Forçage d'extension en argumentation abstraite par optimisation booléenne

Sylvie Coste-Marquis Sébastien Konieczny Jean-Guy Mailly Pierre Marquis
CRIL, Université d'Artois - CNRS
Lens, France
{coste,konieczny,mailly,marquis}@cril.fr

Résumé

La dynamique des systèmes d'argumentation constitue un sujet de recherche très développé actuellement. En particulier, le fait de modifier un système d'argumentation de façon à ce qu'un ensemble d'arguments E donné devienne une extension (ou soit inclus dans une extension), appelé problème du forçage (*enforcement*) de E, a reçu une certaine attention ces dernières années. Dans cet article, nous définissons une nouvelle famille d'opérateurs de forçage pour lesquels le forçage peut être réalisé en ajoutant de nouveaux arguments et de nouvelles attaques au système initial (comme c'est le cas dans les approches existantes), mais aussi en remettant en question les attaques (et non-attaques) de ce système. Cette famille d'opérateurs de forçage inclut les opérateurs existants, mais aussi de nouveaux opérateurs, pour lesquels le succès de l'opération de forçage est garanti. Nous montrons également comment le problème du forçage peut être modélisé et résolu via une traduction en problème d'optimisation booléenne. Une étude expérimentale conclut l'article et montre que notre approche se revèle efficace en pratique.

1 Introduction

Le travail de Dung sur l'argumentation abstraite [10] est à l'origine d'un cadre simple et puissant pour représenter et raisonner sur des arguments. Dans ce cadre, les arguments sont associés aux nœuds d'un graphe dirigé, appelé système d'argumentation (AF), et les arcs entre les nœuds encodent les attaques entre arguments. Plusieurs sémantiques d'acceptabilité ont été définies dans le but de déterminer des ensembles d'arguments qui peuvent être acceptés conjointement.

Les systèmes d'argumentation sont utiles pour modéliser et résoudre de nombreux problèmes, par exemple pour représenter et raisonner au sujet de débats dans un système multiagent. Quel que soit le problème considéré, la dynamique des systèmes d'argumentation, c'est-à-dire l'évolution d'un AF pour intégrer une nouvelle information, est une question importante. De ce fait, ce sujet a été largement étudié au cours des dernières années [7, 4, 6, 5, 8, 9].

Dans cet article, nous nous intéressons en particulier au problème du forçage : la question centrale est de déterminer s'il est possible de changer un AF pour assurer qu'un ensemble d'arguments particulier soit une extension, ou soit au moins inclus dans une extension. Ce problème a été étudié dans [2, 1]. Dans ces travaux, diverses contraintes concernent les changements qui sont autorisés : forcer un ensemble d'arguments peut être effectué via l'ajout de nouveaux arguments et de nouvelles attaques entre ces nouveaux arguments et les arguments du système. Cependant, aucun changement parmi les attaques du système de départ n'est permis.

De tels opérateurs de forçage peuvent être utiles pour modéliser un débat argumentatif entre plusieurs agents, car l'arrivée de nouveaux arguments dans le débat peut remettre en question les extensions existantes, et un agent peut alors souhaiter déterminer quels arguments et attaques doivent être ajoutés pour forcer l'ensemble d'arguments qu'il préfère. Cependant, dans de nombreux autres scénarios, il est impossible aux agents d'utiliser de nouveaux arguments pour justifier le changement, et il est alors utile de remettre en question les attaques entre les arguments existants [8, 9].

Dans cet article, nous définissons une nouvelle famille d'opérateurs de forçage, pour lesquels le forçage peut être réalisé en ajoutant de nouveaux arguments (et attaques) au système F donné (comme dans les approches précédentes), mais aussi en remet-

tant en question certaines attaques et (non-attaques) de F. Cette famille inclut les opérateurs de forçage précédents comme cas particuliers, mais aussi de nouveaux opérateurs pour lesquels le succès de l'opération de forçage est garanti. Nous montrons comment le problème de forçage pour les opérateurs de cette famille peut être modélisé comme un problème d'optimisation pseudo-booléenne. D'intenses expérimentations montrent que la méthode est applicable en pratique, et passe à l'échelle.

L'article est organisé de la façon suivante. Les notions de base d'argumentation abstraite sont rappelées en section 2. La section 3 présente les définitions des principales familles de forçage définies dans les travaux précédents par Baumann et Brewka, met en évidence certaines de leurs limites du point de vue de l'impossibilité, et ensuite définit une nouvelle famille d'opérateurs de forçage pour lesquels le succès est garanti. En section 4, nous expliquons comment réaliser le forçage en résolvant des problèmes d'optimisation. Avant la conclusion, la section 5 présente la méthode utilisée pour implémenter les opérateurs de forçage, et donne des résultats expérimentaux.

2 Préliminaires formels

Les définitions suivantes viennent de [10].

Définition 1. *Un système d'argumentation abstrait (AF) F est un graphe dirigé $\langle A, R\rangle$ où A est un ensemble d'entités atomiques appelées* arguments *et $R \subseteq A \times A$ est la* relation d'attaque.

La signification intuitive de la relation d'attaque est que $(a_i, a_j) \in R$ si, lorsque a_i est accepté par l'agent, alors a_j doit être rejeté. On dit qu'un ensemble d'arguments $E \subseteq A$ attaque un argument a_i si et seulement si $\exists a_j \in E$ tel que $(a_j, a_i) \in R$. Un argument a_i (respectivement un ensemble d'arguments E) défend l'argument a_j contre a_k tel que $(a_k, a_j) \in R$ si a_i (respectivement E) attaque a_k.

Dung a défini plusieurs *sémantiques d'acceptabilité*, qui visent à définir des *extensions* : une extension est un ensemble d'arguments qui peuvent être acceptés conjointement par l'agent. Quelle que soit la sémantique σ, $Ext_\sigma(F)$ est l'ensemble des σ-extensions du système F. Les différentes sémantiques reflètent certaines propriétés qui doivent être satisfaites par les extensions. Par exemple, $E \subseteq A$ est un ensemble sans conflit de $F = \langle A, R\rangle$ si et seulement s'il n'y a pas d'arguments $a_i, a_j \in E$ tels que $(a_i, a_j) \in R$. Ensuite, $E \subseteq A$ est une extension complète de $F = \langle A, R\rangle$ si et seulement si E est sans conflit et E contient chaque argument $a_k \in A$ qui est défendu par E. L'extension de base est l'élément minimal (selon $\subseteq$) parmi les extensions complètes. $E \subseteq A$ est une extension stable de $F = \langle A, R\rangle$ si et seulement si E est sans conflit et E attaque tout argument $a_k \in A \backslash E$.

3 Forçage d'extension

Forcer un ensemble d'arguments E est défini dans [2] comme un changement d'un AF F vers un autre F' de façon à ce que E soit une extension de F' ou soit au moins inclus dans une extension de F'. Plusieurs méthodes de forçages sont présentées, basées sur la notion d'*expansion* d'un AF. Une expansion est l'ajout de nouveaux arguments et nouvelles attaques à un AF, respectant diverses contraintes. Le forçage de E dans F est alors défini comme une expansion de F telle que E en est une extension. Trois sortes d'expansions sont considérées :

- L'expansion *normale* : de nouveaux arguments sont ajoutés, ainsi que de nouvelles attaques telles qu'au moins un des arguments considérés par chaque nouvelle attaque est un des nouveaux arguments (il n'y a pas de changement dans les attaques entre les arguments déjà présents).
- L'expansion *faible* est une expansion normale telle qu'aucune attaque n'est dirigée d'un nouvel argument vers un ancien argument. Les nouveaux arguments sont alors appelés *arguments faibles*.
- L'expansion *forte* est une expansion normale telle qu'aucune attaque n'est dirigée d'un ancien argument vers un nouveau. Les nouveaux arguments sont alors appelés *arguments forts*.

En plus de la nature de l'expansion, deux paramètres additionnels interviennent dans la définition d'un opérateur de forçage. Tout d'abord, le forçage peut être *strict* lorsque l'on attend que l'ensemble d'arguments E soit exactement une extension du système résultant du forçage, ou *non strict* lorsqu'il suffit que l'ensemble E soit inclus dans une extension du système. De plus, le forçage peut être *conservatif* si la sémantique reste la même ou *libéral* dans le cas contraire. Dans la suite, nous définissons des opérateurs de forçage qui peuvent être utilisés aussi bien pour le forçage conservatif que libéral, puisque la sémantique associée au système d'entrée n'est pas spécifiée dans la définition des opérateurs. Cependant pour des raisons de simplicité, nous nous concentrons sur des cas de forçage conservatif.

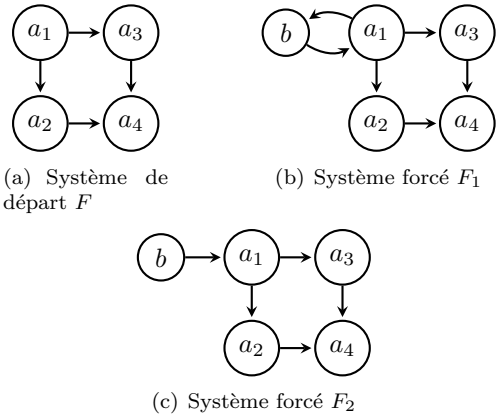
Définition 2. *Soit $F = \langle A, R\rangle$ un AF, σ une sémantique d'acceptabilité, $E \subseteq A$ un ensemble d'arguments, et A' un ensemble d'arguments tel que $A \cap A' = \emptyset$.* L'opérateur de forçage normal (respectivement normal strict) $+^N_\sigma$ *(resp. $+^N_{\sigma,s}$) est défini comme une fonction*

qui associe à F et E un AF $F' = \langle A \cup A', R \cup R_{A'} \rangle$, avec $R_{A'}$ un ensemble d'attaques (a_i, a_j) telles que $a_i \in A'$ ou $a_j \in A'$, et tel que E est inclus dans (resp. est exactement) une extension de F'. De plus,

- *si $R_{A'} \cap (A' \times A) = \emptyset$, alors $+_\sigma^{N,W}$ (resp. $+_{\sigma,s}^{N,W}$) est* l' opérateur de forçage faible (resp. strict faible) *;*
- *si $R_{A'} \cap (A \times A') = \emptyset$, alors $+_\sigma^{N,S}$ (resp. $+_{\sigma,s}^{N,S}$) est* l' opérateur de forçage fort (resp. strict fort).

Nous illustrons ci-dessous l'approche de forçage fort.

Exemple 1. *Soit F le système donné à la figure 1(a). Son ensemble d'extensions stables est $Ext_{st}(F) = \{\{a_1, a_4\}\}$. L'extension que l'on souhaite forcer est $E = \{a_2, a_3\}$. Un forçage normal possible est présenté à la figure 1(b) : les extensions stables de F_1 sont $Ext_{st}(F_1) = \{\{a_2, a_3, b\}, \{a_1, a_4\}\}$, et donc l'ensemble E est forçé. F_2, présenté à la figure 1(c), est un résultat possible de forçage fort, étant donné que $Ext_{st}(F_2) = \{\{a_2, a_3, b\}\}$. Le forçage faible n'est en*



(a) Système de départ F

(b) Système forcé F_1

(c) Système forcé F_2

FIGURE 1 – Processus de forçage normal et fort

revanche pas possible dans ce cas, puisqu'il faut rejeter a_1 pour obtenir a_2 et a_3 dans une extension, ce qui n'est pas réalisable en ajoutant des arguments faibles.

Comme illustré par l'exemple précédent dans le cas du forçage faible, il est important de noter que quel que soit l'opérateur de forçage considéré, l'opération peut échouer. Par exemple, considérons l'exemple simple du forçage de $E = \{a_1, a_2\}$ dans le système $F = \langle A, R \rangle$ tel que $(a_1, a_2) \in R$. Il est évident que le forçage de E n'est possible avec aucun des opérateurs de forçage décrits précédemment. Le théorème 2 et le théorème 3 de [2] présentent divers résultats d'impossibilité au sujet des opérateurs de forçage strict. Il est toutefois intéressant de noter le résultat qui indique que pour tout système F, il est possible de forcer tout ensemble d'arguments E qui est sans conflit dans F, via un opérateur de forçage fort non strict, et qui garantit aussi que l'ajout d'un unique argument est suffisant [2]. Cela signifie que le forçage fort non strict peut être réalisé en ajoutant seulement un seul argument, ce qui est particulièrement utile pour implémenter les encodages correspondants (voir section 4).

Notons que la présence de conflits dans l'ensemble E est une condition suffisante, mais pas nécessaire, pour provoquer l'échec du forçage. Dans le but de rendre ceci plus formel, introduisons la notion d'ensemble d'arguments non trivial pour une sémantique donnée :

Définition 3. *Soit $F = \langle A, R \rangle$ un système d'argumentation, et σ une sémantique. $E \subseteq A$ est un ensemble d'arguments non trivial pour σ dans F si et seulement si E est sans conflit dans F et $E \notin Ext_\sigma(F)$.*

Nous supposons que l'ensemble E à forcer est non trivial pour σ, afin d'éviter les cas évidents pour lesquels le forçage est d'emblée satisfait car E est une σ-extension de F, ou impossible en raison des conflicts. Cependant, cette hypothèse n'est pas suffisante pour éviter l'échec du forçage pour toute sémantique :

Proposition 1. *Pour tout $F = \langle A, R \rangle$ et $E \subseteq A$ un ensemble d'arguments non trivial pour la sémantique stable dans F, il n'y a aucun forçage strict de E dans F pour la sémantique stable.*

Proposition 2. *Pour tout $F = \langle A, R \rangle$, et $E \subseteq A$ un ensemble d'arguments non trivial pour la sémantique complète dans F,*

1. *si E ne se défend pas contre tout attaquant, alors il n'y a aucun forçage strict de E dans F pour la sémantique stable ;*
2. *sinon, si E défend un argument $a_i \in A \backslash E$, alors*
   (a) *il n'y a pas de forçage strict faible de E dans F pour la sémantique complète.*
   (b) *si les cycles de longueur impaire ne sont pas autorisés, alors il n'y a aucun forçage strict fort de E dans F pour la sémantique complète.*

Proposition 3. *Pour tout $F = \langle A, R \rangle$ et $E \subseteq A$ un ensemble d'arguments non trivial pour la sémantique de base dans F, si $Ext_{gr}(F) = \{\emptyset\}$, alors il n'y a pas de forçage strict de E dans F pour la sémantique de base.*

Forçage à arguments fixés. Dans les approches précédentes pour le forçage d'un ensemble d'arguments, il est supposé que de nouveaux arguments peuvent être ajoutés, et que les interactions entre les arguments préexistants ne changent pas. Cette méthode est particulièrement sensée lorsque le forçage est supposé être

le résultat d'un dialogue : étant donné un système d'argumentation qui représente l'état du dialogue, un agent ajoute de nouveaux arguments lorsqu'il veut convaincre un autre agent d'accepter un ensemble d'arguments particulier. Le fait d'interdire tout changement entre les attaques du système de départ est la raison des résultats d'impossibilité donnés ci-dessus. Il est intéressant de noter que le cas « contraire », c'est-à-dire si l'on considère des situations pour lesquelles on ne peut changer l'ensemble d'arguments, mais il est possible de faire évoluer la relation d'attaque, a aussi du sens. Par exemple, quand l'agent constate qu'un ensemble d'arguments est une extension dans le résultat du processus argumentation, mais ne correspond pas au résultat du système personnel de l'agent. Dans de tels cas, sans connaissance de nouveaux arguments, l'agent doit changer ses croyances au sujet de la relation d'attaque pour être cohérent avec l'ensemble d'arguments observé.

Définition 4. *Soit $F = \langle A, R\rangle$ un AF, σ une sémantique d'acceptabilité, et $E \subseteq A$ un ensemble d'arguments.* L' opérateur de forçage à arguments fixés (resp. strict à arguments fixés) $+^A_\sigma$ *(resp.* $+^A_{\sigma,s}$*) est défini comme une fonction qui associe à F et E à un AF $F' = \langle A, R'\rangle$, avec $R' \subseteq A \times A$, et tel que E est inclus dans (resp. est exactement) une extension de F'.*

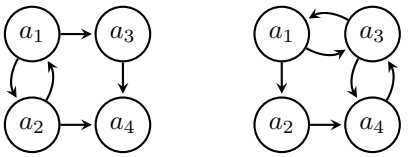
L'opérateur de forçage à arguments fixés garantit le succès de l'opération, même dans le cas strict :

Proposition 4. *Soit $F = \langle A, R\rangle$ un AF, σ une sémantique d'acceptabilité et $E \subseteq A$ un ensemble d'arguments. Il existe un forçage strict F' de E dans F.*

Bien sûr, les deux idées (ajouter des arguments et changer les attaques) peuvent être combinées :

Définition 5. *Soit $F = \langle A, R\rangle$ un AF, σ une sémantique d'acceptabilité, et $E \subseteq A$ un ensemble d'arguments.* L' opérateur de forçage général (resp. strict général) $+_\sigma$ *(resp.* $+_{\sigma,s}$*) est défini comme une fonction qui associe à F et E un AF $F' = \langle A \cup A', R'\rangle$, avec $R' \subseteq A \times A'$, et tel que E est inclus dans (resp. est exactement) une extension de F'.*

Exemple 2. *Considérons à nouveau l'AF F décrit à la figure 1(a). Nous avons donné un exemple de forçage non strict fort, mais comme montré par la proposition 1, il est impossible de réaliser un forçage strict sous la sémantique stable en utilisant les approches de Baumann (normal, fort et faible). Utilisons à présent le forçage à arguments fixés pour obtenir un forçage strict de l'ensemble d'arguments $E = \{\{a_2, a_3\}\}$. Un résultat possible est le système F_3 décrit à la figure 2(a), dont les extensions stables sont $Ext_{st}(F_3) = \{\{a_1, a_4\}, \{a_2, a_3\}\}$, et donc E est forcé en tant qu'extension stable du résultat. Un autre résultat possible est F_4 donné à la figure 2(b), dont les extensions stables sont les mêmes : $Ext_{st}(F_4) = \{\{a_1, a_4\}, \{a_2, a_3\}\}$.*



(a) F_3 (b) F_4

FIGURE 2 – Deux résultats possibles du forçage à arguments fixés

Dans l'exemple précédent, les deux options présentées pour le forçage, F_3 et F_4, ne sont pas équivalentes du point de vue du nombre d'attaques qu'il a été nécessaire d'ajouter pour parvenir au résultat. En effet, par rapport au système de départ F, F_3 ajoute une unique attaque (a_2, a_1), tandis que F_4 ajoute deux attaques (a_3, a_1) et (a_4, a_3). Cela nous conduit à nous poser la question du changement minimal dans le processus de forçage.

Changement minimal. Comme expliqué à la proposition 4, la possibilité de forcer un ensemble d'arguments est assurée lorsque des changements de la relation d'attaque sont autorisés. D'un point de vue pratique, cela offre une garantie de succès, ce qui est une propriété importante pour un tel opérateur. Une autre propriété attendue est le changement minimal, emprunté à la théorie des changements de croyances. Les processus de forçage peuvent conduire à plusieurs résultats, et donc les opérateurs de forçage définis précédemment doivent sélectionner un des systèmes possibles comme résultat. [1] a déjà étudié une telle notion de proximité pour les approches de forçage basées sur l'expansion normale. Il définit le changement minimal comme la minimisation du nombre d'attaques qui sont ajoutées au système durant le processus de forçage. Nous généralisons cette notion de changement minimal, en utilisant la distance Hamming pour mesurer à quel point deux systèmes sont différents.

Définition 6. *La distance de Hamming d_h entre deux systèmes d'argumentation $F = \langle A, R\rangle$ et $F' = \langle A', R'\rangle$ est définie par :*

$$d_h(F, F') = |(R \backslash R') \cup (R' \backslash R)|$$

Pour tout opérateur de forçage $+$, l'opérateur minimisant associé $\underset{\min}{+}$ est tel que le système résultant F' minimise la distance de Hamming avec le système de départ F parmi l'ensemble des systèmes qui sont un résultat possible de $+$.

4 Le forçage via la satisfaisabilité et l'optimisation

Une première observation au sujet du calcul du forçage d'un ensemble d'arguments est que lorsque l'on limite le nombre de changements autorisés dans la relation d'attaque, le forçage est difficile d'un point de vue calculatoire dans le cas général :

Proposition 5. *Soit $F = \langle A, R\rangle$ un AF, $E \subseteq A$, et k un entier. Déterminer s'il est possible de forcer E dans F pour la sémantique stable avec au plus k changements (ajouts ou retraits) d'attaques est* NP*-difficile.*

La proposition 5 assure que (sauf si P = NP) il n'y a pas d'algorithme en temps polynomial qui réalise le forçage avec changement minimal dans le cas général. Pour cette raison, il est tout à fait sensé de traiter les problème de forçage (resp. de forçage avec changement minimal) en utilisant des algorithmes développés pour résoudre (resp. optimiser) des problèmes NP-difficiles. C'est pourquoi nous procédons de la façon suivante : nous réduisons l'enforcement à un problème de satisfaisabilité propositionnelle, et le forçage avec changement minimal à un problème d'optimisation pseudo-booléenne.

Le forçage via la satisfaisabilité booléenne. Notre approche par traduction est basée sur la possibilité d'associer à un AF F et une sémantique σ une formule propositionnelle dont les modèles correspondent exactement aux σ-extensions de F.

Définition 7. *Étant donné F un AF et σ une sémantique, Φ_σ^F est une formule propositionnelle construite sur l'ensemble des variables booléennes $\{x_a \mid a \in A\}$, telle que $\{x_{a_1}, \ldots, x_{a_k}\}$ est un modèle de Φ_σ^F si et seulement si $\{a_1, \ldots, a_k\}$ est une σ-extension de F.*

Dans la suite, dans un souci de simplification, et comme aucune ambiguïté n'est possible, nous écrivons les formules en utilisant le symbole a_i au lieu de x_{a_i}. Nous nous focalisons sur l'encodage Φ_{st} de la sémantique stable, tel que présenté dans [3]. Étant donné $F = \langle A, R\rangle$, Φ_{st} est défini par

$$\Phi_{st} = \bigwedge_{a_i \in A} [a_i \Leftrightarrow (\bigwedge_{a_j : (a_j, a_i) \in R} \neg a_j)]$$

Ensuite, vérifier si un ensemble d'arguments E est une extension stable de F est équivalent à vérifier si la formule $\Phi_{st,s}^E = \Phi_{st} \wedge (\bigwedge_{a_k \in E} a_k) \wedge (\bigwedge_{a_l \notin E} \neg a_l)$ est cohérente. Pour réaliser un forçage non strict, il est nécessaire d'avoir un moyen de s'assurer que E est inclus dans une extension. Il suffit pour cela de retirer la conjonction $(\bigwedge_{a_l \notin E} \neg a_l)$ de la formule, $\Phi_{st,s}^E$, ce qui donne précisément la formule Φ_{st}^E dont nous avons besoin.

Dans le but d'établir un lien entre la sémantique et la structure du système d'argumentation dans les modèles de la formule, nous introduisons les variables booléennes att_{a_i,a_j} qui signifient qu'il y a une attaque de a_i vers a_j dans le système F considéré. Les formules données précédemment peuvent être généralisées en :

$$\Phi_{st}^{A,E} = \bigwedge_{a_i \in A} [a_i \Leftrightarrow (\bigwedge_{a_j \in A} (att_{a_j,a_i} \Rightarrow \neg a_j)] \wedge (\bigwedge_{a_k \in E} a_k)$$

et

$$\Phi_{st,s}^{A,E} = \bigwedge_{a_i \in A} [a_i \Leftrightarrow (\bigwedge_{a_j \in A} (att_{a_j,a_i} \Rightarrow \neg a_j)] \wedge (\bigwedge_{a_k \in E} a_k) \wedge (\bigwedge_{a_l \notin E} \neg a_l)$$

Assez clairement, propager les valeurs de vérité des variables att_{a_i,a_j} dans ces formules est suffisant pour obtenir la formule $\Phi_{st,s}^E$ et sa contrepartie non stricte. Cette formule est la base de nos encodages propositionnels des opérateurs de forçage d'extension. Il nous reste à introduire deux fonctions qui permettent de « décoder » de telles formules pour obtenir des systèmes d'argumentation :

- $Proj_{att}^A(\Phi) = \{m \cap \{att_{a_i,a_j} \mid a_i, a_j \in A\} \mid m \models \Phi\}$ est l'ensemble des modèles de la formule Φ projetés sur les variables att_{a_i,a_j}.
- $arg^A(m) = \langle A, R\rangle$ tel que $(a_i, a_j) \in R$ si et seulement si $att_{a_i,a_j} \in m$, avec m un modèle projeté sur les variables att_{a_i,a_j}, est le système correspondant à l'affectation des variables att_{a_i,a_j}. Ensuite, avec M un ensemble de tels modèles, $arg^A(M) = \{arg^A(m) \mid m \in M\}$.

Nous avons aussi besoin d'un encodage pour la structure d'un AF $F = \langle A, R\rangle$:

$$struct_{A'}(F) = (\bigwedge_{(a_i,a_j) \in R} att_{a_i,a_j}) \wedge (\bigwedge_{(a_i,a_j) \notin R} \neg att_{a_i,a_j})$$

où $a_i, a_j \in A \cup A'$. $struct(F)$ est une notation pour $struct_\emptyset(F)$.

Finalement, $\delta : \{F_1, \ldots, F_k\} \mapsto F_j$ tel que $F_j \in \{F_1, \ldots, F_k\}$ est une règle de *tie-break* qui sélectionne un unique système parmi un ensemble de systèmes.

Sur cette base, tout opérateur de forçage défini dans la section précédente peut être encodé comme un problème de satisfaisabilité d'une formule propositionnelle. En effet, par construction, chaque modèle de la formule $\Phi_\sigma^{A \cup A', E}$, lorsqu'il est projeté sur les variables att_{a_i,a_j} donne un AF qui est un forçage normal de E. Il nous suffit d'ajouter les bonnes contraintes pour assurer que le forçage est fort ou faible. Dans le but d'éviter l'introduction de nouveaux arguments, et obtenir les

opérateurs à arguments fixés, il nous suffit de considérer la formule $\Phi_\sigma^{A,E}$ dans l'encodage de l'opérateur. De façon similaire, les formules $\Phi_{\sigma,s}^{A\cup A',E}$ et $\Phi_{\sigma,s}^{A,E}$ peuvent être utilisées pour définir les contreparties strictes des opérateurs de forçage.

Définition 8. *Pour tout AF $F = \langle A, R\rangle$, tout ensemble d'arguments $E \subseteq A$, toute sémantique σ, et $X = \sigma$ ou $X = \sigma, s$:*

$F +_X^N E = \delta(arg^{A\cup A'}(Proj_{att}^{A\cup A'}(\Phi_X^{A\cup A',E} \wedge struct(F)))$

$F +_X^{N,W} E = \delta(arg^{A\cup A'}(Proj_{att}^{A\cup A'}(\Phi_X^{A\cup A',E} \wedge struct(F) \wedge (\bigwedge_{(a_i,a_j)\in A'\times A} \neg att_{a_i,a_j}))))$

$F +_X^{N,S} E = \delta(arg^{A\cup A'}(Proj_{att}^{A\cup A'}(\Phi_X^{A\cup A',E} \wedge struct(F) \wedge (\bigwedge_{(a_i,a_j)\in A\times A'} \neg att_{a_i,a_j}))))$

$F +_X^A E = \delta(arg^A(Proj_{att}^A(\Phi_X^{A,E})))$

$F +_X E = \delta(arg^{A\cup A'}(Proj_{att}^{A\cup A'}(\Phi_X^{A\cup A',E}))).$

Pour chacun de ces opérateurs de forçage $+$, tout AF F et tout ensemble d'arguments E, $Enc(F + E)$ représente l'encodage propositionel correspondant. Par exemple, $Enc(F +_\sigma^N E)$ est la formule propositionnnelle $\Phi_\sigma^{A\cup A',E} \wedge struct(F)$. Il suffit d'utiliser n'importe quel prouveur SAT pour trouver un modèle de $Enc(F + E)$ et ensuite décoder les valeurs de vérité des variables att_{a_i,a_j} pour déterminer un forçage de E dans F.

Le forçage avec changement minimal via l'optimisation pseudo-booléenne. Comme expliqué précédemment, [1] considère un principe de changement minimal dans le processus de forçage. Dans ce travail, la minimalité fait référence à la minimalité du nombre d'attaques à ajouter au système pour réaliser une expansion normale. Un moyen possible d'assurer le changement minimal est de définir une règle de *tie-break* δ qui sélectionne un des systèmes d'argumentation résultant du forçage qui minimise cette mesure. Dans le but de tirer avantage de logiciels d'optimisation disponibles, une approche alternative consiste à encoder le critère de minimalité en une fonction objectif pseudo-booléenne :

$$newAtt(A\cup A') = \sum_{(a_i,a_j)\in((A\cup A')\times(A\cup A'))\setminus(A\times A)} att_{a_i,a_j}$$

Pour le forçage fort ou faible, cette représentation de la fonction objectif peut être simplifiée, puisque les variables att_{a_i,a_j} qui correspondent à des attaques interdites ont nécessairement la valeur *faux*.

Le changement minimal pour le forçage à arguments fixés et le forçage général n'est pas aisé à encoder en utilisant les variables booléennes disponibles. Afin d'obtenir l'encodage voulu, nous considérons des variables additionnelles pour représenter l'état du système avant le forçage. On minimise ensuite le nombre de différences entre les valeurs de vérité de ces variables et celles qui leur correspondent dans le nouveau système. Formellement, pour tout couple d'arguments $(a_i, a_j) \in (A \cup A') \times (A \cup A')$, la variable $prev_{a_i,a_j}$ a la valeur *vrai* si et seulement si $(a_i, a_j) \in R$. Ainsi, $prev_{a_i,a_j} \oplus att_{a_i,a_j}$, où $\oplus$ désigne le connecteur « ou exclusif » habituel, donne l'information au sujet du changement de l'attaque (a_i, a_j) : s'il y avait précédemment une attaque de a_i à a_j, et que cette attaque n'est plus présente après le forçage, $prev_{a_i,a_j} \oplus att_{a_i,a_j}$ est *vrai*. C'est aussi *vrai* s'il n'y avait pas attaque avant le forçage, et qu'il y en a une après le forçage. L'encodage de la structure du système F doit aussi être revu pour tenir compte des variables $prev_{a_i,a_j}$:

$$struct_{A'}^{prev} = (struct_{A'}(F)_{|att_{a_i,a_j}\leftarrow prev_{a_i,a_j}})$$

Minimiser les différences sur la relation d'attaque est alors équivalent à minimiser la fonction objectif

$$attChange(A\cup A') = \sum_{a\in(A\cup A'),b\in(A\cup A')} prev_{a_i,a_j} \oplus att_{a_i,a_j}$$

Clairement, cette somme compte 1 pour toute attaque (a_i, a_j) dans le résultat qui concerne un argument de l'ensemble A', puisque $prev_{a_i,a_j}$ est toujours *faux* si $a_i \in A'$ ou $a_j \in A'$. Ainsi, cette approche peut aussi être utilisée pour le forçage général.

Résumons à présent les définitions des opérateurs de forçage avec changement minimal :

Définition 9. *Pour tout AF $F = \langle A, R\rangle$, tout ensemble d'arguments $E \subseteq A$,*

- *si $+_{\min}$ est un opérateur de forçage normal, fort ou faible (ou sa contrepartie stricte) avec changement minimal, alors forcer E dans F est équivalent à satisfaire $Enc(F +_{\min} E)$ tout en minimisant $newAtt(A \cup A')$;*
- *si $+_{\min}$ est un opérateur de forçage à arguments fixés ou général (ou sa contrepartie strictre) avec changement minimal, alors forcer E dans F est équivalent à satisfaire $Enc(F +_{\min} E) \wedge struct_{A'}^{prev}(F)$ tout en minimisant $attChange(A\cup A')$.*

Nous remarquons que la seconde méthode, qui fait intervenir la fonction objectif $attChange(A \cup A')$, est suffisante pour chaque opérateur de forçage. Cependant, comme cette méthode requiert l'ajout de variables booléennes $prev_{a_i,a_j}$, nous ne l'utilisons pas si ce n'est pas réellement nécessaire, pour éviter toute perte d'efficacité calculatoire.

Le cadre formel adapté à notre problème d'optimisation est le problème d'optmisation pseudo-booléenne (PB), qui est une extension du problème de satisfaisabilité booléenne.

Définition 10. *Étant donné un ensemble de variables booléennes $V = \{x_1, \ldots, x_n\}$ et une fonction $\mathcal{O} : \{0,1\}^n \to \mathbb{R}$, un problème* PB-*Opt* $\mathcal{P} = (\mathcal{C} = \{c_1, \ldots, c_m\}, \mathcal{O})$ *sur V est la recherche d'une affectation de chaque variable dans V telle que les contraintes*

$$\begin{array}{rc} c_1 : & w_1^1 x_1 + \cdots + w_n^1 x_n \geq k^1 \\ \vdots & \vdots \\ c_m : & w_1^m x_1 + \cdots + w_n^m x_n \geq k^m \end{array}$$

sont satisfaites, et la fonction objectif $\mathcal{O}$ atteint sa valeur optimale.

Dans notre cas, la valeur optimale de la fonction objectif est sa valeur minimale. Il est bien connu que toute formule propositionnelle peut être transformée en une formule équivalente sous forme normale conjonctive (CNF), et toute clause d'une formule CNF peut être ré-écrite comme une contrainte PB : la clause $x_1 \vee x_2 \vee \cdots \vee x_n$ est satisfaite si et seulement si la contrainte PB $x_1 + x_2 + \cdots + x_n \geq 1$ est satisfaite. Ainsi, les problèmes d'optimisation décrits précédemments peuvent être aisément traduits dans le formalisme PB.

5 Résultats expérimentaux

Dans notre étude expérimentale, nous nous sommes focalisés sur le problème de forçage avec changement minimal. Nous avons implémenté la famille d'opérateurs de forçage décrits dans cet article, en utilisant l'outil `CPlex` [13] comme moteur d'optimisation sous-jacent. Nous présentons uniquement les résultats obtenus pour trois approches : le forçage non strict fort de [2], et les versions stricte et non stricte de notre opérateur de forçage à arguments fixés. Dans chaque cas, nous avons utilisé la sémantique stable.

Nous étudions une classe de systèmes d'argumentation aléatoires [11, 12]. Étant donné un ensemble de n arguments, chaque attaque entre deux arguments est générée en utilisant une probabilité fixée p. Dans nos expérimentations, n varie de 200 jusque 500 arguments. Pour chaque n, les graphes sont divisés en quatre familles, correspondant à quatre valeurs de p distinctes. Nous avons utilisé les familles de systèmes de [11], où $p \in \{0.4, 0.65, 0.9\}$. Nous avons aussi généré des systèmes avec une probabilité $p = 0.1$. Il apparait dans nos expérimentations que le choix de p ne change pas de façon significative les performances de l'algorithme de forçage par traduction, donc nous reportons uniquement les résultats pour $p = 0.1$.

Nous avons calculé le forçage avec changement minimal d'ensembles d'arguments E dans des systèmes d'argumentation F contenant n arguments, avec $n \in \{200, 300, 400, 500\}$. Pour chaque système F avec n arguments, nous avons considéré des ensembles d'arguments E à forcer tels que $|E| = m$, m variant entre 1 et $\frac{35}{100}n$. Pour chaque couple de valeurs (n, m), nous avons généré 10 requêtes de forçage [1]. Sur la figure 3, l'ordonnée de chaque point des courbes correspond au temps de calcul moyen pour tous les couples (F, E) qui ont été considérés, où le nombre n d'arguments de F est indiqué en abscisse.

Le premier résultat intéressant qui ressort de nos expérimentations est que le forçage est traitable en pratique sur de tels systèmes d'argumentation générés aléatoirement, ce qui n'était pas évident, étant donné que le forçage est NP-difficile. Comme illustré à la figure 3, le temps de calcul augmente de façon raisonable avec le nombre d'arguments n, jusqu'à une valeur moyenne de 13.76 secondes (pour un écart-type de 0.48) obtenue avec des systèmes d'argumentation à 500 arguments, quand le forçage strict à argument fixés est considéré (courbe +), et jusqu'à une moyenne de 16.61 secondes (pour un écart-type de 5.31) quand le forçage fort est considéré (courbe ×).

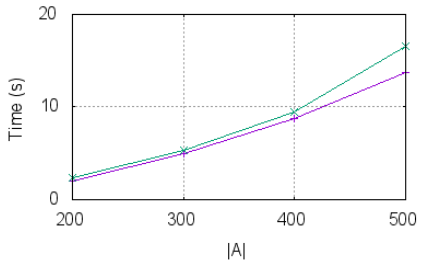
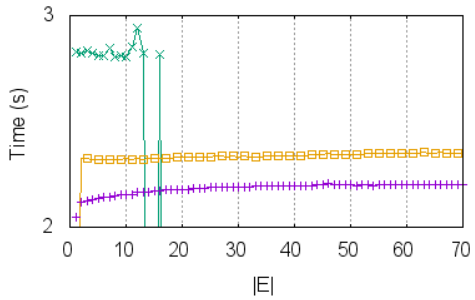


FIGURE 3 – Temps moyen pour le forçage fort (courbe ×) et strict à arguments fixés (courbe +), n variant de 200 à 500

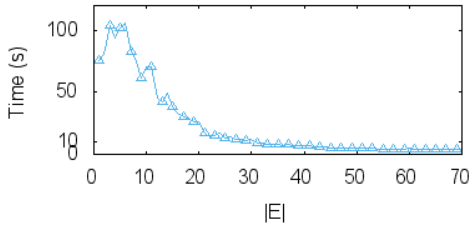
Ensuite, nous avons comparé les trois différentes approches sur des familles de systèmes d'argumentation à 200 arguments, laissant le cardinal de E varier de 1 à 70. Le but de cette comparaison est d'étudier l'impact du cardinal de E sur le comportement des opérateurs de forçage. Nous n'avons pas retiré les ensembles d'arguments triviaux des expérimentations, puisque notre approche peut retirer des attaques, rendant sans conflit un ensemble au départ conflictuel. Cela nous permet d'illustrer le taux d'échec du forçage fort, qui est (sans surprise) élevé, puisque le forçage est impossible dès que la requête de forçage est un ensemble conflictuel dans le système de départ. En effet, avec une probabilité p qu'une attaque apparaisse entre deux arguments, la probabilité qu'un ensemble d'arguments E de cardinal m soit sans conflit est $(1-p)^m$. Donc,

1. Nous appelons « requête de forçage » l'ensemble d'arguments E dont on attend qu'il soit une extension.

plus le cardinal de la requête de forçage est élevé, plus la probabilité que le forçage soit possible est faible. En particulier, dans nos expérimentations, le forçage fort échoue toujours losque $m > 20$. Clairement, le taux d'échec du forçage fort croît exponentiellement en fonction de m.



(a) Forçage strict à arguments fixés (courbe +), et succès (courbe ×) et échecs (courbe □) pour le forçage fort



(b) Forçage non strict à argument fixés

FIGURE 4 – Temps moyens, $n = 200$, m variant de 1 à 70

Nous avons comparé le temps de calcul pour le forçage pour les trois approches (voir la figure 4). La courbe × représente le temps moyen pour réaliser le forçage fort, tandis que la courbe □ correspond au temps nécessaire à l'algorithme pour reporter l'échec quand le forçage est impossible. Pour le forçage fort aussi bien que pour le forçage à arguments fixés (courbe +), il apparaît que le temps requis pour obtenir le résultat est le même quel que soit le cardinal de la requête de forçage et la probabilité d'attaques dans le système : entre 2 et 3 secondes. Le cardinal a plus d'influence sur l'opérateur de forçage non strict à arguments fixés. En effet, plus la requête de forçage est petite, plus le résultat est difficile à obtenir. Quand le cardinal croît, le temps de calcul décroît jusqu'à quelques secondes.

Enfin, nous avons mesuré les efforts requis du point de vue du changement (c'est-à-dire le nombre d'attaques à ajouter ou retirer) pour forcer E dans le système d'argumentation (voir figure 5). Clairement, l'effort requis croît avec le cardinal de la requête de forçage pour l'opérateur fort (courbe ×) et non strict à arguments fixés (courbes △). La courbe pour le forçage fort s'arrête bien avant les autres (après 14 arguments) en raison du haut taux d'échec que nous avons mentionné. Jusqu'à ce point, comme on peut l'observer, la courbe × est presque identique à la courbe △. Le forçage strict à arguments fixés (courbe +) requiert plus de changements de la relation d'attaque. Comme l'ensemble forcé doit être exactement une extension stable, même un petit ensemble d'arguments requiert l'addition d'un grand nombre d'attaques. Par exemple, avec $E = \{a_i\}$, il est nécessaire que a_i attaque chaque autre argument pour assurer que E est une extension stable. Quand le cardinal m de E croît, les efforts requis par les versions stricte et non stricte de l'opérateur à arguments fixés convergent.

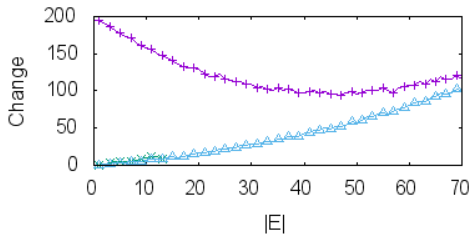


FIGURE 5 – Changement moyen pour le forçage fort (courbe ×), strict à arguments fixés (courbe +) et non strict à arguments fixés (courbe △)

6 Conclusion

Dans cet article, nous avons étudié le problème de forçage d'un ensemble d'arguments en tant qu'une extension d'un système d'argumentation. Notre contribution est multiple. Premièrement, nous avons indiqué plusieurs cas pour lesquels les approches de forçage existantes peuvent échouer, même si l'ensemble à forcer est sans conflit. Pour pallier cette faiblesse, nous avons défini de nouvelles méthodes de forçage, pour lesquelles le succès du processus est garanti. Pour chacune de ces méthodes, nous avons donné des encodages booléens qui permettent de tirer parti des logiciels de satisfaction et d'optimisation afin de procéder au forçage. Nous avons utilisé un outil d'optimisation connu pour implémenter une bibliothèque d'opérateurs de forçage, et nous avons expérimenté certains d'entre eux sur une large classe d'instances. Les expérimentations ont montré que notre approche est applicable en pratique et supporte le passage à l'échelle.

Il est intéressant de noter que d'autres types d'opérations de changement sont des forçages d'extension spécifiques. Par exemple, l'explication crédule étudiée dans [5] est le forçage d'un singleton. De façon simi-

laire, certains types de changements dirigés par les buts de [14] et certains opérateurs de révision de [8, 9] sont des opérateurs de forçage. Ainsi, notre approche de forçage est aussi utile pour calculer le résultat de tels opérateurs de changement dans les systèmes d'argumentation.

Ce travail ouvre plusieurs perspectives pour de futures recherches. Pour autant que nous le sachions, aucun des travaux existants au sujet du changement dans les systèmes d'argumentation n'a conduit à l'implémentation de logiciels (efficaces). Cependant, l'implémentation de systèmes logiciels pour l'argumentation est actuellement un sujet actif (dans la même veine, voir l'organisation d'une compétition de prouveurs d'argumentation [15]). La création de notre logiciel de forçage est issue de la même volonté de fournir des outils de raisonnement argumentatif, ce qui est aujourd'hui une étape nécessaire pour faire progresser ce domaine. Nous prévoyons ainsi d'encoder et d'implémenter des opérateurs de forçage pour d'autres sémantiques. Cette tâche est directe pour d'autres sémantiques pour lesquelles le calcul d'une extension est un problème (au pire) NP-complet. En effet, de telles sémantiques σ peuvent être encodées par des formules propositionelles Φ_σ similaires à la formule présentée ici pour la sémantique stable. Les sémantiques dont la complexité est plus élevée, comme par exemple la sémantique préférée, nécessitent un travail plus approndi, étant donné qu'il faut passer par un cadre formel différent (comme les formules booléennes quantifées par exemple), ou accepter une croissance exponentielle de la taille des encodages. D'autres extensions de nos approches peuvent être envisagées, comme la prise en compte du changement minimal du statut des arguments ou l'ajout de contraintes d'intégrité.

Remerciements

Ce travail a bénéficié d'une aide de l'Agence Nationale de la Recherche portant la référence ANR-13-BS02-0004 dans le cadre du projet AMANDE.

Références

[1] Ringo Baumann. What does it take to enforce an argument ? Minimal change in abstract argumentation. In *ECAI'12*, pages 127–132, 2012.

[2] Ringo Baumann and Gerhard Brewka. Expanding argumentation frameworks : Enforcing and monotonicity results. In *COMMA'10*, pages 75–86, 2010.

[3] Philippe Besnard and Sylvie Doutre. Checking the acceptability of a set of arguments. In *NMR'04*, pages 59–64, 2004.

[4] Pierre Bisquert, Claudette Cayrol, Florence Dupin de Saint Cyr, and Marie-Christine Lagasquie-Schiex. Change in argumentation systems : exploring the interest of removing an argument. In *SUM'11*, pages 275–288, 2011.

[5] Richard Booth, Dov Gabbay, Souhila Kaci, Tjitze Rienstra, and Leendert van der Torre. Abduction and dialogical proof in argumentation and logic programming. In *ECAI'14*, pages 117–122, 2014.

[6] Richard Booth, Souhila Kaci, Tjitze Rienstra, and Leendert van der Torre. A logical theory about dynamics in abstract argumentation. In *SUM'13*, pages 148–161, 2013.

[7] Claudette Cayrol, Florence Dupin de Saint Cyr, and Marie-Christine Lagasquie-Schiex. Change in abstract argumentation frameworks : Adding an argument. *J. Artif. Intell. Res.*, 38 :49–84, 2010.

[8] Sylvie Coste-Marquis, Sébastien Konieczny, Jean-Guy Mailly, and Pierre Marquis. On the revision of argumentation systems : Minimal change of arguments statuses. In *KR'14*, pages 52–61, 2014.

[9] Sylvie Coste-Marquis, Sébastien Konieczny, Jean-Guy Mailly, and Pierre Marquis. A translation-based approach for revision of argumentation frameworks. In *JELIA'14*, pages 397–411, 2014.

[10] Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming, and n-person games. *Artif. Intell.*, 77(2) :321–357, 1995.

[11] Wolfgang Dvorák, Matti Järvisalo, Johannes Peter Wallner, and Stefan Woltran, 2011. see `http://www.dbai.tuwien.ac.at/research/project/argumentation/dynpartix/examples/`.

[12] Wolfgang Dvorák, Matti Järvisalo, Johannes Peter Wallner, and Stefan Woltran. Complexity-sensitive decision procedures for abstract argumentation. *Artif. Intell.*, 206 :53–78, 2014.

[13] IBM. IBM ILOG CPLEX Optimization Studio : Optimization model development toolkit for mathematical and constraint programming, 2014.

[14] Dionysios Kontarinis, Elise Bonzon, Nicolas Maudet, Alan Perotti, Leon van der Torre, and Serena Villata. Rewriting rules for the computation of goal-oriented changes in an argumentation system. In *CLIMA XIV*, pages 51–68, 2013.

[15] Matthias Thimm and Serena Villata. First International Competition on Computational Models of Argumentation (ICCMA'15), 2015. see `http://argumentationcompetition.org/2015/`.