Alignement d'ontologies pour OWL-Lite : l'apport d'un classifieur sémantique

Raphaël Troncy, Umberto Straccia, Henrik Nottelmann

Alignement d'ontologies pour OWL-Lite : l'apport d'un classifieur sémantique*

Raphaël Troncy[1], Umberto Straccia[1], Henrik Nottelmann[2]

[1] ISTI-CNR, Via G. Moruzzi 1, 56124 Pisa, Italy
{troncy,straccia}@isti.cnr.it
http://nmis.isti.cnr.it/troncy/

[2] Institute of Informatics and Interactive Systems,
University of Duisburg-Essen, 47048 Duisbourg, Germany
nottelmann@uni-duisbourg.de

Résumé : Cet article introduit un cadre logique pour apprendre automatiquement à aligner des ontologies formelles, une tâche cruciale pour le Web Sémantique. Notre approche est basée sur la théorie des probabilités, ce qui permet de gérer à la fois l'incertitude intrinsèque au processus de comparaison et les correspondances entre entités ontologiques qui ne sont pas exactes. Plusieurs composants spécialisés sont combinés pour trouver les différents alignements possibles (avec leurs poids). En particulier, un nouveau classifieur utilisant pleinement la sémantique des définitions est proposé. Les ensembles de règles de correspondance (ou *mapping*) sont alors évalués, la probabilité maximale désignant finalement le "meilleur" alignement.

Mots-clés : Alignement d'ontologies, logique probabiliste, apprentissage, classifieur sémantique, OWL

1 Introduction

La recommandation récente par le W3C des langages OWL (OWL, 2004) et RDF pour représenter des ontologies et des annotations formelles sur le Web est une étape supplémentaire pour parvenir au Web Sémantique (Berners-Lee *et al.*, 2001). Alors que des efforts importants sont désormais déployés pour accélérer l'adoption de ces nouveaux langages à travers la mise en ligne de ressources et de guides de bonnes pratiques[1], et pour interroger les structures formelles déjà existantes via un langage de requêtes et des mécanismes de règles[2], il est un autre domaine primordial pour permettre une large utilisation des ontologies :

*Ce travail a été soutenu et financé dans le cadre d'une bourse ERCIM.

[1] http://www.w3.org/2001/sw/BestPractices/

[2] http://www.w3.org/2001/sw/DataAccess/

leur comparaison. En effet, les ontologies déjà développées pour un même domaine sont multiples, par exemple dans le cas de la médecine (Charlet *et al.*, 2005). De plus, on peut très facilement imaginer que les ontologies vont être développées dans des environnements distribués qui vont requérir, entre autres fonctionnalités, la possibilité d'intégrer et de fusionner des extraits d'ontologies déjà existantes. Dans ce cadre, l'alignement d'ontologies, c'est-à-dire la comparaison des différentes entités définies dans celles-ci, est une tâche centrale.

Les méthodes de comparaison d'ontologies vont permettre de traduire des requêtes formulées selon une base de connaissance *source* vers une base de connaissance *cible*. Ainsi, des documents annotés formellement selon cette dernière base de connaissance pourront être retrouvés. D'une manière générale, l'alignement d'ontologies sera une des premières tâches pour les agents les consommant pour pouvoir rendre des services sur le web. A l'extrême, ces méthodes de comparaison pourraient être utilisées entre les différentes versions d'une même ontologie afin de retrouver une trace des changements intervenus lors de son évolution et de détecter des éventuelles inconsistances. Enfin, cette tâche doit pouvoir s'effectuer de manière complètement automatique dans la mesure où il n'est pas réaliste de maintenir des mises en correspondance d'ontologies effectuées à la main, a fortiori si ces ontologies dépassent une certaine taille ou complexité. Dans cet article, nous proposons un cadre logique et probabiliste permettant de comparer automatiquement des ontologies. En particulier, nous présentons un classifieur utilisant pleinement la sémantique des définitions et des axiomes OWL pour établir des relations d'équivalence et de subsomptions entre les concepts et les relations (ou propriétés) des ontologies.

L'article est structuré de la manière suivante. Nous introduisons dans la section 2 notre cadre logique et probabiliste qui a pour but de trouver les meilleures correspondances (ou *mappings*) entre entités définies dans des ontologies représentées en OWL. La section 3 détaille notre approche théorique pour apprendre les différents alignements possibles, l'évaluation finale reposant sur la combinaison de prédictions de divers classifieurs. Nous donnons alors une liste non exhaustive de classifieurs pouvant être utilisés. Ceux-ci se basant essentiellement sur les données terminologiques des entités ontologiques à comparer ou sur leurs instances, nous présentons un nouveau classifieur utilisant pleinement la sémantique formelle des définitions et des axiomes OWL (section 4). La section 5 revient sur les différents travaux liés à notre approche. Finalement, la section 6 nous permet de conclure et d'ouvrir quelques perspectives.

2 Un cadre logique pour l'alignement d'ontologies

Cette section introduit notre cadre logique pour comparer automatiquement des ontologies. Celui-ci s'inspire des travaux formels menés autour de l'échange d'information (Fagin *et al.*, 2003) et emprunte à d'autres approches comme GLUE (Doan *et al.*, 2003) l'idée de combiner plusieurs composants spécialisés pour obtenir le meilleur résultat. De plus, notre approche gère l'incertitude intrinsèque au processus de comparaison en se basant sur Datalog probabiliste,

pour lequel des outils sont disponibles. Il reprend donc la formalisation proposée dans (Nottelmann & Straccia, 2005) et constitue une extension du système **sPLMAP** (*Schema Probabilistic Learning Mappings*) pour l'alignement d'ontologies.

2.1 Datalog probabiliste

Datalog probabiliste (Fuhr, 2000) (ou *pDatalog*) est une extension de *Datalog* qui est lui même une variante de la logique des prédicats, basé sur des clauses de Horn sans fonction en argument des prédicats (*i.e.* les arguments sont des constantes ou des variables). Dans pDatalog, tous les faits et règles sont préfixés par un poids probabiliste $0 < \alpha \leq 1$:

$$\alpha\ A \quad \leftarrow \quad B_1, \ldots, B_n$$

où A (la tête de la règle), et $B_1, \ldots, B_n$ $(n \geq 0)$ (les sous-buts du corps de la règle) sont des formules atomiques. Le poids $\alpha = 1$ est facultatif. Une règle αr s'interprète alors comme *"la probabilité qu'une instanciation de la règle r soit vraie est α"*. Par exemple, le programme pDatalog ci-dessous indique qu'une personne est une femme avec une probabilité de 50%.

$$\begin{aligned} \texttt{personne(ana)} \quad &\leftarrow \\ 0.8\ \texttt{personne(raphaël)} \quad &\leftarrow \\ 0.5\ \texttt{femme(X)} \quad &\leftarrow \quad \texttt{personne(X)} \end{aligned}$$

On peut alors calculer $Pr(\texttt{femme(ana)}) = 0.5$, et $Pr(\texttt{femme(raphaël)}) = 0.8 \times 0.5 = 0.4$. Formellement, une structure d'interprétation[3] dans pDatalog est un couple $\mathscr{I} = (\mathcal{W}, \mu)$ où $\mathcal{W}$ représente les mondes possibles (*i.e.* l'instanciation de la partie déterministe d'un programme pDatalog plus un sous-ensemble de la partie probabiliste où toutes les probabilités sont enlevées) et μ est une distribution des probabilités sur $\mathcal{W}$. Une interprétation est un couple $I = (\mathscr{I}, w)$ tel que $w \in \mathcal{W}$. La valeur de vérité d'une formule, compte tenu d'une interprétation et d'un monde possible, peut être définie récursivement de la manière suivante :

$$\begin{aligned} (\mathscr{I}, w) \quad &\models \quad A \text{ ssi } A \in w\ , \\ (\mathscr{I}, w) \quad &\models \quad A \leftarrow B_1, \ldots, B_n \text{ ssi } (\mathscr{I}, w) \models B_1, \ldots, B_n \Rightarrow (\mathscr{I}, w) \models A\ , \\ (\mathscr{I}, w) \quad &\models \quad \alpha r \text{ ssi } \mu(\{w' \in \mathcal{W} \colon (\mathscr{I}, w') \models r\}) = \alpha\ . \end{aligned}$$

Une interprétation est alors un modèle pour un programme pDatalog si et seulement si elle implique tous les faits et règles.

2.2 Ontologies et alignement

Ontologies. Par ontologie, nous entendons la représentation conceptuelle et formelle d'un domaine pour une application donnée. Les ontologies fournissent donc le vocabulaire propre à un domaine et définissent formellement des concepts et des relations entre ceux-ci. Ces concepts (et ces relations) sont organisés *via* la

[3] Dans pDatalog, seules les interprétations dans l'univers d'Herbrand sont considérées.

relation de subsomption pour former une taxonomie. L'ontologie peut contenir en outre des axiomes formels pour permettre des calculs d'inférences additionnels, et des individus (*i.e.* des instances des concepts représentés). Nous considérons dans la suite que ces ontologies sont représentées dans les langages OWL-Lite ou OWL-DL (OWL, 2004) qui sont dérivés de logiques de descriptions bien étudiées (resp. $\mathcal{SHIF}(\mathcal{D})$ et $\mathcal{SHOIN}(\mathcal{D})$) (Horrocks *et al.*, 2003). Le tableau 1 donne ainsi un extrait de deux ontologies **S** et **T** en utilisant une syntaxe usuelle en logiques de descriptions.

Ontologie source **S** :		
`Directions`	$\doteq$	`(`$\leq$ `1 town xsd:string)`
`Congress`	$\doteq$	`(and (`$\leq$ `1 id xsd:string) (`$\leq$ `1 acronym xsd:string)` `(all place Directions))`
Ontologie cible **T** :		
`Address`	$\doteq$	`(`$\leq$ `1 city xsd:string)`
`Conference`	$\doteq$	`(and (`$\leq$ `1 name xsd:string) (`$\leq$ `1 shortName xsd:string)` `(all location Address))`

TAB. 1 – Extrait de deux ontologies **S** et **T**

La partie terminologique $\mathcal{TF}$ d'une ontologie contient des axiomes de la forme $A \sqsubseteq C$ et $A \doteq C$, où A est un nom de concept et C une expression. Nous transformons alors ces axiomes en forme normale pour la négation (NNF) en utilisant les règles de ré-écriture suivantes sur les différentes composantes d'une définition jusqu'à ce que le connecteur de négation ne soit appliqué qu'à des formules atomiques :

$\neg\top$ devient $\bot$ $\quad$ $\neg\bot$ devient $\top$ $\quad$ $\neg\neg C$ devient C
$\neg(C \sqcap D)$ devient $\neg C \sqcup \neg D$ $\quad$ $\neg(C \sqcup D)$ devient $\neg C \sqcap \neg D$
$\neg\exists R.C$ devient $\forall R.\neg C$ $\quad$ $\neg\forall R.C$ devient $\exists R.\neg C$
$\neg(\geq n\ R.C)$ devient $(\leq n-1\ R.C)$ si $n \geq 1$. Pour $n = 0$ $\neg(\geq n\ R.C)$ devient $\bot$
$\neg(\leq n\ R.C)$ devient $(\geq n+1\ R.C)$

De plus, les définitions de concept de la forme $A \sqsubseteq C$ sont remplacés par $A \doteq C \sqcap A^*$, où A^* est un nouveau nom de concept. Finalement, $\mathcal{TF}$ est *saturée* (*i.e.* complétée par l'ensemble des déductions possibles compte tenu de la sémantique du langage OWL) pour donner $\mathcal{TF}^* = \{A \doteq C\}$ où A est un nom de concept et C une expression en NNF.

Alignement. Notre but est de déterminer automatiquement des relations d'équivalence (ou de subsomption) entre des entités définies dans des ontologies OWL. Par exemple, étant donné les deux ontologies définies dans le tableau 1, nous voudrions établir une correspondance entre les concepts `Congress` et `Conference`. Théoriquement, l'alignement d'ontologies dans notre cadre suit l'approche GLaV[4] détaillée dans (Lenzerini, 2002). Un *mapping* est un tuple $\mathcal{M} = (\mathbf{T}, \mathbf{S}, \Sigma)$ où **S** et **T** sont respectivement les ontologies source et cible, et

[4] Approche mixte *Global and Local as View* développée pour l'interopérabilité des bases de données.

Σ est un ensemble de contraintes (*i.e.* des règles pDatalog) de la forme :

$$\alpha_{j,i}\ T_j(x) \quad \leftarrow \quad S_i(x)\ .$$

Cette règle signifie que le concept (resp. la relation) S_i dans l'ontologie source correspond au concept (resp. la relation) T_j dans l'ontologie cible, et que la probabilité que cet appariement soit effectivement vrai est $\alpha_{j,i}$. Notons qu'aucune contrainte supplémentaire n'est posée sur ces mises en correspondance : un concept de l'ontologie source peut correspondre à 0 ou plusieurs concepts dans l'ontologie cible et réciproquement.

Soit un *mapping* $\mathcal{M} = (\mathbf{T}, \mathbf{S}, \Sigma)$ et un modèle I pour $\mathbf{S}$, un modèle J pour $\mathbf{T}$ est une *solution* pour I sous $\mathcal{M}$ si et seulement si $\langle J, I\rangle$ (l'interprétation combinée sur $\mathbf{T}$ et $\mathbf{S}$) est un modèle pour Σ. La solution minimale, notée $J(I, \Sigma)$, est l'instance de la relation correspondante avec $\mathbf{T}(I, \Sigma)$. En utilisant les règles pDatalog, la solution minimale $\mathbf{T}(I, \Sigma)$ est exactement le résultat de l'application des règles Σ sur les entités de $\mathbf{S}$.

3 Apprendre automatiquement à comparer des ontologies

L'apprentissage des appariements possibles entre ontologies dans notre approche s'effectue en quatre étapes : (i) nous devinons un alignement potentiel, c'est-à-dire un ensemble de règles Σ_k de la forme $T_j(x) \leftarrow S_i(x)$ (règles sans poids) ; (ii) nous estimons alors la qualité de Σ_k ; (iii) parmi tous les ensembles possibles Σ_k, nous sélectionnons le "meilleur" selon une certaine mesure de qualité ; et finalement (iv) nous estimons le poids α pour les règles appartenant à l'alignement sélectionné.

3.1 Estimation de la qualité d'un ensemble de correspondances entre entités ontologiques

Soit une ontologie source $\mathbf{S} = \langle S_1, \ldots, S_s\rangle$, une ontologie cible $\mathbf{T} = \langle T_1, \ldots, T_t\rangle$, et deux interprétations I pour $\mathbf{S}$ et J pour $\mathbf{T}$. Notre but est de trouver le "meilleur" ensemble Σ de règles établissant des correspondances entre entités ontologiques, c'est-à-dire l'ensemble qui maximise la probabilité $Pr(\Sigma, J, I)$ que les objets appartenant à la solution minimale $T(I, \Sigma)$ avec $\mathcal{M} = (T, S, \Sigma)$ et les objets de T soient similaires. $Pr(\Sigma, J, I)$ estime donc la probabilité qu'un tuple de $T(I, \Sigma)$ soit une valeur plausible pour T et vice versa. En utilisant la théorie de Bayes, nous obtenons[5] :

$$\begin{aligned} Pr(\Sigma, J, I) &= Pr(T(I,\Sigma)|J) \cdot Pr(J|T(I,\Sigma)) \\ &= Pr(T(I,\Sigma)|J)^2 \cdot \frac{Pr(T)}{Pr(T(I,\Sigma))} \\ &= Pr(T(I,\Sigma)|J)^2 \cdot \frac{|J|/|U|}{|T(I,\Sigma)|/|U|} \end{aligned}$$

[5]Où U est l'ensemble des tuples et $Pr(T)$ est la probabilité qu'un tuple donné soit dans J.

$$= \quad Pr(T(I,\Sigma)|J)^2 \cdot \frac{|J|}{|T(I,\Sigma)|} \tag{1}$$

Nous appelons ensuite $T_j(I,\Sigma)$ la restriction de $T(I,\Sigma)$ à T_j. L'ensemble Σ peut être partitionné en sous-ensembles Σ_j formés des règles de *mapping* ayant T_j comme objet commun en tête. Nous avons alors :

$$Pr(T(I,\Sigma)|J) = \sum_j Pr(T_j(I,\Sigma_j)|J_j) \cdot \frac{Pr(J_j)}{Pr(J)} \tag{2}$$

Pour s entités dans l'ontologie source et un nombre j fixe, il y a aussi s ensembles possibles $\Sigma_{j,i}$, et donc $2^s - 1$ combinaisons non vides (union) de ces ensembles qui forment tous les sous-ensembles Σ_j possibles[6]. Pour simplifier la notation, nous posons dans la suite $S_i = T_j(I,\Sigma_{j,i})$. Ainsi, si Σ_j est formé par les r sous-ensemble de règles $\Sigma_{j,i_1},\ldots,\Sigma_{j,i_r}$, nous obtenons :

$$Pr(T_j(I,\Sigma_j)|J_j) = \sum_{k=1}^{r} Pr(S_{i_k}|J_j) \tag{3}$$

La probabilité $Pr(\Sigma,J,I)$ peut donc être obtenue à partir des $\mathcal{O}(s \cdot t)$ probabilités $Pr(S_i|J_j)$. La section suivante montre comment calculer la probabilité de ces règles.

3.2 Estimation de la probabilité d'une règle

De la même manière que GLUE (Doan *et al.*, 2003), nous estimons la probabilité $Pr(S_i|J_j)$ en combinant les prédictions de différents classifieurs $CL_1,\ldots,CL_n$. Chaque classifier calcule un poids[7] $w(S_i,J_j,CL_k)$ qui est ensuite normalisé et transformé en $Pr(S_i|J_j,CL_k) = f(w(S_i,J_j,CL_k))$, l'approximation de $Pr(S_i|J_j)$ pour CL_k. Nous utilisons les fonctions f de normalisation suivantes :

$$\begin{aligned}
w(S_i,J_j,CL_k) &\mapsto Pr(S_i|J_j)\ , \\
f_{id}(x) &= x\ , \\
f_{sum}(x) &= \frac{x}{\sum_{i'} w(S_{i'},J_j,C_k)}\ , \\
f_{lin}(x) &= c_0 + c_1 \cdot x\ , \\
f_{log}(x) &= \frac{\exp(b_0 + b_1 \cdot x)}{1+\exp(b_0 + b_1 \cdot x)}\ .
\end{aligned}$$

Les fonctions f_{id}, f_{sum} et f_{log} retournent des valeurs dans l'intervalle $[0,1]$. Pour la fonction linéaire, les résultats inférieurs à 0 (resp. supérieurs à 1) doivent être normalisés à 0 (resp. 1). La fonction f_{sum} assure que chaque valeur appartient à l'intervalle $[0,1]$ et que leur somme vaut 1. Son principal avantage est qu'elle n'utilise pas de paramètres extérieurs. En revanche, les paramètres des fonctions linéaire et logistique doivent être appris par régression sur un jeu d'entraînement. Cette phase n'étant nécessaire qu'une seule fois, le résultat peut ensuite

[6] Nous discutons de cette combinatoire apparemment rédhibitoire dans la section 4.3.

[7] Dans la suite, w représentera toujours un poids.

être utilisé pour comparer n'importe quelles ontologies. Les fonctions de normalisation peuvent naturellement être combinées. Par exemple, il est souvent utile d'amener les poids des classifieurs dans un même intervalle (en utilisant f_{sum}), puis d'appliquer une autre fonction de normalisation (*e.g.* f_{log}).

Pour la probabilité finale $Pr(S_i|J_j, CL_k)$, nous avons la contrainte

$$0 \leq Pr(S_i|J_j, CL_k) \leq \frac{\min(|S_i|, |J_j|)}{|J_j|} = \min(\frac{|S_i|}{|J_j|}, 1)$$

Par conséquent, la valeur normalisée (qui appartient à $[0, 1]$) est multipliée par $\min(|S_i|/|J_j|, 1)$ dans une seconde étape de normalisation.

Les prédictions finales $Pr(S_i|J_j, CL_k)$ sont alors combinées en utilisant le Théorème des Probabilités Totales (Mood *et al.*, 1974) qui donne la somme pondérée suivante :

$$Pr(S_i|J_j) \approx \sum_{k=1}^{n} Pr(S_i|J_j, CL_k) \cdot Pr(CL_k) \tag{4}$$

La probabilité $Pr(CL_k)$ reflète la confiance et donc l'importance que l'on donne au classifieur CL_k. Dans la suite, nous avons simplement utilisé $Pr(CL_k) = \frac{1}{n}$ pour $1 \leq k \leq n$, considérant donc que tous les classifieurs sont équiprobables. Nous présentons dans la section 4 différents classifieurs en mettant l'accent sur un nouveau classifieur utilisant pleinement la structure et la sémantique des définitions des concepts et des relations de l'ontologie.

4 Les différents classifieurs utilisés

Nous commençons par présenter quelques classifieurs classiques pouvant être utilisés dans notre cadre (section 4.1). Cependant, la plupart de ces classifieurs nécessitent, pour fonctionner, d'avoir des instances des deux ontologies que l'on cherche à comparer. Nous introduisons donc un nouveau classifieur n'ayant pas ce pré-requis et utilisant pleinement la sémantique des concepts et des relations définis dans les ontologies (section 4.2). Nous montrons finalement à l'aide d'un exemple son fonctionnement et nous discutons de son implémentation (section 4.3).

4.1 Les classifieurs classiques

4.1.1 Même nom de classe ou de relation

Ce classifieur binaire CL_S retourne le poids 1 si et seulement si les deux concepts (ou relations) que l'on cherche à comparer ont le même nom ou le même *stem* (la racine du terme privée de ses suffixes), et 0 dans les autres cas :

$$w(S_i, J_j, CL_S) = \begin{cases} 1 & \text{si } S_i, J_j \text{ ont le même } \textit{stem}, \\ 0 & \text{sinon} \end{cases}$$

4.1.2 Classifieur kNN

L'algorithme des k plus proches voisins (kNN) repose sur le calcul des distances entre une forme inconnue et toutes les formes d'une base de référence. Il est particulièrement populaire pour la classification de textes (Sebastiani, 2002). Dans notre classifieur CL_{kNN}, chaque concept ou relation S_l représente ainsi une catégorie et des ensembles d'entraînement sont formés à partir des instances i' de S_l :

$$Train = \bigcup_{l=1}^{s} \{(S_l, i') : i' \in S_l\}$$

Une variante probabiliste du produit scalaire est alors utilisée pour calculer les valeurs de similarités. Soient t et t' des ensembles de mots, $Pr(m|S_i)$ et $Pr(m|J_j)$ sont calculés comme la fréquence normalisée de ces mots dans les instances :

$$RSV(t, t') = \sum_{m \in t \cap t'} Pr(m|S_i) \cdot Pr(m|J_j)$$

4.1.3 Classifieur naïf de Bayes

Le classifieur naïf de Bayes CL_B est lui aussi très prisé pour la classification de textes (Sebastiani, 2002). De la même manière que précédemment, chaque concept ou relation S_i représente une catégorie, et leurs instances sont considérées comme des ensembles de mots (la fréquence des mots est normalisée pour les estimations de probabilité). La formule de calcul est finalement :

$$w(S_i, J_j, CL_B) = Pr(S_i) \cdot \sum_{x \in J_j} \prod_{m \in x} Pr(m|S_i)$$

4.2 Le classifieur structurel et sémantique

Outre ces algorithmes bien connus en classification, nous introduisons un autre classifieur, CL_{Sem}, capable d'utiliser pleinement la sémantique des définitions OWL et guidé par la syntaxe de ces définitions. Son utilisation dans notre cadre peut s'effectuer a posteriori (auquel cas, il utilisera en entrée le résultat des autres classifieurs), ou indépendamment (dans le cas où les ontologies n'ont pas d'instances par exemple). Dans la suite, nous notons $Pr'(A_T|A_S, \Sigma)$ (resp. $Pr'(R_T|R_S, \Sigma)$) la probabilité de la règle $A_T(x) \leftarrow A_S(x)$ (resp. $R_T(x, y) \leftarrow R_S(x, y)$) estimée par les autres classifieurs, où A_S (resp. A_T) est un nom de concept de l'ontologie source (resp. cible), et R_S (resp. R_T) est un nom de propriété de l'ontologie source (resp. cible). Nous rappelons également que la probabilité $\alpha_{j,i}$ d'une règle $T_j(x) \leftarrow S_i(x)$ est donnée par :

$$\alpha_{j,i} = Pr(T_j|S_i) = Pr(S_i|T_j) \cdot \frac{Pr(T_j)}{Pr(S_i)} = Pr(S_i|T_j) \cdot \frac{|T_j|}{|T_j(I, \Sigma)|}$$

La définition formelle et récursive de CL_{Sem} est alors :

1. Si R_S et R_T sont des noms de propriétés :

$$w(R_S, R_T, \Sigma) = \begin{cases} 0 & \text{si } R_T \leftarrow R_S \notin \Sigma \\ Pr'(R_T|R_S, \Sigma) & \text{sinon} \end{cases}$$

2. Si A_S et A_T sont des noms de concepts : soit $\mathcal{D} = \mathcal{D}(A_S) \times \mathcal{D}(A_T)$[8]

$$w(A_S, A_T, \Sigma) = \begin{cases} 0 & \text{si } A_T \leftarrow A_S \notin \Sigma \\ Pr'(A_T|A_S, \Sigma) & \text{si } |\mathcal{D}| = 0 \text{ et } A_T \leftarrow A_S \in \Sigma \\ \frac{1}{(|\mathcal{D}|+1)} \cdot \left(Pr'(A_T|A_S, \Sigma) + \max_{Set} \left(\sum_{(C_i, D_j) \in Set} w(C_i, D_j, \Sigma) \right) \right) & \text{sinon} \end{cases}$$

3. Soit $C_S = (QR.C)$ et $D_T = (Q'R'.D)$, où Q, Q' sont les quantifieurs $\forall$ ou $\exists$ ou une restriction sur la cardinalité, R, R' sont des noms de propriétés et C, D des expressions, alors :

$$w(C_S, D_T, \Sigma) = w_Q(Q, Q') \cdot w(R, R', \Sigma) \cdot w(C, D, \Sigma)$$

4. Soit $C_S = (op\ C_1 \dots C_m)$ et $D_T = (op'\ D_1 \dots D_m)$, où les concepts C_S, D_T sont en notation préfixée et op, op' sont des constructeurs de concept parmi $\sqcap, \sqcup, \neg$ et $n, m \geq 1$, alors :

$$w(C_S, D_T, \Sigma) = w_{op}(op, op') \cdot \frac{\max_{Set} \left(\sum_{(C_i, D_j) \in Set} w(C_i, D_j, \Sigma) \right)}{\min(m, n)} \quad \text{avec}$$

- $Set \subseteq \{C_1 \dots C_m\} \times \{D_1 \dots D_n\}$, et $|Set| = \min(m, n)$,
- $(C, D) \in Set, (C', D') \in Set \Rightarrow C \neq C', D \neq D'$

Plutôt que de prendre la valeur maximale, la valeur moyenne peut être utilisée pour simplifier les calculs. En effet, dans ce cas là, l'estimation du classifier est donnée par la résolution d'un seul système d'équations linéaires, alors que dans le premier cas, il faut résoudre autant de systèmes qu'il y a de choix pour Set. Des valeurs pour les poids w_{op} et w_Q sont données à titre indicatif dans le tableau 2.

w_{op} est défini par :

	$\sqcap$	$\sqcup$	$\neg$
$\sqcap$	1	1/4	0
$\sqcup$		1	0
$\neg$			1

w_Q est défini par :

	$\exists$	$\forall$
$\exists$	1	1/4
$\forall$		1

	$\leq n$	$\geq n$
$\leq m$	1	1/3
$\geq m$		1

TAB. 2 – Valeurs possibles pour les poids w_{op} et w_Q

4.3 Exemple et implémentation

Nous illustrons maintenant le fonctionnement de ce classifieur en reprenant une version simplifiée de l'exemple donné dans le tableau 1. Supposons que les autres classifieurs aient fournis les estimations suivantes :

$$\begin{array}{lcll} w(town, city, \Sigma) & = & Pr(city|town, \Sigma) & = & 0.8 \\ w(id, name, \Sigma) & = & Pr(name|id, \Sigma) & = & 1 \\ w(place, location, \Sigma) & = & Pr(location|place, \Sigma) & = & 0.6 \end{array}$$

[8] $\mathcal{D}(A_S)$ représente l'ensemble des concepts directement parents de A_S.

Le classifieur CL_{Sem} estimera $w(Directions, Address, \Sigma)$ par :

$$\begin{aligned} w(D_S, A_T, \Sigma) &= \frac{1}{2} \cdot (\alpha + w_Q(\leq 1, \leq 1) \cdot w(town, city, \Sigma) \cdot w(string, string, \Sigma)) \\ &= \frac{1}{2} \cdot (\alpha + 1 \cdot 0.8 \cdot 1) = \frac{\alpha + 0.8}{2} = 0.9 \text{ (si } \alpha = 1) \end{aligned}$$

Nous définissons ensuite Set sur $\{C_1 \dots C_m\} \times \{D_1 \dots D_n\}$:

$$\begin{aligned} Set &= \{(Cong_1, Conf_1), (Cong_2, Conf_2)\} \text{ ou} \\ Set &= \{(Cong_1, Conf_2), (Cong_2, Conf_1)\} \end{aligned}$$

On obtient alors :

$$\begin{array}{lcll} w(Cong_1, Conf_1, \Sigma) & = & 1 \cdot w(id, name, \Sigma) \cdot w(string, string, \Sigma) & = & 1 \\ w(Cong_1, Conf_2, \Sigma) & = & 1 \cdot w(id, location, \Sigma) \cdot w(string, Add, \Sigma) & = & 0 \\ w(Cong_2, Conf_1, \Sigma) & = & 1 \cdot w(place, name, \Sigma) \cdot w(Dir, string, \Sigma) & = & 0 \\ w(Cong_2, Conf_2, \Sigma) & = & 1 \cdot w(place, location, \Sigma) \cdot w(Dir, Add, \Sigma) & & \\ & = & 1 \cdot 0.6 \cdot \frac{\alpha+0.8}{2} & = & \frac{15 \cdot \alpha + 12}{50} \end{array}$$

Finalement, CL_{Sem} estimera $w(Congress, Conference, \Sigma)$ par :

$$\begin{aligned} w(C_S, C_T, \Sigma) &= \frac{1}{2} \cdot \left(\beta + w_{op}(\sqcap, \sqcap) \cdot \frac{\max_{Set} \left(\sum_{(Cong_i, Conf_j) \in Set} w(Cong_i, Conf_j) \right)}{\min(2, 2)} \right) \\ &= \frac{1}{2} \cdot \left(\beta + 1 \cdot \frac{\max\left((1 + \frac{15 \cdot \alpha + 12}{50}), (0 + 0) \right)}{2} \right) \\ &= \frac{1}{2} \cdot \left(\beta + \frac{15 \cdot \alpha + 62}{100} \right) = \frac{100 \cdot \beta + 15 \cdot \alpha + 62}{200} = 0.88 \text{ (si } \alpha = \beta = 1) \end{aligned}$$

où α et β représentent la moyenne pondérée des estimations fournies par les autres classifieurs, et valent 1 si le classifieur sémantique est utilisé seul.

Comme nous l'avons vu dans la section 3.1, notre approche commence par considérer tous les appariements possibles (supposons $m \times n$), et génère ensuite tous les sous-ensembles Σ_k contenant ces règles, soit $2^{m \times n}$. Cette combinatoire est évidemment très coûteuse en temps de calcul. Cependant, il est possible de la réduire fortement en ajoutant des contraintes sur la manière dont les sous-ensemble Σ_k sont formés. Par exemple, nous imposons que chaque ensemble Σ_k contienne exactement une règle appariant chacune des entités ontologiques, le nombre de Σ_k est alors réduit à $\frac{Max(m,n)!}{|m-n|!}$. D'autres contraintes utilisant plus fortement les taxonomies des ontologies à comparer ont aussi été implémentées pour réduire le nombre de ces sous-ensembles.

5 Travaux existants

Parmi les travaux liés à l'alignement d'ontologies, (Ehrig & Staab, 2004) proposent également de combiner différentes mesures de similarité calculées à partir de règles générales établies à la main. Cependant, outre le fait que ces règles sont relativement discutables, cette approche n'est pas complètement automatique, contrairement à la notre. (Noy & Musen, 2001) ont développé une approche très intéressante : ils partent de paires de concepts qui semblent proches (découverts

de façon automatique ou proposés manuellement) et calculent leur similarité *hors contexte* (les ancres de la recherche, peuvent être éloignées) en étudiant les chemins (dans la taxonomie) qui relient les paires de concepts. Cette méthode, implémentée dans l'outil ANCHOR-PROMPT, affiche pour l'instant les meilleurs résultats. (Euzenat & Valtchev, 2004) ont adapté, eux, des travaux sur le calcul de similarité dans la représentation de connaissance par objets aux langages web actuels pour représenter des ontologies. Une mesure de similarité globale prenant en compte l'ensemble des caractéristiques du langage OWL-Lite est ainsi proposée, capable à la fois de traiter les définitions circulaires et les collections. Pour un état de l'art complet sur les différentes méthodes proposées jusqu'à présent, nous renvoyons le lecteur à (KW, 2004).

Ces techniques d'alignement d'ontologies étant donc relativement différentes, il est difficile de les comparer théoriquement. Une approche empirique, sous la forme de *benchmarks* communs, est préférable pour évaluer tous ces algorithmes. C'est dans cet esprit qu'a eu lieu le challenge EON *Ontology Alignment Contest* fournissant des jeux de tests communs avec les alignements à deviner (Sure *et al.*, 2004). Bien qu'hétérogènes, les résultats fournis par les participants sont très intéressants et de nouvelles idées ont été proposées pour améliorer les futures compétitions du même type. Signalons enfin les travaux de (Euzenat, 2004) qui propose une API commune pour exprimer les résultats des méthodes d'alignement d'ontologies. Nous utilisons également ce format ce qui devrait faciliter les futures comparaisons avec les autres approches.

6 Conclusion et perspectives

Dans cet article, nous avons introduit un cadre logique pour comparer automatiquement des ontologies formelles. Les alignements possibles y sont définis comme des règles dans Datalog probabiliste. Nous avons également présenté un nouveau classifieur utilisant pleinement la sémantique des définitions OWL-Lite des concepts et des relations et des axiomes de l'ontologie. L'implémentation de ce classifieur et son intégration dans le cadre global est en cours de finalisation, mais les premiers résultats sont déjà très encourageants.

Il nous reste désormais à prendre en compte la définition des concepts en extension (par l'énumération de ses instances) pour que notre approche puisse considérer tout OWL-DL. Une autre piste à l'étude consiste à ajouter des classifieurs supplémentaires (utilisation de ressources terminologiques ou d'autres mesures de distance) et à multiplier les tests en les combinant. Nous planifions enfin de participer activement aux prochaines campagnes d'évaluation des différentes approches existantes pour l'alignement d'ontologies, dans le même esprit que le dernier atelier EON (Sure *et al.*, 2004) par exemple.

Références

BERNERS-LEE T., HENDLER J. & LASSILA O. (2001). The Semantic Web.

Scientific American, **284**(5), 34–43.

CHARLET J., BACHIMONT B. & TRONCY R. (2005). Ontologies pour le Web sémantique. *Revue I3, numéro HS "Web sémantique"* (à paraitre).

DOAN A., MADHAVAN J., DHAMANKAR R., DOMINGOS P. & HALEVY A. (2003). Learning to Match Ontologies on the Semantic Web. *The VLDB Journal*, **12**(4), 303–319.

EHRIG M. & STAAB S. (2004). QOM - quick ontology mapping. In *3rd International Semantic Web Conference (ISWC'04)*, p. 683–697, Hiroshima, Japon.

EUZENAT J. (2004). An API for ontology alignment. In *3rd International Semantic Web Conference (ISWC'04)*, p. 698–712, Hiroshima, Japon.

EUZENAT J. & VALTCHEV P. (2004). Similarity-based ontology alignment in OWL-Lite. In *15th European Conference on Artificial Intelligence (ECAI'04)*, p. 333–337, Valence, Espagne.

FAGIN R., KOLAITIS P. G., MILER R. J. & POPA L. (2003). Data Exchange : Semantics and Query Answering. In *9th International Conference on Database Theory (ICDT'03)*, p. 207–224, Sienne, Italie.

FUHR N. (2000). Probabilistic Datalog : Implementing logical information retrieval for advanced applications. *Journal of the American Society for Information Science*, **51**(2), 95–110.

HORROCKS I., PATEL-SCHNEIDER P. F. & VAN HARMELEN F. (2003). From SHIQ and RDF to OWL : The making of a web ontology language. *Journal of Web Semantics*, **1**(1), 7–26.

KW C. (2004). State of the Art on Ontology Alignment. Deliverable Knowledge Web 2.2.3, FP6-507482.

LENZERINI M. (2002). Data integration : A theoretical perspective. In *21st ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS'02)*, p. 233–246, Madison, Wisconsin, USA.

MOOD A. M., GRAYBILL F. A. & BOES D. C. (1974). *Introduction to the Theory of Statistics*. McGraw-Hill.

NOTTELMANN H. & STRACCIA U. (2005). sPLMap : A probabilistic approach to schema matching. In *27th European Conference on Information Retrieval (ECIR'05)*, p. 81–95, Santiago de Compostela, Espagne.

NOY N. F. & MUSEN M. A. (2001). Anchor-PROMPT : Using non-local context for semantic matching. In *Workshop on Ontologies and Information Sharing at IJCAI'01*, Seattle, Washington, USA.

OWL (2004). Web Ontology Language Reference. W3C Recommendation (10 Février). `http://www.w3.org/TR/owl-ref/`.

SEBASTIANI F. (2002). Machine learning in automated text categorization. *ACM Comuting Surveys*, **34**(1), 1–47.

Y. SURE, O. CORCHO, J. EUZENAT & T. HUGHES, Eds. (2004). *3rd International Workshop on Evaluation of Ontology-based Tools (EON'04)*, Hiroshima, Japon.