Définition de la sémantique des clés dans le Web sémantique : un point de vue théorique

Michel Chein, Madalina Croitoru, Michel Leclère, Nathalie Pernelle, Fatiha Saïs, Danai Symeonidou

Définition de la sémantique des clés dans le Web sémantique : un point de vue théorique

Michel Chein[1], Madalina Croitoru[1], Michel Leclere[1], Nathalie Pernelle[2], Fatiha Saïs[2], Danai Symeonidou[2]

[1] UNIVERSITÉ MONTPELLIER 2 Montpellier, France
chein@lirmm.fr, croitoru@lirmm.fr, leclere@lirmm.fr

[2] UNIVERSITÉ PARIS SUD Paris, France
pernelle@lri.fr, sais@lri.fr, symeonidou@lri.fr

Résumé : De nombreuses approches ont été définies pour permettre le liage automatique de sources de données RDF publiées sur le Web. Certaines de ces approches sont basées sur la sélection des plus petits ensembles de propriétés pertinentes pour comparer deux données. Ces ensembles forment des clés et cette notion est similaire aux clés définies pour les bases de données relationnelles. Dans cet article, nous proposons d'explorer différentes sémantiques de clés qui peuvent être utilisées dans le cadre du Web sémantique.

1 Introduction

De nombreuses approches ont été définies pour permettre le liage automatique de sources de données RDF publiées sur le web (voir [2] pour un état de l'art). La plupart de ces approches exploitent des règles de liage qui spécifient les conditions que doivent remplir les descriptions de deux entités pour que celles-ci soient liées par un lien d'identité. Certaines de ces approches exploitent des sources de données pour apprendre des règles de liage expressives qui comportent des transformations, des mesures de similarité et des fonctions d'agrégation. Ces règles sont spécifiques au vocabulaire utilisé dans les sources de données exploitées. D'autres approches utilisent les axiomes définis dans l'ontologie tels que les clés ou les propriétés (inverse) fonctionnelles, en particulier depuis que OWL 2 permet, via le constructeur *owl :hasKey*, de déclarer qu'un ensemble de propriétés forment une clé pour une classe définie dans une ontologie. Ces clés peuvent être utilisées comme des règles logiques pour inférer des liens d'identité ou pour guider la construction de fonctions de similarité plus complexes pour lesquels des mesures de similarité élémentaires peuvent être choisies par un expert [4, 7, 9]. Certaines méthodes utilisent des clés pour réduire le nombre de paires d'instances à comparer par un outil de liage [5].

Ces clés n'étant pas toujours déclarées dans l'ontologie, certaines approches se sont intéressées à la découverte de clés à partir de données RDF. Le problème de découverte de clés à partir de sources de données RDF est similaire au problème de découverte de clés dans les bases de données relationnelles. Dans les deux cas, il s'agit d'un sous-problème de celui consistant à découvrir des dépendances fonctionnelles (DF). Une DF exprime le fait que la valeur d'un attribut est uniquement déterminée par les valeurs attribuées à un autre ensemble d'attributs. La découverte de clés à partir de données RDF diffère cependant du cadre habituel des bases de données relationnelles car, d'une part, certaines propriétés sont multivaluées et, d'autre part, certaines propriétés ne sont pas renseignées pour certaines données (i.e. valeurs nulles).

Dans [8], les auteurs découvrent des propriétés (inverse) fonctionnelles dans des sources de données où l'hypothèse du nom unique (UNA) est vérifiée. D'autres approches telles que [6, 1] découvrent des clés composées de plusieurs propriétés à partir de données RDF. Cependant, les clés découvertes n'ont pas la même sémantique. En effet, dans [6], deux données ayant au moins une valeur commune pour chaque propriété de la clé sont considérées comme référant à la même entité. Dans [1], les auteurs considèrent que deux données doivent être décrites par les mêmes ensembles de valeurs pour toutes les propriétés de la clé.

Dans cet article nous proposons de formaliser différentes notions de clés utilisables dans des applications de liage de données ou de détection de redondances dans une source de données. Plus précisement, nous définissons les notions d'interprétation et de déduction en logique du premier ordre pour ces deux sémantiques de clés. Nous montrons ensuite que pour définir l'ensemble des clés pouvant être découvertes (clés observées) dans une base de connaissances en terme de satisfiabilité, certains faits d'égalité et de différence doivent être explicités.

Après une étude de l'existant, nous définissons les deux sémantiques de clés. Puis, nous présentons leur utilisation dans un mécanisme de déduction. Nous présentons ensuite, comment nous définissons l'ensemble des clés observées pour ces deux sémantiques.

2 Etude de l'existant

La notion de clé a été introduite dans le modèle relationnel des bases de données. Pour une relation donnée, nous pouvons distinguer son schéma, l'ensemble des attributs, des ensembles de tuples qui forment les instances de la relation. Une clé est un ensemble d'attributs dont les valeurs définissent un seul tuple de la relation. Quand aucune clé n'a été définie, ou quand les clés impliquent un ensemble d'attributs trop grand ou des attributs trop complexes (e.g. une longue valeur textuelle), un nouvel attribut construit automatiquement peut être généré dont l'unicité est garantie. Dans le modèle relationnel, une clé doit être minimale i.e aucun sous-ensemble non vide de la clé ne doit être une clé.

Il est possible de représenter les instances d'un modèle relationnel en logique du premier ordre (FOL) en ordonnant les attributs de chaque relation et en traduisant chaque tuple en un atome ayant pour nom le nom de la relation et pour termes les valeurs des attributs dans l'ordre choisi. La conjonction des atomes obtenus forme alors la description de l'ensemble des instances de la relation. Cependant le contexte du web sémantique (WS) est différent du contexte des bases de données relationnelles. Tout d'abord, en base de données, un ensemble d'instances est représenté en intension par le schéma de la relation et en extension par l'ensemble des tuples. Dans le cadre du WS, un ensemble d'entités est représenté en intension par une classe (ou une expression de classe) et les instances représentées par cette classe forment seulement une partie de cette extension.

Deuxièmement, dans le cadre du WS, certaines données ne sont pas conformes à un schéma : rien n'interdit d'utiliser n'importe quel vocabulaire pour décrire les données. Il est cependant possible de déclarer le vocabulaire utilisé dans une ontologie.

Troisièmement, le WS ne fait pas l'hypothèse du monde fermé (Closed Word Assumption) qui pose que toute connaissance non démontrée est fausse. Aussi, si une instance de classe i n'est pas associée à une valeur v par la propriété p, cela n'entraîne pas que cette connaissance est fausse.

Enfin, le WS ne fait pas l'hypothèse du nom unique (UNA) qui exprime le fait que deux constantes sont sémantiquement égales si et seulement si elles sont syntaxiquement égales. Ainsi, deux instances syntaxiquement distinctes peuvent être déclarées égales ou différentes.

2.1 Définitions de clés dans le Web sémantique

Il existe dans le Web sémantique deux notions de clés : les axiomes OWL *owl :InverseFunctionalProperty* et *owl :hasKey* (*owl :InverseFunctionalProperty* existe dans la première version de OWL et `owl:hasKey` a été introduit dans OWL 2). L'axiome OWL *owl :InverseFunctionalProperty* permet de déclarer qu'une propriété est clé dans une base de connaissances RDF. Si cet axiome est déclaré pour une propriété p, alors sa sémantique logique pourrait être traduite par la règle en logique des prédicats suivante :

$$\forall x \forall y \forall z (p(x,z) \wedge p(y,z) \rightarrow x = y)$$

Dans le web sémantique, la déclaration d'un ensemble de propriétés $\{P_1, ...P_n\}$ comme étant clé pour une classe d'expression C donnée, peut être exprimée par la règle logique suivante :

$$\textbf{R1} : \forall x \forall y \forall z_1 ... z_n (C(x) \wedge C(y) \wedge \bigwedge_{i=1}^{n} (P_i(x,z_i) \wedge P_i(y,z_i)) \rightarrow x = y)$$

Cette définition est cohérente avec la sémantique de *owl :hasKey* dans OWL 2 fondée sur RDF [1]. Néanmoins, cette définition n'est cohérente, ni avec la présentation formelle de l'axiome *owl :hasKey* [2], ni avec la sémantique directe de *owl :hasKey* [3]. Dans cette définition une condition supplémentaire contraint les individus considérés à être nommés (*i.e.* ils doivent être des URIs ou des littéraux, mais ne peuvent pas être des noeuds blancs). Considérons un prédicat unaire $Const$ qui vaut $vrai$ si son argument est une constante sinon il vaut $faux$. Cette seconde sémantique de *owl :hasKey* peut être exprimée par la règle suivante :

$$\begin{aligned} \textbf{R2} : \forall x \forall y \forall z_1 ... z_n (C(x) \wedge C(y) \wedge Const(x) \wedge Const(y) \wedge \\ \bigwedge_{i=1}^{n} (P_i(x,z_i) \wedge P_i(y,z_i) \wedge Const(z_i)) \rightarrow x = y) \end{aligned}$$

Si on généralise cette notion, en gardant la cohérence avec OWL, on obtiendrait ce qui suit :

$$\begin{aligned} \textbf{R3} : \forall x \forall y \forall z_1 ... z_n (C[x] \wedge C[y] \wedge \bigwedge_{i=1}^{n} (P_i(x,z_i) \wedge P_i(y,z_i)) \\ \rightarrow x = y) \end{aligned}$$

où $C[.]$ est une formule avec exactement une variable libre qui représente une classe définie concernée par la clé.

1. Voir section 5.14 de http ://www.w3.org/TR/owl2-rdf-based-semantics/
2. Voir section 9.5 de http ://www.w3.org/TR/owl2-syntax/
3. Voir section 2.3.5 de http ://www.w3.org/TR/owl2-direct-semantics/

2.2 Validité d'une clé dans une base de connaissances

Si un ensemble de propriétés $K = \{p_1, \ldots, p_n\}$ est déclaré clé pour une classe C, alors ne pas respecter la clé K dans une source de données peut être dû à des erreurs ou à la présence de données dupliquées. Nous nommons abusivement "exceptions" les paires d'instances conduisant à des incohérences lorsque une clé K est déclarée. En se basant sur la sémantique des clés définie dans OWL2, une exception peut être vue comme une paire d'instances distinctes x et y de la classe C pour lesquelles il existe un ensemble d'instances ou de valeurs litérales $o_1, \ldots, o_n$ tel que x et y sont tous deux associés à o_i par la propriété p_i ($i = 1, \ldots, n$). On peut noter qu'il n'est pas nécessaire que x et y coïncide pour toutes les valeurs de p_i (excepté si p_i est fonctionnelle).

Cela implique, ce qui correspond à l'esprit du Web sémantique, que même si de nouvelles informations sont apprises sur x et y, il s'agira toujours d'exceptions pour K.

Néanmoins, lorsque les clés sont utilisées pour nettoyer ou lier des données, cela peut également être utile de considérer des clés pour lesquelles x et y seront des exceptions à K si toutes les valeurs de chacune des propriétés $p_i \in K$ coincident i.e s'il existe o_i tel que x est lié à o_i par p_i alors y doit aussi être lié à o_i par p_i, et vice versa. Avec une telle notion de clé, x et y peuvent cesser d'être des exceptions à K si de nouvelles connaissances sur x et y sont apprises.

2.3 Exemple Illustratif

La Figure 1 représente un exemple de graphe RDF décrivant quatre instances de la classe $Person$. La traduction de cet exemple en FOL est la suivante :

$F = Person(i_1) \wedge name(i_1, \textit{'Qié'}) \wedge firstName(i_1, \textit{'Wong'}) \wedge lab(i_1, \textit{'LRI'}) \wedge mail(i_1, \textit{'qie@ls.fr'})$
$\wedge Person(i_2) \wedge name(i_2, \textit{'Qie'}) \wedge firstName(i_2, \textit{'Tim'}) \wedge mail(i_2, \textit{'t.qie@orange.fr'})$
$\wedge lab(i_2, \textit{'LRI'}) \wedge Person(i_3) \wedge name(i_3, \textit{'Sting'}) \wedge mail(i_3, \textit{'sting@thales.fr'}) \wedge lab(i_3, \textit{'LIMSI'})$
$\wedge Person(i_4) \wedge name(i_4, \textit{'Sting'}) \wedge name(i_4, \textit{'Mills'}) \wedge lab(i_4, \textit{'LIMSI'}) \wedge mail(i_4, \textit{'st.mills@ls.fr'})$

Nous allons illustrer les différences entre deux notions de clés ainsi que les hypothèses qui peuvent être considérées lors de la découverte des clés ayant ces deux sémantiques.

Nous sommes intéressés par la dècouverte des clés pour une classe dans une base de connaissances OWL2 représentée en Logique du premier ordre (FOL).

Si l'on adopte le point de vu habituel des bases de données, on peut conclure que la propriété $\{mail\}$ est une clé observée mais que $\{lab\}$ n'est pas une clé observée mais nous ne savons pas comment considérer les deux autres propriétés $\{firstName\}$ et $\{Name\}$ à cause de la multivaluation ou de l'absence de valeurs. Une clef observée peut être vue comme un ensemble de prédicats tel que les valeurs de ces prédicats sont suffisantes pour identifier de manière unique les instances apparaissant dans la base de connaissances. Cette observation se base sur certaines hypothèses implicites. Pour conclure que $\{lab\}$ n'est pas une clef observée, nous faisons l'hypothèse implicite que i_1 et i_2 sont differents. De même, pour conclure que $\{mail\}$ est une clé, nous faisons l'hypothèse implicite que *'qie@ls.fr'* et *'t.qie@orange.fr'* sont différents. En fait, pour découvrir des clés en exploitant un fait, nous avons théoriquement besoin de détenir une connaissance complète sur les informations d'égalité ou de différence entre les termes apparais-

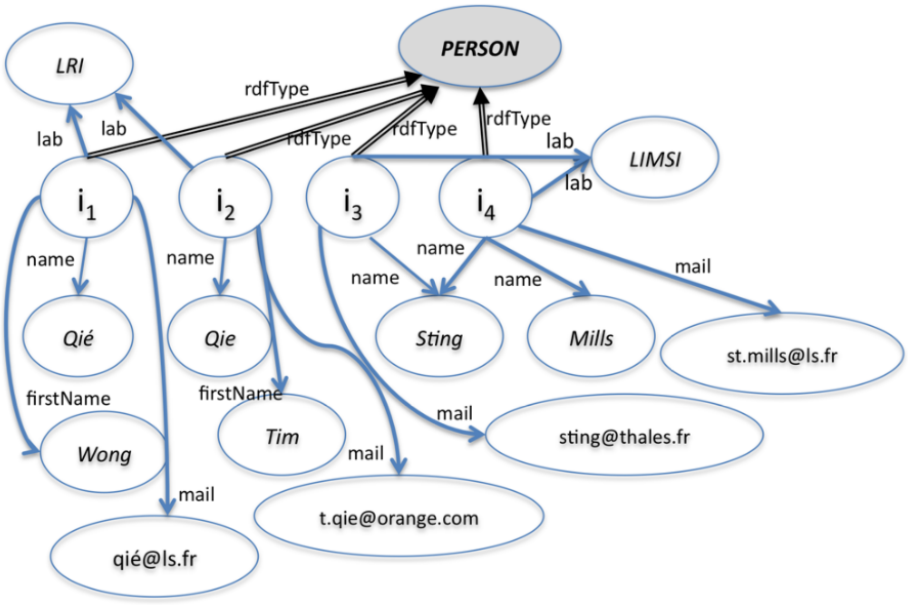

FIGURE 1 – La base de connaissances (KB_1).

sant dans le fait que ce soit pour les instances de classes ou les litéraux. Si ces informations ne sont pas connues, une heuristique simple consiste à choisir la relation d'égalité syntaxique et de ce fait à faire l'hypothèse du nom unique (UNA).

Dans certains domaines d'études, il est raisonnable de penser que pour certaines propriétés, les informations décrites pour une instance sont complètes. Cette connaissance peut découler de contraintes définies sur la propriété comme les contraintes de cardinalité ou de connaissances sur la source de données. Ainsi, on peut avoir déclaré dans l'ontologie qu'à une personne ne peut être associé qu'un laboratoire de recherche. De même, on peut savoir que dans la source de données DBLP les auteurs d'une publication sont décrits de manière exhaustive. Quand cette complétude locale est connue (i.e ce que l'on peut nommer "propriété fermée"), les clés peuvent être adaptées pour en tirer bénéfice. C'est cette intuition que nous formaliserons dans les sections suivantes sous le terme de F-règle.

Enfin, dans la sémantique que OWL2 a défini pour les clés, le prédicat "sameAs" est logiquement interprété comme une égalité. Cela n'est pas toujours pertinent sur le web de données où cette interprétation peut conduire à de nombreuses inconsistances [3]. Ainsi si l'individu "Tim Qié" de KB1, dont l'emploi principal (unique) est au laboratoire LRI, est déclaré identique à l'individu "Tim Qié" d'une autre base de connaissance KB2 dans lequel il est déclaré comme travaillant pour le LIP6, on risque en appliquant la sémantique du sameAs et l'axiome de fonctionnalité déclaré pour la propriété $\{lab\}$ d'inférer que ces deux laboratoires sont identiques. Remplacer l'égalité par une relation de similarité permet de définir une notion de clé applicable à de réels usages possibles sur le Web.

Dans la suite de cet article, nous explorons théoriquement ces différentes notions et comparons les deux notions possibles de clé.

3 Définitions de Clés

3.1 Deux définitions généralisées fondées sur les règles

L'utilisation d'une relation de similarité permet de généraliser la notion d'égalité. En effet, l'utilisation de la similarité est plus adaptée aux usages réels du Web où les similarités sont utilisées au lieu de "l'égalité logique pure".

Prenons $Sim_1, ..., Sim_n$ des relations de similarité utilisées pour comparer les valeurs de propriétés. L'introduction des relations de similarité conduit à une extension simple de la règle $R3$ (c.f. section 2.1)

Definition 1 (S-règle)
La règle de similarité ***"Some"*** *(ou* ***S-règle****) pour une classe C et pour les relations de similarité $Sim_{S_1}, \ldots, Sim_{S_n}$ est la règle définie comme suit :*

$$\forall x \forall y (C[x] \wedge C[y] \wedge$$
$$\bigwedge_{i=1}^{n} \exists z_i \exists w_i (p_i(x, z_i) \wedge p_i(y, w_i) \wedge Sim_{S_i}(z_i, w_i)) \rightarrow Sim_C(x, y))$$

Sim_{S_i} représente les relations de similarité utilisées dans la S-règle pour comparer z_i et w_i, i.e. les valeurs de la propriété p_i (e.g. mesures de similarité entre chaînes de caractères comme Levenshtein ou Jaro-Winkler). Une S-règle pour une classe C avec les relations $Sim_1, ..., Sim_n$ est notée S[(C, Sim_C), (p_1, Sim_{S_1}), ..., (p_n, Sim_{S_n})] ou, lorsqu'il n'y a pas de confusion possible, (C, Sim_C), (p_1, Sim_{S_1}), ..., (p_n, Sim_{S_n}).

Example 1
Considérons KB_1 (Figure 1). Supposons que $(Person, Sim_{Person}), (name, Sim_{S_1})$ est une S-règle et que nous avons Sim_{S_1}('Qié', 'Qie'). A partir de ces connaissances, nous pouvons inférer les similarités : $\{Sim_{Person}(i_1, i_2), Sim_{Person}(i_3, i_4)\}$

Une S-règle S est une extension d'une clé OWL dans le sens où : si tous les prédicats de similarité dans S, Sim_{S_i} et Sim_C, représentent l'égalité alors S et R_3 sont des formules équivalentes.

Proposition 1
La S-règle $S[(C, =)$; $(p_1, =)$, ..., $(p_n, =)]$ est logiquement équivalente à R_3, i.e. $R_3 \leftrightarrow S$ est valide.

Notons que pour prouver la propriété précédente, il est nécessaire de considérer la propriété de substitution pour l'égalité logique, ce qui n'est pas toujours pertinent pour `owl:sameAs` et n'est pas supposé dans cet article pour les relations de similarité.

Un deuxième type de régle de similarité peut être défini pour exprimer la seconde sémantique de clés fondée sur la comparaison des ensembles de valeurs de propriétés.

Definition 2 (F-règle)
La règle de similarité ***"Forall"*** *(ou* ***F-règle****) pour une classe C et pour les relations de similarité $Sim_{F_1}, ..., Sim_{F_n}$ est définie comme suit :*
$\forall x \forall y (C[x] \wedge C[y] \wedge \bigwedge_{i=1}^{n} (\forall z_i \exists w_i (p_i(y, z_i) \rightarrow p_i(x, w_i) \wedge Sim_{F_i}(z_i, w_i)) \wedge$
$(\forall u_i \exists v_i (p_i(x, u_i) \rightarrow p_i(y, v_i) \wedge Sim_{F_i}(u_i, v_i))) \rightarrow Sim_C(x, y)))$

Example 2
Considérons KB_1. Supposons maintenant que $(Person, Sim_{Person}), (name, Sim_{F_1})$ est une F-règle et que nous avons Sim_{F_1}('Qié', 'Qie'). A partir de ces connaissances, nous inferons la similarité : $\{Sim_{Person}(i_1, i_2)\}$

Une F-règle pour une classe C et pour les relations $Sim_{F_1}, ..., Sim_{F_n}$ est notée $F[(C, Sim_C), (p_1, Sim_{F_1}), ..., (p_n, Sim_{F_n})]$ ou simplement $(C, Sim_C), (p_1, Sim_{F_1}), ..., (p_n, Sim_{F_n})$ lorsqu'il ne peut pas y avoir de confusion.

La proposition suivante déclare que les S-règles et les F-règles sont monotones par rapport à la relation d'inclusion de propriétés.

Proposition 2
Soit R une S-règle (resp. F-règle) $[(C, Sim_C), (p_1, Sim_{R_1}), ..., (p_n, Sim_{R_n})]$ et R' une autre S-règle (resp. F-règle) $[(p_1, Sim_{R_1}), ..., (p_n, Sim_{R_n}), (p_{n+1}, Sim_{R_{n+1}}), ...]$. Alors : $R \models R'$, i.e. R' est logiquement déduite de R.

Les relations entre les S-règles et les F-règles sont étudieés dans ce qui suit.

3.2 Les modèles des S-règles et des F-règles

Dans cette section, nous présentons les propriétés des interprétations (i.e. modèles) satisfaisant les S-règles et les F-règles.

Nous montrons également que leur définition logique représente la sémantique en logique du premier ordre des S-règles et des F-règles apprises par des algorithmes existants (Propositions 3 et 4).

Des algorithmes existent dans la littérature qui permettent de découvrir des clés en s'appuyant sur la comparaison d'ensembles de valeurs. Il existe deux approches : la première [1] considère l'égalité entre ensembles de valeurs de propriétés et la seconde [6] consiste à vérifier que l'intersection entre les ensembles de valeurs de propriétés est non vide.

Definition 3
Soit $\mathcal{C}$ un ensemble, P une relation binaire sur $\mathcal{C}$, $\mathcal{S}$ une relation de similarité reflexive et symétrique définie sur le co-domaine de P, et c, c' appartenant à $\mathcal{C}$.

- *$P(c)$ indique l'ensemble d'éléments dans $\mathcal{C}$ liés à c par P, i.e. $P(c) = \{u \mid (c, u) \in P\}$;*
- *$P(c) \subseteq_s P(c')$ exprime que pour tout élément u dans $P(c)$ il existe un élément v dans $P(c')$ tel que u et v sont similaires par rapport à $\mathcal{S}$, i.e. $(u, v) \in \mathcal{S}$;*
- *$P(c) =_s P(c')$ exprime que $P(c) \subseteq_s P(c')$ et $P(c') \subseteq_s P(c)$*
- *$P(c) \cap_s P(c')$ est égale à l'ensemble $\{u \in P(c) | \exists v \in P(c')$ tel que $(u, v) \in \mathcal{S}\}$*

Example 3
Dans KB_1, nous avons par exemple, pour la relation binaire $name$ et la relation de similarité simple S (comparaison en ignorant les accents), $name(i_1) = \{$Qié$\}$, $name(i_3) \subseteq_s name(i_4), name(i_1) =_s name(i_2), name(i_2) \cap_s name(i_4) = \emptyset$.

Notons que $\cap_s$ n'est pas commutatif, néanmoins $P(c) \cap_s P(c') \neq \emptyset$ ssi $P(c') \cap_s P(c) \neq \emptyset$. Remarquons aussi que si S est la relation d'égalité alors $\subseteq_s$ est l'inclusion d'ensembles, $\cap_s$ est l'intersection d'ensembles et $=_s$ est l'égalité entre ensembles.

Definition 4
Une interprétation I en logique des prédicats dans le domaine Δ^I des prédicats apparaissant dans une S-règle, ou une F-règle, est exprimée par :
- $C^I \subseteq \Delta^I$ *l'interprétation de la classe* C *;*
- $p_i^I \subseteq \Delta^I \times \Delta^I$ *l'interprétation des predicats* p_i *;*
- $Sim_i^I \subseteq \Delta^I \times \Delta^I$ *l'interprétation des predicats* Sim_i *;*
- $Sim_C^I \subseteq \Delta^I \times \Delta^I$ *l'interépretation du predicat* Sim_C*.*

Proposition 3
Une interprétation I est un modèle pour une S-règle (C, Sim_C), (p_1, Sim_{S_1}), ..., (p_n, Sim_{S_n}) ssi quels que soient c et c' appartenant à C^I et tels que pour tout $i = 1, \ldots, k$, $p_i^I(c) \cap_S p_i^I(c') \neq \emptyset$, on a $(c, c') \in Sim_C^I$.

Notons que lorsqu'une propriété p_i^I est une fonction totale et lorsque la relation de similarité est la relation d'égalité alors une S-règle minimale (par rapport à l'inclusion de propriétés) est une clé dans le cadre des bases de données relationnelles.

Proposition 4
Une interprétation I est un modèle de F-règle (C, Sim_C), (p_1, Sim_{F_1}), ..., (p_n, Sim_{F_n}) ssi quels que soient c et c' appartenant à C^I et tels que pour tout $i = 1, \ldots, k$, $p_i^I(c) =_S p_i^I(c')$, on a $(c, c') \in Sim_C^I$.

Les relations entre les S-règles et les F-règles sont données dans la Proposition 5. La première propriété indique que la notion de S-règle est plus restrictive que la notion de F-règle, lorsque l'on considère des interpretations dans lesquelles pour tout p_i^I il existe au plus un élement de C^I n'ayant pas de valeur. Dans les bases de données relationnelles ce cas correspond au cas où il existe au plus une valeur nulle.

La deuxième propriété exprime le cas contraire lorsque l'on considère des interpretations dans lesquelles toutes les propriétés p_i^I sont fonctionnelles. Dans les bases de données relationnelles ce cas correspond au cas où toutes les propriétés sont mono-valuées.

Enfin, la dernière propriété exprime le fait que ces deux notions sont équivalentes lorsque l'interpretation de toute propriété p_i, i.e. p_i^I, est une fonction totale. En bases de données relationnelles, ce cas correspond au cas où toutes les propriétés sont renseignées et mono-valuées.

Proposition 5
Soient S-R et F-R une S-règle et une F-règle (respectivement) associées à (C, Sim_C), (p_1, Sim_{S_1}), ..., (p_n, Sim_{S_n}) .
- *Pour toute interprétation I telle que pour tout $i = 1, \ldots, n$ il existe au plus un élément $c \in C^I$ avec $p_i^I(c) = \emptyset$, on a : si I est un modèle de S-R alors I est un modèle de F-R (i.e. si $I \models S$-R alors $I \models F$-R).*
- *Pour toute interprétation I telle que pour tout $c \in C^I$ et pour toute propriété p_i on a $card(p_i^I(c)) \leq 1$, on a : si I est un modèle de F-R alors I est un modèle de S-R (i.e. si $I \models F$-R alors $I \models S$-R).*
- *Pour toute interprétation I telle que pour tout élément $c \in C^I$ et pour toute propriété p_i on a : $card(p_i^I(c)) = 1$, alors I est un modèle de S-R ssi I est un modèle de F-R (i.e. $I \models F$-R ssi $I \models S$-R).*

4 Application des S-règles et des F-règles pour la déduction de similarités

Les F-règles et des S-règles peuvent être utilisées pour inférer des similarités dans le cadre d'applications de liage de données et de nettoyage de données.

Nous présentons dans cette section l'impact sur les similarités déduites du choix de la sémantique des clés, c'est-à-dire, en utilisant les S-règles ou les F-règles.

Dans ce qui suit, on considère un *vocabulaire* V de la logique des prédicats sans symboles de fonctions et contenant au moins un prédicat unaire C, un prédicat binaire Sim_C, pour tout $i = 1, \ldots, k$ un prédicat binaire p_i, un ensemble de prédicats de similarité Sim_1,..., Sim_n réflexifs et symétriques, un ensemble de constantes et un ensemble de variables.

Definition 5 (Fait et Fait simple)

- *Un* fait relatif à une classe C *sur un vocabulaire V est la fermeture existentielle d'une conjonction d'atomes positifs construits sur V telle que les termes apparaissant dans un atome Sim_C apparaissent aussi dans les atomes de C et les termes apparaissant dans un atome Sim_{p_i} apparaissent en seconde position d'un atome p_i.*
- *Un* fait simple *est un fait sans atome Sim_C.*

Example 4
Soit F' le fait relatif à la classe C sur un vocabulaire V donné :

$F' = C(i_1) \wedge p(i_1, d) \wedge q(i_2, e) \wedge r(i_1, a) \wedge s(i_1, d) \wedge t(i_1, a) \wedge t(i_1, d) \wedge C(i_2) \wedge p(i_2, f) \wedge r(i_2, a) \wedge q(i_3, f) \wedge t(i_2, a) \wedge C(i_3) \wedge p(i_3, b) \wedge r(i_3, d) \wedge s(i_3, d) \wedge t(i_3, b) \wedge t(i_3, f) \wedge C(i_4) \wedge p(i_4, d) \wedge p(i_4, g) \wedge C(i_5) \wedge p(i_5, b) \wedge C(i_6) \wedge p(i_6, h) \wedge \exists x\ p(i_7, x) \wedge Sim_Sp(f, h)$

Definition 6 (Saturation par des S-règles)
Soit $\mathcal{F}$ un fait et S une S-règle, $R(\mathcal{F})$ est le fait maximum tel que : $\mathcal{F}, S \models R(\mathcal{F})$.

Example 5
La saturation de F' en utilisant la S-règle $(C, Sim_C), (p, SimSp)$ permet d'obtenir : $R(F') = F' \wedge Sim_C(i_1, i_4) \wedge Sim_C(i_3, i_5) \wedge Sim_C(i_2, i_6)$.

L'utilisation des F-règles pour l'inférence nécessite l'enrichissement du fait $\mathcal{F}$ avec des formules exprimant la fermeture de certains prédicats (cf. [10]).

Definition 7 (Fermeture)
Soit t une instance d'une classe C et p un prédicat binaire. $Closed(t, p)$ est représenté par la formule ci-dessous qui exprime la fermeture de $\mathcal{F}$ sur p relativement à t :

$Closed(t, p) = \forall x(p(t, x) \rightarrow (x = t_1) \vee \ldots \vee (x = t_n) \vee \perp)$ avec $t_1, \ldots, t_n$ sont des termes apparaissant en seconde position de l'atome $p(t, -)$ dans $\mathcal{F}$.

$Closure(\mathcal{F}, \{p_1, \ldots, p_n\})$ est la fermeture existentielle de la conjonction d'atomes de $\mathcal{F}$ et des formules de $Closed(t, p_i)$ construite pour chaque terme t apparaissant dans un atome C et dans chaque atome p_i.

Example 6
$Closed(i_1, t) = \forall x(t(i_1, x) \rightarrow (x = d) \vee (x = a) \vee \perp)$

Definition 8 (Saturation par des F-règles)
Soit $\mathcal{F}$ un fait et F une F-règle (C, Sim_C), (p_1, Sim_{F_1}), ..., (p_n, Sim_{F_n}) , $F(\mathcal{F})$ est le fait maximal tel que :

$$Closure(\mathcal{F}, \{p_1, ..., p_n\}), F \models F(\mathcal{F})$$

Example 7
Soit r_1 une F-régle $(C, Sim_C), (p, SimSp)$. $r_1(F') = F' \wedge Sim_C(i_3, i_5) \wedge Sim_C(i_2, i_6) \wedge (Sim_C(i_7, i_1) \vee Sim_C(i_7, i_2) \vee Sim_C(i_7, i_3) \vee Sim_C(i_7, i_5) \vee Sim_C(i_7, i_6))$

Definition 9 (Deduction de similarities par une règle de similarité)
Soit $\mathcal{F}$ un fait et R une S-règle ou une F-règle. Les similarités déduites *à partir de $\mathcal{F}$ par R, notées $Sim(R, \mathcal{F})$, correspondent à l'ensemble des atomes de $R(\mathcal{F})$ n'appartenant pas à $\mathcal{F}$, i.e. $Sim(R, \mathcal{F}) = R(\mathcal{F}) \setminus \mathcal{F}$.*

Example 8
$Sim(r_1, F') = Sim_C(i_3, i_5) \wedge Sim_C(i_2, i_6)$

Notons que $Sim(R, \mathcal{F})$ produit par une règle R est réflexive et symétrique (*i.e.* Sim_C est réflexive et symétrique sur l'ensemble des termes dans $Sim(R, \mathcal{F})$). De plus, cette relation est transitive pour les F-règles puisque $=_s$ est transitive. Enfin, remarquons que la Proposition 2 conduit à ce qui suit.

Proposition 6
Soit F un fait, $R = (C, Sim_C), (p_1, Sim_{R_1}), ..., (p_n, Sim_{R_n})$.
Soit $R' = (C, Sim_C), (p_1, Sim_{R_1}), ..., (p_n, Sim_{R_n}), (p_{n+1}, Sim_{R_{n+1}})$ une S-règle (ou une F-règle). Alors, $Sim(R', \mathcal{F}) \subseteq Sim(R, \mathcal{F})$.

4.1 Découverte de clés

Definition 10 (Une clé pour un fait)
Soit $\mathcal{F}$ un fait et R une S-règle ou une F-règle : $(C, Sim_C), (p_1, Sim_{R_1}), ..., (p_n, Sim_{R_n})$. La règle R est une S ou F-clé pour C dans $\mathcal{F}$ *si $Sim(R, \mathcal{F}) \subseteq \mathcal{F}$, i.e. toute similarité déduite en appliquant R appartient au fait $\mathcal{F}$.*

Une autre distinction peut également être introduite. Selon que le fait $\mathcal{F}$ contienne ou pas des propriétés non instantiées, deux choix d'interprétation de l'absence de valeurs de propriétés peuvent être considérés : (i) considérer un modèle du contenu avec information minimale (c'est-à-dire, aucune nouvelle information ne peut être ajoutée si les valeurs sont manquantes) ou (ii) un modèle du contenu avec information non minimale (c'est-à-dire, de nouvelles informations peuvent potentiellement être ajoutées si les valeurs sont renseignées). Par abus de langage, nous notons $p_i(c) \notin \mathcal{F}$ le cas où l'instance c n'a pas de valeurs pour la propriété p_i dans le fait $\mathcal{F}$. Nous obtenons alors :

Definition 11 (Clé pour un fait avec contenu à information minimale)
Soient $\mathcal{F}$ un fait et R une S ou une F-règle : $(C, Sim_C), (p_1, Sim_{R_1}), ..., (p_n, Sim_{R_n})$. Nous Notons $\mathcal{F}'_R$ l'ensemble obtenu par la suppression de toutes les instances non complètement instantiées : $\mathcal{F}_r = \mathcal{F} \setminus \{c | \exists p_i$ tel que $p_i(c) \notin \mathcal{F}\}$. R est une S-clé ou une F-clé pour un fait avec un contenu à information minimale C dans $\mathcal{F}$ si elle est une S-clé ou une F-clé pour C dans $\mathcal{F}'_R$.

Notation : Dans ce qui suit, nous nommons ADS-clés, les F-clés pour un fait avec contenu à information minimale (puisqu'elles ont été introduites ainsi dans [1], mais d'un point de vue fonctionnel uniquement).

Supposons que $\mathcal{F}$ est complet par rapport à Sim_C, i.e. quels que soient $C(t)$ et $C(t')$ dans $\mathcal{F}$ et $Sim_C(t,t') \notin \mathcal{F}$ est interprété par le fait que t et t' ne sont pas similaires. Alors, une clé ne doit pas conduire à la déduction d'un atome $Sim_C(t,t') \notin \mathcal{F}$. Nous formalisons ceci dans ce qui suit.

Definition 12 (Fait étendu)
Un fait étendu *est composé d'un fait contenant des atomes Sim_C pouvant être niés.*

Definition 13 (Completion d'un fait)
Un fait étendu $\mathcal{F}$ relatif à une classe C est complet *si pour tout couple de termes (t,t'), non nécessairement différents, qui sont des instances de C, il contient soit $Sim_C(t,t')$ ou $\neg Sim_C(t,t')$. Soit $\mathcal{F}$ un fait, nous notons par $\mathcal{F}^C$ le* fait étendu complet *obtenu par l'ajout d'atomes de la forme $\neg Sim_C(t,t')$ pour chaque couple de termes (t,t') tels que $Sim_C(t,t') \notin \mathcal{F}$.*

Example 9
$F'^C = F' \wedge \neg Sim_C(i_1,i_2) \wedge \neg Sim_C(i_1,i_3) \wedge \ldots \wedge \neg Sim_C(i_6,i_7)$

Il est évident de noter qu'un *fait étendu $\mathcal{F}$ est consistant ssi il n'existe pas de t,t' tels que des atomes $Sim_C(t,t')$ et $\neg Sim_C(t,t')$ appartiennent tous les deux à $\mathcal{F}$.* Il est important de noter qu'un fait étendu complet est consistant.

Proposition 7
Soient $\mathcal{F}$ un fait relatif à une classe C, et S une S-règle $(C, Sim_C), (p_1, Sim_{S_1}), ..., (p_n, Sim_{S_n})$, S est une S-clé pour C dans $\mathcal{F}$ ssi $(\mathcal{F}^C, S)$ est satisfiable.

Proposition 8
Soient $\mathcal{F}$ un fait relatif à la classe C, et F une F-règle $(C, Sim_C), (p_1, Sim_{F_1}), ..., (p_n, Sim_{F_n})$, F est une F-clé pour C dans $\mathcal{F}$ ssi $closure(\mathcal{F}^C, F), F$ est satisfiable.

5 Conclusion

Dans cet article, nous avons exploré et comparé différentes sémantiques de clés que ce soit en terme d'interprétation, de déduction et de découverte. Ces clés peuvent être utilisées dans le contexte du Web de Données. Nous souhaitons, dans un premier temps compléter cette formalisation en fournissant des preuves pour les principales propositions. De plus, nous envisageons de comparer expérimentalement leur utilisation pour lier différents jeux de données de divers domaines.

Remerciements
Ce travail a été financé par l'Agence Nationale de la Recherche (projet QUALINCA-ANR-12-CORD-0012).

Références

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

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

[3] HALPIN H., HAYES P. & THOMPSON H. S. (2011). When owl : sameas isn't the same redux : A preliminary theory of identity and inference on the semantic web. In *Proc of Workshop on Discovering Meaning On the Go in Large Heterogeneous Data*, p. 25–30.

[4] HOGAN A., ZIMMERMANN A., UMBRICH J., POLLERES A. & DECKER S. (2012). Scalable and distributed methods for entity matching, consolidation and disambiguation over linked data corpora. *J. Web Sem.*, **10**, 76–110.

[5] ISELE R., JENTZSCH A. & BIZER C. (2011). Efficient multidimensional blocking for link discovery without losing recall. In *Proceedings of the 14th International Workshop on the Web and Databases 2011, WebDB 2011*.

[6] PERNELLE N., SAÏS F. & SYMEONIDOU D. (2013). An automatic key discovery approach for data linking. *Web Semantics : Science, Services and Agents on the World Wide Web*, **23**, 16 – 30.

[7] 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.

[8] SUCHANEK F. M., ABITEBOUL S. & SENELLART P. (2011). Paris : Probabilistic alignment of relations, instances, and schema. *The Proceedings of the VLDB Endowment*, **5**(3), 157–168.

[9] VOLZ J., BIZER C., GAEDKE M. & KOBILAROV G. (2009). Discovering and maintaining links on the web of data. In *Proceedings of the 8th International Semantic Web Conference*, ISWC '09, p. 650–665, Berlin, Heidelberg : Springer-Verlag.

[10] WAGNER G. (2003). Web rules need two kinds of negation. In *PPSWR*, p. 33–50.