Pour implémenter ces artefacts, nous proposons de réutiliser le composeur FeatureHouse [1] qui permet l'implémentation de la LdP en Java. FeatureHouse associe à chaque caractéristique les éléments de code l'implémentant. Ceci peut contenir l'ajout de nouvelles classes et/ou le raffinement des classes existantes. FeatureHouse utilise par la suite un mécanisme de composition pour générer le code de chaque variante en se basant sur une composition des caractéristiques [1].

Dans ce qui suit, nous illustrons l'implémentation des caractéristiques pour l'emploi du temps. Ensuite, nous présentons la façon dont chaque variante SMA peut être dérivée à partir de cette implémentation.

4.3.1 Implémentation des artefacts

Comme indiqué ci-dessus, les caractéristiques LdP-SMA peuvent concerner quatre dimensions : l'environnement, les agents, l'interaction et l'organisation. Les caractéristiques qui concernent l'environnement pour le SMA de l'emploi du temps telles que *TimeTable*, Course, *Slot, et Contract*, sont directement mappées aux classes Java.

L'implémentation des caractéristiques liées aux agents est basée sur l'extension des classes agents de JADE. Par exemple, le code associé à JADE pour la classe *Teacher-Agent* consiste à définir la classe de façon à ce qu'elle hérite de la classe *Agent* de JADE. Nous avons également besoin de raffiner la classe *Launcher* en ajoutant l'initialisation de l'agent *Teacher-Agent*. En effet, pour chaque implémentation JADE mise en œuvre, une classe de lanceur devrait être définie pour initialiser les différents agents.

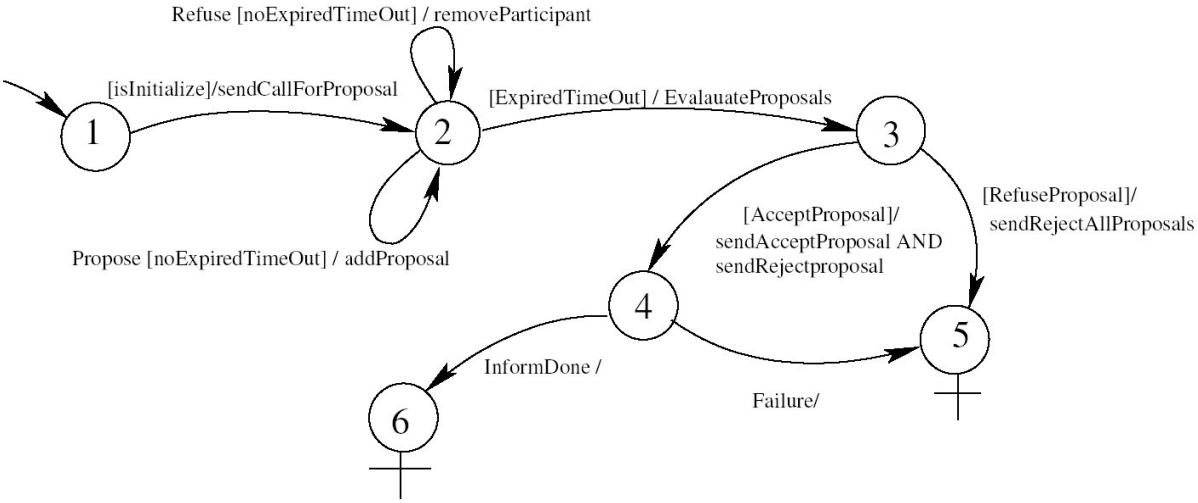


FIG.6 – Machine d'états finis représentant le rôle initiateur du CNP.

La mise en œuvre des caractéristiques *CPN-Initiator* et *CPN-Participant* qui sont les deux rôles liés au CPN, est basée sur la réutilisation de la classe *jade.core.Behaviour* de JADE. Pour rendre le comportement associé générique, nous utilisons le *FSMBehavior*. Chaque rôle est donc mis en œuvre comme une sous-classe de cette classe.

La Figure 6 donne un exemple de ce FSM à travers sa machine d'état associée. Elle s'appuie sur un contrat, une stratégie d'évaluation, une liste de participants et ne nécessite pas une autre entrée pour être exécutée.

4.3.2 *Dérivation de variantes*

A partir du modèle des caractéristiques des SMA et de tous les objets logiciels associés, différentes variantes peuvent être dérivées automatiquement sur la base des mécanismes de composition, comme ceux définis dans FeatureHouse [1].
La dérivation des variantes est implémentée en deux étapes :

- créer une configuration valide du FM qui contient la sélection des fonctionnalités pour une variante spécifique,
- composer les artefacts logiciels qui sont associés à la fonction sélectionnée, et ce, à l'aide du compositeur.

Pour le SMA de l'emploi du temps, plusieurs configurations valides peuvent être créées à partir des caractéristiques du modèle de la figure 5.

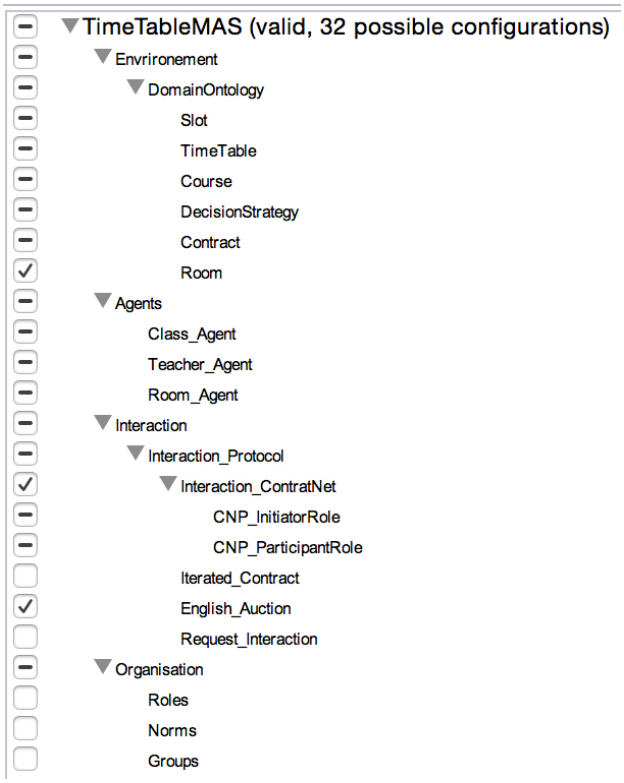


FIG.7 –Exemple de configuration valide pour une variante de l'emploi du temps.

La figure 7 montre un exemple d'une telle configuration valide qui concerne une variante du SMA de l'emploi du temps supportant deux types de protocoles d'interaction : *ContratNet* et *English_Auction*. Cette variante gère également les contraintes de salles parce que les deux caractéristiques *Room* et *Room-Agent* sont sélectionnées. Cependant, toutes les caractéristiques qui sont liées à la dimension Organisation ne sont pas présentes.
De la configuration valide créée, FeatureHouse compose le code Java pour toutes les caractéristiques choisies afin de générer le code source de la variante cible.
La génération de code est basée sur le raffinement du code [1].

5. Conclusion et perspectives

Nous avons introduit dans ce papier une approche LdP-SMA, nommée Id-LdPMAS qui permet le développement incrémental basé sur les caractéristiques et leur implémentation, et qui comble le fossé existant entre la modélisation et l'implémentation des SMA. Pour outiller cette approche, nous avons utilisé FeatureHouse, un framework de LdP qui permet de générer des variantes du système à partir des caractéristiques. Pour l'implémentation des SMA, nous avons utilisé JADE. Chaque caractéristique est associée au code JADE et/ou Java.
Id-LdPMAS utilise un ensemble de caractéristiques qui peuvent être déduites de quelques modèles SMA existants. L'identification de ces caractéristiques et leur catégorisation sont réalisées sur la base du paradigme Voyelles. Par conséquent, l'approche pourrait être adoptée par les développeurs utilisant n'importe quelle méthode pour créer des modèles de SMA, y compris les méthodes situationnelles [7].
Id-LdPMAS a été validé sur l'exemple de l'emploi du temps avec la plate-forme JADE. En perspectives nous projetons : (i) d'analyser l'impact sur l'identification des caractéristiques lors de changement de plateforme ; (ii) de considérer d'autres exemples de SMA qui s'articuleraient autour de modèles organisationnels tel que celui des fourmis et du champs potentiels afin d'enrichir le modèle avec différents mécanismes d'interaction et d'auto-organisation également ; (iii) de

proposer des lignes directrices et des tutoriels accompagnés d'exemples pour les développeurs des applications SMA, afin de les encourager à adopter les LdP-SMA en les aidant à construire leur propre LdP-SMA étape par étape ; (iv) de proposer l'adoption de l'agilité tout en la combinant avec notre approche, ce qui serait intéressant pour concevoir une véritable méthode pour les SMA.

Remerciements

Anarosa A. F. Brandão est financée par une bourse #014/03297-7, São Paulo Research Foundation (FAPESP).

References

[1] S. Apel, C. Kästner, C. Lengauer, Language-Independent and Automated Software Composition: The FeatureHouse Experience. *IEEE Transaction on Software Eng*ineering 39(1): 63-79, 2013

[2] S. Apel, D. Batory, C. Kästner, G. Saake: *Feature-Oriented Software Product Lines - Concepts and Implementation*. Springer, 2013.

[3] D. Benavides, S. Segura, and A. R. Cortés, Automated analysis of feature models 20 years later: A literature review, *Information Systems*, vol. 35, no. 6, pp. 615–636, 2010.

[4] F. Bergenti, M-P. Gleizes and F. Zambonelli, *Methodologies and Software Engineering for Agents Systems*, Kluwer Publishing, 2004.

[5] C. Bernon, M-P Gleizes, P. Glize, G. Picard, Le problème de l'emploi du temps - Cahier des charges, In C. Bernon, V. Camps, M-P. Gleizes, P. Glize, S. Peyruqueou, G. Picard. Ingénierie des AMAS pour l'emploi du temps. ETTO - Emergent TimeTable Organization. *Rapport de recherche, ASA-PRC IA, IRIT*, 2002.

[6] S. Casare, A. A. F. Brandão, Z. Guessoum, J. Sichman, Medee Method Framework: a situational approach for organization-centered MAS. Autonomous Agents and Multi-Agent Systems, 28(3): 430-473, 2014.

[7] E. Cirilo, I. Nunes, U. Kulesza, C. Lucena, Developping Multi-Agent System Product Lines: From Requirements to Code. *IJAOSE*, pp. 197-216, 2011.

[8] Clements and L. Northrop, *Software product lines: practices and patterns*. Addison-Wesley Longman Publishing Co., 2001.

[9] J. Dehlinger, R. Lutz, Gaia-PL: A Product Line Engineer-ing Approach for Efficiently Designing Multi-Agent Systems. *ACM Transaction on Software Engineering Methodol*ogies 20(4): 17 (2011)

[10] Y. Demazeau, From interactions to collective behavior in agent-based systems. *Proceedings of the 1st. European Conference on Cognitive Science*. Saint-Malo, p. 117-132, 1995.

[11] J. Ferber, O. Gutknecht, F. Michel, From Agents to Organizations: An Organizational View of Multi-agent Systems. In P. Giorgini, J.P. Müller, J. Odell (Eds.): *Agent-Oriented Software Engineering*, LNCS 2935, pp. 214–230, Springer, 2004.

[12] Z. Guessoum, M. Cossentino and J. Pavon Mestras. A Roadmap of Agent-Oriented Software Engineering: The European Agentlink Perspective. In "*Methodologies and Software Engineering for Agents Systems*", Bergenti ey al (eds.), Kluwer pp. 430-450, 2004.

[13] T. Jarraya, Z. Guessoum, Towards a Model Driven Process for Multi-Agent System. *Proc.of CEEMAS 2007*: 256-265

[14] K. Kang, S. Cohen, J. Hess, W. Nowak, S. Peterson, Feature-Oriented Domain Analysis (FODA) Feasibility Study (Report). *CMU/SEI-90-TR-21*.

[15] I. Nunes, C. Lucena, D. Cowan, U. Kulesza, P. Alencar, C. Nunes, Developing Multi-agent System Product Lines: From Requirements to Code, Agent-Oriented Software Engineering, 4(4), 353-389, 2011.

[16] J. Peña, M. Hinchey, M. Resinas, R. Sterritt, J. Rash, Designing and managing evolving systems using a MAS product line approach. Sci. Comput. Program. 66(1), 71-86, 2007

[17] J. Peña, M. Hinchey, A. Ruiz-Cortés, Multi-agent system product lines: challenges and benefits. Communications of the ACM, Vol. 49 No. 12, Pages 82-84. 2006.

[18] M. Cossentino. N. Gaud, V. Hilaire, S. Galland, A. Koukam. ASPECS: an agent-oriented software process for engineering complex systems. In Autonomous Agents and Multi-Agent Systems, 20(2), 260-304, 2010.

Simulation de smart grids avec MECSYCO

J. Vaubourg[a] Julien.Vaubourg@inria.fr
Y. Presse[a] Yannick.Presse@inria.fr
B. Camus[b] Benjamin.Camus@loria.fr
L. Ciarletta[a,b] Laurent.Ciarletta@loria.fr
V. Chevrier[b] Vincent.Chevrier@loria.fr
J-P. Tavella [c] Jean-Philippe.Tavella@edf.fr
B. Deneuville [c] Boris.Deneuville@edf.fr
Olivier.Chilard [c] Olivier.Chilard@edf.fr

[a]Centre de Recherche INRIA Nancy Grand Est,
Villers-les-Nancy France

[b]LORIA UMR 7503
Vandoeuvre-les-Nancy, France

[c]EDF R&D,
Clamart, France

Résumé

Cette démonstration présente l'application de la plateforme de simulation MECSYCO (Multi-agent Environment for Complex SYstems CO-simulation), antérieurement appelée AA4MM, dans le contexte des smart grids. Nous montrons, notamment à partir d'un cas d'usage tiré d'un scénario réel, comment cette plateforme permet d'intégrer des simulateurs hétérogènes existants et de simuler l'ensemble de manière cohérente et complètement décentralisée.

Mots-clés : *Simulation multi-agent, smart grids, multi-modélisation, agents et artéfacts*

Abstract

This demonstration presents the use of the MECSYCO simulation platform, previously named AA4MM. From use cases based on real scenarios, we show how this platform is able to integrate existing heterogeneous simulators and simulate the whole coherently and in a decentralized way.

Keywords: *Multi-agent simulation, smart grids, multi-modeling, agents and artifacts*

1 Simulation de smart grids

Simuler un smart grid implique d'intégrer plusieurs domaines d'expertise ; a minima, le réseau électrique de distribution, les télécommunications entre composants et les systèmes d'information et de décision. Chaque domaine possède ses propres outils de simulation, et plusieurs outils peuvent cohabiter dans un même domaine. Le défi central est alors la multi-simulation, i.e. exécuter cet ensemble d'outils de simulation hétérogènes comme un tout cohérent.

Nous montrons les apports d'une approche multi-agent pour la multi-simulation au travers de la plateforme MECSYCO dans le contexte du projet MS4SG (Multi-Simulation for Smart Grids) entre le LORIA-INRIA et EDF R&D.

MECSYCO utilise les concepts d'Agents et Artéfacts pour décrire une multi-simulation : l'ensemble des simulateurs en interaction correspond à un système multi-agent. Chaque agent (appelé M-agent) gère un simulateur (initialisation, avancement du temps de simulation, gestion des entrées/sorties) grace à un artéfact de modèle et ses interactions avec les autres simulateurs grâce à des artéfacts de couplage. Ces différents concepts sont formalisés en DEVS (Discrete Event System Specification), formalisme qui assure l'intégration de formalismes hétérogènes. La coordination entre les différents M-agents correspond à l'implantation du protocole de simulation parallèle conservatif de DEVS basé sur l'algorithme de Chandy-Misra permettant alors une coordination complètement décentralisée.

Ces concepts sont instanciés sous la forme de bibliothèques Java et C++ qui autorisent plusieurs déploiements de la multi-simulation : d'un processus unique (en Java ou C++) à une exécution hybride (Java/C++), distribuée (plusieurs machines) et décentralisée.

2 Le cas Concept-Grid

Concept-Grid est un cas d'usage couvrant à la fois la modélisation et la simulation d'un scénario d'effacement cascado-cyclique et sa réalisation, en grandeur réelle, sur le réseau électrique expérimental Concept-Grid à EDF Lab Les Renardières.

Le système (figure 1) comprend cinq maison-

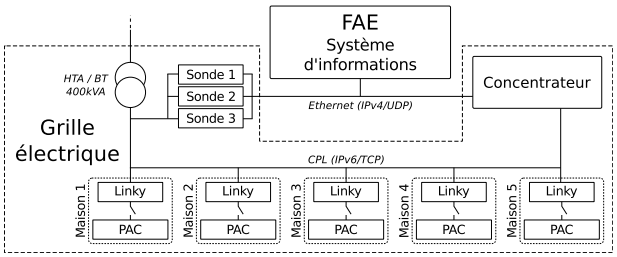


FIGURE 1 – Vue simplifiée de Concept-Grid.

Domaines	Composants	Simulateurs	Plate-formes	Langages	Formalismes	Echelles Temporelles
Télécom	TCP/IPv6/CPL	NS-3	GNU/Linux	C++	Evénement	Nanoseconde
	UDP/IPv4/Ethernet					
	Noeuds (FAE, Linkys, Sondes, Concentrateur)					
SI	FAE	Ad-hoc	Indifférent	Java	Automate	Seconde
	Linky	FMUs	Windows	Java	Equationnel	Seconde
Réseau électrique	PACs	FMU	Windows	Java	Equationnel	Seconde
	Transfo. HTA/BT					
	Lignes électriques					
	Charges (maisons)					
	Sondes					

FIGURE 2 – Les différents types d'hétérogénéité.

nettes, chacune équipée d'un compteur intelligent (Linky) raccordé à une pompe à chaleur (PAC) : on peut connecter/déconnecter la PAC en envoyant un ordre au compteur à partir d'un centre décisionnel (Fonction d'Effacement Avancée). La communication entre centre de décision et compteur se fait par courant porteur en ligne (CPL). Le scénario consiste à effacer les pompes à chaleur des maisonnettes une à une en suivant un ou plusieurs cycles d'effacement et vérifier à partir de remontée d'informations que l'effacement permet effectivement de limiter la consommation.

Concept-Grid est un cas représentatif des défis de la multi-simulation car il combine simultanément plusieurs types d'hétérogénéité : trois domaines d'expertise interviennent ; plusieurs types de simulateurs doivent interagir ; ces simulateurs s'appuient sur des formalismes différents (équationnel ou événementiel) ; il utilise des échelles temporelles différentes (secondes et nanosecondes). De plus, des contraintes logicielles et matérielles doivent être respectées en terme de plateforme (Windows ou GNU/Linux) et de langage d'interfaçage (Java/C++). La figure 2 récapitule ces différents points.

3 Démonstration

La démonstration a pour objectif de montrer comment l'approche Agents et Artéfacts utilisée

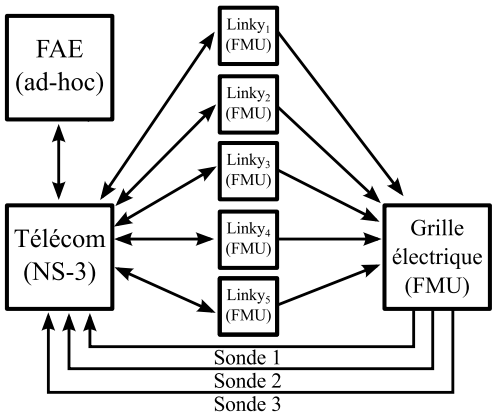


FIGURE 3 – Schéma d'échanges entre simulateurs.

par MECSYCO permet de gérer simultanément ces différents défis au travers d'un cas concret applicatif de smart grids. De plus, des preuves de concept compléteront la démonstration.

La simulation de ce cas fait intervenir plusieurs simulateurs existants : le réseau de télécommunication est simulé avec le logiciel NS-3 qui intègre et modélise les communications sur un lien ethernet en UDP/IPv4 et les communications sur le lien CPL en TCP/IPv6 ; le réseau de distribution et les PACs sont simulés avec une FMU [1] ; le système de décision est un automate écrit de manière ad-hoc en Java. Les compteurs Linky sont modélisés dans les trois domaines à la fois : au niveau électrique dans la FMU du réseau de distribution, au niveau télécom dans NS-3 et au niveau applicatif grâce à une instance de FMU dédiée pour chaque compteur.

Les échanges d'informations entre simulateurs correspondent aux flux proposés en figure 3.

4 Conclusion

Une version Java du cœur avec sa documentation ainsi que des exemples de mise en œuvre (preuves de concept) sont disponibles depuis début 2015, sous licence AGPL 3.0.

Un ensemble d'outils de méta-modélisation, de visualisation durant l'exécution et post-mortem, des wrappers FMI, une version du cœur C++ seront mis à disposition au fur et à mesure de l'année 2015 pour expérimentation et validation.

Ce travail a été en partie financé par EDF R&D par le biais du projet MS4SG.

1. Functional Mock-up Units : il s'agit d'un format d'export d'un ensemble d'équations associé à un solveur.

Croyances volatiles pour l'adaptation collective de véhicules autonomes : application à une cellule de production flexible.

E. Adam
emmanuel.adam@univ-valenciennes.fr

Laboratoire d'Automatique, de Mécanique et d'Informatique industrielle et Humaine, UMR CNRS
Université de Valenciennes et du Hainaut-Cambrésis, France

1 Introduction

Pour être compétitives, les industries manufacturières doivent s'adapter aux conditions changeantes imposées par le marché. Au cours des dernières décennies, les progrès scientifiques dans le domaine de la gestion de production ont mené à la définition de nouvelles architectures distribuées qui jouent un rôle de premier plan dans les FMS (Flexible Manufacturing Systems). Plusieurs approches bio-inspirées ont été proposées, et sont proposées pour permettre l'adaptation des éléments d'un système de production aux contraintes d'un environnement dynamique ; nous proposons une approche permettant d'éviter l'utilisation d'un environnement partagé et d'une couche de contrôle. En effet, dans la perspective d'une mise en œuvre réelle physiquement distribuée au sein de véhicules autonomes, qui évolue selon un graphe uni-directionnel, et qui peuvent être concurrents (comme les convoyeurs automatisés dans les cellules de production flexibles), nous tentons d'éviter au plus l'utilisation d'un environnement partagé, et d'une couche de contrôle.

En se basant sur la notion de stigmergie, nous proposons la notion de croyance volatile. En fait un coefficient de dégradation est appliqué non pas à la valeur de la phéromone, mais à la confiance accordée à la croyance. Chaque agent dégrade ainsi les confiances accordées à ses croyances. Lorsqu'une croyance a un niveau de confiance trop bas, l'agent la retire de sa liste des croyances.

2 Croyances volatiles pour la gestion de Cellule de Production Flexibles

Nous considérons une cellule de production (qui existe réellement dans notre université) qui est composée de navettes autonomes recevant des ordres (commandes) de produits (blocs s_1, ..., s_8 dans la figure 1). Un produit est composé de parties assemblées par des stations de travail sur une navette, il y est retiré après avoir été vérifié. Une station (blocs noirs dans la figure 1) peut poser des parties de types différents avec un temps variable selon ses capacités. Les rails utilisés par les navettes pour joindre les stations sont simples et uni-directionnels. Des aiguillages (disques N_1, ..., N_{11} dans la figure 1) pilotables à distance par les navettes leurs permettent de se diriger vers leurs meilleurs chemins. En effet, les navettes tentent de trouver la meilleure station permettant d'obtenir la prochaine partie du produit. La station la plus utile pouvant être une combinaison des facteurs de proximités, de temps de traitement et de file d'attente au pied des stations. Dans notre approche, les navettes s'échangent leurs prochaines intentions (le nom de la prochaine station choisie), ainsi que les informations sur les événements (dégradation/amélioration du temps de traitement sur une station).

Pour donner un très bref aperçu de notre définition d'une croyance, une croyance sur un objet ou un agent est une vue partielle de cet élément. Elle est également définie par sa : date de création ; son niveau de confiance ; un degré de dégradation de celle-ci ; un seuil de confiance sous lequel la croyance est supprimée ; un indicateur de pérennité ; un mode d'agrégation. Un agent peut posséder plusieurs croyances sur un même objet avec des valeurs de confiance différentes. C'est à dire qu'un agent peut posséder plusieurs points de vue différents d'un même objet. Dans le cas d'agents plongés dans un environnement industriel, il n'y a pas de découverte ; à la base un agent possède une image initiale du système ; cette croyance est pérenne, l'agent ne l'oubliera jamais. A chaque cycle de vie d'un agent, il dégrade ses croyances non pérennes selon le degré de dégradation associé. Par conséquent, si aucun nouvel événement intervient, il ne reste

dans l'agent que les croyances initiales et pérennes sur l'environnement.

Relativement à la gestion de cellule de production, deux types de croyances sont utilisées : croyances sur l'utilisation future d'une ressource par d'autres agents (en ce cas, l'agent cumule les temps d'utilisation des autres agents pour en déduire le nouveau temps nécessaire pour lui pour subir un traitement sur cette ressource) ; croyance sur un changement important d'état d'un objet (l'agent prend alors en considération la croyance la plus récente sur l'objet).

3 Simulation.

Nous avons développé un simulateur en JADE (l'objectif étant de porter les agents développé et les placer directement sur les navettes réelles dont nous disposons) ; avec une interface en JavaFX ; et en utilisant le logiciel JOSM pour créer le graphe représentant l'atelier, ses stations et leurs spécialités.

Il est possible de paramétrer les commandes de produits, le nombre d'agents devant les réaliser. Il est également possible de modifier les coefficients de dégradations, le seuil d'oubli ; ainsi que les informations communiquées (future utilisation d'une station, utilisation courante, problème ou restauration d'une station) ; et les moments où les messages sont échangés (en entrée et/ou sortie de station, aux passages sur les aiguillages).

Chaque agent prend la responsabilité d'une commande (construire un mot, composé de lettres, composées de parties). Il crée la liste des stratégies possibles permettant de réaliser cette commande (listes de chemins entre stations de travail) et choisit la stratégie la plus courte. Sans communication entre eux, les agents se gênent et créé des files aux stations les plus intéressantes initialement. Pour un scénarion simple, composé de 8 mots, d'environ 4 lettres de 7 parties chacun (donc une commande nécessitant la pose de 164 parties), la communication des objectifs et le recalcul des temps associés aux stratégies (à chaque passage sur un point d'aiguillage et en sortie de station) améliorent le temps pour la création de tous les produits de 17%. Lorsqu'une panne est provoquée sur une des machines (en l'occurrence la machine S_2) pendant un temps assez long ($1/5^{eme}$ du temps total), sans communication l'impact négatif est de 54% !, tandis que sur le mode avec communication et croyances volatiles, l'impact n'est que

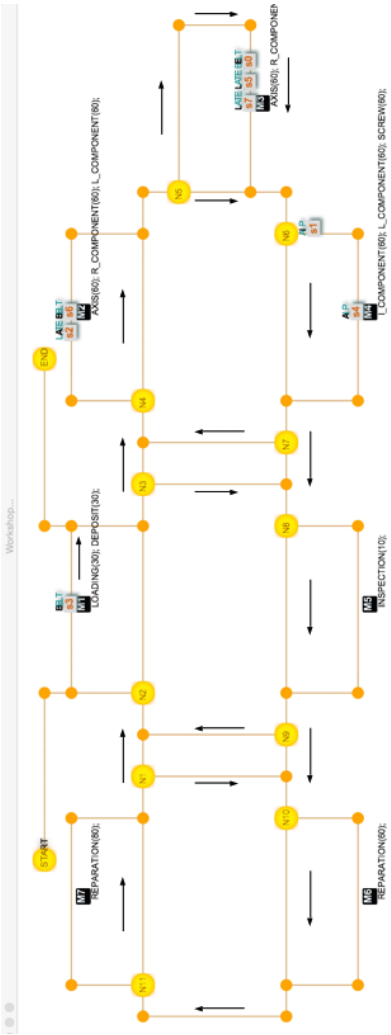


FIGURE 1 – Copie d'écran du simulateur de cellule de production.

de 32%. En "jouant" avec le coefficient de dégradation et en le diminuant, l'impact négatif de la panne n'est plus que de 25%. Ceci est dû au cas particulier d'une panne longue à rétablir ; les agents oubliant trop vite qu'il y a une panne estime de nouveau qu'un passage par la station comme est plus utile qu'une autre solution et s'y dirigent alors qu'elle n'est pas réparée. Inversement, si la panne se rétablit vite et que les agents y croient toujours, ils continuent d'éviter une station pourtant en fonctionnement normal.

Les résultats montrent donc que cette nouvelle approche donne des résultats corrects même en mode dégradé, et de manière réactive.

Nous avons également testé cette approche avec succès sur des robots physiques Wifibots qui évoluent dans un environnement perturbé, sans contrôle centralisé.

Yet Another Traffic Simulator : Outil multi-agent de prototypage pour les niveaux opérationnels et tactiques de la conduite

Julien Saunier
julien.saunier@insa-rouen.fr

LITIS, INSA de Rouen,
Avenue de l'Université - BP8, 76801 Saint-Étienne-du-Rouvray Cedex - France

Mots-clés : *Systèmes multi-agents, simulation de trafic, agent incarné, environnement, architecture*

1 Motivation

YATS (*Yet Another Traffic Simulator* [1]) est un simulateur de trafic utilisé pour le prototypage de modèles de trafic en 1,5D (réseau représenté sous forme de graphe à largeur non contrainte pour permettre l'étude des comportements latéraux) selon une approche incarnée permettant la différenciation entre modèle de décision au niveau stratégique/tactique et sa mise en œuvre au niveau opérationnel.

La mise à disponibilité de données de plus en plus précises sur le trafic, notamment la base NGSIM [2], ou l'extraction de trajectoires aux carrefours par capteurs vidéos, permet d'obtenir des informations fines sur la trajectoire des véhicules. Les modèles de simulation microscopiques qui étaient validés sur des données agrégées de trafic peuvent désormais être confrontés à ces nouvelles données.

Traditionnellement en simulation de trafic, l'unité considérée est le couple conducteur-véhicule qui se déplace sur un réseau. Les modèles de trafic doivent ainsi prendre en compte à la fois des paramètres physiques du véhicule (capacité d'accélération, de freinage...) et des paramètres comportementaux (capacités du conducteur, cycle cognitif...). Décomposer ces deux niveaux dans un modèle agent a alors pour objectif de valider individuellement les différents niveaux entrant en jeu dans le déplacement. Pour cela, nous proposons de considérer les niveaux tactiques (décisions de court terme) et opérationnels (actions) comme représentatifs respectivement du conducteur et du véhicule. Ceci sera appliqué à l'étude du comportement latéral du véhicule au sein de sa voie, généralement non considérée dans les modèles de trafic classiques.

L'approche incarnée [6] a, depuis les années 80, apporté une nouvelle façon de concevoir l'Intelligence Artificielle (IA), en contrepoint de l'approche cognitiviste traditionnelle. Quand cette dernière conçoit l'intelligence du système comme celle de son unité de contrôle, l'incarnation démontre la possibilité de simplifier le contrôle par une utilisation des capacités du corps, des lois de l'environnement et de l'interaction entre ces trois éléments (esprit, corps, environnement).

2 Principes

Le simulateur YATS repose sur la médiation des interactions entre l'agent et l'environnement par un troisième composant, le corps de l'agent (voir figure 1). De cette façon, à la manière de [1], les propriétés et capacités des agents sont stockées dans leur corps de façon à externaliser la gestion de leur hétérogénéité. La possibilité de réussite partielle d'une tentative d'action par l'agent [3, 4, 5] implique la nécessité de rétroactions, de façon à ce que l'agent s'adapte aux capacités de son corps.

Dans le cadre de la conduite, les variations latérales au sein des voies ne peuvent être prises en compte que par une modélisation de l'espace en 2 dimensions. Pour simplifier la dynamique des véhicules, nous proposons de nous reposer sur une structure de graphe dans laquelle le véhicule peut se déplacer latéralement de façon relative au segment de route qui le porte.

La gestion des virages utilise la notion d'affordance [2] : en arrivant sur un noeud entre deux arcs, le noeud contient lui-même une proposition de vitesse maximale permettant de le passer en sécurité. Le véhicule adapte alors sa vitesse

1. https://github.com/Ifsttar/YATS
2. http://ngsim-community.org/

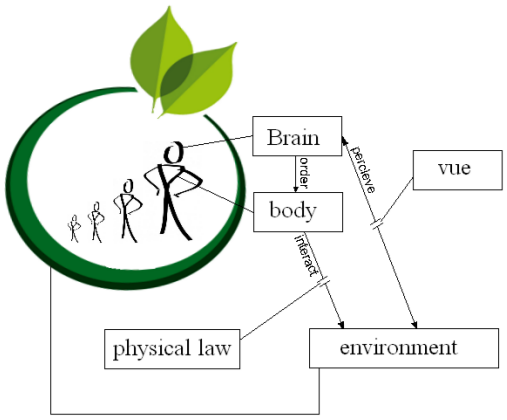


FIGURE 1 – Esprit, corps et environnement dans une architecture pour la simulation de trafic.

en tenant compte de cette affordance et de ses propriétés, tandis que la trajectoire résultante est contrôlée par l'environnement en respectant les grandeurs physiques mises en oeuvre (capacités de freinage, d'angle de braquage, d'adhérence...).

3 Éléments actuels et futurs

YATS est un outil de prototypage rapide diffusé sous licence GPL 3.0. Il est implémenté en C++, et utilise les bibliothèques Boost (pour les calculs physiques) et SDL (pour la partie graphique). La génération des perceptions et la gestion des prises de virage est implémentée, ainsi qu'une première version des interactions entre véhicules sur une même voie. Un ensemble de circuits permettant la vérification de comportements unitaires est également disponible, ainsi que les points d'apparition et disparition des véhicules.

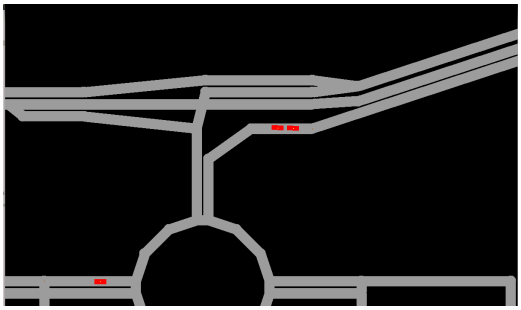


FIGURE 2 – Exemple de circuit.

À court terme, la gestion des interactions en intersections, ainsi que le cas multi-voie doivent être traités. Ceci permettra d'utiliser YATS pour la validation de l'effet de friction entre véhicules : l'espace latéral disponible, avec l'infrastructure et les autres véhicules, contraint et modifie le comportement longitudinal du conducteur.

À moyen terme, l'utilisation d'un format standard -tel que RoadXML ou OpenRoad- permettrait l'import rapide de nouveaux circuits.

Remerciements

Le développement du simulateur YATS entre dans le cadre du projet PROFIL, opération réalisée avec l'aide financière de la fondation Sécurité Routière, en partenariat avec le CEREMA (DTer Ouest et Dter Normandie Centre) et l'IFSTTAR (laboratoire COSYS-LEPSIS).

Références

[1] F. Behe, S. Galland, N. Gaud, C. Nicolle, and A. Koukam. An ontology-based metamodel for multiagent-based simulations. *International Journal on Simulation Modelling, Practice, and Theory*, 40 :64–85, Jan. 2014.

[2] J. J. Gibson. *The Ecological Approach to Visual Perception*. Lawrence Erlbaum Associates, 1979.

[3] F. Michel. Le modèle irm4s, de l'utilisation des notions d'influence et de réaction pour la simulation de systèmes multi-agents. *Revue d'Intelligence Artificielle*, 21(5-6) :757–779, 2007.

[4] E. Platon, N. Sabouret, and S. Honiden. Tag interactions in multiagent systems : Environment support. In *Proceedings of Environment for Multi-Agent Systems, Workshop held at the Fifth Joint Conference in Autonomous Agents and Multi-Agent Systems*, volume 4389 of *Lecture Notes in Artificial Intelligence*, pages 106–123. Springer Verlag, 2007.

[5] J.-C. Soulié. *Vers une approche multi-environnements pour les agents*. PhD thesis, Université de la Réunion, 2001.

[6] F. J. Varela, E. Rosch, and E. Thompson. *The embodied mind : Cognitive science and human experience*. MIT press, 1992.

Composition dynamique de services Web: une approche agents fondée sur la confiance et les coalitions

A. Louati
amine.louati@lamsade.dauphine.fr

J. El Haddad
elhaddad@lamsade.dauphine.fr

S. Pinson
pinson@lamsade.dauphine.fr

PSL, Université Paris-Dauphine, CNRS LAMSADE UMR 7243, France

Nous nous intéressons dans ce travail à la composition de services Web. Dans un environnement distribué comme les réseaux sociaux, la composition de services est un processus complexe qui nécessite le raisonnement et la coopération. La technologie multi-agents est bien adaptée à la modélisation de ce processus car elle permet de promouvoir l'interaction, la représentation des connaissances et le raisonnement, ainsi que des métaphores sociales comme la confiance.

La composition de services Web implique une phase de découverte de services. Dans nos travaux précédents [4], nous avons proposé une approche de découverte de services fondée sur la confiance entre agents. À partir des relations entre les agents, de leurs expériences antérieures et des informations extraites du réseau social, nous définissons un modèle de confiance comme étant un concept compositionnel formé de trois composantes. (1) Une composante sociale évaluant la confiance en la crédibilité sociale $SC(a_k, a_j)$ d'un agent a_j selon le point de vue d'un agent a_k en agrégeant trois mesures : sa position sociale $SPo(a_j)$, sa proximité sociale $SPr(a_k, a_j)$ et sa similarité sociale $SSi(a_k, a_j) = NS(a_k, a_j) \times PS(a_k, a_j)$ qui est le produit de leur similarité de voisinages et leur similarité de profils. (2) Une composante évaluant la confiance en expertise $EC(a_k, a_j, s_{jl})$ d'un agent a_j pour un service s_{jl} selon le point de l'agent a_k. C'est le produit de trois attributs de QoS : la spécialisation $Sp(s_{jl})$, la fiabilité $Re(s_{jl})$ et la qualité $Eval_x(a_k, s_{jl})$ (pour plus de détail, voir [4]). (3) Une composante évaluant la confiance en la recommandation $RC(a_k, a_j, s_{pl})$ d'un agent a_j pour un service s_{pl} selon le point de l'agent a_k. C'est une agrégation de deux mesures : une mesure objective $[r_{kj}|s_{pl}] \in [0, 1]$, qui indique la performance de a_j par rapport au nombre de bonnes recommandations du service s_{pl} et une mesure subjective $[q_{kj}|s_{pl}] \in [0, 1]$, qui reflète la satisfaction de l'agent a_k concernant les recommandations de a_j pour le service s_{pl}. La propagation de la recherche dans le réseau social se fait d'une manière progressive à l'aide d'un système de références où les agents utilisent ce modèle de confiance pour évaluer la fiabilité d'autres agents et décider par la suite d'interagir avec eux ou pas. Plus précisément, Pour chaque agent, nous adoptons une architecture délibérative composée de cinq modules : un module de raisonnement, un module de confiance, un module de contrôle, un module de planification et un module d'interaction et deux bases de données : une base de croyances et une bibliothèque de plans et une pile de buts.

À la fin de cette étape de découverte, le demandeur de services a_r garde localement un ensemble de fournisseurs dignes de confiance P ainsi que les services qu'ils offrent S sur lequel il va lancer un processus de formation de coalitions pour répondre à ses besoins. Peu de travaux [1, 2, 3, 5, 6] ont proposé un processus de formation de coalitions adaptative pour la composition de services. Dans ces travaux, le processus comprend un modèle de négociation basé sur les attributs de QoS afin de fournir un service composite qui répond aux besoins du demandeur. Les agents sont autonomes au niveau de leurs décisions : participer ou non à une coalition et accepter ou refuser l'adhésion d'un candidat mais ils n'intègrent pas la contribution de la dimension sociale dans la sélection des membres. La sélection d'un membre se fonde uniquement sur les valeurs de QoS des services qu'il offre. Cependant, les demandeurs préfèrent souvent les fournisseurs qui proposent non seulement des services requis mais aussi qui sont dignes de confiance. La capacité à réaliser une sélection efficace est une exigence majeure pour la composition de services. La confiance est reconnue comme étant un mécanisme efficace pour réduire le risque perçu dans les interactions entre les agents. C'est dans cette perspective que Bourdon et al. ont introduit dans [1] une approche de formation de coalition cen-

trée sur la confiance des fournisseurs pour une composition de services. Leur approche proposée est statique et effectuée par un protocole de négociation inspiré d'une gestion multi-agent de confiance permettant aux fournisseurs de choisir leurs partenaires dans une composition de services. Comme Bourdon et al., nous proposons dans ce travail un modèle multi-agents basé sur un broker pour la composition des services. La composition de service est effectuée par des agents égoïstes équipés d'un ensemble de services avec leurs valeurs de QoS. Ces agents sont engagés dans une formation de coalition où la prise de décision est guidée par la confiance. Cependant, nous voulons aller au-delà du caractère statique de la composition en intégrant le dynamisme et la dimension sociale afin d'améliorer la flexibilité et l'adaptation de la solution. Pour ce faire, nous proposons une nouvelle mesure de confiance appelée la confiance en la coopération (CC) permettant aux agents de raisonner sur la fiabilité d'autres agents avant de coopérer avec eux. Basée sur l'historique des anciennes coalitions, la valeur de confiance $CC(a_k, a_j)$ d'un agent a_j selon le point de vue d'un autre agent a_k est définie par : $CC(a_k, a_j) = \frac{Nb_{adh_{jk}}}{Nb_{soll_{kj}}}$ où $Nb_{adh_{jk}}$ est le nombre d'adhésions de a_j à la coalition dans laquelle a_k est un membre et $Nb_{soll_{kj}}$ est le nombre de sollicitations de a_k à a_j pour rejoindre sa coalition. Notre processus de formation de coalitions (PFC) est un processus :

– Incrémental : la construction d'une coalition se fait d'une manière itérative où à chaque itération un seul agent peut rejoindre la coalition.
– Dynamique : les agents peuvent rejoindre et quitter une coalition d'une manière autonome à tout moment en fonction de leurs décisions locales.
– Recouvrante : chaque agent peut être membre de plusieurs coalitions à la fois.

Le PFC se compose de trois phases organisées en ordre séquentiel décrit par l'algorithme 1. Dans la première phase, le demandeur de services a_r génère un ensemble de coalition initiales singletons $\mathbb{C}$ formées chacune d'un seul membre de son ensemble d'accointances. Dans la deuxième phase, il essaye de compléter chacune des coalition initiales singletons $c_i \in \mathbb{C}$ par des membres fournissant les services nécessaires à la composition de services. Notons que les agents sont complètement autonomes dans leur décisions de rejoindre/quitter une coalition ou d'accepter l'adhésion d'un candidat. Dans la dernière phase, a_r va sélectionner la meilleure coalition parmi celles générées dans la phase précédente. Pour ce faire, il établit un classement selon la confiance en l'expertise de chacune de ces coalitions. La confiance en l'expertise d'une coalition donnée est définie comme étant la somme de celles de ses membres.

Algorithm 1: Algorithme de formation de coalitions

Variables: P est l'ensemble des fournisseurs, S est l'ensemble des services découverts, $\mathbb{C}$ est l'ensemble des coalitions et c_i^* est la meilleure coalition.

```
1 ℂ ← coalitions_generation(P, S) ;
2 for all (c_i ∈ ℂ) do
3     members_selection(c_i) ;
4 c_i* ← pick_best(ℂ) ;
```

Dans ce travail, nous avons présenté un mécanisme original pour une composition dynamique de services satisfaisant les besoins complexes d'un demandeur dans les environnements distribués comme les réseaux sociaux. Ce mécanisme est effectuée par une approche agents fondée sur la confiance dans laquelle les agents coopèrent dans des coalitions en se fondant sur des connaissances limitées et une prise de décision décentralisée.

Références

[1] J. Bourdon, L. Vercouter, and T. Ishida. A multiagent model for provider-centered trust in composite web services. In *PRIMA*, pages 216–228. Springer, 2009.

[2] V. Ermolayev, N. Keberle, and S. Plaksin. Towards agent-based rational service composition - racing approach. In *ICWS-Europe*, pages 167–182. Springer, 2003.

[3] N. Griffiths and M. Luck. Coalition formation through motivation and trust. In *AAMAS*, pages 17–24. ACM, 2003.

[4] A. Louati, J. El Haddad, and S. Pinson. A multilevel agent-based approach for trustworthy service selection in social networks. In *IAT*, pages 214–221. IEEE Computer Society, 2014.

[5] I. Muller, R. Kowalczyk, and P. Braun. Towards agent-based coalition formation for service composition. In *Proceedings of the IEEE/WIC/ACM international conference on Intelligent Agent Technology*, pages 73–80. IEEE Computer Society, 2006.

[6] H. Tong, J. Cao, S. Zhang, and M. Li. A distributed agent coalition algorithm for web service composition. *2014 IEEE World Congress on Services*, pages 62–69, 2009.

Utilisation de la Reconnaissance d'Intentions pour Choisir Prudement une Stratégie Collective dans le Contexte des SMA Embarqués

E. M. Khalfi J-P. Jamont M. Occello

Université de Grenoble Alpes, LCIS, F-26000 Valence, FRANCE
el-mehdi.khalfi@lcis.grenoble-inp.fr

1 Motivation

Les systèmes multi-agents s'intéressent aux problèmes dans lesquels des entités en interaction cherchent à accomplir des buts individuels et collectifs. Dans le contexte particulier des SMA plongés dans le monde "physique", l'environnement est caractérisé par la possible cohabitation, au sein d'un même espace, de plusieurs systèmes distincts. L'environnement d'un SMA n'est donc pas considéré comme son unique propriété, il est partagé par plusieurs autres SMA.

Les agents sont souvent amenés à interagir pour surmonter leurs limitations lorsqu'ils répondent aux objectifs individuels ou collectifs. Si un agent coopère régulièrement avec les autres, ou que des agents expriment clairement leurs intentions, la coopération reposera sur la transmission de messages ou l'interaction par l'environnement. Si cette communication n'est pas possible, limitée, coûteuse, ou indésirable, un agent peut se trouver devant des situations d'interaction où il coopère avec des agents d'autres systèmes sans n'avoir ni une idée précise de leurs intentions, ni une connaissance des conséquences de cette coopération sur la réalisation de ses propres objectifs.

Dans ce travail qui s'inscrit dans le cadre de l'ANR ASAWoO [1] (Web des Objets), le partage de l'environnement physique et donc des ressources exploitables remet en question l'hypothèse de la pleine coopération. Notre conviction est que la reconnaissance d'intention est un bon moyen qui permet aux agents de SMA hétérogènes de s'entendre, voire de coopérer : un agent doit avoir connaissance de ce que les autres veulent réaliser pour pouvoir adapter son propre comportement.

2 Contribution Visée

En présence d'agents appartenant à des SMA différents, l'agent doit trouver des moyens pour identifier la catégorie d'interaction qu'il doit maintenir ou qu'il est susceptible de démarrer. Nous proposons une approche de sélection de stratégie collective basée sur la reconnaissance des intentions des autres agents. Un agent ne peut pas observer, directement, une intention, mais il peut l'inférer à partir de l'observation des actions, des interactions et des activités sociales des autres agents.

L'approche que nous proposons est articulée autour de 4 phases :

Observations. Dans cette phase, l'agent ne s'intéresse qu'aux actions et aux messages en relation avec son contexte, c-à-d, ceux qui peuvent changer les propriétés physiques qu'il souhaite maitriser de son environnement. Il maintient une trace des actions effectuées. Dans le cas d'une communication, l'agent observateur s'intéresse aux relations entre les agents, les équipes, les sous-équipes, et les coalitions formées.

Reconnaissance d'Intentions. L'objectif de cette phase est de reconnaître l'intention de l'agent ou de l'équipe observée. Dans notre contexte applicatif, des hypothèses simplificatrices peuvent être introduites comme l'existence de librairies de plans, de modèles d'actions, ou la possible accessibilité à la définition des rôles d'agents. Selon les observables disponibles, on peut alors reconnaitre un but, un plan, un engagement même si c'est avec un doute.

Validation. Après avoir fait des hypothèses sur les intentions de l'agent observé, l'agent observateur évalue chacune de ses hypothèses : il vérifie s'il y a des contradictions avec ce qu'il a dans sa base de connaissances. Cette évaluation se base sur les relations entres les agents, les équipes, les rôles joués par les agents, la disponibilité des ressources, la dernière représentation que l'on a d'un état mental et le plan de l'agent observé. Il peut aussi interagir avec les agents de son propre système ou ceux qu'il juge dignes de confiance pour lever un doute ou confronter son analyse.

Choix de Stratégie Collective. Un agent qui souhaite accomplir un but connait généralement les compétences qui lui manquent, sait s'il est en compétition pour l'accès aux ressources et il est capable d'identifier les agents qui peuvent combler ces manquements. À l'aide des phases précédentes, il a une représentation des buts, des plans et des intentions des autres agents. En fonction de la compatibilité de ses buts avec ceux qu'il a estimé, il peut identifier sa situation d'interaction vis-à-vis des agents observés. Cette identification l'aide à choisir plus prudemment, que s'il était dans un état d'esprit purement coopératif, des stratégies collectives.

1. http://liris.cnrs.fr/asawoo/

Une approche méthodologique de la coopération des collectifs de systèmes multi-agents hétérogènes

A. Khenifar Bessadi[a]
a_khenifar@esi.dz

J-P. Jamont[b]
jean-paul.jamont@lcis.grenoble-inp.fr

M. Occello[b]
michel.occello@lcis.grenoble-inp.fr

M. Koudil[a]
m_koudil@esi.dz

[a] Laboratoire de Modélisation et de Conception des Systèmes, Ecole Nationale Supérieure d'Informatique, Alger, Algérie

[b] Laboratoire de Conception et d'Intégration des Systèmes Université Grenoble Alpes, Valence, France

Résumé

Nous visons dans ce travail la coopération de Systèmes Multi-Agents (SMA). Une approche possible est de considérer au niveau macroscopique chaque SMA comme une entité indépendante. Cela permet d'augmenter l'intelligibilité de la solution globale par une réduction de la complexité. Nous expliquons dans cet article le problème visé ainsi que la démarche suivie.

Mots-clés *: SMA, coopération, production collective, méthode, conception.*

Abstract

We aim in this research to cooperate Multi-Agent Systems (MAS) at the macroscopic level. This is by considering each MAS as an independent entity, which can increase the intelligibility of the overall solution by decreasing the complexity. We explain in this article the problem addressed and the approach.

Keywords*: MAS, cooperation, collective production, method, design.*

1 Introduction

Le produit collectif d'un système multi-agent (SMA) est le résultat d'interactions entre agents d'un même SMA œuvrant pour un objectif commun. La coopération à partir des produits collectifs de différents SMA peut être motivée par des objectifs globaux que n'arrivent pas à satisfaire au moins un des systèmes.

Dans la littérature, le problème de coopération des SMA est souvent reformulé en un problème de coopération de leurs composants de niveau microscopique (i.e. les agents) [1-4]. Cependant, un agent n'a pas toujours connaissance de l'objectif global de son SMA [5]. Dans ce travail, nous défendons une approche centrée sur la coopération sur la base des produits collectifs fournis par différents SMA : une coopération de niveau « macro ». Notre conviction est qu'elle permettra à la fois :

- une meilleure intelligibilité du produit collectif qui répond aux objectifs à atteindre en masquant la complexité interne de chacun des SMA,
- une meilleure résolution collective des problèmes tout en préservant l'indépendance opérationnelle et l'autonomie de gestion des systèmes mis-en-jeux,
- une meilleure dynamique d'adaptation à l'environnement.

Ce travail de thèse vise la proposition d'une démarche méthodologique qui assistera les concepteurs qui souhaitent l'exploitation de productions collectives de SMA déployés par d'autres SMA.

2 Démarche suivie

Approche : Nous divisons la coopération des niveaux collectifs de SMA en trois principales

phases comme l'illustre la figure 1 :

Phase P1: Cette phase consiste à permettre l'identification, le nommage et la compréhension d'un collectif. Elle peut être réalisée manuellement par un observateur humain ou automatiquement par un observateur artificiel.

Phase P2: Une fois les collectifs identifiés, on sélectionne les collectifs pertinents pour la coopération. Il convient de s'assurer de la compatibilité des objectifs des différents collectifs mis en jeu. Puis on applique une stratégie de coopération.

Phase P3: La stratégie de coopération et le produit collectif sont évalués (analyse du retour d'expérience) par l'utilisateur, qu'il soit humain ou artificiel.

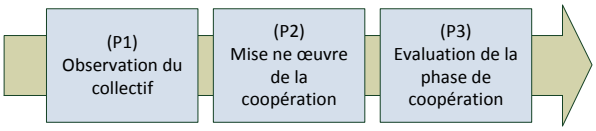


FIG. 1 – Les phases du processus de coopération des collectifs

Fiche de conception : Nous avons défini une grille d'analyse qui pose différentes questions permettant de fixer les limites de l'étude et de faire la synthèse des collectifs. Elle peut être utilisée à différents niveaux du cycle de vie du SMA :
1) *L'étape d'analyse par un concepteur* : la grille agit comme une check-list des questions à se poser pour chacun des SMA afin de comprendre les collectifs;
2) *L'étape d'identification automatique des collectifs par d'autres SMA*: cette grille est alors exposée par un collectif afin de le décrire. Elle peut être complétée de manière automatique par un collectif observant en utilisant des techniques de reconnaissance d'intentions, d'activités etc.
À titre d'exemple certaines des questions soulevées qui figurent dans cette fiche sont:
Question 1 : Pourquoi faire coopérer les SMA au niveau collectif (P2, P3)? – Pour agir sur l'environnement, maintenir un état ou assurer la résilience, se confronter avec un autre point de vue, renforcer une compétence, créer une nouvelle compétence. Cette question porte sur la motivation des SMA à coopérer.

Question 2: Quels types de phénomènes collectifs met en jeu la coopération des collectifs des SMA (P1, P2) ? Structures, propriétés, comportements/services. Cette question s'intéresse à la nature des produits collectifs existants.

Question 3: Comment peuvent interagir deux SMA pour coopérer à un niveau collectif (P1, P2) ? [10] –Communication, Action directe, Interaction par l'environnement.

Beaucoup d'autres questions, que nous ne pouvons pas toutes citer ici, sont posées

3 Conclusion

Dans cet article, nous avons proposé et défendu la coopération des SMA au niveau collectif de SMA. Une concrétisation de l'approche proposée par un modèle est la prochaine étape de notre travail. Nous visons aussi à instancier l'architecture du modèle et à l'implémenter afin de tester et de valider l'approche proposée.

Références

[1] W. Alshabi, S. Ramaswamy, M. Itmi, and H. Abdulrab, "Coordination, cooperation and conflict resolution in multi-agent systems," in *Innovations and Advanced Techniques in Computer and Information Sciences and Engineering*, ed: Springer, 2007, pp. 495-500.

[2] F. Buccafurri, D. Rosaci, G. M. Sarnè, and L. Palopoli, "Modeling cooperation in multi-agent communities," *Cognitive Systems Research*, vol. 5, pp. 171-190, 2004.

[3] J. E. Doran, S. Franklin, N. R. Jennings, and T. J. Norman, "On cooperation in multi-agent systems," *The Knowledge Engineering Review*, vol. 12, pp. 309-314, 1997.

[4] T. Ebadi, M. Purvis, and M. Purvis, "A framework for facilitating cooperation in multi-agent systems," *The Journal of Supercomputing*, vol. 51, pp. 393-417, 2010.

[5] I. Rochlin, D. Sarne, and M. Mash, "Joint search with self-interested agents and the failure of cooperation enhancers," *Artificial Intelligence*, vol. 214, pp. 45-65, 2014.

Les stratégies d'acteurs pour la Gouvernance Communautaire des Ressources Naturelles :

Expérimentation d'un outil d'aide à la Décision à base de Systèmes Multi-Agents appliqué à l'Océan Indien

A. Gaudieux[a] Aurelie.gaudieux@univ-reunion.fr

J. Kwan[b] Joel.Kwan@univ-reunion.fr

R. Courdier[b] Remy.Courdier@univ-reunion.fr

[a] Centre d'Economie et de Management de l'Océan Indien et Laboratoire d'Informatique et de Mathématiques, Université de La Réunion, France

[b] Laboratoire d'Informatique et de Mathématiques, Université de La Réunion, France

Résumé

L'article décrit le système SIEGMAS (Stakeholders Interactions in Environmental Governance by a Multi-Agent System) de simulation des interactions entre les parties prenantes dans la gouvernance communautaire des ressources naturelles dans les îles de l'Océan Indien. Ce système d'aide à la décision repose sur un modèle d'agents permettant d'étudier les interactions entre les agents agissant sur un territoire sous un aspect économique via une interface agronomique.

Le travail présenté ici constitue une extension d'un premier système SIEGMAS étendu au travers d'une interface interactive et dynamique dotant notre système de propriétés puissantes de configuration par cartes et d'interprétation des résultats. L'objectif de SIEGMAS est double. D'une part, il s'agit d'offrir un modèle dédié à la communauté économique pour la gestion des ressources naturelles. D'autre part, l'objectif consiste à d'offrir un cadre de solution informatique adaptée aux décideurs du milieu économique et politique.

Mots-clés : gouvernance communautaire des ressources naturelles, Interactions, SMA.

Abstract

This paper presents the system SIEGMAS (Stakeholders Interactions in Environmental Governance by a Multi-Agent System) design to simulate interactions between stakeholders in Common Pool Resources in Indian Ocean Islands. This help system tool is based on a model of agents allowing to study the interactions between agents which act on a territory below an economic aspect thanks to an agronomic interface. The work presented here is an extension of a first extended SIEGMAS system through an interactive and dynamic interface providing our system of powerful properties by cards' configuration and interpretation of results. The goal is double. On one hand, we want to offer a tool devote to the economists community to the Common Pool Resources. On the other hand, we want to present a complete informatics solution for results proposals for decision-makers in business and politics.

Keywords: *Agent-based simulation, Common Pool Resources, Interactions.*

1 Introduction

Dans un monde asservi par les crises et un contexte géopolitique où les dysfonctionnements entre les rapports de forces assujettissent les populations à s'adapter, l'évocation par les instances diverses de la gouvernance sous ces formes multiples via le fleurissement de multiples indicateurs de mesures semblent être une

solution palliative pour résorber les multiples problèmes et tout particulièrement ceux relatifs à la gestion de ressources naturelles.
Les ressources naturelles ont, dans un premier temps, été considérées comme illimités dans les siècles derniers [13]. Puis, avec un accroissement de la population et de la consommation, la notion de limitation des ressources est apparue dans les discours institutionnels. La finitude des ressources incite donc, les sociétés à assurer une gestion plus soucieuse de l'environnement des nuisances et des risques générés. Le système SIEGMAS (Stakeholders Interactions in Environmental Governance by a Multi-Agents System) que nous avons conçu est un outil d'aide à la décision permettant d'étudier les interactions entre les différentes parties prenantes dans la gouvernance communautaire des ressources naturelles.
Bien que cette problématique soit très présente dans l'actualité et dans de nombreuses recherches pluridisciplinaires, la génération de simulation portant spécifiquement sur les interactions entre les parties prenantes dans la gouvernance communautaire des ressources naturelles dans les îles de l'Océan Indien n'avait à ce jour jamais été réalisée. L'approche multi-agents, utilisée dans ce travail, permet d'appréhender cette problématique selon un angle nouveau, complémentaire aux outils économiques actuels.
Notre objectif consiste à apporter une contribution à la communauté scientifique travaillant sur les modèles économiques par la proposition d'un environnement informatique constitué d'une chaine d'outils dont le cœur est constitué d'un modèle multi-agents réutilisable adaptés aux différents territoire de la zone océan indien. Ainsi nos expérimentations réalisées avec SIEGMAS portent tout autant sur la grande Ile de Madagascar, dont certaines règles de gestion sont liés à des bailleurs internationaux (FMI,…) qu'à la petite Ile française de la Réunion régie par le code européen.
L'article présente la problématique sensible de gouvernance communautaire et justifie l'intérêt d'une approche SMA. Il décrit le modèle conceptuel de représentation multi-agents associé à notre système, puis introduit des extensions originales constituées d'une part d'un outil de configuration MASC (MAp Sector Creator) qui permet de générer de façon flexible un environnement de simulations en partant de cartes numériques de représentation de données d'experts et d'autre part présente notre démarche d'interprétation des résultats basés sur une séparation complète des données de génération par rapport aux données de visualisation.

2 Le contexte scientifique

La gouvernance des ressources naturelles désigne la gestion locale efficace en commun des ressources par plusieurs parties prenantes [6]. Les stratégies de gouvernance communautaire s'axent autant sur la participation active de parties prenantes tels que les entreprises et les citoyens que sur la mise en œuvre de politiques par des institutions représentantes d'autorités. Cette seconde forme de gouvernance (celle des autorités) est alors qualifiée de gouvernance passive et s'exerce conjointement à la gouvernance active menée par les entreprises et les citoyens.

En économie, plusieurs méthodes sont envisageables pour l'étude des ressources naturelles et des transferts de gestion [2][5]. Rappelons que les transferts de gestion désignent une passation de pouvoir du gouvernement aux collectivités décentralisées et à d'autres parties prenantes dans le cadre de la gouvernance commune des biens plus communément désignée par le terme « Common Pool Resources » [16].
Toutefois, ces méthodes modélisent difficilement les interactions et la cognition entre parties prenantes. L'incorporation de la biodiversité dans les méthodes de calculs économiques demeure controversée et limitée. Les analyses économiques *ex ante* et *ex post* de la biodiversité portent soit sur la

rationalité de la conservation (analyse coût bénéfice et le bien-être social gagné pour chaque euro investi) soit sur l'efficacité de la stratégie de conservation retenue. Les services éco-systémiques menant à la recherche d'un optimum social relèvent donc de l'évaluation coût–avantage, supplantée par l'analyse coût–efficacité. Le développement économique dépend de la préservation des ressources et de la minimisation de des impacts relevant de son usage. Plusieurs méthodes économiques et transversales ont été envisagées pour créer cette simulation.

- Les méthodes d'efficience du marché et d'économie général qui portent sur les analyses relatives à la valeur du marché et à la valeur, les théories de la croissance et le Modèle d'Equilibre Général Calculable ;
- Les méthodes environnementales employées qui portent sur l'analyse mono ou multicritère(s), les mesures compensatoires ou restauratrices et l'éco-certification et l'éco-potentialité ;
- Les méthodes relatives à l'économie écologique qui portent sur des indicateurs de biodiversité, du calcul de l'emprunt écologique et des indicateurs relatifs à la comptabilité nationale.

Comme l'a montré Stuart Kaufmann [8], de nombreux systèmes dynamiques, dont notamment les systèmes sociaux dans lequel des humains interviennent (tel que dans notre situation) reposent sur des dynamiques non-linéaires. L'utilisation des méthodes classiques de l'économie peut alors masquer la présence de dynamiques complexes non prise en considération jusqu'à présent dans les études réalisées, tels par exemple la sensibilité aux conditions initiales, la prise en compte de phénomènes d'émergence correspondant à des comportements collectifs, combiné à la prise en compte de conditions spatiales et temporelles.

Ainsi, les méthodes économiques présentées ci-avant analysent donc difficilement la cognition de systèmes dynamiques sociaux, d'où la pertinence à ce stade de s'intéresser aux systèmes multi-agents qui ont leurs preuves notamment dans les solutions informatiques s'appuyant sur la simulation.

Notre démarche, s'inscrit dans l'exploration de nouvelle propositions d'outils et de montrer l'intérêt de l'approche multi-agents dans une solution informatique complète supportant les critères d'évaluation de la communauté scientifiques des économistes.

Notre travail présenté ici a deux objectifs :

- proposer un modèle agent générique réutilisable attaché à la problématique d'aide à la décision dans le domaine du « Common Pool ressources ».
- définir une chaines d'outils adaptés au monde économique pour proposer une solution cohérente et exploitable de bout en bout pour des économistes ; Technique d'initialisation, simulation, visualisation et exploitation des résultats de simulation.

Afin de valider notre modèle, sa généricité et la prise en solidité des choix de solution informatique réalisés, les expérimentations mise en œuvre sur le système SIEGMAS ont été appliquées à des territoires différents de Madagascar et de La Réunion [7].

3 SIEGMAS, un système de simulation en trois étapes

SIEGMAS est un système d'aide à la décision permettant d'étudier les interactions entre les parties prenantes dans la gouvernance communautaire des ressources naturelles (fig. 1). Actuellement, ce système constitue le seul outil conçu dans le champ économique dédié à la problématique de transferts de gestion dans les îles de l'Océan Indien et l'étude bibliographique réalisée ne nous a pas permis de pouvoir appuyer notre étude sur un système informatique existant.

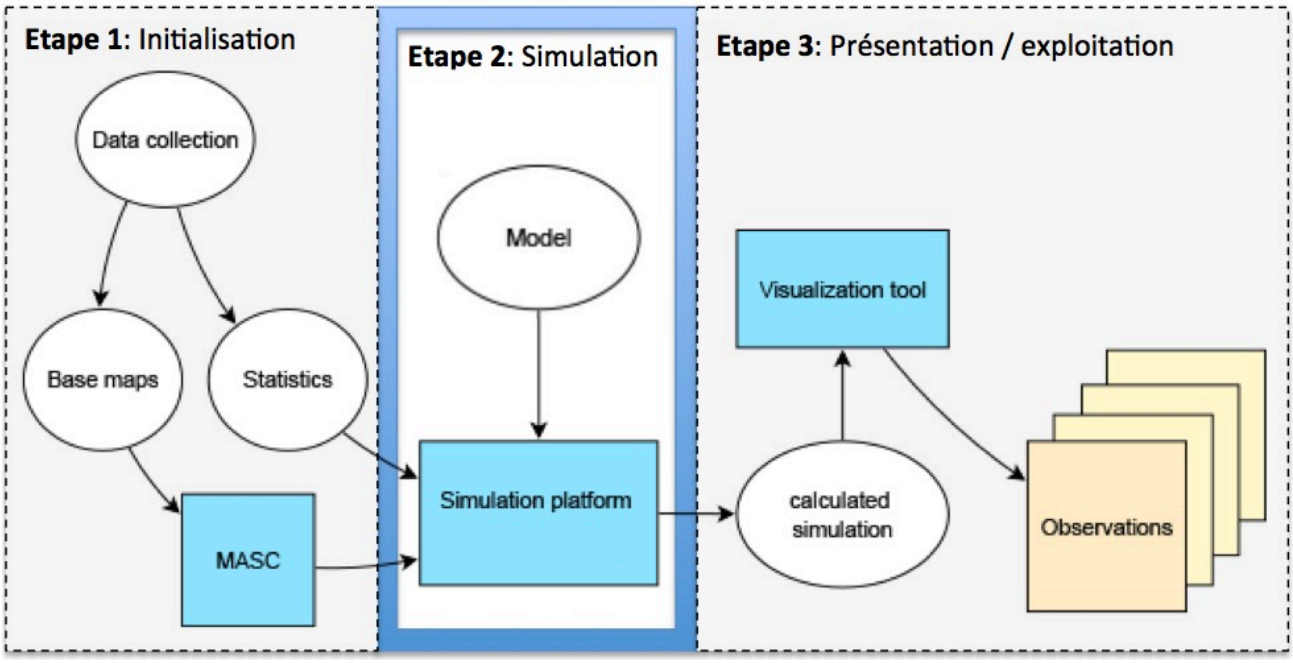


FIGURE 1 - SIEGMAS, un système en trois étapes

L'étape d'initialisation, calibrage :

Dans le domaine de la gestion de ressources naturelles de très nombreuses données d'information géographique sont disponibles quelque soient les territoires, et même dans des contrées reculées de Madagascar, l'imagerie satellitaire permet d'initialiser un système en respectant des critères génériques. C'est l'objet du module MASC (MAp Sector Creator) de notre système, qui peut être considéré comme un générateur d'environnement de simulations autorisant à utiliser des cartes numérisées de représentation des ressources naturelles directement utilisable dans une simulation multi-agents. MASC [7].

L'étape de simulation :

La difficulté de l'exercice de définition du modèle de simulation tient au fait que l'étude des politiques de gouvernance environnementale nécessite tant une vision agronomique, économique, juridique, sociétale et cognitive des interactions entre les parties prenantes. Le modèle présenté en détail dans la section 4 de cet article.

Le modèle s'appuie en entrée sur des données cartographiques issue de l'étape 1. Le paramétrage des échelles spatiales et temporelles prise en compte durant las simulation sont de pourvoir réaliser différentes simulations possédant un même cadre d'initialisation mais en considérant des échelles variées.

L'étape de présentation / exploitation :

La simulation a pour objectifs de répondre à des questions que se pose un observateur sur un monde réel qu'il ne peut étudier directement pour des raisons de temps, de coûts ou de faisabilité. La simulation produit comme résultat une qualité d'information dont la taille dépend de quatre principaux facteurs, i. La taille du système représenté, ii. La précision de représentation du système, iii. La complexité des entités du système, iv. La richesse et la variété des interactions entre les entités. L'enjeux de cette étape et de permettre la compréhension les interactions entre les nombreux acteurs au sein de leur environnement et dont l'évolution est sensible à des conditions environnementales et non prévisible à priori. La complexité de cette étape tient au fait que la complexité des situations réelles représentées produit une masse de résultats qu'il est paradoxalement aussi difficile à observer et analyser que le système réel initial [14]. Nous proposons pour cette étape d'utiliser des techniques de représentation interactive de données riches et complexes ou data visualization (dataviz) [15]. La solution mise en œuvre consiste d'une part du point de vue structurel à séparer la génération des données/résultats de simulation de leurs visualisations. D'autre part, du point de vue conceptuel la démarche s'appuie sur les résul-

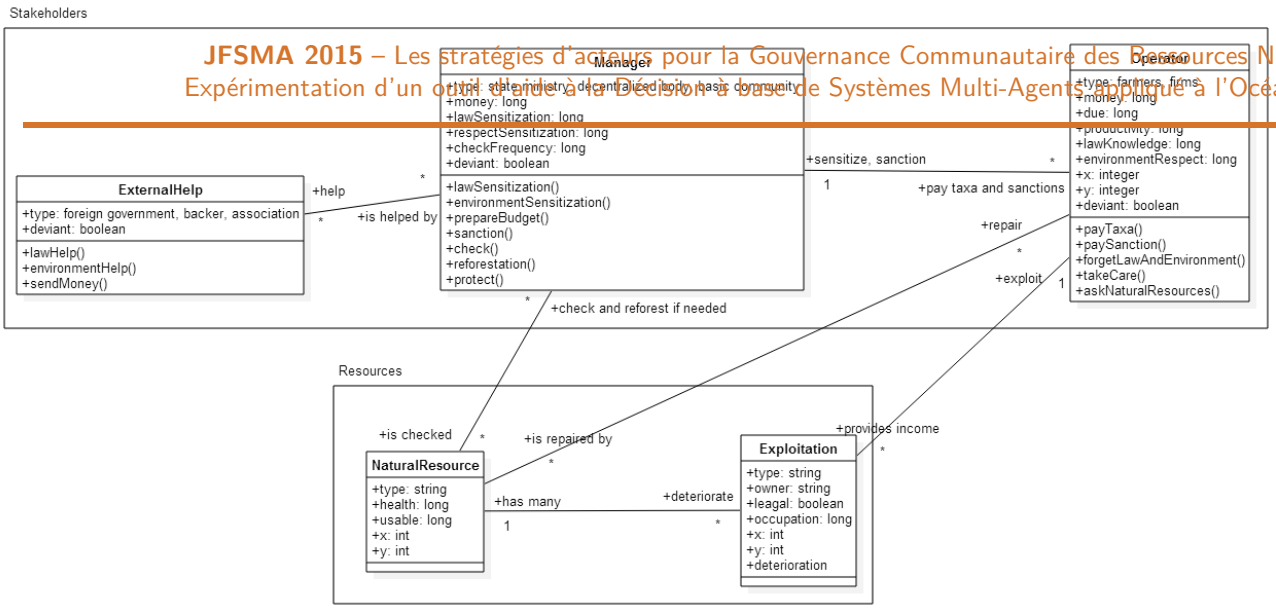


FIGURE 2. Model structurel SIEGMAS (format UML)

tats des solutions basées sur les techniques de graphismes interactifs par la proposition d'outil dédiés à la présentation de données économiques en rapprochant dynamiquement la présentation de graphiques à des contextes sociaux/spatiaux et en simplifiant la l'exploitation et des données aux travers de tableaux dynamique [8].

4 Modèle agent de SIEGMAS

4.1 Modélisation : aspects structurels

On distingue dans SIEGMAS deux catégories d'agents (figure 2) :

Un ensemble d'agents « Resources » de représentation des ressources naturelles et un ensemble d'agents « *StakeHolders* » de gestion des ressources naturelles.

Au niveau « Resources », nous avons distingué deux types d'agents :

- L'agent *NaturalResource*, entité de modélisation des zones physiques de territoire (parcelle agricole, foret, ou friche). Un agent *NaturalResource* subit des pratiques d'exploitation de la ressource, *il* est contrôlé par un *Manager* (e.g. un ministère) et peut être géré par un *Operator* (e.g. un agriculteur).
- L'agent *Exploitation*, entité de modélisation des pratiques d'exploitations appliquées à une ressource naturelle (pratique biologique, pratique conventionnelle, brûlis, …). Un agent *Exploitation* s'applique à un agent NaturalResource, il est sous l'influence d'un *Operateur* pouvant mettre en œuvre cette pratique.

Au niveau « *StakeHolders* », trois types d'agents ont été définis :

- L'agent *Manager*, entité de modélisation de l'État, de ministères, d'instances de gestion décentralisées, ainsi que de communautés citoyennes. Un agent Manager est assisté par des ExternalHelp (e.g. des bailleurs de fond), il contrôle les *NaturalResource* et peut sanctionner des Operator (e.g une entreprise minière).
- L'agent *Operator,* entité de modélisation des agriculteurs et des entreprises d'exploitation de territoires. L'Operator met en œuvre des pratiques d'*Exploitation,* paye des taxes et des sanctions à un *Manager* et peut mener des actions de protection de *NaturalResource*.
- L'agent *ExternalHelp,* entité de modélisation des gouvernements étrangers, bailleurs de fonds et associations. Un EternalHelp porte assistance à des *Manager.*

Il est à noté que nous sommes dans un contexte économique et que chaque partie prenante décrite ci-dessus va dynamiquement en fonction des influences (pressions) qu'elle subira au cours de la simulation pouvoir adopter un comportement *déviant* non conforme aux respects de règles collectives et pouvant donner lieu à des sanctions.
L'un des objectifs principaux du système est d'analyser la sensibilité aux sanctions des

acteurs et l'impact que cela induit sur les ressources naturelles.

But des agents de notre modèle
Les différents buts des agents de notre modèle sont explicités dans le tableau ci-dessous. Ces buts sont paramétrés dans le modèle en fonction d'une typologie qui est également précisés dans le tableau.

Agents	But	Typologie
Manager	Assumer et veiller à l'application d'une politique de gestion de ressources naturelles partagées. Un Manager peut adopter un comportement déviant qui représente alors une institution corrompue.	StateMinistry Decentralizedbody BasicCommunity
Operator	Définir des pratiques économiquement viables sur une ressource naturelle en tenant compte du cadre fixé par les institutions et des risques encourus en cas de non respect de ce cadre en utilisant un comportement déviant.	Farmer Firm
ExternalHelp	Veiller aux ressources naturelles protégées par les instances internationales. Veiller à des intérêts économiques, notamment à créer les conditions d'accès à des ressources naturelles. Un ExternalHelp peut adopter un comportement déviant dans un but stratégique.	ForeignGovernment Backer Association
Exploitation	Appliquer une pratique agricole (biologique, conventionnelles, sur brûlis) sur une ressource naturelle et évaluer la détérioration engendrée sur la ressource dans le temps.	Bio Con Snb
NaturalResource	Le but d'une ressource naturelle Limiter son niveau de détérioration. Les règles appliquées vont dépendre du type de ressource. Siegmas considère actuellement 3 type ressources (Friche / FallowLand, Terre arable / ArableLand, Forêt/Forest)	FallowLand ArableLand Forest

4.2 Modélisation : aspects spatio-temporels

La spatialité dépend de la carte générée dans MASC. En effet, lors de la passation de la carte dans MASC, celle-ci est mise en correspondance avec un espace en deux dimensions discrétisé dans une grille. La finesse de le discrétisation est alors décidée par les modélisateurs. Il est alors possible de réaliser des simulations à différents niveaux d'échelle (microéconomique, méso-économique ou macroéconomique) en fonction des objectifs de simulation fixés.

Sur le plan temporelle. Pour l'interprétation des résultats de simulation d'un système dynamique et le paramétrage des actions, il est primordial de pouvoir disposer d'une base temporelle adaptée à notre application.

La base de conversion entre le temps processeur et le temps de simulation, correspond à l'échelle de temps choisie pour le modèle (plus petit intervalle de temps en deçà duquel on ignore les éventuels changements). Ce paramètre influe sur la vitesse de simulation et la finesse des résultats obtenus.

Dans le modèle SIEGMAS, nous avons choisi qu'un pas de temps de simulation correspond à un mois en unités de temps astronomique.

4.3 Modélisation : aspect comportementaux

Plusieurs règles et stratégies régissent le modèle SIEGMAS.

Pour l'exploitation des ressources naturelles, les parties prenantes exploitent ou tentent d'exploiter tant les ressources avoisinant leur territoire que les ressources éloignées géographiquement. Les agents (*Operators*, *ExternalHelp*, *Manager*) interagissent sur le marché interne et international pour l'achat et la vente des biens et services. Concernant les biens collectifs se pose le problème du passager clandestin [13] qui permet aux agents d'obtenir 100% de gains pour un citoyen qui utilise les biens publics sans s'acquitter d'impôts. Les investissements et les préférences individuelles sont déterminants pour mesurer l'utilité collective. S'agissant des externalités (positives ou négatives), elle dépend des paiements des taxes/impôts, de l'application des peines et des dons. Les choix publics relèvent des politiques et stratégies environnementales, du coût du contrôle, de la productivité, du coût de la réalisation des lois et de son application. La coopération s'ajoute à ces choix publics pour les agents *ExternalHelp*.

Les sanctions : les agents peuvent respecter ou transgresser la législation envers la

protection des ressources naturelles selon leur appartenance au groupe des agents sains, intermédiaires ou déviants. Les lois nationales et spécifiques aux territoires étudiées réglementent la gestion des ressources.

Classification des stratégies

Une classification de stratégies a été construite dans le modèle SIEGMAS. Ces stratégies sont adoptées dynamiquement par les agents. On notera que certaines conditions peuvent conduire les agents à appliquer des stratégies combinées.

stratégies	**Principales actions**	**Objectifs visés**
De prix	Produits discounts de valeurs comparables à celles des concurrents.	Offrir des produits moins chers sur un marché
Hybride	Produits avec plus de valeur et moins coûteux que ceux des concurrents.	Offrir des produits avec plus de valeur à des prix réduits.
Innovation	Produits et services innovants	Maintenir sa situation ou l'améliorer
Alignement	Par rapport aux autres régions nationales et internationales	Risques de diminution des profits
communicatio n	Auprès des acteurs de la gouvernance (citoyens, firmes)	Développement futurs/ bilans
Encrage	Stratégie ciblée, politique de prix	Dominer dans un secteur ou sur une niche de marché
différentiation	Produits locaux ou de valeurs différentes de celles des concurrents	Part de marché, niche
Epuration (stratégie de différentiation vers le bas)	Produits et services dont la valeur est inférieure à celle des concurrents.	Offrir des produits à prix réduits.
Sophistication (stratégie de différentiation vers le haut)	Produits et services dont la valeur est supérieure à celle des concurrents.	Offrir des produits à prix supérieurs aux concurrents.
Focalisation (ou de niche)	Produits différents pour attirer une part des consommateurs.	Offrir des produits différentiés.
D'agglomérati on	Stratégie de court terme	Attirer et agglomérer les pôles économiques
De concentration	Sur le moyen terme	Être plus concurrentielle et ériger des barrières à l'entrée

Prise en compte des propriétés des Systèmes multi-agents dans le modèle SIEGMAS

Considération des propriétés classiquement conférées aux *Agents* constitutifs d'un SMA :

Agents	Buts: les parties prenantes sont dotées d'un but qui sera polarisé.
Autonome	Décide et contrôle ces actions en faveur de la Gouvernance des Ressources Naturelle (GRN).
Proactif	L'agent est proactif et opportuniste et n'hésite pas à choisir des méthodes allant à l'encontre des RN et de transgresser les lois.
Situé	Les « entrées sensorielles » orientent ces décisions sur la GRN.
Flexible	L'agent s'adapte aux nouvelles informations perçues sur la GRN et les actions des autres agents pour atteindre ces buts et ne pas se faire sanctionner.
Just in time	L'agent émet une réponse dans le temps idéal dès qu'il perçoit les actions des autres agents ou-et dès qu'il dispose d'une nouvelle information sur la GRN.
Social	Chaque agent à une capacité d'interaction avec les agents GRN pour concourir à des buts collectifs (améliorer la gouvernance des RN) ou individuels (s'assurer ou accroitre son revenu, rendre son exploitation productif, équilibre des prix sur le marché des biens et services).
Adaptatif	Les agents sont capables de changer de stratégie pour atteindre ces buts. Ainsi, un agent peut changer de méthodes de préservation de ressources naturelles ou peut passer de la mise en œuvre d'une mesure de protection de l'environnement à l'adoption d'une mesure/ pratique contraire à la sauvegarde de l'environnement.

Considération des éléments de représentation mentale classiquement utilisés par les modèles SMA :

Notions mentales de l'agent	**Agents**
Connaissances	Les agents savent que l'environnement doit être protégé et qu'il ne faut pas transgresser les lois.
Croyances	Certains agents protègent l'environnement parce qu'ils savent que les ressources sont limitées et qu'ils seront sanctionnés.
Buts et désirs	Les agents désirent percevoir un profil et une image positive ou de ne pas se faire sanctionner par l'Etat.
Intentions	Certains agents ont l'intention de mettre en œuvre des mesures pour l'environnement ou d'en appliquer, d'autres désirent exploiter illicitement les ressources.
Choix & décisions	Les agents ont décidé de mettre en place les mesures de la Gouvernance des Ressources Naturelles ou pas.
Engagements	Les agents protègent ou pas l'environnement tant que le contrat de transfert de gestion est en vigueur ou que

	la zone reste protégée.
Conventions	L'Etat peut exercer sa fonction de contrôle de sensibilisation et de sanction.
Obligations	Si les agents ne respectent pas les lois certains dispositifs des sanctions sont applicables (amendes et peines de prison).

5 Autres outils de SIEGMAS

5.1 Le générateur d'environnement MASC

Dans l'article MASC [7], le sujet principal de la discussion s'est focalisé sur la création d'un générateur d'environnement de simulations qui permet d'utiliser des cartes numérisée pour avoir une représentation de la zone considérée directement utilisable dans une simulation multi-agents.
Afin d'éviter à tout modélisateur de devoir installer un outil informatique trop lourd et pour limiter les mises à jours des postes informatiques, nous avons conçu MASC selon une architecture de type client/serveur basée sur un processus qui se décompose en trois étapes (figure 4).

La première étape consiste à acquérir toutes les informations nécessaires à l'outil. Cette étape se déroule principalement du coté client (poste modélisateur). L'information de base requise pour le bon fonctionnement de l'outil est un ensemble de fichiers d'images au format normalisé par les standards du W3C, qui représente le territoire simulé.

Le modélisateur est conduit alors à définir le niveau d'échelle spatiale le plus fin pouvant être raisonnablement considéré dans une simulation. MASC extrait alors les couleurs de l'image et permet de façon interactive une association dynamique de plages de couleurs avec des ressources considérées par le modèle (e.g. forêt, friche, etc).

Dans une seconde étape, ces données sont envoyées à serveur pour qu'il puisse extraire une grille qui servira comme point de départ pour générer le code « snippet » utilisé pour l'initialisation de système muli-agents. Pour cette étape, le modélisateur n'a pas d'opérations à réaliser, tout se fait uniquement au niveau du serveur.

La dernière étape constitue un retour de résultats visuels de l'opération faite par la partie serveur. D'une part, l'utilisateur a un retour graphique et d'autre part, il peut légèrement modifier la grille générée ou changer manuellement certaines couleurs de secteurs avant la génération de code final. On notera que cette génération de code final est un module « plug&play » qui autorise l'utilisation de MASC par d'autres plateformes Agents.

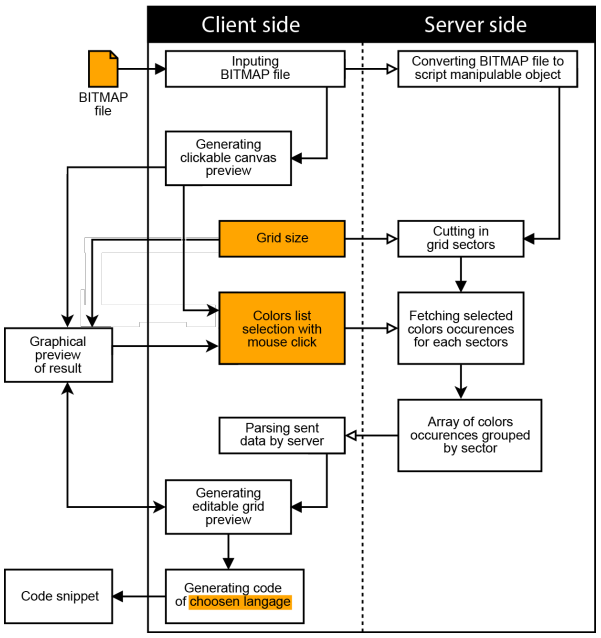


FIGURE 4 - Workflow détaillé de MASC

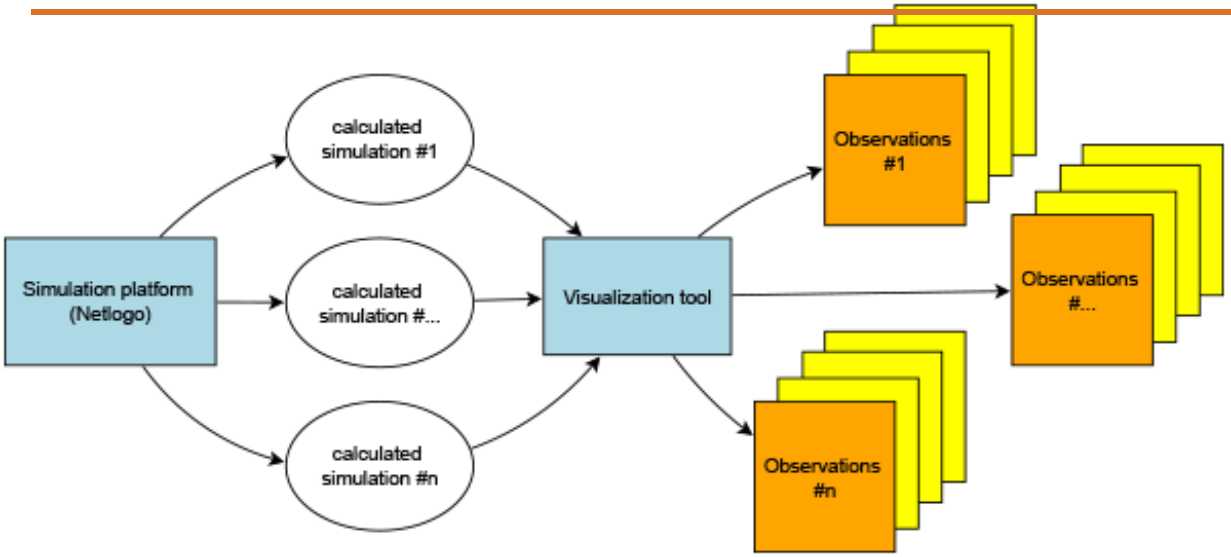


FIGURE 5 – Tableaux de bord dynamique construit sur la base de multiples simulations

5.2 Présentation de résultats d'observations de simulations

Le modèle multi-agent de SIEGMAS a été implanté sur la plateforme NetLogo qui offre un certain nombre de fonctionnalités d'observation de résultats de simulation sommaires pour la mise au point du système (figure 6).

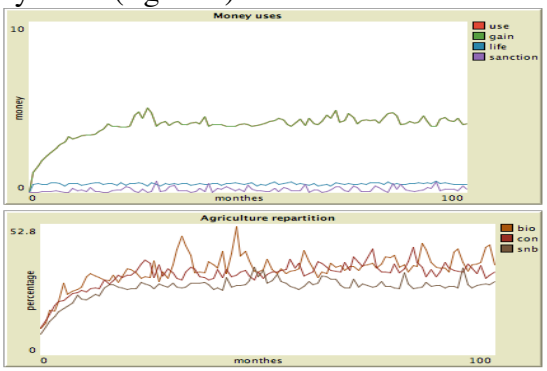


FIGURE 6. Présentation de résultats De simulation sous NetLogo

Cependant, seuls quelques résultats peuvent être exprimés alors que compte tenu de la masse d'information générée au cours d'une simulation de nombreuses formes d'interprétations peuvent être offertes et construites de façon interactive par un économiste.
Nous avons donc externalisé le maximum de données issues de séries de simulations afin que celles-ci soient exploitables « en différé » par des outils de gestion de tableau de bord intercatifs qui sont des outils très prisés par les acteurs de l'économie.

Il s'agit d'offrir aux décideurs plusieurs axes de vues pour les données calculées. En effet, il y a toujours des difficultés de choisir la « chaîne » de données à suivre pendant le déroulement de la simulation et conduisant à la mise en valeur de phénomènes observables. L'approche mise en œuvre, fournit la possibilité de visualiser la simulation dans plusieurs directions et axes et aussi autorise d'extraire les observations sans relancer les calculs plusieurs fois. De plus, il fournit également une chronologie pour avancer ou rembobiner la visualisation de résultats.

A ce niveau, une application est réalisée avec l'environnement « tableau public »[1] ne nécessitant pas de connaissance informatique pour la gestion et présentation interactive de données complexes. D'autres expérimentations sont en cours d'étude avec des outils de type « *data-driven* », utilisés pour la présentation de grandes quantités de données et utilisés dans le contexte Big Data[2].

[1] Tableau de bord interactif
http://www.tableau.com/public/

[2] Bibliothèque Data-Driven Document
http://d3js.org

6 Conclusion et perspectives

La réalisation d'un outil d'aide à la décision pluridisciplinaire pour la gouvernance communautaire, SIEGMAS, apporte une réponse à l'étude des interactions dans ce domaine, difficilement modélisable par des méthodes économiques traditionnelles. Dans [6] nous avons pu montrer que cette démarche permettait d'obtenir des résultats prometteurs, transposable à différents territoires, en considérant trois régions de l'Océan Indien: Analamanga et Itasy (Madagascar) et l'île de La Réunion. Dans ce papier nous avons détaillé et argumenté l'intérêt d'un modèle multi-agents. Nous avons également enrichi notre proposition en spécifiant un cadre complet de définition d'une architecture informatique adaptée à la problématique de gouvernance des « *Common Pool ressources* » en tirant profit des propriétés de l'approche système multi-agents.

Dans l'avenir, nous comptons affiner notre modèle agent et nos différents outils au travers d'expérimentations menées sur d'autres territoires insulaires de l'Océan Indien (Seychelles, Mayotte, Comores et Maurice). Il s'agira alors de créer des synergies positives résultants de la valorisation des ressources naturelles pour d'améliorer les politiques de gouvernance communautaire des ressources naturelles dans un contexte communautaire international.
Grâce à des simulations prospectivistes, un tel système devrait favoriser des actions d'anticipation sur la gestion de ressources naturelles et conduire à un comportement proactif des acteurs impliqués dans l'ensemble des niveaux de la chaine de décisions.

Références

[1] L. Alden Wily, Can We Really Own the Forest? A Critical Examination of Tenure Development in Community Forestry in Africa, Paper 251d, *Tenth Biennial Conference, International Association for the Study of Common Property (IASCP)*, p.15, 2004.

[2] S. Aubert, S. Razafianrison, A. Bertrand, *Déforestation et systèmes agraires à Madagascar : les dynamiques des tavy sur la côte orientale*, CIRAD ; CITE : FOFIFA, 2003.

[3] O. Brandouy, P. Mathieu et I. Veryzhenko, Optimal Portfolio Diversification? A Multi-agent Ecological Competition Analysis. *PAAMS (Special Sessions)*, pp. 323-331, 2012.

[4] P. Bontems, G. Rotillon, *L'économie de l'environnement*, Repères, 2007.

[5] F. Bousquet, O. Barreteau, C. Mullon, J. Weber, Modélisation d'accompagnement : système multi-agents et gestion des ressources renouvelables, in Actes internationales, *Quel environnement au 21ème siècle ? Environnement, maîtrise du long terme et démocratie*, Germes, 2001.

[6] A. Gaudieux, Y. Gangat, J. Kwan, R. Courdier. (2014). *Study of the interactions between stakeholders by a multi-agents system : application to the governance of natural resources in Miarinarivo District (Madagascar)*, conference MAS 2014, Bordeaux, September Full Paper, 2014.

[7] A. Gaudieux, R. Courdier, Y. Gangat, J. Kwan, *MASC : MAp Sectors Creator A tool to help at the configuration of multi-agents systems for everyone,* SIMULTECH 2014, Special Session on Applications of Modeling and Simulation to Climatic Change and Environmental Sciences - MSCCEC 2014, 28-30 August, 2014 - Vienna, Austria, Full Paper, pp. 836-844, 2014.

[8] Alain Fernandez, *Les nouveaux tableaux de bord des managers, Le projet Business Intelligence clés en main*, Ey-

rolles, 6e édition, ISBN 978-2-212-55647-6, 2013

[9] S. Kaufman, *Investigations*, Oxford University Press, 2000.

[10] P. Mathieu, S. Picault, Intérêt de la simulation centrée interactions pour les sciences humaines et sociales, *Revue des nouvelles technologies de l'information*, pp. 15-30, 2012.

[11] J-P. Müller et S. Aubert, Formaliser les rôles et les territoires par les systèmes multi-agents institutionnels, *JFSMA 2012*, Cépaduès, pp. 13-22, 2012.

[12] M. Olson, *The logic of collective action. Public goods and the theory of groups*, Cambridge, Massachusetts, Harvard University Press, 1965.

[13] E. Ostrom, *Gouvernance des biens communs*, Boeck, 2010.

[14] T. Ralambondrainy, J.-M. Médoc, R. Courdier, F. Guerrin, Tools to visualise the structure of multiagents' conversations at various levels of analysis, International Congress on Modelling and Simulation (MODSIM07), Christchurch, New Zealand, Dec. 10-13, 2007.

[15] Phil Simon. *The Visual Organization: Data Visualization, Big Data, and the Quest for Better Decisions* (1st ed.). Wiley Publishing. 2014

[16] R. Wade, The management of common property ressources : collective action as an alternative to privatisation or state regulation, *Cambridge Journal of Economics*, 11 : 95- 106, 1987.

Dynamique des relations de confiance dans une équipe d'agents virtuels

L. Callebert[a] lucile.callebert@hds.utc.fr
D. Lourdeaux[a] domitile.lourdeaux@hds.utc.fr
JP. Barthès[a] barthes@utc.fr

[a]Sorbonne universités, Université de Technologie de Compiègne,
CNRS, Heudiasyc UMR 7253

Résumé

De nombreuses études en psychologie sociale montrent que les relations de confiance interindividuelles influencent les dynamiques de groupe et la performance d'une équipe. Pour l'émergence de comportements collectifs cohérents dans un groupe d'agents, nous proposons un modèle d'agent cognitif dont le processus décisionnel individuel prend en compte ses relations de confiance, dont la modélisation et la dynamique est inspirée du modèle proposé par [Mayer et al., 1995].

Mots-clés : *système multi-agents, psychologie-sociale, modèle cognitif de la confiance*

Abstract

Studies in social psychology showed that trust relationships among people influence both group dynamics and team performance. For coherent collective behaviors to emerge among a group of agents, we propose a cognitive agent model whose decisional process takes into account the trust relationships. Trust relationships and their dynamics are defined following the model of organizational trust proposed by [Mayer et al., 1995].

Keywords: *multi-agents system, social psychology, cognitive model of trust*

1 Introduction

Les environnements sociotechniques sont aujourd'hui de plus en plus complexes, et le recours à la simulation peut être un bon outil de formation pour les personnes qui s'y trouvent confrontées. Dans le cadre de formation technique à la gestion de situations complexes ou critiques, des facteurs humains entrent en compte : les personnes sont amenées à faire face à des situations stressantes, à gérer leurs relations aux autres, etc. En effet, dans ces environnements complexes, les personnes ne travaillent pas seules mais sont bien souvent interdépendantes et le pouvoir de décision est partagé. Les compétences mises en jeu ne sont alors plus seulement techniques mais également sociales, et incluent des capacités spécifiquement liées au travail en équipe telles que le leadership ou la gestion des communications. L'un des paramètres clés influençant la mise en oeuvre de ces capacités est celui de la confiance : un climat de confiance dans une équipe permet notamment une répartition plus souple des tâches, un échange plus libre d'informations lors des communications, un meilleur investissement de chacun des membres de l'équipe [Jones and George, 1998]. Dans ce cadre de formation à des compétences non techniques, nous cherchons à modéliser le processus de prise de décision d'agents virtuels évoluant dans des environnements sociotechniques complexes en prenant en compte ce facteur de la confiance.

Pour peupler environnements ces evironnement sociotechniques complexes d'agents dont les comportements rendent compte de l'activité humaine telle qu'observée en environnement réel, nous proposons dans cet article une formalisation du modèle et de la dynamique de la confiance de [Mayer et al., 1995]. Nous expliquons ensuite comment la confiance est utilisée dans le processus décisionnel individuel des agents pour l'émergence d'une activité collective. Enfin nous présentons un exemple simple de la dynamique de la confiance avant de conclure.

2 Travaux connexes

2.1 Confiance et ACA

En informatique affective, la notion de confiance a principalement été abordée dans le domaine des agents conversationnels animés (ACA), en référence à la relation de confiance qu'un ACA doit établir avec l'utilisateur. Partant du principe qu'établir une relation de confiance agent-utilisateur va améliorer la qualité de l'interaction et l'appréciation de l'agent

par l'utilisateur, les travaux de recherche sont centrés sur les facteurs qui favorisent cette confiance. Par exemple [Antos et al., 2011] et [de Melo et al., 2013] montrent qu'un utilisateur fait plus confiance et s'engage plus facilement dans une relation de coopération avec un agent exprimant des émotions qu'avec un agent n'en exprimant pas.

Dans l'optique de l'amélioration de la relation agent-utilisateur,[Bickmore and Cassell, 2001], [Bickmore and Cassell, 2005] ont développé un modèle de dialogue social visant créer une relation de confiance avec l'utilisateur. Se basant sur des travaux en psychologie sociale, ils identifient plusieurs dimensions importantes dans une relation liant deux personnes : la *solidarité*, qui désigne le degré de similarité entre les dispositions comportementales de deux personnes ; la *familiarité*, qui désigne le degré d'échange d'information ; et la textitdimension affective, qui désigne le degré d'appréciation. Ces dimensions sont utilisées pour la définition de stratégies permettant à l'agent d'améliorer sa relation avec l'utilisateur. A l'issue de l'interaction avec l'agent, les sujets remplissent un questionnaire pour estimer le niveau de confiance qu'ils ont en l'agent : ce niveau de confiance reportée est utilisé comme critère d'évaluation. Dans ces travaux, aucune opérationnalisation de la relation de confiance n'est faite.

[Sansonnet and Bouchet, 2011], dans leurs travaux sur les ACA dotés de comportements rationnels et psychologiques, se servent du concept de la confiance : chaque agent a une confiance en soit, qui est dynamique et représente l'assurance cognitive de l'agent. Pour modéliser la relation à l'autre de l'agent, les auteurs se servent d'un vecteur à trois dimensions : la $dominance_{i,j}$, qui représente la puissance de i par rapport à j, $cooperation_i$ qui correspond à la tendance de i à se montrer coopératif et $trust_{i,j}$ qui caractérise la confiance de i et j. Toutes ces valeurs sont mises à jour au cours de l'interaction avec l'utilisateur, et sont utilisées pour influencer le processus de décision rationnelle de l'agent. Dans ces travaux comme dans les précédents, la confiance fait référence à un concept global se rapportant au climat qui règne entre l'utilisateur et l'agent. Cette notion vague englobe plusieurs questions : l'utilisateur pense-t-il que les informations données par l'agents sont fiables ? L'utilisateur pense-t-il que l'agent va agir dans son intérêt, a l'intention de l'aider ? Ce manque de différenciation limite les possibilités de mise en place de stratégies visant à l'amélioration de la confiance.

2.2 Confiance et négociation

La notion de confiance a également été étudiée dans les travaux sur la négociation : l'hypothèse est communément faite que le succès de la négociation dépend de la capacité du négociateur à mettre l'autre parti en confiance. Dans leurs travaux pour l'entrainement à la négociation, [Traum et al., 2005] modélisent la confiance selon les axes suivants : la *familiarité*, la *solidarité*, et la textitcrédibilité, représentant le degré de partage des croyances. La confiance est opérationnalisée comme une combinaison de ces trois dimensions sous la forme d'une valeur appartenant à l'intervalle $[0; 1]$ mise à jour à chaque interaction, et fait référence à la situation globale. La confiance est ensuite utilisée par l'agent virtuel pour décider de croire ou non les informations données par son interlocuteur et pour décider d'une stratégie de négociation. Dans des travaux ultérieurs, [Traum et al., 2008a], [Traum et al., 2008b] étendent leur modèle pour permettre une négociation multipartis : la confiance n'est plus globale mais devient spécifique à un agent.

2.3 Confiance et simulation sociale

Des travaux se sont également intéressés à la confiance dans le domaine de la simulation sociale : la relation de confiance est utilisée pour modéliser des agents aux comportements plus crédibles. La confiance est opérationnalisée par [Marsella et al., 2004] : une relation de confiance (ou méfiance) lie chaque paire d'agents. Cette relation est mise à jour à chaque interaction entre les agents et sert en partie à déterminer la valeur accordée par un agent aux messages reçus de la part d'un autre. D'une façon similaire, [Silverman et al., 2011] utilisent la confiance comme l'une des composantes des relations entre agents. Elle est opérationnalisée comme la *familiarité*, influencée par les actions des agents et utilisée pour déterminer la quantité d'information à donner lors d'un échange entre agents. Dans ces modèles, le seul 'type' de confiance utilisé fait référence à la fiabilité des informations échangées. Un autre manque peut être évoqué : aucune distinction n'est faite en fonction du contexte du dialogue. Or dans un environnement complexe, les domaines de compétences mis en jeu sont divers et les compé-

tences de chaque agent sont limitées à certains domaines. L'information donnée par un agent n'a donc pas la même valeur selon son niveau de compétence dans le domaine considéré.

Cette notion de contextualisation de la confiance a été étudiée par [Castelfranchi and Falcone, 1998], dont les travaux s'inscrivent à la fois dans les domaines des sciences cognitives et de l'informatique. [Castelfranchi and Falcone, 2001] proposent un modèle computationnel de la confiance pour l'étude des phénomènes sociaux. Un agent cognitif i attribue une valeur de confiance à j pour la réalisation d'un but identifié g en considérant principalement les dimensions suivantes :
– la capacité de j à réaliser g.
– la nécessité pour i que g soit réalisé par j.
– l'engagement de j par rapport à g.
– la motivation de j à aider i pour réaliser g.
La notion centrale de ces travaux est la délégation : un niveau de confiance suffisant de i à j permet à l'agent i de déléguer, de manière explicite ou implicite, une partie de l'activité à l'agent j. De cette notion de délégation émerge le concept d'activité collective. Dans cette approche, la confiance est entièrement définie par rapport à un but identifié et est donc contextualisée. La relation de confiance liée à l'appréciation de j pour i est également considérée ici : de part le paramètre de motivation. Cependant cette motivation est dirigée uniquement vers i ; or dans un contexte de travail collectif, la motivation de j peut être dirigée vers son équipe de travail et pas forcément spécifiquement vers l'un des membres de l'équipe, avec qui il peut par ailleurs ne pas entretenir d'affinités particulières.

Nous proposons une formalisation et une dynamique de la confiance basées sur le modèle de [Mayer et al., 1995] prenant en compte la double contextualisation de la confiance : par rapport à une personne particulière et dans un contexte identifié.

3 Modèle de la confiance

Le modèle de la confiance proposé par [Mayer et al., 1995] est issu d'études en psychologie sociale et organisationnelle. Les auteurs ont étudié différents modèles pour en construire un s'appliquant à tous les niveaux organisationnels et dont les dimensions englobent celles proposées dans la littérature. La relation de confiance de A à B est alors caractérisée par des éléments propres à celui qui fait confiance, mais aussi à sa représentation de l'autre. D'une part la *disposition* de A à faire confiance en règle générale correspond à la part irrationnelle de la confiance et est fortement liée à la personnalité de A ; elle va être déterminante dans le cas où A ne dispose pas d'informations sur B. D'autre part les éléments liés à la représentation de l'autre sont les suivants :
– la *bienveillance* : A doit croire que B veut son bien et ne va pas agir uniquement dans son propre intérêt. La bienveillance est liée à une relation particulière d'attachement entre A et B et est proche de la notion d'affinité ou d'appréciation.
– l'*intégrité* : A et B doivent partager des valeurs communes, et A doit penser que B va agir en accord avec ces valeurs. Dans le monde du travail, il est préférable qu'un employé partage les valeurs de son entreprise.
– les *capacités*. Cette notion est centrale puisqu'elle permet d'une part de découpler la notion de confiance de celle d'affinité (*e.g.* A peut apprécier B sans pour autant lui faire confiance) et d'autre part de contextualiser la confiance (*e.g.* A peut faire confiance à B pour faire la cuisine mais pas pour faire le ménage).
La confiance constitute une relation asymétrique entre deux personnes et est contextualisée : elle n'existe que par rapport à une personne identifiée sur une tâche particulière. Le modèle de [Mayer et al., 1995] est un modèle cognitif de la confiance : une relation de confiance implique des croyances issues d'une évaluation cognitive de la situation. La figure 1 illustre le modèle proposé par [Mayer et al., 1995].

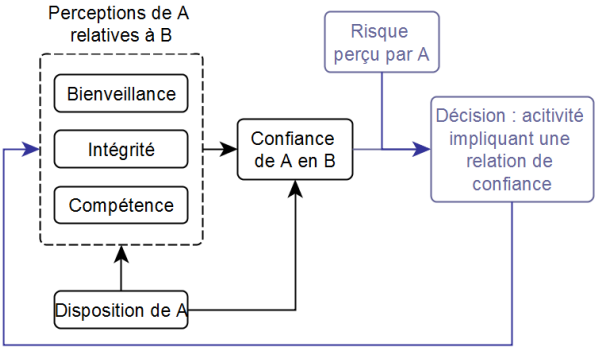


FIGURE 1 – Modèle de la confiance proposé par [Mayer et al., 1995]

Suivant le modèle de [Mayer et al., 1995], nous proposons de modéliser cette confiance $trust$ d'un agent i à un agent j pour une tâche k à un instant t comme une combinaison des éléments

de disposition, intégrité, bienveillance et capacité :

$$trust_{i,j,k} = \Psi(d_i, v_{b_{i,j}} b_{i,j}, v_{i_{i,j}} i_{i,j}, v_{c_{i,j,k}} c_{i,j,k})$$
$$\text{avec } c_{i,j,k} = \left\langle c_{1_{i,j}}, c_{2_{i,j}}, ..., c_{n_{i,j}} \right\rangle$$

où

- $d_i(t) \in D : [-1;1]$ représente la disposition de i à faire confiance. Une valeur positive réprésente quelqu'un de confiant par nature tandis qu'une valeur négative caractérise une personne de naturel méfiant.
- $b_{i,j} \in B : [-1;1]$ représente la confiance de i en la bienveillance de j à son égard. Une valeur positive indique que i pense que j lui veut du bien alors qu'une valeur négative indique que i croit que j veut lui nuire. La valeur 0 représente une relation neutre.
- $i_{i,j} \in I : [-1;1]$ représente la confiance de i en l'intégrité de j. De même que pour la bienveillance, une valeur positive (*resp.* négative) indique des intentions positives (*resp.* négatives) de la part de j à l'égard de l'équipe.
- $c_{i,j,k}(t) \in C : [0;1]$ représente la confiance de i en les capacités de j sur l'ensemble des compétences nécessaires à la tâche k. Une valeur de 1 indique que j maîtrise parfaitement les compétences alors que 0 indique qu'il ne les maitrise pas du tout.
- $v_X \in [0;1]$ représente une valeur de croyance associée à chacune des dimensions liée à la représentation de l'autre (*i.e.* $b_{i,j}$, $i_{i,j}$ et $c_{i,j,k}$). En effet, ces valeurs représentent une croyance : l'agent A *pense* qu'il peut avoir confiance en la bienveillance de B. L'agent A peut avoir un degré de certitude par rapport à ce qu'il pense plus ou moins élevé, par exemple selon la source de l'information : l'information peut être issue d'une observation directe ou être rapportée verbalement. v_X est égale à 1 si l'agent est certain, alors que v_X vaut 0 si l'agent ne sait pas du tout.

4 Dynamique de la confiance

La confiance entre deux personnes est évolutive : les niveaux de confiance entre deux personnes déterminent si elles s'engagent dans une collaboration impliquant une relation de confiance, et l'évaluation des événements et interactions liés à cette collaboration vont réflexivement modifier les niveaux de confiance entre ces personnes. La figure 2 illutre ce processus. Par ailleurs, d'après [Mayer et al., 1995], [Karsenty, 2010], en l'absence d'évenements dans l'environnement, la confiance reste stable. Il n'y a donc pas de dynamique temporelle interne de la confiance. La dynamique de la confiance est dont liée aux événements qui se produisent dans l'environnement.

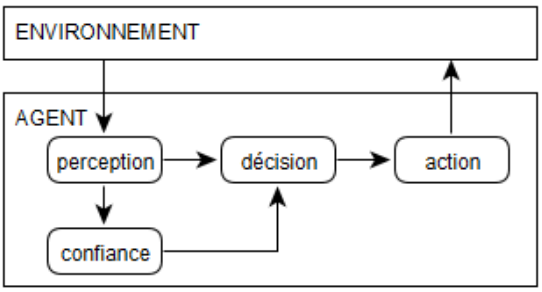


FIGURE 2 – Cycle d'un agent

La disposition à la confiance étant liée à la personnalité et donc stable, les composantes de la confiance qui sont dynamiques sont celles liées à la représentation de l'autre.

4.1 Dynamique des capacités

Les croyances de l'agent A sur les capacités de B évoluent dynamiquement lorsque A obtient, de manière directe ou indirecte, des informations relatives à un comportement de B :

- A obtient des informations de manière directe sur un comportement de B lorsqu'il observe B effectuer ce comportement dans l'environnement. A oeut alors évaluer la qualité d'exécution de la tâche en fonction du résultat et/ou du temps d'exécution. La réussite (*resp.* l'échec) d'une tâche k par j entraîne une augmentation (*resp.* une diminution) de la confiance de i en les compétences nécessaires à la réalisation de k. A est alors *sûr* de la compétence de B : $v_{c_{i,j,k}}$ est égale à 1.
- A obtient des informations de manière indirecte sur un comportement de B lors d'un dialogue avec un autre agent (pas obligatoirement B). La valeur accordée à l'information est alors pondérée par la confiance que A a en la source de l'information : $v_{c_{i,j,k}} \sim trust_{i,source}$.

C étant l'ensemble de définition des capacités, nous proposons la fonction :

$\mu : C \rightarrow C$
$\mu(c_{i,j,k}, comportement_j)$: représente la variation de la confiance de i en les compétences de j nécessaires à la réalisation de la tâche k.

4.2 Dynamique de la bienveillance et l'intégrité

Dans une équipe d'agents ayant des intentions communes, l'agent A va modifier ses croyances sur la bienveillance et l'intégrité de B en fonction du comportement de B. Le comportement de B est évalué par A en référence aux intentions propres de A et aux intentions communes à l'équipe. Le calcul de la confiance étant un processus cognitif, nous nous basons sur les théories de l'évaluation cognitive ordinnairement utilisées pour le calcul des émotions. L'évaluation d'un événement se fait selon plusieurs dimensions dont nous retenons la congruence motivationnelle et l'attribution de la responsabilité. Un comportement de B ayant un impact positif sur l'équipe (*resp.* sur A) sera évalué positivement par A, ce qui augmente la confiance de A en la l'intégité (*resp.* la bienveillance) de B. Cette évaluation est à modérer en fonction de l'attribution de la responsabilité à B dans le comportement qu'il exhibe : B avait-il conscience des conséquences de son comportement ? B était-il libre ou contraint de son comportement ?

B étant l'ensemble de définition de la bienveillance, I celui de l'intégrité et INT celui des intentions, nous proposons les fonctions suivantes :

$\Phi 1 : B \times INT \rightarrow B$
$\Phi 1(b_{i,j}, intentions_i, comportement_j)$: représente la variation de la confiance de i en la bienveillance de j liée à la perception et l'évaluation cognitive par i d'un comportement de j.

$\Phi 2 : I \times INT \rightarrow I$
$\Phi 2(i_{i,j}, intentions_{team}, comportement_j)$: représente la variation de la confiance de i en l'intégrité de j liée à la perception et l'évaluation cognitive par i d'un comportement de j.

De la même manière que pour les capacités, les valeurs de croyance associées à la confiance de A en la bienveillance et l'intégrité de B est mise à jour selon la manière dont l'information est obtenue (*e.g.* par observation directe ou rapportée aucours d'un dialogue) et selon la source de l'information.

5 Prise de décision

Nous avons présenté dans la partie précédente la dynamique de la confiance : des processus cognitifs permettent à l'agent d'évaluer la situation et de mettre à jour ses niveaux de confiance à chaque interaction. Les niveaux de confiance sont ensuite utilisés par l'agent pour prendre une décision, comme le montre la figure 2.

5.1 Prise de décision et plans partagés

Nous nous intéressons aux équipes d'agents évoluant dans des domaines complexes : les agents sont amenés à collaborer et il est nécessaire que chaque agent prenne en compte les autres dans sa décision individuelle. Lorsque les rôles de chacun des agents sont attribués *à priori* et les procédures de travail pré établies, comme dans [Rickel and Johnson, 1999], la prise en compte des autres est limitée : certaines des actions doivent être synchronisées et chaque agent doit savoir qui joue quel rôle, mais aucune décision n'est à prendre quant à la construction d'un plan commun ou aux tâches dont chacun est responsable. D'autres travaux se sont intéressés à la construction de tels plans partagés permettant de supporter une activité collective : [Pollack, 1986] et [Grosz and Sidner, 1988] ont été parmis les premiers à formaliser cette notion dans le contexte d'un dialogue agent-utilisateur. [Sidner, 1994] a introduit le concept de *négociation collaborative* pour désigner le processus par lequel deux agents A et B négocient la construction d'un plan partagé. [Chu-Carroll and Carberry, 1994], [Chu-Carroll and Carberry, 2000] introduisent la notion de *préférence* par rapport à une action pour la construction d'un plan partagé agent-utilisateur : l'agent prend en compte les préférences de l'utilisateur avant de lui faire des propositions de plans.

Tous ces travaux se placent dans le contexte d'une collaboration entre agents, mais deux principales limites sont à relever :
– les agents sont par définition collaboratifs et n'ont pas la possibilité de refuser la collaboration. Or en réalité les comportements humains ne sont pas toujours motivés par de bonnes intentions et la collaboration est rarement la seule option ;
– la notion de compétence n'est pas prise en compte : les agents sont conosidérés comme compétents par défaut sur l'ensemble des tâches composant le plan partagé.

5.2 La confiance pour la construction de plans partagés

[Mayer et al., 1995] distinguent dans leur modèle la confiance en elle-même et l'engagement

dans une activité (collaborative ou non) impliquant une relation de confiance. En effet, il peut y avoir collaboration sans confiance, par exemple lorsque la collaboration est contrainte. L'inverse est vrai également : une relation de confiance n'implique pas forcément que les agents s'engagent dans une activité collaborative : par exemple s'ils n'en ont pas l'utilité, ou parce que les risques liés à la collaboration sont trop élevés. Par ailleurs, une fois que les agents ont décidé de collaborer, la confiance intervient à chaque étape de construction d'un plan partagé : il faut prendre en compte les compétences de chacun avant de proposer un plan et une répartition des tâches.

L'engagement dans une activité collaborative et la construction d'un plan partagé nécessite donc, pour l'agent, de considérer :

– d'une part ses niveaux de confiance en les agents concernés. Le contexte de la collaboration intervient en premier lieu à ce moment puisque la confiance en les capacités est contextuelle et dépend des compétences mises en jeu ;
– d'autre part une évaluation des risques liés en particulier à cette collaboration. Les risques sont indépendants de la confiance et liés à la situation : la prise en compte du contexte est encore une fois indispensable.

C'est la combinaison de ces deux paramètres qui va permettre à l'agent de prendre sa décision. D'après [Mayer et al., 1995], une relation de confiance implique l'*envie* de prendre le risque de collaborer avec la personne, et l'évaluation des risques permet d'engager un comportement de confiance (*i.e. trusting behavior*, ici une activité collaborative) en toute connaissance de cause. L'évaluation des risques distingue les comportements de 'confiance aveugle' de ceux de 'confiance consciente' (distinction entre *confidance* et *trust* dans [Mayer et al., 1995]).

6 Illustration : collaboration pour aménager un bureau

Pour illustrer la dynamique des relations sociales, nous avons déroulé un exemple sur un scénario simple impliquant deux agents : A et B faisant partie d'une nouvelle équipe de travail et ne se connaissant pas veulent aménager le bureau qu'ils partagent. A et B vont donc collaborer pour mener à bien cette tâche, et établir des relations de confiance. Dans cet exemple, nous nous plaçons du point de vue de l'agent A.

La colonne de droite du tableau 1 décrit les événements qui se produisent dans l'environnement. La colonne de gauche présente les évolutions des niveaux de confiance de A en B suite à l'évaluation de la situation par A. A étant de nature confiante, $disposition_A > 0$. A $t0$, l'agent A ne connait pas l'agent B et n'a donc pas d'informations sur sa bienveillance, son intégrité ou ses compétences. La décision de A de se lancer dans une activité collaborative avec B est donc basée sur $disposition_A$. A $t1$, A et collaborent pour monter le bureau ($T1$). A met à jour ses niveaux de confiance : B participe à l'activité donc B est compétent pour $T1$, et $competence_{A,B,T1}$ augmente. B n'est pas contraint de réaliser $T1$ et permet d'atteindre un but de l'équipe : les intentions des B sont compatibles avec celles de l'équipe. L'intégrité de B est donc évaluée psitivement par A : $integrite_{A,B}$ augmente. L'action de B n'est âs pertinente du point de vue des intentions personnelles de A, A n'a donc toujours pas d'information sur $bienveillance_{A,B}$. A $t2$, B casse un pied du bureau : B n'était donc pas compétent pour $T1$ et $competence_{A,B,T1}$ chute. A étant de nature confiante évalue cependant l'action de B comme involontaire. $integrite_{A,B}$ reste stable. A ne dispose toujours pas d'information sur $bienveillance_{A,B}$. A termine de monter le bureau seul à $t3$ et n'a pas d'interaction avec B : les niveaux de confiance restent stables en l'abscence d'interaction entre les agents. A $t4$ B prépare un café pour A. B est donc compétent sur $T2$: $competence_{A,B,T2}$. De nature confiante, A interprète cette événement comme une bonne intention de à son égard (A aime le café donc action compatible avec les intentions de A) : $bienveillance_{A,B}$ augemente. Cette action n'est pas pertinente du point de vue de l'équipe, $integrite_{A,B}$ reste stable.

FIGURE 3 – Exemple de dynamique de la confiance

La figure 3 présente les courbes d'évolution des niveaux de confiance de A en B aucours de ce scénario. Les décisions futures de A seront prises par rapport à ces niveaux de confiance. Par soucis de simplicité, ce scénario à été conçu

TABLE 1 – Exemple de scénario illustrant la dynamique de la confiance

Situation initiale	**Confiance initiale de A à B**
A et B ne se connaissent pas mais savent qu'ils ont un but commun. A est de nature confiante.	$disposition_A > 0$ Pas d'information sur $competence_{A,B,x}$? $integrite_{A,B}$? $bienveillance_{A,B}$?
Evénement	**Evaluation par l'agent A : impact sur la relation de confiance entre A et B**
A et B collaborent pour monter le bureau (T1)	$competence_{A,B,T1}$ $\nearrow$: B est compétent. $integrite_{A,B}$ $\nearrow$ et $bienveillance_{A,B}$? : Congruence motivationnelle : réalisation d'un but commun. Responsabilité : B aide l'équipe et B n'est pas contraint.
B casse un pied du bureau	$competence_{A,B,T1}$ $\searrow$: Action T1 échouée : B n'est pas compétent. $integrite_{A,B}stable$ et $bienveillance_{A,B}$? : Congruence motivationnelle : contre la réalisation d'un but commun. Responsabilité : action involontaire.
A monte le bureau seul	$competence_{A,B,T1}$, $integrite_{A,B}$ et $bienveillance_{A,B}stables$: Pas d'interactions entre A et B.
B prépare un café pour A	$competence_{A,B,T2}$ $\nearrow$: Action réussie, B est compétent. $integrite_{A,B}stable$ et $bienveillance_{A,B}$ $\nearrow$: Congruence motivationnelle : A aime le café. Responsabilité : B aide A et n'est pas contraint.

pour présenter une situation à deux agents où les événements sont observés par A directement dans l'environnement : nous n'avons pas représenté ici la valeur de certitude que l'agent A accordre à chacun de ces niveaux de confiance. L'agent A est sûr que l'agent B est compétent pour prépaper le café puisqu'il le voit faire. Cependant, dans le cas d'un travail en équipe, beaucoup d'informations sont échangées entre les agents par le dialogue. Dans ce cas la valeur de certitude prend toute son importance : si B *propose* à A de lui préparer un café, A *pense* que B est compétent pour préparer un café, mais n'en est pas *sûr* temps que B n'a pas effectivement préparer le café. L'importance accordée par A à $competenceA, B, T2$ dans sa prise de décision (*e.g.* A accepte-t-il la proposition de B ?) sera moindre.

Cet exemple simple illustre également l'importance de la contextualisation : la confiance de A en B ne fait pas référence à une valeur globale pouvant s'appliquer de la même manière dans toutes les situations. A distingue son niveau de confiance en B selon les compétences mises en jeu ; et l'échec de B sur $T1$ n'empêchera pas A de collaborer à nouveau avec B sur des tâches mettant en jeu d'autres compétences. Cette distinction est extrêment importante dans un environnement de travail complexe où les compétence mises en jeu sont diverses et variées. Par ailleurs, un niveau de confiance faible sur une compétence précise n'est pas forcément un point négatif pour la collaboration : c'est une source d'information qui permet d'adapter les comportements (*e.g.* A peut surveiller l'agent sur les tâches dont il a la charge mais pour lesquelles ses compétences sont faibles) et ainsi limiter les erreur [Karsenty, 2010], [Amalberti, 2001].

7 Conclusion

Pour la modélisation d'agents sociaux évoluant dans des domaines complexes et supportant une activité collaborative, nous proposons de prendre en compte le facteur de la confiance. Nous avons présenté dans cet article un modèle des relations de confiance et de leur dynamique. Les relations de confiance entre un agent A et un agent B sont constituées de trois dimensions : la confiance de A en la bienveillance de B, en l'intégrité de B et en les compétences de B. La prise en compte du contexte, indispensable dans le cadre des domaines complexes, est faite à deux niveaux : la confiance est une relation spécifiquement dirigée d'un agent à un autre, et est re-

lative à une compétence identifée.

La relation de confiance évolue au fur et à mesure des observations par i des comportements des autres agents : la dynamique de la confiance est basée sur l'évaluation par i des comportements de j. La confiance est ensuite utilisée par les agents pour prendre une décision relative à une activité collaborative : tout d'abord la décision de s'engager dans une collaboration est basée sur la confiance, puis la confiance est également utilisée pour établir un plan partagé.

Références

[Amalberti, 2001] Amalberti, R. (2001). La maÃ®trise des situations dynamiques. *Psychologie franÃ§aise*, 46(2) :107–118.

[Antos et al., 2011] Antos, D., De Melo, C., Gratch, J., and Grosz, B. J. (2011). The Influence of Emotion Expression on Perceptions of Trustworthiness in Negotiation. In *AAAI*.

[Bickmore and Cassell, 2001] Bickmore, T. and Cassell, J. (2001). Relational agents : a model and implementation of building user trust. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 396–403. ACM.

[Bickmore and Cassell, 2005] Bickmore, T. and Cassell, J. (2005). Social Dialongue with Embodied Conversational Agents. In *Advances in natural multimodal dialogue systems*, pages 23–54. Springer.

[Castelfranchi and Falcone, 1998] Castelfranchi, C. and Falcone, R. (1998). Principles of trust for MAS : Cognitive anatomy, social importance, and quantification. In *Multi Agent Systems, 1998. Proceedings. International Conference on*, pages 72–79. IEEE.

[Castelfranchi and Falcone, 2001] Castelfranchi, C. and Falcone, R. (2001). Social Trust : A Cognitive Aproach. *J. Pitt. London : Wiley*.

[Chu-Carroll and Carberry, 1994] Chu-Carroll, J. and Carberry, S. (1994). A plan-based model for response generation in collaborative task-oriented dialogues. *arXiv preprint cmp-lg/9405011*.

[Chu-Carroll and Carberry, 2000] Chu-Carroll, J. and Carberry, S. (2000). Conflict resolution in collaborative planning dialogs. *International Journal of Human-Computer Studies*, 53(6) :969–1015.

[de Melo et al., 2013] de Melo, C., Carnevale, P., and Gratch, J. (2013). People's biased decisions to trust and cooperate with agents that express emotions. In *Proc. AAMAS*.

[Grosz and Sidner, 1988] Grosz, B. J. and Sidner, C. L. (1988). Plans for discourse. Technical Report BBN-6728, BBN LABS INC CAMBRIDGE MA.

[Jones and George, 1998] Jones, G. R. and George, J. M. (1998). The experience and evolution of trust : Implications for cooperation and teamwork. *Academy of management review*, 23(3) :531–546.

[Karsenty, 2010] Karsenty, L. (2010). Comment faire confiance dans les situations Ã risque ?

[Marsella et al., 2004] Marsella, S. C., Pynadath, D. V., and Read, S. J. (2004). PsychSim : Agent-based modeling of social interactions and influence. In *Proceedings of the international conference on cognitive modeling*, volume 36, pages 243–248. Citeseer.

[Mayer et al., 1995] Mayer, R. C., Davis, J. H., and Schoorman, F. D. (1995). An Integrative Model of Organizational Trust. *The Academy of Management Review*, 20(3) :709.

[Pollack, 1986] Pollack, M. E. (1986). A model of plan inference that distinguishes between the beliefs of actors and observers. In *Proceedings of the 24th annual meeting on Association for Computational Linguistics*. Association for Computational Linguistics.

[Rickel and Johnson, 1999] Rickel, J. and Johnson, W. L. (1999). Virtual humans for team training in virtual reality. In *Proceedings of the ninth international conference on artificial intelligence in education*, volume 578, page 585. Citeseer.

[Sansonnet and Bouchet, 2011] Sansonnet, J.-P. and Bouchet, F. (2011). Personnification d'entitÃ©s par Agents Conversationnels.

[Sidner, 1994] Sidner, C. L. (1994). An artificial discourse language for collaborative negotiation. In *AAAI*, volume 94, pages 814–819.

[Silverman et al., 2011] Silverman, B. G., Pietrocola, D., Nye, B., Weyer, N., Osin, O., Johnson, D., and Weaver, R. (2011). Rich socio-cognitive agents for immersive training environments : case of NonKin Village. *Autonomous Agents and Multi-Agent Systems*, 24(2) :312–343.

[Traum et al., 2008a] Traum, D., Marsella, S. C., Gratch, J., Lee, J., and Hartholt, A. (2008a). Multi-party, multi-issue, multi-strategy negotiation for multi-modal virtual agents. In *Intelligent Virtual Agents*, pages 117–130. Springer.

[Traum et al., 2008b] Traum, D., Swartout, W., Gratch, J., and Marsella, S. (2008b). A virtual human dialogue model for non-team interaction. In *Recent trends in discourse and dialogue*, pages 45–67. Springer.

[Traum et al., 2005] Traum, D., Swartout, W., Marsella, S. C., and Gratch, J. (2005). Fight, flight, or negotiate : Believable strategies for conversing under crisis. pages 52–54. Springer Berlin Heidelberg.

Caractérisation de la collaboration entre objets connectés mobiles par modélisation-simulation orientée agent

L. Lucien[a] laurent.lucien@femto-st.fr
C. Lang[a] christophe.lang@femto-st.fr
N. Marilleau[b] nicolas.marilleau@ird.fr
L. Philippe[a] laurent.philippe@femto-st.fr

[a]UMR CNRS 6184 Femto-ST / DISC,
Université des Sciences et Technologies de Besançon, France

[b]UMI 209 UMMISCO,
IRD/UPMC Bondy, France

Résumé

Aujourd'hui de plus en plus d'objets sont susceptibles de communiquer entre eux, comme les smartphones par exemple mais aussi, plus généralement, des objets du quotidien porteurs de capteurs multiples. Dans le cas d'entités mobiles autonomes évoluant dans un espace physique, la collaboration devient une nécessité pour qu'elles atteignent des objectifs complexes. Sa mise en oeuvre suppose de prévoir des schémas d'organisation mais aussi des protocoles d'échanges de données élaborés. Il faut alors prendre en compte les caractéristiques propres de ces objets simulés, modifier les protocoles de communication et choisir les informations à transmettre en fonction du domaine d'application dans le but de définir un vocabulaire adapté et d'optimiser les échanges.
Nous nous attachons à définir ce qu'est la collaboration et ce qui la caractérise. Sur la base de cette contribution, nous nous interrogeons sur quelques approches intéressantes pour la représenter tout en tenant compte des contraintes réelles des applications. Pour étudier nos propositions, nous utilisons les systèmes multi-agents qui constituent un paradigme adéquat pour la modélisation et la simulation de systèmes complexes mettant en oeuvre des entités mobiles (contexte de simulation continue). Nous proposons une architecture et un schéma de conception permettant l'intégration de la communication entre agents. Nous illustrons l'intérêt de la collaboration à travers l'exemple d'un réseau de drones ayant une mission collective globale.

Mots-clés : *Systèmes complexes, systèmes multi-agents, collaboration, modélisation*

Abstract

Nowadays more and more objects are able to communicate. We are thinking about smartphones but about usual objects carrying sensors too. In the case of autonomous mobile objects that evolve in a physical space, the collaboration is mandatory for reaching complex objectives. It needs specific organization scheme and specific communication protocols able to permit elaborated exchanges. We have to take object characteristics into account and to adapt both data to be transmitted and communication protocol to the application domain, finally to define a vocabulary and to optimize exchanges.
We start by the definition of the collaboration with all his characteristics. Then, we focus on some approaches in order to deploy it takeing into account the specificities of each case. We use multiagent systems to study our proposition. They are suitable for the modelling and the simulation of complex systems using mobile entities (continuous simulation context). We propose an agent architecture and design schema that permist to integrate collaboration into agents. We illustrate our propositions with a concrete example of a drone network with a global objective.

Keywords: *Complex systems, Multiagent systems, Collaboration, Modelling*

1 Introduction

Aujourd'hui, nous vivons dans un monde connecté où pléthore d'objets sont susceptibles de communiquer entre eux. Cela passe par les smartphones par exemple mais aussi, plus généralement, par un certain nombre d'objets du quotidien porteurs de capteurs multiples (de l'alarme à la voiture). Dans le cas précis d'entités mobiles autonomes évoluant dans un environnement donné, la collaboration devient une fonctionnalité clef pour atteindre des objectifs complexes au service des utilisateurs. Par exemple les trains de véhicules cherchent amé-

liorer la sécurité des conducteurs [5]. Sa mise en œuvre suppose de prévoir des schémas d'organisation mais aussi des protocoles d'échange de données susceptibles de permettre des échanges élaborés. Il faut alors prendre en compte les caractéristiques propres de ces objets simulés. Par exemple, des véhicules peuvent être amenés à se croiser très rapidement. Ainsi, collaborer ne peut se faire qu'en adaptant les protocoles de communication et les informations à transmettre en fonction du domaine d'application.

Nous nous attachons d'abord à définir ce qu'est la collaboration et ce qui la caractérise. Sur la base de cette contribution, nous nous interrogeons sur les modèles les plus pertinents pour la représenter tout en prenant en considération les contraintes réelles d'application. L'organisation des données et la structuration des objectifs doivent en effet permettre de pouvoir agir dans un environnement contraint par les durées des communications entre entités.

Pour étudier nos propositions, nous utilisons les systèmes multi-agents qui constituent un paradigme adéquat pour la modélisation et la simulation des systèmes complexes faisant intervenir des entités mobiles. Nous nous plaçons donc dans un contexte de simulation continue. La nature même des agents étant propice à l'autonomie de déplacement et de communication, ceux-ci sont donc bien adaptés à notre problématique. Dans cet environnement multi-agents, nous illustrons l'intérêt de la collaboration à travers l'exemple d'un réseau de drones (avec une mission collective globale telle que la détection d'incendies de forêt) ou encore un ensemble de véhicules connectés (avec des objectifs propres à chacun) évoluant dans une ville.

2 Collaboration

Avant de définir ce qu'est la collaboration dans le contexte de notre étude, nous donnons dans un premier temps un point de vue général en nous appuyant sur des définitions standards et sur différentes approches.

2.1 Concept et définition

La "*Collaboration*" est un concept courant qui veut dire "*action de collaborer, participer à un travail en commun. Collaborer, c'est travailler avec quelqu'un d'autre, l'aider dans ses objectifs, participer avec un ou plusieurs autres à une tâche commune*". Et même si les définitions usuelles sont proches, il ne faut pas confondre avec la "*coopération*" qui signifie "*action de coopérer, participer à une tâche commune. Coopérer, c'est prendre part, contribuer à une tâche commune.*" [1]

L'approche éducative [2] décrit la collaboration comme un processus qui va au-delà d'une simple coopération. On retrouve le partage des tâches, une certaine coordination des efforts et donc une synchronisation des résultats mais surtout une "*conception partagée*". Cela sous-entend que ce processus demande un travail de découpage du problème et surtout une compréhension réciproque des tenants et aboutissants de la part des intervenants.

Selon Marie-France Blanquet[1], le travail coopératif est accompli par une division du travail dans laquelle chaque personne est responsable d'une partie de résolution d'un problème. La collaboration implique un engagement mutuel des participants dans un effort coordonné pour résoudre ensemble le problème. Deux ou plusieurs personnes travaillant de manière synchrone ou asynchrone, dans le même milieu ou dans des lieux différents, échangent des points de vue sur des informations existantes, organisent leur travail collectif, définissent des objectifs communs en vue de construire ensemble un texte, une encyclopédie, des savoirs.

Marie-France Blanquet [1] insiste sur le concept d'engagement mutuel (forme de contrat moral) entre les collaborateurs. Le travail coopératif semble presque mécanique ou automatique tandis que le travail collaboratif implique une analyse commune du problème pour que le processus soit couronné de succès.

Dans le domaine des systèmes multi-agents, les définitions de la coopération et de la collaboration mettent en évidence l'interaction entre les agents et les principes de cognition : quelques actions de coordination et des algorithmes de résolution des conflits pour réaliser les tâches sont nécessaires. Selon Jacques Ferber [6], "*la collaboration est une forme d'interaction qui s'intéresse à la manière de répartir le travail entre plusieurs agents, qu'il s'agisse de techniques centralisées ou distribuées.*" Cependant, il reste très ambigu sur la définition et la distinction entre coopération et collaboration : "*La coopération demeure l'apanage des êtres capables d'avoir un projet explicite donc des agents cognitifs. Il est possible de parler d'une coopéra-*

1. http://www.larousse.fr/dictionnaires/francais
2. http://edutechwiki.unige.ch

tion de manière aussi bien réactive que cognitive si l'on envisage uniquement le résultat des actions et non les intentions des agents."

La collaboration doit donc être considérée comme une coopération de haut niveau avec le développement d'une compréhension mutuelle, associée à un partage de point de vue sur les tâches à accomplir par plusieurs individus en interaction [16, 1].

2.2 Comment qualifier la collaboration entre entités mobiles ?

Dans le contexte des objets mobiles intelligents, nous posons la définition suivante : *la collaboration est une interaction et un échange d'information entre deux entités (ou plus) en vue d'atteindre un résultat partiel commun qui participe à la réalisation d'un objectif plus global. Il s'agit d'un processus intentionnel et cognitif, une volonté de la part des entités qui collaborent avec un effort de partage des informations nécessaires et suffisantes ainsi qu'une vision commune du résultat à atteindre*.

Compte tenu de cette définition, répondre aux questions suivantes est une première façon d'illustrer et de qualifier la collaboration dans un système complexe composé d'objets intelligents :

- **Qui collabore ?** – Au moins deux entités susceptibles de se comprendre pour échanger des informations et/ou mettre en commun des ressources.
- **Pourquoi collaborer ?** – Pour atteindre plus facilement et plus rapidement un objectif ou pour satisfaire un besoin ou un désir.
- **Quand collaborer ?** – Quand une situation de blocage empêche la réalisation des objectifs ou quand survient simplement une occasion de collaborer en vue d'une optimisation.
- **Comment collaborer ?** – Il faut nécessairement que les entités soient en mesure de communiquer efficacement entre elles. Cela veut dire disposer d'un langage suffisamment évolué pour véhiculer les informations à échanger ainsi que les résultats à atteindre et formaliser les problèmes.
- **Quelles informations doit-on transmettre ?** – Toute information susceptible d'identifier un problème puis de participer à sa résolution. Il faut donc considérer chaque type d'information à échanger (degré de confidentialité selon les entités, caractéristiques des entités, etc).

3 Architecture d'agent collaboratif

Après avoir présenté la classification actuelle et quelques exemples d'architecture, nous discutons de l'architecture d'agent la plus efficace pour modéliser un processus de collaboration.

3.1 Travaux antérieurs

Depuis les années 90, la communauté scientifique a l'habitude de distinguer l'agent réactif de l'agent cognitif et d'en proposer une classification "classique" (voir Table 1) où les principales caractéristiques sont présentées. L'agent réactif est un "*agent qui ne dispose que d'un protocole simple et d'une capacité de communication réduite afin de répondre seulement à un stimulus par une action.*" [11]. L'agent cognitif est "*un agent possédant une base de connaissances avec toutes les informations et l'expertise nécessaires pour mener à bien sa tâche et sachant gérer les interactions avec d'autres agents et son environnement*"[6].

L'expérience a montré la nécessité de développer des agents intermédiaires qui ne sont ni purement réactifs ni purement cognitifs. Wooldridge [17] introduit ce nouveau type d'agent en présentant les architectures hybrides et un nouveau mode d'organisation de raisonnement (avec un agent doté d'un raisonnement déductif et un agent pourvu d'un raisonnement pratique). Aujourd'hui, il existe de nombreux exemples d'architecture réactive, cognitive et hybride [6, 17, 3]. Chacun d'eux donne un point de vue sur la réaction et la cognition ainsi que les moyens de les concevoir.

Boissier [3] propose un autre point de vue en comparant les différents types d'agents avec une vision raisonnement (agent réactif, hybride et délibératif) et coordination (agent autonome, interagissant ou social) dans la Table 2. Il énumère les différents modèles et plateformes connues au moment de la rédaction de l'article. Cependant il reste prudent sur la notion d'*agent cognitif* car seuls les *agents délibératifs* sont introduits dans ses travaux. Il les considère comme des agents cognitifs n'exécutant que des plans d'actions prédéterminés. Nous pouvons également remarquer que, pour lui, il n'existe pas encore d'agents hybrides avec un comportement social. C'est exactement cette voie que nous voulons explorer pour la mise en oeuvre de la collaboration comme nous définissons que celle-ci se fonde sur une interaction voulue entre deux entités intelligentes dans le cadre d'un processus

Agents réactifs	**Agents cognitifs**
Pas de représentation explicite de l'environnement	Représentation explicite
Pas de mémoire locale	Peut enregistrer des évènements et les exploiter
Comportement stimulus/action	Comportement complexe

TABLE 1 – Agents réactifs contre agents cognitifs [6]

Raisonnement / **Coordination**	**Agents réactifs**	**Agents hybrides**	**Agents délibératifs**
Agent autonome	Subsumption, PENGI, MANTA	Touring Machines	*IRMA, PRS/dMARS*
Agent interagissant	PACO, SMARRPS	*InteRRap*	IMAGINE, ARCHON, *AOP, COSY, GRATE*
Agent social	ECO, PACORG		*ADEPT, ASIC*, STARS, SAM, DIMA, CIA

TABLE 2 – Classification de Boissier des modèles et plateformes agents selon leur niveau de cognition et de sociabilité (les architectures de type BDI sont en italique) [3]

intentionnel.

Plusieurs architectures hybrides ont déjà été proposées pour permettre une communication élaborée entre agents et permettre la mise en oeuvre de processus collaboratifs. Par exemple, nous pouvons citer, sur la base de l'architecture BDI Beliefs-Desires-Intentions (Croyances, Désirs, Intentions) de Rao et Georgeff [14], les TouringMachines de Ferguson [7], le modèle InteRRaP de Müller [12] ou DIMA de Zahia Guessoum [10].

Le **modèle d'architecture BDI** [14] met en oeuvre les principaux aspects de la théorie de Michael Bratman sur le raisonnement pratique humain avec les notions de croyance, de désir et d'intention [2]. Nous retrouvons la notion de plan qui est une succession d'étapes, certaines d'entre elles pouvant invoquer d'autres plans. Le modèle BDI est étroitement associé à des modèles d'agents intelligents mais il n'inclut pas encore toutes les caractéristiques de ce type d'agent. Par exemple, il ne gère pas les aspects de communication entre agents. Ce modèle d'architecture est donc une première tentative pour résoudre un problème avec des plans d'actions mais il n'est pas suffisant pour réellement mettre en oeuvre un agent intelligent.

Les modèles des **TouringMachines [7]** (voir la Figure 1) ou **InteRRaP [12]** (voir la Figure 2) présentent des architectures multicouches avec une suite de couches dédiées : une couche réactive, une couche de planification ainsi qu'une de modélisation ou une couche coopérative. Pour le modèle InteRRaP, chaque couche utilise une partie d'une base de données (de type hiérarchique) contenant toutes les informations requises pour la bonne exécution de celle-ci.

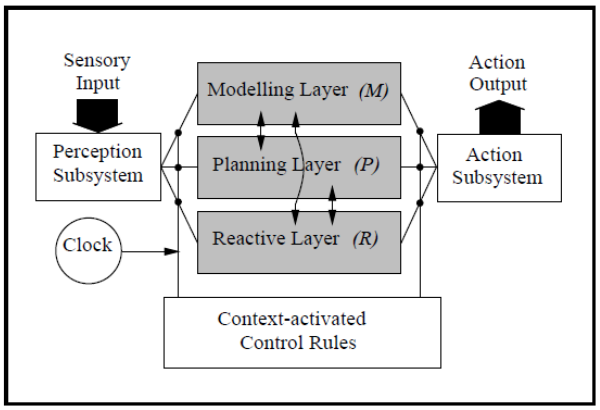


FIGURE 1 – TouringMachines (Fergusson)

Un autre exemple, le modèle **DIMA [10]** dispose d'une architecture modulaire où chaque module intègre tous les protocoles de communication, les algorithmes, les sous-programmes réactifs et cognitifs. C'est donc un modèle "évolutif" permettant l'ajout de modules en fonction des besoins. Dans le cadre de la modélisation d'échanges entre entités mobiles, le principal problème réside dans la gestion des communications entre ces différents éléments afin de transmettre les bonnes informations rapidement selon le contexte d'exécution.

3.2 La collaboration dans les SMA

Tous les modèles présentés précédemment proposent une architecture multi-couches avec une couche pour la représentation du monde, une

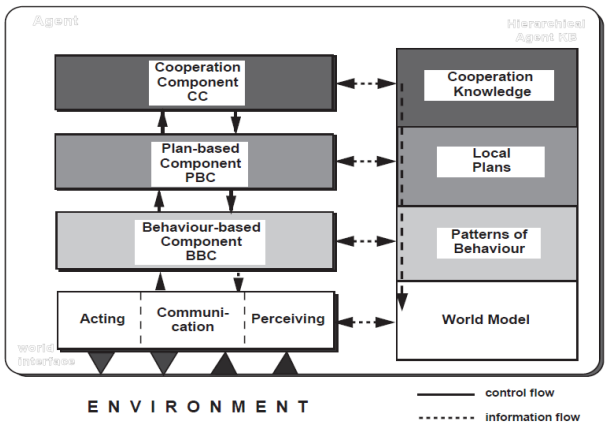


FIGURE 2 – InteRRaP (Müller)

couche pour les comportements de base (ou réactifs), une autre pour les comportements planifiés et une dernière pour les processus de communication et de collaboration. Les possibilités de communications entre agents sont d'ailleurs généralement limitées à cette couche dédiée (la couche supérieure la plus "cognitive").

Cette approche implique que l'agent doit utiliser par construction successivement toutes les possibilités de toutes les couches avant de demander de l'aide à un autre agent, quelque part dans l'environnement de simulation . Mais, dans les cas où des agents représentent des objets mobiles rapides, cette contrainte s'avère pénalisante en raison des temps de traitement.

Pour modéliser des échanges collaboratifs entre des entités mobiles, une **architecture d'agent hybride** est donc nécessaire pour répondre rapidement aux stimulii de l'environnement (partie réactive) mais aussi avec une certaine capacité de stockage pour enregistrer des expériences et une gestion de l'abstraction pour les objectifs et les priorités (partie cognitive).

Pour mettre en oeuvre des comportements collaboratifs entre agents, nous devons évoquer toutes les difficultés que nous pouvons rencontrer : la définition de la bonne architecture d'agent, l'enregistrement des informations (représentation du monde, les buts à atteindre, les interactions entre les agents), le protocole de communication à utiliser et, en fonction de la plate-forme de développement multi-agents, la gestion correcte des échelles de temps (échelle de simulation et échelle de communication par exemple).

Comment organiser les données dans une structure d'agent ? Ce n'est pas un défi trivial car cela dépend du type d'agent que nous voulons créer. La structure doit être adaptée au contexte. Dans un domaine proche, une large littérature portant sur les *ontologies* existe et propose différents moyens de représentation (de connaissances et de faits) : de simples fichiers texte avec des listes, un fichier XML pour les connaissances ou faits, des graphes (orientés ou non) ou une structure spécifique.

Il est nécessaire de définir le juste besoin afin d'éviter de surcharger les règles de comportement de l'agent (par exemple l'analyse non-optimisée d'un fichier XML de plusieurs centaines de lignes prend un temps non-négligeable). Une bonne structure de données permet une lecture facile des règles et donc d'optimiser la détermination et le suivi des objectifs.

Afin de ne pas compliquer la tâche, nous modélisons des entités autonomes homogènes avec des listes de faits, des règles et des objectifs limités. Ce travail est d'autant plus nécessaire dans un contexte d'échange d'informations avec des mises à jour réciproques des faits et des bases de connaissances des entités communicantes. La structure de données choisie devra intégrer toutes ces contraintes.

Concernant la **représentation du monde** pour l'agent, nous distinguons quatre approches possibles : de simples variables pour signifier l'état de l'agent, un graphe (orienté ou non) pour la représentation mentale, une matrice pour la représentation de l'environnement ou une structure spécifique (en fonction du problème à modéliser).

En fonction de l'espace modélisé, les structures de données seront choisies. Par exemple, une matrice sera pertinente pour le stockage des environnements rectangulaires mais un graphe sera plus intéressant pour enregistrer et matérialiser les liens entre les entités. Nous pouvons également concevoir une structure globale pour enregistrer toutes les informations : représentation du monde, entités rencontrées, objectifs, etc en utilisant par exemple une structure d'hypergraphe.

Pour la **gestion des communications**, nous devrions utiliser les ACL pour les échanges structurés dans les systèmes multi-agents. Il existe

différentes possibilités : KQML [8], FIPA/ACL [13], un langage ou un protocole spécifique.

Aussi bien pour KQML que pour FIPA/ACL, nous devons définir un vocabulaire suffisant pour permettre l'échange d'informations et de données entre les agents. Le choix du langage dépendra des possibilités offertes (conditions préalables, post-conditions, etc).

A propos de la **gestion du temps**, il est nécessaire de gérer une ou plusieurs échelles de temps (une pour la simulation du modèle et l'autre pour les aspects de communication entre les agents par exemple). Les outils de modélisation tels que NetLogo ou GAMA, ne parviennent pas à intégrer et à gérer nativement plusieurs échelles de temps.

Nous devons malgré tout conserver l'échelle de temps de la simulation comme l'échelle principale et pour chaque pas de temps, gérer un ou plusieurs niveaux d'échelles de temps internes. Ce mécanisme est nécessaire principalement pour le traitement des messages entre les agents mobiles (qui peuvent ne plus être en contact d'un pas de temps à l'autre alors que les messages ne sont pas encore totalement parvenus). Ce problème est secondaire pour les agents statiques.

4 Un modèle pour la collaboration

Nous devons définir une architecture multi-couche et multi-protocole pour communiquer avec d'autres agents (réaction, planification, abstraction, etc) en utilisant une organisation hiérarchisée des données (utilisant des dictionnaires définis et des données limitées, pas d'ontologie). Mais il faut également considérer la modélisation d'agents ayant des temps d'échanges (de communication) très courts dans certains cas particuliers, par exemple, deux véhicules se croisant à 130 km/h sur les autoroutes.

Nous proposons dans cette section une classification des propriétés d'une collaboration (gradient et types) et un schéma général présentant les différentes étapes du processus.

4.1 Gradient de collaboration

Le gradient de collaboration caractérise le genre de relations établies entre les entités collaborant et le rôle joué par chacune d'elles. Cette classification représente les fondations de notre future architecture d'agent. Nous pouvons distinguer quatre catégories :

Collaboration partielle – Simple échange partiel d'informations, juste nécessaire pour que chaque entité puisse continuer à satisfaire ses objectifs et ses besoins dans l'environnement. Ce genre de collaboration peut faire penser à un processus de coopération car quelques informations spécifiques sont seulement transmises aux autres agents. Il n'y a pas besoin d'échanges complexes. Par exemple, nous pourrions utiliser le principe de stigmergie où seules les informations spatiales sont communiquées.

Collaboration complète – Processus standard de collaboration avec partage du problème, découpage en sous-tâches, vision commune, coordination d'action et capitalisation des résultats. Il s'appuie sur une connaissance de l'organisation, un partage de l'information et des algorithmes de négociation (décrit dans 4.3). Par exemple, deux drones communiquent leur propre carte interne pour compléter leur vision réciproque de leur environnement. Puis ils échangent leurs objectifs afin d'optimiser l'utilisation de leurs ressources (batterie) ou d'utiliser leurs compétences spécifiques à bon escient.

Collaboration altruiste – Offre de service, entraide potentiellement sans contrepartie, facilitant la réalisation des objectifs sur le concept du bénévolat ou de la bienveillance [9]. Cela ne peut être mis en oeuvre que si l'entité offrant ses services n'a pas objectifs forts à atteindre (et se trouve donc dans une phase d'attente). Par exemple, une entité sans affectation préalable d'une mission avec un niveau élevé de batterie et en possession d'un grand nombre de informations utiles à transmettre, peut proposer ses services aux autres entités.

Collaboration forcée – L'une des deux entités n'est pas en mesure de refuser l'échange ou le partage de l'information car c'est une question de "vie ou de mort", au-delà des priorités propres ou des objectifs en cours de réalisation. Pour ce genre de collaboration, nous utilisons le concept de violation des règles et des comportements planifiés. Mais cela affectera le comportement général de l'agent. Par exemple, un drone n'a plus de batterie pour rentrer à la base mais

il doit transmettre une information importante (comme l'emplacement d'un feu de forêt par exemple). Il pourrait ainsi forcer un autre drone à finir cet objectif pour transmettre l'information dès que possible.

4.2 Types de collaboration

Nous venons de définir le gradient de collaboration. Nous pouvons également définir plusieurs types de collaboration (qui peuvent d'ailleurs être regroupés selon les usages) : synchrone et formel, informel et direct, etc.

Collaboration synchrone/asynchrone : dans un processus de collaboration synchrone, deux entités (ou plus) communiquent en temps réel afin de coordonner leurs actions. Dans le cas d'un processus asynchrone, les entités passent par des relais de communication (tableau blanc, d'autres entités, boite aux lettres, etc).

Collaboration formelle/informelle : une collaboration formelle structure l'échange d'informations et la méthode de communication (découpage du problème, partage des résultats, capitalisation des résultats). Une collaboration informelle n'utilise que des connaissances partielles de l'autre sans la mise en commun de l'expertise et des ressources, en prenant juste le minimum pour répondre à un besoin individuel.

Collaboration directe/indirecte : une collaboration directe se définit dans un même environnement, en communication directe avec l'autre : unité de lieu, d'espace et de temps. Une collaboration indirecte passe par des intermédiaires relayant l'information.

Collaboration locale/distante : une collaboration locale est une collaboration entre deux entités (ou plus) dans un contexte d'exécution locale avec les ressources et les informations directement disponibles. Une collaboration distante est une collaboration entre deux entités (ou plus) n'évoluant pas dans les mêmes lieux mais partageant une unité de temps.

Ces propriétés maintenant définies (gradient et types) décrivent la collaboration de manière statique. Le schéma général (dans le paragraphe suivant) en décrit la dynamique.

4.3 Schéma général

Nous définissons dans cette section le schéma général d'une collaboration avec sa phase d'initialisation et sa phase d'exécution. Le processus de collaboration est donc proposé en cinq étapes : la prise de contact, la communication préalable, la planification des tâches, l'exécution effective du processus de collaboration et sa conclusion.

La prise de contact est la première étape lorsque deux entités se rencontrent dans l'environnement. Ce contact initial peut être découpé en deux phases : une **interaction** et une **évaluation de l'entité rencontrée**. Nous pouvons distinguer trois types d'interactions. Tout d'abord, l'***interaction fortuite*** : il s'agit d'une simple rencontre due au hasard dans l'environnement où évolue les entités. Deuxièmement, l'***interaction provoquée*** : une des deux entités cherche à entrer en contact car elle a identifié un problème qu'elle ne sait pas résoudre et se met en quête d'une autre entité capable de l'aider. Troisièmement, l'***interaction induite*** : l'entité rencontrée ou contactée à dessein ne peut pas répondre directement mais elle connaît une autre entité pouvant le faire. Elle se met en contact avec ou met en relation les deux autres entités. Pour l'**évaluation de l'entité rencontrée**, différents aspects sont évalués : capacités cognitives, aptitude à communiquer, etc. Bien évidemment, si l'entité rencontrée n'est pas jugée capable, il y a interruption du cycle et reprise des investigations.

La deuxième étape concerne la communication : informations communiquées, objectifs détaillés, moyens de résoudre le problème, une **évaluation de la capacité à résoudre le problème** et la mission globale afin de vérifier que les entités peuvent collaborer ensemble et être utiles l'une pour l'autre.

La planification réelle de la collaboration est élaborée par les deux entités avec, d'une part, l'**organisation de la collaboration** qui comprend le découpage du problème, les affectations, les attentes et les processus de négociation et d'autre part, la **distribution des rôles** qui définit les objectifs de la collaboration, quels sont les récompenses pour chacune des entités collaborantes (algorithme de coûts et bénéfices) ?

La prise de contact, la communication et la planification sont des étapes d'initialisation. La partie principale de l'algorithme global est l'**étape d'exécution**. Chaque entité suit le plan

de tâche assigné. L'**évaluation des résultats**, en les comparant au plan initial, permet de valider si les objectifs sont atteints. Puis un autre processus permet de décider de la **poursuite de la collaboration** en utilisant les stratégies d'engagement [4], utiles pour évaluer s'il est nécessaire, indispensable ou inutile de chercher plus avant. Premièrement, l'***engagement aveugle*** *(blind commitment)* où l'intention de parvenir à une solution est plus forte que tout le reste au mépris des contraintes externes. Deuxièmement, l'***engagement "monomaniaque"*** *(single-minded commitment)* où une entité continue à maintenir une intention jusqu'à ce qu'elle estime que celle-ci a été satisfaite ou qu'il n'est plus possible de poursuivre. Troisièmement, l'***engagement "ouvert d'esprit"*** *(open-minded commitment)* lorsque l'entité continue aussi longtemps qu'elle pense qu'il est possible d'obtenir un résultat.

Pour finir, nous définissons trois autres étapes pour finaliser correctement le processus de collaboration : l'**évaluation de la collaboration** où les résultats sont examinés au regard des objectifs identifiés au début du processus, la **mise à jour de la base de faits et de connaissances** avec les informations recueillies et ajoutées aux connaissances respectives des entités ayant participées au processus de collaboration et, afin de pouvoir capitaliser efficacement, l'**historisation des échanges et des réalisations obtenues** pour effectuer une synthèse des relations entre entités lors de la collaboration.

5 Collaboration entre drones

Pour réaliser les premiers tests d'algorithmes de collaboration, nous avons décidé de mettre en oeuvre un réseau de drones avec NetLogo [15].

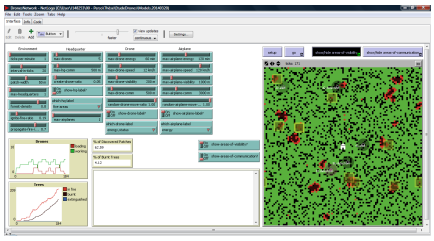


FIGURE 3 – Simulation d'un réseau de drones en NetLogo

5.1 Description

La mission principal des drones est de surveiller une forêt pour détecter les départs d'incendie et alerter la base. Mais il faut deux observations : une première détection et une confirmation afin d'être sûr que les pompiers peuvent être envoyés sur la zone incendiée. Nous avons programmé deux versions du modèle.

La première version des drones est assez simple. En effet, chacun d'entre eux possède certaines caractéristiques principales comme une vitesse moyenne de déplacement, un niveau de batterie (avec un algorithme de retour immédiat pour les forcer à rentrer à la base pour recharger), un champ de vision et des moyens de communication (uniquement avec la base pour alerter) limités. Ces premiers drones ne sont que des agents réactifs.

Pour la seconde implémentation, nous avons codé un comportement cognitif basique : enregistrement d'informations sur l'environnement (comme la carte du milieu forestier et les coordonnées des zones d'incendie) et possibilité de communiquer, non seulement avec la base mais aussi avec les autres drones situés à proximité immédiate. A noter que pour cette implémentation, nous avons choisi d'utiliser une matrice pour enregistrer ce que le drone voit de son environnement. Nous avons également géré un facteur d'obsolescence de l'information. Concernant la communication, les drones n'échangent que les coordonnées des zones d'incendies détectés. Les missions des drones sont également pris en charge (détection, confirmation, retour à la base).

5.2 Premiers résultats

Avec la première mise en oeuvre (Figure 3), on remarque que toute la zone est explorée, les mouvements aléatoires semblant efficaces sur un petit espace. Mais avec un environnement plus large, les drones n'explorent plus la totalité de la zone. Les mouvements aléatoires montrent ainsi leurs limites et les incendies couvrent rapidement des surfaces de plus en plus grandes. Dans ces conditions, la mission principale des drones qui est de surveiller la zone forestière et d'alerter les secours le plus vite possible devient difficile à remplir et la nécessité de la collaboration est mise en évidence.

Dans la deuxième implémentation, nous fixons un cap initial pour chaque drone. Le processus de déplacement est une combinaison de mouvements aléatoires et orientés. Avec cet algorithme simple, toute la zone est désormais couverte. Mais après la mise en oeuvre du nouvel algorithme de communication, nous pou-

vons observer un phénomène nouveau : l'effet "*salon de thé*". En effet, chaque drone semble vouloir communiquer avec tous les drones qu'il peut rencontrer. Les drones passent ainsi plus de temps à communiquer et à échanger des informations avec les autres qu'à explorer leur environnement. Ils finissent donc par rester à proximité quasi-immédiate de la base car il n'y a pas d'arbitrage entre les fonctions de déplacement et de communication.

Dans une future implémentation, nous devrons adapter cet algorithme afin de décider du comportement de chaque drone au regard de sa mission principale et de son environnement immédiat. Il y a en effet un temps pour communiquer et un temps pour explorer, le but étant de détecter les feux de forêt. Cela démontre l'importance de la mise en oeuvre des algorithmes de gestion des objectifs et des moyens de communication pour s'assurer que la mission principale est remplie de manière efficace.

6 Conclusion

Une analyse approfondie des travaux de la littérature montre que la collaboration est un processus intentionnel et cognitif qui nécessite de partager l'information nécessaire et qui oblige à avoir une vision globale de l'objectif à atteindre. Nous avons ensuite présenté les verrous techniques liés à l'implémentation de comportements de collaboration dans un système multi-agents. Notre contribution consiste en la définition des différents types de collaboration mais aussi des rôles et des obligations des entités qui prennent part à une collaboration.

Notre premier objectif est de proposer une solution globale afin d'implémenter un environnement de collaboration pour les agents. Partant de nos définitions, et dans nos travaux à venir, nous allons développer une architecture collaborative pour les agents basée sur une structure hybride multi-couches où chacune de ces couches sera capable de communiquer avec une autre couche ou avec les agents. La représentation du monde, et donc l'organisation des données, doivent être adaptées à ce contexte. Il sera probablement nécessaire de confronter les solutions où toutes les données seront fédérées dans une entité supérieure et celles où elle seront distribuées entre les agents, afin de garantir ou faciliter l'accès à l'information.

Grâce à ces propositions nous pourrons débattre sur l'intérêt de la collaboration dans le cadre de l'optimisation des systèmes complexes. Ensuite, nous proposerons une architecture technique complète avec toutes les fonctionnalités pour enregistrer et organiser les données. Nous utiliserons pour cela une implémentation sur la plate-forme GAMA (sur la base de notre premier exemple avec le réseau de drones qui a permis de mettre en évidence l'importance de la caractérisation de la collaboration et sa mise en oeuvre). Nous nous concentrerons ensuite sur les véhicules communicants notamment pour les problématiques d'aide à la conduite et d'amélioration du trafic routier.

Références

[1] M.-F. Blanquet. Web collaboratif, web coopératif, web 2.0. : quelles interrogations pour l'enseignant documentaliste. *Formation des personnes ressources en documentation*, 2009.

[2] M. Bratman. *What is Intention ?* Number n°27 ;n°69 in Report (Center for the Study of Language and Information (U.S.))). Stanford University, 1987.

[3] Jean-Pierre Briot, Yves Demazeau, and others. *Principes et architecture des systèmes multi-agents*, volume 217. Hermès Science Publications, 2001.

[4] Philip R. Cohen and Hector J. Levesque. Intention is Choice with Commitment. *Artif. Intell.*, 42(2-3) :213–261, March 1990.

[5] Jean Michel Contet, Franck Gechter, Pablo Gruer, and Abder Koukam. Multiagent system model for vehicle platooning with merge and split capabilities. In *Third International Conference on Autonomous Robots and Agents*, pages 41–46, 2006.

[6] Jacques Ferber. *Les systèmes multi-agents : vers une intelligence collective*. Informatique, Intelligence Artificielle. InterÉditions, 1995.

[7] Innes A. Ferguson. Touring Machines : Autonomous Agents with Attitudes. *Computer*, 25(5) :51–55, May 1992.

[8] Tim Finin, Richard Fritzson, Don McKay, and Robin McEntire. KQML as an agent communication language. In *Proceedings of the third international conference on Information and knowledge management*, pages 456–463. ACM, 1994.

[9] Claudia V Goldman and Jeffrey S Rosenschein. Emergent coordination through the use of cooperative state-changing rules. In *AAAI*, pages 408–413, 1994.

[10] Zahia Guessoum. *Un environnement opérationnel de conception et de réalisation de systèmes multi-agents*. PhD thesis, Paris 6, Grenoble, 1996. Th. : informatique.

[11] Sofiane Labidi and Wided Lejouad. De l'intelligence artificielle distribuée aux systèmes multi-agents. 1993.

[12] Jörg P. Müller and Markus Pischel. The Agent Architecture InteRRaP : Concept and Application. Technical report, 1993.

[13] Stefan Poslad, Phil Buckle, and Rob Hadingham. The FIPA-OS agent platform : Open source for open standards. In *Proceedings of the 5th International Conference and Exhibition on the Practical Application of Intelligent Agents and Multi-Agents*, volume 355, page 368, 2000.

[14] Anand S. Rao and Michael P. Georgeff. BDI Agents : From Theory to Practice. In *IN PROCEEDINGS OF THE FIRST INTERNATIONAL CONFERENCE ON MULTI-AGENT SYSTEMS (ICMAS-95*, pages 312–319, 1995.

[15] Ilias Sakellariou, Petros Kefalas, and Ioanna Stamatopoulou. An Intelligent Agents and Multi-Agent Systems Course Involving NetLogo. *Multi-Agent Systems for Education and Interactive Entertainment : Design, Use and Experience : Design, Use and Experience*, page 26, 2010.

[16] Gerhard Weiss. *Multiagent systems : a modern approach to distributed artificial intelligence*. MIT press, 1999.

[17] Michael Wooldridge. *An introduction to multiagent systems*. John Wiley & Sons, 2009.

De l'intérêt de l'éthique collective pour les systèmes multi-agents *

Nicolas Cointe[1] Grégory Bonnet[2] Olivier Boissier[1]

[1] Institut Henri Fayol – Mines Saint-Étienne, Laboratoire Hubert Curien, UMR CNRS 5516
[2] Normandie Université, GREYC, CNRS UMR 6072, F-14032 Caen, France

Mines Saint-Étienne, 158 cours Fauriel 42023 Saint-Étienne Cedex 2
prenom.nom@{emse.fr, unicaen.fr}

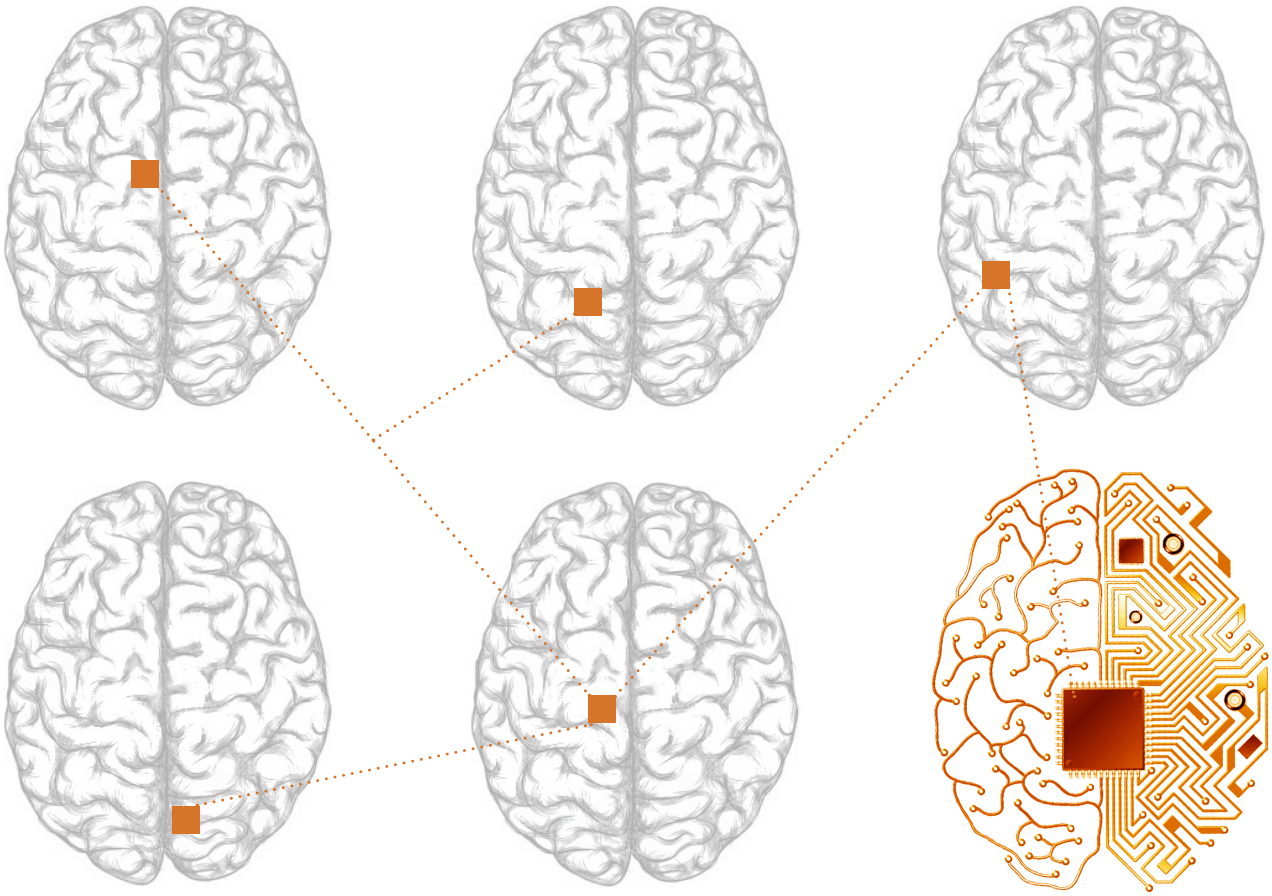
L'introduction d'agents autonomes artificiels dans des domaines tels que le milieu hospitalier, le trading haute fréquence ou encore le transport pourrait soulever de nombreux problèmes si ces agents ne sont pas en mesure de comprendre et suivre certaines règles morales. Par exemple, des agents capables de comprendre et utiliser le code de déontologie médical pourraient s'appuyer sur des motivations éthiques afin de choisir quelles informations diffuser, à qui et sous quelles conditions, conformément au principe du secret médical. L'intérêt pour le comportement éthique des agents autonomes semble être récemment apparu dans la communauté comme en témoignent les nombreux articles et conférences. Cependant, ces travaux s'intéressent uniquement à l'éthique à l'échelle individuelle de l'agent.

Or dans un système multi-agent (ou SMA), cette représentation individuelle ne permet à un agent que de se comporter individuellement de manière éthique dans un collectif mais le laisse démuni lorsqu'il doit tenir compte de l'éthique des autres agents. Par exemple, un agent gestionnaire d'investissements financiers pourrait être tout à fait capable de se comporter selon des principes de gestion responsable sans pour autant être capable de constater si ses partenaires ont ou non les mêmes scrupules. Prendre en considération la dimension multi-agent de ce problème nécessite l'exploration de nouvelles pistes telles que la création d'une ou plusieurs éthiques collectives ou la prise de décisions en coopération face à des problèmes d'éthique. Ces questions ont d'autant plus d'importance dans le contexte actuel de déploiement d'un nombre croissant d'agents dans notre environnement, collaborant entre eux ou avec des humains. Cet article a pour but de proposer des définitions et des questions mettant en évidence la problématique des éthiques collectives dans les systèmes multi-agents. Elles permettront dans des travaux futurs de proposer une formalisation des notions de philosophie pour la représentation explicite de concepts éthiques au sein d'agents autonomes.

La première étape de notre démarche est la proposition de définitions pratiques des notions philosophiques relatives à l'éthique. Ces définitions, étayées par des travaux philosophiques reconnus, nous permettent de proposer une première approche intuitive basée sur un système de règles morales employées au sein de principes éthiques dans le processus de décision de l'agent.

En nous appuyant sur un tel modèle abstrait d'éthique individuelle, nous cherchons à explorer un vaste ensemble de questions structurées autour des interactions entre agents, interactions entre agents et organisations et interactions entre plusieurs organisations. Nous avons en particulier identifié trois questions clés pour un agent : comment représenter l'éthique des autres agents, les juger et comment prendre en compte ce jugement dans les mécanismes de décision. Nous avons aussi identifié trois questions clés pour une organisation : comment construire, fusionner ou scinder des éthiques collectives, comment les faire respecter, comment les faire cohabiter avec des éthiques individuelles. Après s'être doté d'un premier modèle plus précis de l'éthique individuelle, nous envisageons de définir des mesures de similarité et de mécanismes de jugement qui sont les éléments nécessaires et fondamentaux pour envisager des éthiques collectives.

*Ces travaux sont financés par l'Agence Nationale de la Recherche (ANR) projet ETHICAA ANR-13-CORD-0006.



PFIA 2015

http://pfia2015.inria.fr

Plate-forme Intelligence Artificielle

Rennes du 29 juin au 3 juillet 2015