C-SAKey : une approche de découverte de clés conditionnelles dans des données RDF

Nathalie Pernelle[1], Danai Symeonidou[2], Fatiha Saïs[1]

[1] LRI - UNIVERSITÉ PARIS SUD, Bâtiment Ada Lovelace, Orsay, France
prénom.nom@lri.fr

[2] Telecom Paris Tech, Paris, France
danai.symeonidou@telecom-paristech.fr

Résumé : L'exploitation des liens d'identité entre ressources RDF permet aux applications de combiner des données issues de différentes sources. Les approches permettant de lier des données sont largement fondées sur l'existence de clés éventuellement composites. Ces clés étant rarement disponibles, des approches récentes se sont intéressées à la découverte automatique de clés à partir de données RDF. Cependant, dans certains domaines, les classes de l'ontologie sont très générales et les clés valides pour tout l'ensemble d'instances d'une classe sont peu nombreuses. Aussi, dans l'approche C-SAKey, nous proposons de détecter des clés conditionnelles qui ne s'appliqueront qu'à un sous- ensemble des instances d'une classe. Nous avons réalisé une première expérimentation sur un jeu de données de l'INA qui montre que les clés découvertes par notre approche peuvent effectivement varier selon les conditions exprimées dans la clé.

Mots-clés : Intégration de données, Liens d'identité, Liage de données, Clés, RDF, OWL

1 Introduction

Les approches de liage de données permettent de générer des liens d'identité entre deux ressources RDF (voir Ferrara *et al.* (2011) pour un état de l'art). Pour la plupart, ces approches s'appuient sur l'utilisation de propriétés discriminantes (Hu *et al.* (2011); Saïs *et al.* (2009); Nikolov & Motta (2010); Al-Bakri *et al.* (2015)). Une clé est un ensemble de propriétés particulièrement discriminant puisque en théorie, les valeurs de ces propriétés permettent d'identifier un objet du monde réel sans ambiguïté. Ces clés peuvent être utilisées par un raisonneur pour inférer logiquement des liens d'identité ou pour construire des fonctions de similarité plus complexes prenant en compte des mesures de similarités élémentaires entre littéraux. Les clés peuvent être déclarées dans une ontologie représentée en OWL mais elles sont rarement disponibles. En effet, si certaines clés peuvent facilement être déclarées par un expert du domaine, telle que la propriété $isbn$ pour un livre, d'autres clés composites sont difficiles à déterminer, même pour un expert : dans quelle mesure un nom de famille, un prénom et une date de naissance suffisent-ils à identifier un chercheur ? Pourtant, les données RDF étant souvent incomplètes, plus les clés disponibles sont nombreuses, plus les liens d'identité générés grâce à ces clés par un outil de liage de données seront nombreux. Aussi, des approches récentes se sont intéressées à l'exploitation de sources de données RDF pour découvrir des clés automatiquement (Atencia *et al.* (2012); Pernelle *et al.* (2013); Soru *et al.* (2015); Symeonidou *et al.* (2014)).
Parfois, pour certains domaines, le nombre de clés découvertes par ces approches est limité, ou les clés sont trop complexes dans le sens où elles combinent un grand nombre de propriétés. Dans de telles applications, différentes stratégies peuvent être appliquées. Certaines approches s'intéressent aux ensembles de propriétés qui peuvent éventuellement comporter un certain nombre d'exceptions (Atencia *et al.* (2012); Symeonidou *et al.* (2014)). En effet, même

si un ensemble de propriétés n'est pas une clé, son utilisation peut permettre de découvrir de nombreux liens d'identité corrects. Ainsi, le téléphone est une propriété qui peut permettre de découvrir de nombreux liens d'identité pour des instances d'une classe $Restaurant$ même s'il existe un complexe hôtelier particulier pour lequel deux restaurants partagent le même numéro de téléphone. Atencia *et al.* (2012); Soru *et al.* (2015) découvrent des clés dont la sémantique ne s'appuie pas sur la sémantique OWL d'une clé qui sont cependant intéressantes quand les propriétés sont localement complètes. SAKey permet de découvrir des clés OWL, appelées *presque-clés*, comportant un nombre d'exceptions défini manuellement (Symeonidou *et al.* (2014)). Ces approches permettent de découvrir des clés valides dans des sous-ensembles d'instances des classes considérées mais ne permettent pas de caractériser ces sous-ensembles. Dans cet article, nous présentons une approche appelée C-SAKey qui permet de découvrir des clés conditionnelles en exploitant des sources de données RDF. Une clé conditionnelle est une clé qui ne s'applique qu'à un sous-ensemble des instances d'une classe, sous-ensemble défini par des contraintes imposées sur les valeurs de certaines propriétés. Une première expérimentation sur un jeu de données de l'Institut National de l'Audiovisuel (INA) montre que les nouvelles clés découvertes peuvent effectivement varier selon les contraintes imposées sur les valeurs des propriétés.
En section 2, nous définissons quelques notions préliminaires. En section 3, nous présentons l'approche C-SAKey puis, en section 4, l'expérimentation que nous avons menée. Enfin, nous conclurons.

2 Préliminaires

Dans cette section, nous présentons tout d'abord le modèle de données puis l'approche SAKey sur laquelle s'appuie l'approche de découverte de clés conditionnelles C-SAKey.

Modèle de données

RDF (Resource Description Framework) est un modèle de données proposé par le W3C pour décrire des faits représentés sous forme de triplets $<subject, property, object>$. Une source de données RDF peut être associée à une ontologie qui va représenter le vocabulaire utilisé pour décrire les ressources. Une ontologie O peut être représentée par le tuple $(\mathcal{C}, \mathcal{P}, \mathcal{A})$ où $\mathcal{C}$ est l'ensemble des classes, $\mathcal{P}$ est l'ensemble des propriétés (relations et attributs) et $\mathcal{A}$ est un ensemble d'axiomes.

En OWL2 [1], il est possible de déclarer dans $\mathcal{A}$ qu'un ensemble de propriétés est une clé pour une expression de classe CE. Plus précisément, hasKey(CE ($ope_1, \ldots, ope_m$) ($dpe_1, \ldots, dpe_n$)) permet de déclarer que chaque instance de l'expression de classe CE est uniquement identifiée par les relations ou les relations inverses ope_i et les attributs dpe_j. Cela signifie qu'il n'existe pas de couple d'instances distinctes de la classe CE qui partagent des valeurs pour l'ensemble des relations et attributs de la clé. Par exemple hasKey($Researcher$ () ($firstName$, $lastName$)) permet d'exprimer que les attributs $firstName$ et $lastName$ forment une clé pour la classe $Researcher$.

1. http ://www.w3.org/TR/owl2-overview

Brève présentation de l'approche SAKey

SAKey est une approche efficace de découverte de clés dans des fichiers RDF (Symeonidou *et al.* (2014)). Afin de découvrir des ensembles de propriétés ayant un très fort pouvoir discriminant, et afin d'être capable de découvrir ces propriétés dans des fichiers RDF où des erreurs ou des duplicats peuvent exister, SAKey recherche des *n-presque clés* minimales : il s'agit des plus petits ensembles de propriétés pour lesquels il existe au plus n instances de la classe considérée qui partagent des valeurs pour cet ensemble de propriétés avec au moins une autre instance de la classe. Ces instances représentent les exceptions de la clé.

Dans l'ensemble de triplets RDF $D1$ présenté dans la figure 1, des exemples de clés découvertes pour la classe $Researcher$ seraient :

– 0-presque clés (pas d'exception) : {firstName, lastName}, {lastName, position} ...
– 2-presque clés (au plus deux exceptions) : {lastName}, {firstName}, ...

L'un des problèmes de la recherche de clés est de pouvoir passer à l'échelle afin de pouvoir apprendre des clés sur des volumes de données importants. Pour être efficace, SAKey recherche d'abord des *(n+1)-non-clés* maximales, des non clés présentant au moins n+1 redondances, et dérive ensuite les *n-presque-clés* minimales à partir de ces non clés. De plus, SAKey filtre les propriétés et leurs valeurs, et élague l'espace de recherche des non clés en s'appuyant sur certaines caractéristiques des données du jeu de données considéré, caractéristiques qui sont dynamiquement découvertes lors de l'application de SAKey.

Researcher(r1), $firstName(r1, "Fatiha")$, $lastName(r1, "Sais")$,
$worksIn(r1, "LRI"), position(r1," Assistant\ professor")$
Researcher(r2), $firstName(r2, "Nathalie")$, $lastName(r2, "Pernelle")$,
$worksIn(r2, "LRI")$, $position(r1," Assistant\ professor")$,
Researcher(r3), $firstName(r3, "Chantal")$, $lastName(r3, "Reynaud")$,
$worksIn(r3, "LRI")$, $position(r3, "Professor")$,
Researcher(r4), $firstName(r4, "Pierre")$, $lastName(r4, "Marquis")$,
$worksIn(r4, "CRIL")$, $position(r4," Professor")$
Researcher(r5), $firstName(r5, "Olivier")$, $lastName(r5, "Roussel")$,
$worksIn(r5, "CRIL")$, $position(r5, "Assistant\ professor")$
Researcher(r6), $firstName(r6, "Pierre")$, $lastName(r6, "Roussel")$,
$worksIn(r6, "CRIL")$, $position(r6, "Research\ engineer")$

FIGURE 1 – Exemple de source de données RDF ($D1$)

L'approche a été testée sur différents jeux de données et les résultats ont montré l'intérêt des *presque clés* pour le liage de données et l'efficacité de leur découverte. L'approche C-SAKey est une extension de cette méthode.

3 C-SAKey : une approche de découverte de clés conditionnelles

De façon à enrichir l'ensemble des clés susceptibles d'être découvertes, nous proposons une approche de découverte de *clés conditionnelles* appelée C-SAKey. Dans l'exemple D1, présenté en figure 1, nous pouvons observer que, sans exceptions, les propriétés $\{lastName, worksIn\}$ ne forment pas une clé pour la classe $Researcher$. Il existe en effet deux chercheurs dont le

nom de famille est $''Roussel''$ et qui travaillent dans le même laboratoire. Dans cet exemple, autoriser deux exceptions est suffisant pour découvrir que $\{lastName, worksIn\}$ peut être considéré comme une clé pour un grand nombre d'instances. Pourtant, dans une source de données de taille importante comportant de nombreux chercheurs, autoriser quelques exceptions ne sera peut-être pas suffisant. De plus, il existe peut-être des laboratoires pour lesquels le nom est suffisant pour identifier un chercheur. Il nous semble intéressant de pouvoir découvrir quels sont ces laboratoires. Dans cet exemple, la propriété $lastName$ est une clé pour les chercheurs travaillant au $''LRI''$, tandis qu'elle ne l'est pas pour les chercheurs du $''CRIL''$. Aussi, nous proposons de découvrir des clés valides pour des sous-ensembles de données caractérisés par une condition. Nous considérons des conditions pouvant être exprimées par une conjonction de propriétés pour lesquelles une valeur constante sera spécifiée. Plus précisément, si l'on considère une variable X représentant une instance de la classe c, et un ensemble de propriétés $Pcd = \{pcd_1, \ldots, pcd_m\}$ tel que $Pcd \subseteq \mathcal{P}$, une condition $cd(X)$ peut être exprimée par $pcd_1(X, v_1) \wedge \ldots \wedge pcd_m(X, v_m)$. Comme nous l'avons montré dans la section précédente, OWL2 permet de déclarer une clé pour une description de classe quelconque CE et donc pour une classe $c \sqcap cd$. Pour exprimer les conditions sur les valeurs des propriétés, les constructeurs *owl :DataHasValue*, que l'on notera $dhv(p, valeur)$, ou *owl :ObjectHasValue* devront être utilisés [2]. La sémantique d'une clé conditionnelle $((c \sqcap cd)\ (p_1, \ldots, p_n))$ (telle que l'ensemble $\{p_1, \ldots, p_n\}$ est disjoint de Pcd) peut alors être définie comme :

$$\forall X, \forall Y, \forall Z_1, \ldots, Z_n, \wedge c(X) \wedge c(Y) \wedge cd(X) \wedge cd(Y) \bigwedge_{i=1}^{n} (p_i(X, Z_i) \wedge p_i(Y, Z_i)) \Rightarrow X = Y$$

Pour une source de données RDF, nous considérons qu'une clé conditionnelle $((c \sqcap cd)\ (p_1, \ldots, p_n))$ est valide si la propriété suivante est vérifiée :

$$\forall X\ \forall Y\ ((X \neq Y) \wedge c(X) \wedge c(Y) \wedge cd(X) \wedge cd(Y)) \Rightarrow \nexists Z\ \nexists j\ (p_j(X, Z) \wedge p_j(X, Z)))$$

Seules les clés conditionnelles minimales sont intéressantes. Une clé $k1$ est plus générale qu'une clé $k2$ (noté $k1 \geq k2$) si la classe $c_1 \sqcap cd_1$ sur laquelle porte $k1$ subsume la classe $c_2 \sqcap cd_2$ de $k2$ et si l'ensemble des propriétés impliquées dans $k1$ est un sous-ensemble de l'ensemble des propriétés impliquées dans $k2$ (y compris les propriétés de la condition) :

$((c_1 \sqcap cd_1)(p_{11}, \ldots, p_{1n})) \geq ((c_2 \sqcap cd_2)(p_{21}, \ldots, p_{2m}))$

ssi $(c_2 \sqcap cd_2) \sqsubseteq (c_1 \sqcap cd_1)$ et $Pcd_1 \cup \{p_{11}, \ldots, p_{1n}\} \subseteq Pcd_2 \cup \{p_{21}, \ldots, p_{2m}\}$

En particulier, une clé définie pour une classe c de l'ontologie (Pcd = $\emptyset$) sera plus générale qu'une clé conditionnelle définie à partir de cette classe et pour le même ensemble de propriétés. On aura par exemple :

$(Researcher(lastName, firstName)) \geq ((Researcher \wedge dhv(worksIn, \textbf{\textit{LRI}})(lastName, firstName))$,

$(Person(lastName, worksIn)) \geq ((Researcher \wedge dhv(worksIn, \textbf{\textit{LRI}})(lastName))$

Une clé conditionnelle ki est minimale pour un jeu de données ssi $\nexists\ kj \neq ki$ tel que $kj \geq ki$.

Envisager de construire pour chaque classe de l'ontologie toutes les conditions, i.e., tous les ensembles de propriétés instanciées par leur ensemble de valeurs possibles, serait très peu efficace. De plus, de nombreuses clés conditionnelles seraient peu pertinentes car elles ne caractériseraient qu'un sous-ensemble de données de très petite taille. Une première approche consiste

2. voir http ://www.w3.org/TR/owl2-syntax/#Class_Expressions pour plus de détails.

à utiliser SAKey pour extraire les clés puis d'utiliser les non-clés maximales découvertes par SAKey pour élaguer l'espace de recherche des non clés conditionnelles. En effet, les propriétés impliquées dans une clé conditionnelle minimale sont incluses dans une non clé. Ensuite, soit :

- Un expert choisit un ensemble des propriétés (non clé) qui lui semblent pertinentes pour représenter des classes qui ne sont pas formalisées dans l'ontologie mais qui caractérisent des instances ayant potentiellement des propriétés instanciées différemment (exemples : regrouper les chercheurs par type de poste, regrouper les villes par régions de France etc.).
- Un pré-traitement sélectionne les propriétés des conditions et les constantes en fonction de la liste des non-clés et du support des classes restreintes par ces conditions candidates (i.e., en fonction du nombre d'instances des nouvelles classes).

Un pré-traitement sélectionne alors les descriptions RDF des instances qui contiennent les valeurs spécifiées dans les conditions pour ces propriétés. Les clés peuvent alors être recherchées en utilisant SAKey sur ce sous-ensemble d'instances. Les clés conditionnelles minimales peuvent ensuite être déduites à partir des non clés conditionnelles maximales en suivant un processus de dérivation similaire à celui introduit dans SAKey.

4 Expérimentations

Nous avons évalué notre approche sur un ensemble de données RDF fourni par l'Institut National de l'Audiovisuel (INA). Il s'agit de descriptions RDF de contenus audiovisuels (classe $Contenu$) et de personnes impliquées dans ces contenus. Les 44 779 instances de la classe $Contenu$ sont décrites par 82 propriétés. Par exemple, la propriété $ina\colon aPourTitreCollection$ représente le titre d'une émission, $ina\colon aPourTitrePropreIntegrale$ représente le titre d'un épisode particulier de l'émission. La propriété $ina\colon aPourGenre$ permet de représenter la catégorie d'un contenu audio-visuel (débat, sketch, série, ...). Sans exceptions, SAKey ne peut découvrir de clés dans ce corpus. Cela est dû à la présence de contenus ayant exactement les mêmes valeurs pour les 10 propriétés les plus instanciées que nous supposons être les plus pertinentes (fréquence>30%). Nous avons exécuté C-SAKey sur ces données en choisissant d'exploiter la propriété $ina\colon aPourGenre$ pour laquelle il existe 42 catégories de contenus : les conditions sont donc de la forme $ina\colon aPourGenre$=*"Sketch"*, $ina\colon aPourGenre$=*"Reportage"*, etc. Les clés découvertes sont présentées dans le tableau 1. Nous présentons au plus deux exemples de catégories pour chaque ensemble de clés trouvé. Par exemple, quand la condition $aPourGenre$=*"Sketch"* ou $aPourGenre$=*"Magazine"* est exploitée, la clé conditionnelle $\{ina\colon TitreCollection, ina\colon Participant, ina\colon TitrePropreIntegrale, ina\colon DateDiffusion\}$ est découverte.

Les résultats montrent que des clés conditionnelles peuvent être découvertes pour chacune des 42 catégories excepté pour la catégorie *Débat* qui contient les duplicats. De plus, nous pouvons observer que les clés conditionnelles peuvent varier en fonction de la catégorie même si certaines catégories partagent les mêmes clés. Une analyse qualitative des clés est bien sûr nécessaire. Pour cela, une évaluation par des experts de l'INA ainsi que l'utilisation des clés pour générer des liens d'identitéentre contenus pourraient être réalisées.

5 Conclusion

Nous avons présenté l'approche C-SAKey qui permet d'étendre une approche de découverte de clés pour découvrir des clés conditionnelles minimales, valides pour un sous-ensemble d'ins-

Condition/Valeur (ina: $aPourGenre$=...)	Clés conditionnelles découvertes
$Sketch$ $Magazine\ldots$	$\{\{ina: TitreCollection, ina: Participant,$ $ina: TitrePropreIntegrale, ina: DateDiffusion\}\}$
$Interview$ $Serie\ldots$	$\{\{ina: TitreCollection, ina: Participant, ina: TitrePropreIntegrale,$ $ina: Duree, ina: DateCreationNotice, ina: DateDiffusion\}\}$
$Chronique$ $Extrait$ $\ldots$	$\{\{ina: TitreCollection, ina: Participant, ina: TitrePropreIntegrale,$ $ina: Duree, ina: DateCreationNotice, ina: DateDiffusion\},$ $\{ina: TitreCollection, ina: Participant, ina: Theme,$ $ina: TitrePropreIntegrale, ina: Duree, ina: DateCreationNotice\}\}$
$Reportage$	$\{\{ina: TitreCollection, ina: Participant, ina: TitrePropreIntegrale,$ $ina: Duree, ina: DateCreationNotice\}\}$

TABLE 1 – Clés conditionnelles pour les 44 779 instances de la classe $Contenu$

tances de classe, sous-ensemble décrit par des conditions sur les valeurs de certaines propriétés. Une première expérimentation a été conduite sur des données réelles pour lesquelles aucune clé n'avait pu être découverte. Les résultats montrent la possibilité de découvrir des clés conditionnelles dans un tel contexte. D'autres expérimentations sont à mener et leurs résultats doivent être évalués.

Références

AL-BAKRI M., ATENCIA M., LALANDE S. & ROUSSET M.-C. (2015). Inferring same-as facts from linked data : an iterative import-by-query approach. In *Proceedings of AAAI 2015, to appear*.

ATENCIA M., DAVID J. & SCHARFFE F. (2012). Keys and pseudo-keys detection for web datasets cleansing and interlinking. In *EKAW*, p. 144–153.

FERRARA A., NIKOLOV A. & SCHARFFE F. (2011). Data linking for the semantic web. *Int. J. Semantic Web Inf. Syst.*, **7**(3), 46–76.

HU W., CHEN J. & QU Y. (2011). A self-training approach for resolving object coreference on the semantic web. In *WWW*, p. 87–96.

NIKOLOV A. & MOTTA E. (2010). Data linking : Capturing and utilising implicit schema-level relations. In *Proceedings of Linked Data on the Web workshop at 19th International World Wide Web Conference(WWW)'2010*.

PERNELLE N., SAÏS F. & SYMEONIDOU D. (2013). An automatic key discovery approach for data linking. *Journal of Web Semantics*, **23**, 16–30.

SAÏS F., PERNELLE N. & ROUSSET M.-C. (2009). Combining a logical and a numerical method for data reconciliation. *Journal on Data Semantics*, **12**, 66–94.

SORU T., MARX E. & NGONGA NGOMO A.-C. (2015). ROCKER – a refinement operator for key discovery. In *Proceedings of the 24th International Conference on World Wide Web, WWW 2015*.

SYMEONIDOU D., ARMANT V., PERNELLE N. & SAÏS F. (2014). Sakey : Scalable almost key discovery in RDF data. In *13th International Semantic Web Conference (ISWC), Italy, 2014. Proceedings, Part I*, volume 8796 of *Lecture Notes in Computer Science*, p. 33–49 : Springer.