Thèse

de

L'Université Paris-Sud 11

présentée pour obtenir le grade de

Docteur de l'Université Paris-Sud 11

Spécialité : Informatique

Par

Fayçal HAMDI

Améliorer l'interopérabilité sémantique : Applicabilité et utilité de l'alignement d'ontologies

Jérôme Euzenat, Directeur de recherche, INRIA Rhône-Alpes	Rapporteur
Bernd Amann, Professeur, Université Paris 6	Rapporteur
Jean Charlet, Chargé de mission AP-HP, INSERM	Examinateur
Sébastien Mustière, Chargé d'études et de recherche, IGN	Examinateur
Christine Froidevaux, Professeur, Université Paris-Sud 11	Examinatrice
Chantal Reynaud, Professeur, Université Paris-Sud 11	Directrice de thèse
Brigitte Safar, Maître de conférences, Université Paris-Sud 11	Co-directrice de thèse

Laboratoire de Recherche en Informatique, U.M.R. CNRS 8623
Université Paris-Sud 11, 91405 Orsay Cedex, France
INRIA Saclay-Ile-de-France

Table des matières

Introduction

Les technologies internet ont permis de rendre disponible un très grand nombre d'informations au cours de la dernière décennie. Les différentes sources sont a priori facilement accessibles et partageables, mais malheureusement les informations contenues sont souvent manipulables et comprises uniquement par des humains. Ainsi, le challenge du web sémantique est aujourd'hui de permettre à des agents logiciels de comprendre ces sources d'informations d'un point de vue sémantique.

Dans le web actuel, publier et rendre accessible des ressources est une tâche très difficile à réaliser. Les recherches génèrent des volumes de données gigantesques. Il est très difficile d'y trouver l'information recherchée. Par ailleurs, le web d'aujourd'hui est encore très syntaxique. Les pages HTML reliées par des liens hypertextes contiennent beaucoup d'informations mais leur description est limitée. Le marquage actuel des pages HTML concerne le rendu (fonte, taille, couleur). Des hyperliens lient les contenus, mais le contenu sémantique est surtout accessible aux hommes et pas facilement aux machines.

L'idée principale du web sémantique, proposée par Tim Berners-Lee [Berners-Lee et al., 2001], est de donner un sens aux données disponibles sur internet, pour favoriser le développement de services automatisés capables de les comprendre. La vision du web sémantique est donc de permettre à des applications d'accéder automatiquement à des ressources du web. Des standards, de nouveaux langages et de nouvelles technologies ont été définis. Ils devront permettre de trouver les ressources et/ou services recherchés beaucoup plus rapidement et facilement qu'aujourd'hui.

Les ontologies ont été reconnues comme une composante essentielle pour le partage des connaissances et la réalisation de la vision du web sémantique. En définissant des concepts, elles permettent à la fois de décrire le contenu de sources et d'expliciter le vocabulaire utilisable dans les requêtes des utilisateurs. Toutefois, il est peu probable qu'une unique ontologie couvre l'ensemble des sources sur lesquelles des applications sont développées. Dans la pratique, de nombreuses ontologies sont construites indépendamment les unes des autres par des communautés différentes. Comme les applications développées aujourd'hui nécessitent surtout le partage des données, il devient alors essentiel d'établir des correspondances sémantiques entre les ontologies qui les décrivent.

La tâche d'alignement d'ontologies (recherche de mappings, appariements ou mises en correspondance entre les concepts de deux ontologies) est particulièrement importante puisqu'elle autorise la prise en compte conjointe de ressources décrites par des ontologies différentes. Ce thème de recherche a donné lieu à de très nombreux travaux [Euzenat and Shvaiko, 2007]. Un certain nombre de ces travaux ont développé des systèmes d'alignement d'ontologies qui adaptent leur processus d'alignement en fonction des ontologies alignées. Ces systèmes d'alignement per-

mettent de choisir de manière manuelle ou automatique, l'outil d'alignement le plus adapté, la combinaison d'outils d'alignement la plus appropriée, et le réglage des paramètres (seuils, coefficient de formules, etc.) utilisés au sein des outils d'alignement mis en œuvre.

Problématique

Ces travaux de thèse sont centrés sur l'alignement d'ontologies mais contrairement à beaucoup d'autres, notre préoccupation n'a pas été de proposer un nouvel outil. Notre travail repose sur un outil d'alignement existant, TaxoMap. Nos propositions, faites dans le cadre de ce système mais généralisables, sont doubles.

Elles concernent d'une part l'adaptation du processus d'alignement aux caractéristiques des ontologies alignées, qu'il s'agisse de caractéristiques quantitatives telles que leur volume ou de caractéristiques particulières liées par exemple à la façon dont les labels des concepts sont construits. En effet, différents tests et campagnes d'évaluation [Euzenat et al., 2007] [Caracciolo et al., 2008] ont prouvé que les outils actuels d'alignement ne sont pas performants sur tous les domaines ni quelles que soient les ontologies. Ils sont très bons dans certains cas, moins bons dans d'autres. Notre travail répond, pour partie, à ce constat avec pour objectif d'aboutir à des résultats d'alignement de qualité.

D'autre part, nos travaux concernent l'utilisation, en ingénierie ontologique, des résultats d'alignement produits par des outils d'alignement. Nous proposons dans cette thèse une illustration de cette utilisation pour l'enrichissement d'ontologies.

Approches proposées

Pour répondre au fait que les outils d'alignement actuels ne peuvent pas s'adapter à toutes les spécificités des ontologies alignées, nous proposons de procéder en deux temps. Dans un premier temps, nous proposons de générer un alignement par application de techniques génériques mises en œuvre au sein d'un environnement permettant une grande souplesse d'utilisation et offrant un grand degré d'automatisation. Bien que cette première étape soit conçue pour offrir une grande flexibilité du point de vue de l'alignement mis en œuvre, les particularités des ontologies alignées ne peuvent néanmoins pas être considérées dans toute leur finesse. La phase d'alignement est alors complétée par une phase de raffinement des mappings générés. Cette phase est semi-automatique. Elle permet à l'expert du domaine des ontologies concernées, en collaboration avec l'ingénieur-informaticien, de spécifier, de façon déclarative, les modifications de mappings souhaitées. L'approche se veut être particulièrement utile en présence d'un nombre très important de mappings à examiner.

Pour l'alignement des ontologies de très grande taille, nous proposons deux méthodes de partitionnement d'ontologies qui prennent en compte dès le départ l'objectif d'alignement dans le processus de partition. Ces méthodes génèrent, en entrée du processus d'alignement, des sous ensembles de taille raisonnable des deux ontologies à aligner, ce qui permet d'améliorer et de rendre possible (en terme de temps d'exécution, de taille mémoire utilisée ou de la qualité des

résultats obtenus) leur alignement.

Concernant la réutilisation des résultats d'alignement pour l'ingénierie ontologique, notre travail s'est focalisé sur l'utilisation de l'alignement pour l'enrichissement d'ontologies. Nous proposons tout d'abord une approche d'enrichissement d'une ontologie (1) à partir d'une ontologie du même domaine et (2) à partir de parties de petite taille [1] extraites d'une ontologie généraliste. Nous proposons ensuite une seconde approche d'enrichissement à partir d'une ontologie très volumineuse (telle que YAGO) basée sur l'utilisation d'algorithmes de partitionnement d'ontologies.

Contributions

Ainsi, les contributions de notre travail sont les suivantes :

1. L'adaptation du système d'alignement TaxoMap [Kefi, 2006] pour qu'il soit le plus paramétrable et le plus modulaire possible. Les travaux d'extension de TaxoMap incluent le développement de nouvelles techniques d'alignement syntaxiques et structurelles.
2. La proposition d'un environnement permettant aux experts d'un domaine particulier de spécifier et d'effectuer des traitements de raffinement sur des alignements obtenus antérieurement. Nous avons montré comment cet environnement est utilisable pour raffiner un alignement généré par TaxoMap. Par ailleurs, cet environnement a été utilisé pour l'enrichissement d'ontologies.
3. La proposition de deux algorithmes de partitionnement d'ontologies qui permettent à l'outil TaxoMap d'aligner des ontologies volumineuses et généralistes.
4. Une implémentation des approches proposées dans cette thèse au sein de l'environnement TaxoMap Framework.
5. Un ensemble d'expérimentations réalisées grâce à l'environnement TaxoMap Framework. Ces expérimentations permettent de valider l'ensemble des propositions de cette thèse.

Plan du Manuscrit

Ce manuscrit comporte 6 chapitres.

Chapitre 1 : État de l'art

Ce premier chapitre est un état de l'art correspondant aux travaux que nous avons réalisés. Deux catégories de travaux sont décrites, les approches portant sur la prise en compte des spécificités des ontologies au sein même du processus d'alignement et les approches portant sur la prise en compte des spécificités des ontologies via des traitements portant sur un alignement préalablement généré. Dans ce chapitre, nous positionnons également nos travaux par rapport à l'existant.

1. Des ontologies de quelques dizaines de concepts

Chapitre 2 : TaxoMap : un module d'alignement générique, modulaire et paramétrable

Ce chapitre décrit les adaptations que nous avons apportées à l'outil d'alignement TaxoMap de façon à le rendre davantage modulaire, plus efficace et à améliorer la qualité des résultats. Nous décrivons, dans un premier temps, TaxoMap dans sa version courante V_2 en mettant en évidence les adaptations récentes réalisées dont nous avons eu la charge. Nous montrons ensuite que l'approche mise en œuvre dans cette version V_2 de l'outil est une approche modulaire et paramétrable.

Chapitre 3 : Vers un alignement adapté aux spécifications des ontologies alignées

Ce chapitre est dédié à l'approche de raffinement. Nous présentons tout d'abord l'architecture de l'environnement TaxoMap Framework qui permet la spécification de traitements de raffinement à partir de primitives prédéfinies. Nous présentons ensuite le langage de patrons MPL (the Mapping Pattern Language) que nous utilisons pour spécifier les traitements de raffinement. Enfin nous présentons des traitements de raffinement (patrons de raffinement) de mappings appliqués au domaine de la topographie dans le cadre du projet ANR GeOnto.

Chapitre 4 : Le partitionnement : un préalable à l'alignement de très grandes ontologies

Dans ce chapitre nous présentons un état de l'art correspondant aux travaux sur le partitionnement. Nous décrivons ensuite deux méthodes de partitionnement qui sont adaptées à l'alignement des ontologies de très grande taille. Ces méthodes prennent en compte au plus tôt l'information fournie par les *ancres* (les concepts des deux ontologies qui ont exactement les mêmes labels) pour l'utiliser dès le départ dans la phase de partitionnement, ce qui en fait des approches orientées alignement.

Chapitre 5 : L'alignement utilisable pour différentes tâches d'ingénierie ontologique : illustration sur la tâche d'enrichissement

Dans ce chapitre nous décrivons, dans un premier temps, un état de l'art portant sur l'enrichissement d'ontologies. Nous étudions ensuite l'apport des techniques d'alignement de TaxoMap pour l'enrichissement et l'impact des caractéristiques de la ressource externe utilisée comme source d'enrichissement sur l'efficacité de ces techniques. Dans les trois sections suivantes, nous présentons quelques-uns des traitements complémentaires identifiés comme nécessaires suivant les différents contextes d'enrichissement expérimentés : (1) pour une ontologie de même domaine et de même taille que l'ontologie à enrichir, (2) pour une ontologie de taille réduite et issue d'une source généraliste, (3) pour une ontologie généraliste très volumineuse.

Chapitre 6 : Développements et validations expérimentales

Dans ce dernier chapitre, nous présentons la façon dont l'environnement TaxoMap Framework a été implémenté. Ce prototype nous a permis de réaliser des expérimentations pour valider

nos propositions. Nous présentons ensuite certaines de ces expérimentations.

Enfin nous concluons et nous donnons quelques perspectives.

Chapitre 1

État de l'art

Sommaire

Introduction

Dans ce chapitre, nous présentons un état de l'art correspondant aux travaux que nous avons réalisés. Le problème principal auquel nous nous sommes intéressés est l'amélioration de l'interopérabilité sémantique en utilisant des techniques d'alignement d'ontologies. L'alignement d'ontologies est au centre de notre travail. Une première partie de cet état de l'art est ainsi consacrée à la présentation du processus d'alignement d'ontologies et des types de techniques qui nous ont intéressés. Ayant plus particulièrement étudié le problème d'alignement d'ontologies par rapport à la prise en considération des spécificités des ontologies à aligner, nous nous limitons ensuite à la description des travaux portant sur la découverte de mappings qui ont également cette préoccupation. Enfin, dans une dernière partie, nous positionnons notre travail par rapport à l'existant.

Notons que notre travail, bien que centré sur le problème de l'alignement, il touche aussi à d'autres domaines : le partitionnement d'ontologies et l'enrichissement d'ontologies. Des états de l'art sur ces différents aspects se trouvent respectivement dans les chapitres 4 et 5 de cette thèse.

1.1 L'alignement d'ontologies

Dans le contexte informatique de l'ingénierie des connaissances, une ontologie est définie comme un modèle structuré des objets d'un domaine d'application. Plus précisément, il s'agit d'une conceptualisation d'un domaine, comprenant la définition de ses concepts, de leurs propriétés et des relations qu'ils entretiennent entre eux.

[Euzenat and Shvaiko, 2007] définit une ontologie comme suit :

Définition 1 (Une ontologie) *Une ontologie est un tuple* $o = \langle C, I, R, T, V, \preceq, \perp, \in \rangle$ *tel que :*
C *est un ensemble de concepts*
I *est un ensemble d'instances*
R *est un ensemble de relations*
T *est un ensemble de types de données*
V *est ensemble de valeurs* (C, I, R, T, V *étant deux à deux disjoints)*
$\preceq$ *est une relation dans* $(C \times C) \cup (R \times R) \cup (T \times T)$ *dite de spécialisation*
$\perp$ *est une relation dans* $(C \times C) \cup (R \times R) \cup (T \times T)$ *dites d'exclusion*
$\in$ *est une relation sur* $(I \times C) \cup (V \times T)$ *dite d'instanciation*

Fournir les propriétés des concepts et les relations qu'ils ont avec d'autres concepts permet de les définir précisément. Cependant, une ontologie ne donne pas toujours ce niveau de détail. Ainsi, certaines ontologies se limitent à des hiérarchies de concepts, c'est-à-dire des ensembles de concepts uniquement reliés par des relations de subsomption, ce qui limite d'autant leur description.

L'alignement d'ontologies (recherche de mappings, appariements ou mises en correspondance) est aujourd'hui un problème bien connu. Ce thème de recherche, très actif, a donné lieu à de très nombreux travaux [Shvaiko and Euzenat, 2005]. Cette section sera tout d'abord consacrée à la définition de ce qu'est un alignement, en précisant notamment les étapes qui compose ce processus. Nous présenterons ensuite un état de l'art des techniques terminologiques et structurelles.

1.1.1 Le processus d'alignement d'ontologies

Soit la définition du processus d'alignement citée ci-dessous.

Définition 2 (Le processus d'alignement) *L'alignement d'ontologies est une fonction* f *qui, appliquée à deux ontologies* o *et* o'*, à un ensemble d'appariements initial* A*, à partir d'un ensemble de paramètres* p *s'appliquant aux techniques d'alignement mises en œuvre dans le processus et d'un ensemble de ressources externes* r*, produit un ensemble d'appariements* A' *entre les deux ontologies [Euzenat and Shvaiko, 2007].*

$$A' = f(o, o', A, p, r)$$

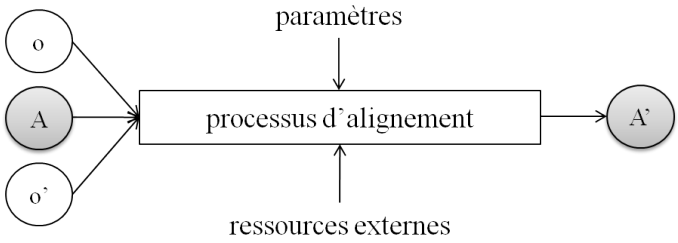

FIG. 1.1 – Processus d'alignement

Le processus d'alignement d'ontologies se compose de différentes étapes que l'on retrouve dans la majorité des systèmes implémentés [Ehrig, 2007]. Ces étapes sont schématisées dans la figure 1.2.

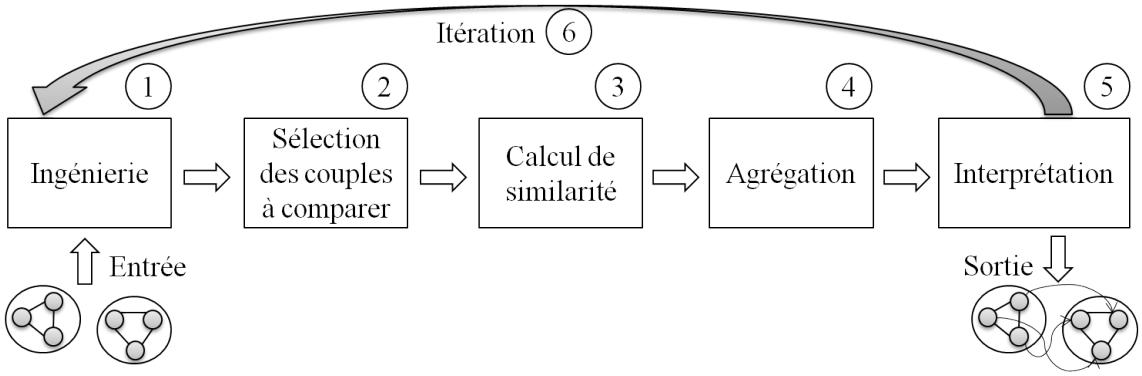

FIG. 1.2 – Les étapes du processus d'alignement

L'entrée du processus comprend deux ou plusieurs ontologies à aligner. En plus, comme la figure 1.1 le montre, elle peut comprendre des alignements.

L'étape d'ingénierie consiste à extraire des entrées, les données sur lesquelles les techniques d'alignement seront appliquées. Ces données seront ensuite comparées dans une étape ultérieure du processus d'alignement. Par exemple, les techniques de mise en correspondance peuvent n'exploiter qu'un sous-ensemble de primitives OWL qui peut représenter la taxonomie des concepts, ou même simplement les noms des éléments de l'ontologie.

L'étape de sélection de couples à comparer consiste à choisir les couples d'éléments à comparer dans le processus d'alignement. Dans cette étape, il est possible de ne considérer que certains couples et d'en ignorer d'autres. Généralement les outils d'alignement existant comparent tous les éléments d'une ontologie avec tous les éléments de l'autre, plus précisément, tous les concepts de l'une avec tous les concepts de l'autre, toutes les propriétés de l'une avec toutes les propriétés de l'autre et toutes les instances de l'une avec toutes les instances de l'autre.

Les éléments de chaque couple sélectionné dans l'étape 2 sont comparés dans la troisième étape du processus d'alignement. Cette comparaison renvoie une valeur numérique qui indique si les deux éléments ont un degré élevé ou faible de similitude. Il existe de nombreuses fonctions de similarité qui se basent souvent sur une distance pouvant être calculée de différentes

façons. Les techniques d'alignement qui exploitent les mesures de similarité sont appliquées sur les données extraites à l'étape 1. Selon [Euzenat and Shvaiko, 2007], on distingue les techniques terminologiques (exploitant les termes ou les labels des différentes composantes des ontologies), les techniques structurelles (exploitant la structure des ontologies), les techniques extensionnelles (exploitant les instances ou les données des ontologies) et les techniques sémantiques (exploitant les modèles des éléments considérés). Notons également que la similarité d'un couple d'éléments peut influencer le calcul de la similarité d'un autre couple d'éléments. C'est le cas, par exemple, de la similarité de deux classes qui peut dépendre de la similarité entre leurs instances. La mise en œuvre de telles approches, dites globales, peut alors passer par la propagation des mesures de similarité dans un graphe.

Plusieurs valeurs de similarité peuvent être calculées pour un même couple d'éléments, par exemple, une valeur de similarité terminologique entre leurs labels et une valeur de similarité structurelle prenant en compte leurs relations avec d'autres éléments. Ces différentes valeurs de similarité doivent être combinées dans la quatrième étape, afin de n'avoir à la fin qu'une seule valeur. L'agrégation de ces valeurs est une fonction difficile à définir, car elle suppose d'avoir fixé la valeur des paramètres donnant un poids plus ou moins fort aux mesures combinées.

Enfin, l'étape d'interprétation du processus d'alignement consiste à obtenir des alignements entre les éléments des ontologies à partir des valeurs de similarité individuelles ou agrégées préalablement calculées. Les similarités sont alors interprétées. Les approches les plus connues utilisent les seuils ou combinent les critères des différentes techniques d'alignement. Un alignement (ou non) entre les couples d'éléments sélectionnés est proposé. Lorsqu'une approche globale est adoptée, il est nécessaire de procéder par itérations. La similarité d'un couple d'éléments est calculée à chaque itération en tenant compte de la similarité des couples d'éléments voisins.

De nombreux systèmes d'alignement adoptent ce processus d'alignement en utilisant des techniques d'alignement variées. Dans ce qui ce suit, nous ne présentons que les techniques terminologiques et structurelles que nous utilisons dans notre travail de thèse, en nous basant sur la classification proposée par [Euzenat and Shvaiko, 2007].

1.1.2 Techniques d'alignement terminologiques et structurelles

Cette section est limitée à la présentation des techniques terminologiques et structurelles car nos travaux s'appuient sur l'outil d'alignement TaxoMap et seuls ces deux types de techniques composent cet outil. En effet, TaxoMap se donne pour objectif d'aligner des ontologies dites légères correspondant à des taxonomies. Nous ne disposons pas d'instances et la description des concepts est limitée au contenu de taxonomies.

Techniques terminologiques

Les techniques terminologiques comparent les noms, les labels et les commentaires utilisés pour décrire les éléments composant les ontologies. On distingue deux familles : les techniques syntaxiques et les techniques linguistiques.

Techniques syntaxiques : Ces techniques sont basées sur la comparaison des chaînes de caractères. Deux éléments sont considérés comme sémantiquement proches si les termes qui les

désignent sont syntaxiquement proches. Il existe plusieurs techniques de comparaison de chaînes de caractères se basant sur le calcul d'une distance. Selon le cas, la chaîne de caractères est considérée comme une suite exacte de caractères, une séquence erronée de lettres, un ensemble de caractères ou de mots. Nous présentons, dans ce qui suit, quelques exemples de ces techniques :

- **distance d'édition** (ou distance de Levenshtein [Levenshtein, 1966]) : la distance d'édition $ed(s_1, s_2)$ est le coût minimal des opérations d'édition nécessaires pour transformer s_1 en s_2. Un coût est ainsi associé à chaque opération d'édition de chaînes (insertion, suppression et modification d'un caractère) sauf s'il s'agit d'une substitution de caractères identiques.

- **mesure de Lin** [Lin, 1998] : la mesure de Lin est une adaptation de la distance de Jaccard [Jaccard, 1901] à la technique des n-grammes [Shannon, 1948].
  La mesure de Jaccard calcule la similarité entre deux ensembles d'éléments en rapportant le nombre d'éléments communs au nombre total d'éléments appartenant aux deux ensembles. Une valeur comprise entre 0 et 1 est obtenue, où 1 correspond aux ensembles identiques.
  En ce qui concerne la technique des n-grammes, la similarité est calculée entre deux chaînes de caractères, en comparant des sous-chaînes de n caractères. Ces dernières sont obtenues par le déplacement, sur ces chaînes, d'une fenêtre de n cases. Plusieurs travaux [Langdon, 2000] [Bonafonte and Mariño, 1996] [Siu and Ostendorf, 2000] se sont intéressés à la détermination d'une valeur optimale de n. Les expérimentations effectuées dans [Langdon, 2000] ont montré que les meilleurs résultats sont obtenus pour $n = 3$. On parle alors de 3-grammes ou tri-grammes.
  La mesure de Lin évalue la similarité entre deux chaînes de caractères, en calculant le ratio entre le nombre de tri-grammes communs de ces deux chaînes sur leur probabilité d'apparition dans le corpus étudié.

  Définition 3 (Mesure de Lin) *Soient $tri(x)$ l'ensemble des tri-grammes de la chaîne de caractères x et $P(t)$ la probabilité d'apparition d'un tri-gramme t*

$$Sim_{Lin}(x, y) = 2 \frac{\sum_{t \in (tri(x) \cap tri(y))} logP(t)}{\sum_{t \in tri(x)} logP(t) + \sum_{t \in tri(y)} logP(t)}$$

Techniques linguistiques : les techniques linguistiques utilisent des méthodes d'analyse linguistique (TALN) et des ressources externes pour évaluer la similarité entre les labels de deux éléments.
La comparaison des formes standards des labels, obtenus par l'application de méthodes d'analyse linguistique (lemmatisation, étude morphosyntaxique, normalisation, etc.), permet de déterminer leur degré de similarité. Ces méthodes d'analyse permettent aussi l'identification et l'interprétation des préfixes et suffixes dans un label. Ceci peut identifier les liens entres les différents termes de ce label.
L'identification des liens sémantiques existant entre les différents termes à l'aide de ressources linguistiques (dictionnaire, thésaurus) est très utile dans le cas, par exemple, où les termes des labels comparés sont synonymes. Le problème qui peut se poser est la transformation des liens sémantiques entre des termes en liens sémantiques entre des concepts. En effet, deux concepts décrits par des termes synonymes ne sont pas nécessairement équivalents.

Techniques structurelles

Les techniques structurelles consistent à exploiter la structure des éléments devant être mis en correspondance. Il existe deux types de structures : une structure interne qui définit un élément sans faire référence aux autres éléments de l'ontologie, et une structure externe qui définit les relations qu'a un élément avec les autres éléments de l'ontologie.

Structure interne : elle représente les attributs (nom, multiplicité et type de donnée) et les relations (nom, cardinalités, domaine et portée) d'un concept. La comparaison de deux concepts revient à la comparaison de leurs attributs et de leurs relations. Les techniques exploitant la structure interne sont généralement combinées avec d'autres techniques, telles que des techniques terminologiques.

Structure externe (relationnelle) : une ontologie peut être considérée comme un graphe dont les arêtes sont étiquetées par des noms de relation. Identifier les correspondances entre les éléments des deux ontologies correspond à la recherche d'un homomorphisme de graphes [Garey and Johnson, 1979].
Dans la pratique, les techniques exploitant la structure externe, utilisent un ensemble initial de correspondances entre les deux ontologies ou une ontologie de référence. Ceci permet de représenter les ontologies en un seul graphe. Ainsi, la comparaison de deux éléments consiste à mesurer la distance qui les sépare dans ce graphe. Il existe de nombreuses fonctions pour calculer cette distance. Nous présentons, dans ce qui suit, quelques exemples :

- **Wu-Palmer** [Wu and Palmer, 1994] : la mesure de Wu-Palmer prend en compte le fait que si le plus petit généralisant commun de deux éléments est proche de la racine en terme de nombre d'arcs, les deux éléments peuvent être très différents sémantiquement. A l'inverse, deux éléments dont le plus petit généralisant commun est très éloigné de la racine devraient être sémantiquement proches. Cette mesure est définie comme suit :

  Définition 4 (similarité de Wu-Palmer) *Soit $\delta(c_1, c_2)$ le nombre d'arcs séparant les concepts c_1 et c_2, soit $c_1 \wedge c_2$, le plus petit généralisant commun de c_1 et c_2 et soit r la racine de l'ontologie.*

  $$WP(c_1, c_2) = \frac{2 \times \delta(c_1 \wedge c_2, r)}{\delta(c_1, c1 \wedge c_2) + \delta(c_2, c1 \wedge c_2) + 2 \times \delta(c_1 \wedge c_2, r)}$$

- **généralisants/spécialisants de concepts similaires** [Dieng and Hug, 1998] : lorsque deux concepts sont similaires, leurs généralisants et spécialisants respectifs sont également similaires. Cette mesure pose problème dans le cas de multiples spécialisants/généralisants.

Ces techniques sont utilisées au sein d'approches mises en œuvre dans les outils d'alignement. Dans la section qui suit, nous décrivons les approches et outils d'alignement proposant des moyens spécifiques pour prendre en compte les spécificités des ontologies à aligner.

1.2 Approches et outils d'alignement prenant en compte les spécificités des ontologies

Beaucoup d'outils d'alignement existant aujourd'hui génèrent de bons résultats dans certains cas et de moins bons dans d'autres, ces résultats étant fonction des caractéristiques des ontologies alignées. Ce constat oriente les recherches dans trois directions principales : le choix de la technique d'alignement la plus adaptée, la combinaison des techniques d'alignement la plus appropriée, et le réglage des paramètres (seuils, coefficient de formules, etc.) utilisés au sein des techniques d'alignement mises en œuvre.

A côté des caractéristiques des ontologies influant sur la stratégie d'alignement à adopter, en particulier les types d'éléments qui les composent (concepts, relations, propriétés, axiomes), il est, par ailleurs, nécessaire de considérer leur taille. La mise en correspondance des entités de deux ontologies peut, en effet, être inefficace, voire impossible à effectuer, lorsque les ontologies rapprochées sont très volumineuses. Pour résoudre ce problème, il est nécessaire de limiter la taille des ontologies à aligner. Ceci peut se faire en faisant précéder l'alignement d'un processus de partitionnement des ontologies rapprochées. Ce processus génère des sous-parties d'ontologies pouvant être alignées séparément. La taille de ces parties est en général paramétrable et peut donc être adaptée aux outils d'alignement utilisés. Nous présentons au début du chapitre 3 de cette thèse un état de l'art sur le partitionnement d'ontologies.

Nous nous focalisons dans cette partie sur les approches d'alignement permettant l'adaptation du processus d'alignement à un contexte d'application particulier. Nous présentons tout d'abord les approches qui prennent en compte les différentes spécificités des ontologies à aligner au sein même du processus d'alignement. Les outils implémentant ces approches offrent la possibilité de choisir manuellement ou automatiquement les techniques d'alignement à exécuter, leur combinaison et les différents paramètres (seuils, coefficients, etc.). Nous présentons ensuite les approches qui prennent en compte les spécificités des ontologies à aligner via des traitements effectués sur des alignements préalablement générés.

1.2.1 Approches portant sur la prise en compte des spécificités des ontologies au sein même du processus d'alignement

Dans le processus d'alignement (cf. Fig. 1.2), avant que les résultats générés soient interprétés comme des correspondances entre les éléments des ontologies alignées, deux phases importantes sont réalisées. La première phase (phase 3) consiste à choisir les techniques d'alignement à appliquer sur l'ensemble des éléments des deux ontologies alignées et la façon de les exécuter. La deuxième phase (phase 4) consiste à agréger les résultats issus de la phase 3. C'est au niveau de ces deux phases du processus d'alignement que les spécificités des ontologies peuvent être prises en compte. De nombreux outils d'alignement [Do and Rahm, 2007] [Castano et al., 2003] [Sayyadian et al., 2005] mettent en œuvre des stratégies permettant de choisir de manière manuelle, semi-automatique ou automatique (1) les techniques de mise en correspondance (2) leur combinaison et (3) les paramètres les plus appropriés. Il existe aussi des outils qui permettent de définir des patrons pour générer des mappings complexes, dans le cas de rapprochement d'ontologies contenant des éléments qui ne sont pas au même niveau de granularité. Nous présentons, dans un premiers temps, les approches de sélection et de combinaison des techniques d'alignement. Nous présentons ensuite, les approches permettant de choisir les paramètres de ces techniques.

Enfin, nous présentons les approches générant des mappings complexes.

Approches de sélection des techniques d'alignement

Les approches de sélection des techniques d'alignement à appliquer sont basées sur la prise en compte de plusieurs critères [Huza et al., 2006], [Mochol et al., 2006] :

- ***Les caractéristiques et l'usage de la technique utilisée*** **:** [Rahm and Bernstein, 2001] classe les techniques d'alignement en deux catégories : les techniques basées sur l'utilisation d'un algorithme simple générant des mappings en prenant en compte un seul critère d'alignement (par exemple les noms des concepts), et les techniques basées sur la combinaison des algorithmes simples soit par la prise en compte de plusieurs critères d'alignement (par exemple les noms des concepts et les instances), soit par la combinaison des résultats générés par les différents algorithmes simples, exécutés de manière indépendante. La sélection de l'approche utilisée (simple ou combinée) dépend de plusieurs facteurs [Mochol et al., 2006] tels que le temps d'exécution, la qualité des résultats (par exemple choisir une technique simple pour diminuer le temps d'exécution, ou combiner plusieurs techniques pour améliorer les résultats d'alignement).
  En ce qui concerne l'usage de la technique utilisée, l'application de certains algorithmes peut ne pas avoir le même effet, en terme de qualité de mappings produits, sur un domaine et sur un autre [Giunchiglia and Shvaiko, 2003]. Ainsi, avant d'utiliser une approche d'alignement, il est important de savoir si elle est adaptée ou non au domaine d'application.

- ***Les caractéristiques des ontologies à aligner*** **:** en terme de taille, type (schéma de base de données, présence d'instances), structure (une structure sous forme d'arbre, de graphe, composée uniquement de relations de subsomption ou d'un ensemble de relations sémantiques dont le sens est uniquement donné par le nom des relations), format de représentation (une représentation informelle, semi-formelle ou formelle). L'alignement d'ontologies très volumineuses nécessite, par exemple, le choix de techniques d'alignement adaptées au traitement de gros volumes de données.

- ***Les résultats désirés*** **:** l'alignement souhaité peut porter sur l'intégralité ou uniquement des parties des ontologies. Les techniques sélectionnées peuvent aussi dépendre de la nature du résultat d'alignement souhaité : un alignement 1:1 associant un élément d'une ontologie avec un autre de l'autre ontologie ou un alignement 1:n associant un élément d'une ontologie dite source avec plusieurs autres éléments de l'autre ontologie, dite cible ou encore un alignement n :m associant un élément de l'ontologie source avec plusieurs autres éléments de l'ontologie cible et inversement.

La prise en compte des critères énoncés ci-dessus se fait au travers du choix des techniques et des paramètres, qui peut être manuel ou automatique. Lorsque ce choix est automatique, il fait partie intégrante de la définition de la stratégie d'alignement du système. Nous décrivons les travaux rentrant dans cette catégorie dans la partie suivante traitant de la combinaison des techniques d'alignement. Lorsque ce choix est manuel, c'est l'utilisateur qui en est pleinement responsable. Des aides sont utiles. Or, la plupart des prototypes d'alignement développés n'offrent pas d'aide ou seulement une interface graphique rudimentaire. Ainsi, le choix manuel des techniques d'alignement et leur configuration sont des tâches qui restent aujourd'hui difficiles. A notre

connaissance, les seules aides à la configuration du processus d'alignement qui existent sont des interfaces graphiques telles celles proposées dans COMA++ [Do, 2005] et AgreementMaker [Cruz et al., 2009].

L'approche d'alignement mise en œuvre dans COMA++ [Do, 2005] fournit à l'utilisateur une interface graphique lui permettant, (1) de choisir et configurer les techniques qu'il veut utiliser pour aligner ainsi que de la façon de les combiner, (2) de demander à raffiner les résultats générés et (3) de valider les mappings obtenus. COMA++ dispose d'une bibliothèque de techniques présentées à l'utilisateur via son interface graphique. L'utilisateur peut créer une nouvelle technique d'alignement en choisissant, dans cette bibliothèque, les techniques d'alignement simple la constituant ainsi que la stratégie de combinaison à appliquer. Il peut choisir de garder les paramètres par défaut proposés pour chaque technique et pour le processus de combinaison ou les modifier.

L'approche d'alignement mise en œuvre dans AgreementMaker [Cruz et al., 2009] fournit, elle aussi, à l'utilisateur une interface graphique lui permettant de sélectionner les techniques d'alignement et leur combinaison manuellement. L'interface permet, par ailleurs, d'afficher simultanément les mappings générés par différents techniques. Ceci simplifie la comparaison des résultats. L'utilisateur peut, de cette façon, plus facilement choisir les techniques les plus adaptées.

Enfin, dans l'approche d'alignement mise en œuvre dans RiMOM (Risk Minimization based Ontology Mapping) [Li et al., 2009], le choix des techniques peut se faire manuellement, en laissant à l'utilisateur la possibilité de choisir en fonction des spécificités des ontologies qu'il veut aligner, ou bien de manière totalement automatique. Dans ce deuxième cas, afin de comparer les caractéristiques des deux ontologies alignées, RiMOM calcule un facteur lexical et un facteur structurel. Le facteur lexical résulte d'une comparaison des noms des éléments (concepts ou propriétés) et le facteur structurel est issu d'une comparaison des hiérarchies de concepts associées aux ontologies alignées. Si les deux ontologies ont un facteur lexical élevé (supérieur à un certain seuil) et un facteur structurel bas (inférieur à un certain seuil), RiMOM donne plus de poids aux techniques terminologiques exploitant les noms des éléments. Dans le cas contraire, RiMOM donne plus de poids aux techniques structurelles.

Approches de combinaison des techniques d'alignement

Les stratégies d'alignement mises en œuvre dans les outils d'alignement sont de différents types. Les techniques d'alignement peuvent être appliquées séquentiellement ou en parallèle. Lorsque plusieurs techniques sont appliquées à un même couple d'éléments comparés ou lorsque les techniques sont appliquées sur plusieurs caractéristiques des éléments des ontologies (leurs noms, leurs propriétés, les axiomes les concernant), les résultats obtenus doivent ensuite être combinés, c'est l'objet de la phase d'agrégation (phase 4 Fig. 1.2).

L'application séquentielle des techniques d'alignement consiste à exécuter une technique puis une autre (cf. Fig. 1.3). Par exemple, il s'agira d'appliquer, en premier lieu, une (ou des) technique(s) terminologique(s) puis, en second lieu, une (ou des) technique(s) structurelle(s). Ces dernières peuvent être appliquées à tous les couples d'éléments comparés ou uniquement aux couples d'éléments dont les techniques terminologiques n'ont pas permis de déterminer la simi-

larité avec une précision suffisamment grande.

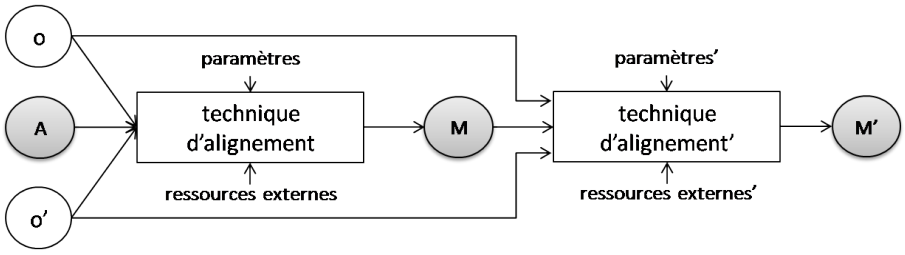

FIG. 1.3 – La composition séquentielle des techniques d'alignement (M représente la matrice de similarité) (d'après [Euzenat and Shvaiko, 2007])

Lorsque les techniques sont appliquées en parallèle, elles sont considérées de manière indépendante (cf. Fig. 1.4) et appliquées à l'ensemble des couples d'éléments des ontologies alignées. Les résultats obtenus par chacune des techniques doivent ensuite être agrégés. Les fonctions d'agrégation utilisées peuvent être, par exemple, le maximum, la moyenne arithmétique ou une moyenne pondérée.

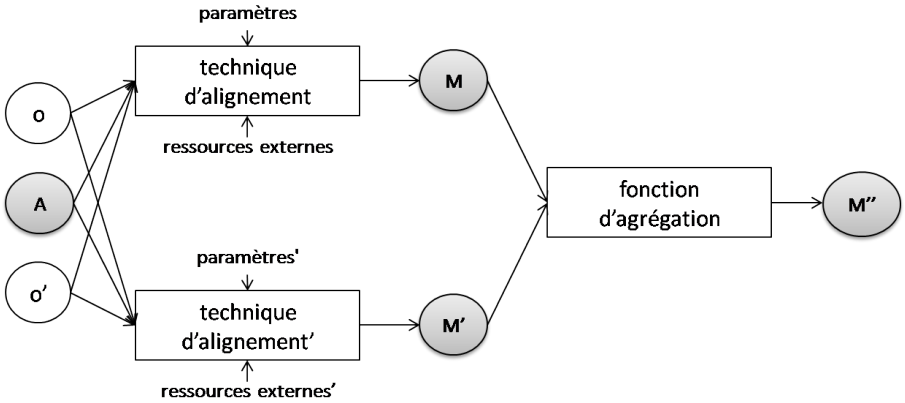

FIG. 1.4 – La composition parallèle des techniques d'alignement (d'après [Euzenat and Shvaiko, 2007])

Lorsque les éléments des ontologies ont plusieurs caractéristiques, les techniques d'alignement sont tout d'abord appliquées sur chacune des caractéristiques, puis les résultats sont agrégés de façon à obtenir une seule valeur. Ainsi, la similarité entre deux éléments peut être obtenue par agrégation des valeurs de similarité de leurs noms et de leurs propriétés. Par exemple, COMA++ [Do and Rahm, 2007] exploite les fonctions d'agrégation suivantes : max, min, moyenne arithmétique et pondérée. La moyenne pondérée consiste à donner un poids à chaque technique en fonction de son importance.

La majorité des outils d'alignement qui existent [Euzenat and Shvaiko, 2007] applique l'une de ces deux stratégies (techniques exécutées de façon séquentielle ou en parallèle), ou combine les deux. Ces stratégies sont parfois très complexes mais elles sont souvent définies une fois pour toutes, quelles que soient les ontologies à aligner.

D'autres travaux ont cherché à offrir plus de flexibilité dans la définition de la stratégie de combinaison des techniques applicables. Cette définition peut être manuelle ou automatique.

Dans le premier cas, les outils d'alignement offrent à l'utilisateur un ensemble de techniques d'alignement. Celui-ci peut, à l'aide d'une interface graphique, sélectionner les techniques qu'il souhaite appliquer et la façon de les combiner (avec des méthodes de combinaison prédéfinies dans ces outils). C'est le cas des outils Rondo [Melnik, 2004], COMA++ [Do and Rahm, 2007], AgreementMaker [Cruz et al., 2009] et Protégé [Noy, 2004] qui offrent un environnement permettant à l'utilisateur de choisir, parmi les stratégies de combinaison implémentées dans ces outils, une stratégie qu'il souhaite appliquer.

Dans le cas d'une définition automatique, les techniques d'alignement à appliquer sont en général fixées mais la façon de les combiner est déterminée dynamiquement en fonction des ontologies alignées et / ou les résultats générés par les techniques au cours du processus d'alignement.

Ainsi, dans Falcon-AO [Jian et al., 2005], deux techniques d'alignement sont systématiquement appliquées pour générer des alignements : une technique linguistique LMO (Linguistic Matching for Ontologies) basée sur la comparaison des labels des éléments des ontologies alignées, et une technique structurelle GOM (Graph Matching for Ontologies) [Hu et al., 2005] basée sur la comparaison de similarités structurelles et qui exploite les mappings générés par LOM et/ou fournis en entrée de la technique. Le poids relatif des résultats générés par ces deux techniques fait l'objet d'une évaluation dynamique en début de traitement. Cette évaluation est basée sur le calcul d'un coefficient de fiabilité, obtenu par une comparaison linguistique et structurelle des deux ontologies alignées. La comparaison linguistique est basée sur le calcul du rapport entre les candidats au mappings et le nombre d'éléments des deux ontologies, tandis que la comparaison structurelle considère le nombre d'occurrences des propriétés prédéfinies des deux ontologies (par exemple rdf:type, rdfs:subClassOf et owl:onProperty). Falcon-AO considère que la similarité linguistique est un peu plus fiable que la similarité structurelle. Ainsi, les mappings générés par LOM sont toujours acceptés. Lorsque le coefficient de fiabilité linguistique est élevé et que le coefficient de fiabilité structurelle est faible, seuls les mappings générés par GOM avec une similarité élevée sont considérés comme fiables et acceptés. En revanche, lorsque le coefficient de fiabilité linguistique est faible, tous les mappings générés par GOM sont acceptés.

L'approche mise en œuvre dans CMC [Tu and Yu, 2005] combine les différentes techniques d'alignement en se basant sur des prédictions. La précision de chaque technique est prédite soit par l'utilisation de règles manuelles, dans le cas où la précision peut être prédite de manière intuitive, soit par un processus d'apprentissage automatique. Chaque technique d'alignement est ainsi associée à un prédicteur de crédibilité qui calcule, de façon dynamique en utilisant des vecteurs de caractéristiques (tels que la longueur et le nombre de mots des noms d'éléments), un poids (une crédibilité) pour chaque résultat généré par cette technique. Les résultats des différentes techniques sont ensuite combinés à l'aide d'une fonction d'agrégation en se basant sur leur crédibilité.

Dans l'algorithme mis en œuvre dans H-Match [Castano et al., 2003], qui est une extension du système d'intégration de données ARTEMIS, la mise en correspondance des concepts est basée sur la comparaison de leur description et de leur contexte. Le contexte d'un concept (cf. Fig. 1.5) est défini comme l'ensemble des ses propriétés et de ses concepts adjacents (les concepts liés par une relation sémantique).

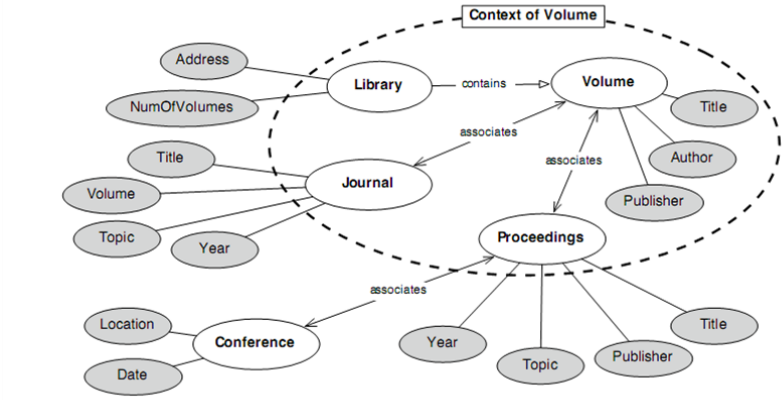

FIG. 1.5 – Exemple de contexte pour le concept *Volume* [Castano et al., 2003]

Dans H-Match, l'évaluation de la similarité sémantique entre concepts est basée sur l'évaluation de l'affinité linguistique et contextuelle. L'affinité linguistique entre concepts consiste à mesurer la similarité linguistique des éléments intervenant dans leur description. Un thesaurus (WordNet) est utilisé pour connaître les relations linguistiques existant entre les différents éléments (relations de synonymie par exemple). L'affinité contextuelle (cf. Fig. 1.6) évalue les contextes. Elle calcule l'affinité linguistique des propriétés composant les contextes ainsi que des concepts adjacents, et évalue le degré de proximité entre les relations sémantiques impliquées dans les contextes des concepts comparés.

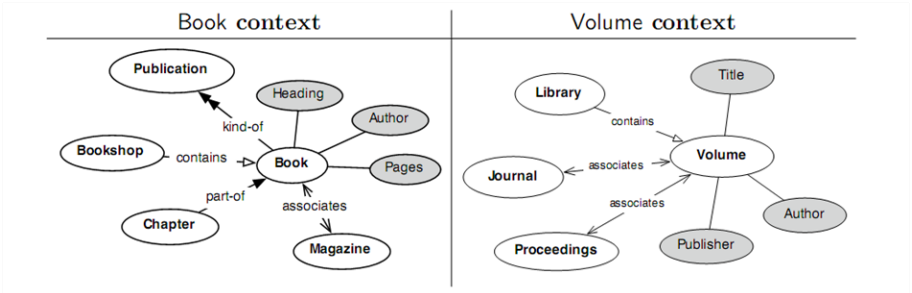

FIG. 1.6 – Exemple de comparaison des contextes des concepts *Book* et *Volume*

L'algorithme d'alignement mis en œuvre dans H-Match est paramétrable. Il permet de donner un poids plus ou moins fort aux aspects linguistiques et contextuels et ainsi de limiter ou ignorer la notion de contexte associé aux concepts. H-match définit trois modèles d'alignement :

– ***Shallow*** **:** L'appariement est effectué avec seulement les informations linguistiques fournies par les noms des concepts et par le thésaurus de référence. La précision de l'appariement dépend du choix des noms des concepts dans la définition des ontologies, qui doivent être significatifs et précis, pour obtenir des bons résultats. Ce modèle d'appariement est performant car il nécessite peu de temps de calcul.

– ***Intermediate*** **:** L'analyse linguistique est étendue aux propriétés des concepts. Ces informations complémentaires sont exploitées pour avoir des résultats plus précis que ceux du modèle "shallow".

– ***Deep*** **:** Ce modèle d'appariement prend en compte tout le contexte des concepts, c'est-à-dire les noms des concepts, les propriétés, les concepts adjacents et les relations avec ces concepts. Les résultats générés avec ce modèle sont d'une grande précision mais le temps de calcul est plus important.

Le modèle appliqué est choisi en fonction de la richesse des descriptions des concepts comparés et de leurs relations avec les autres concepts de l'ontologie.

Approches permettant de choisir les paramètres des techniques d'alignement

Dans un environnement dynamique tel que le web, les changements dans les données traitées sont fréquents. L'adaptation automatique des techniques d'alignement pour gérer ce changement a donc une grande importance. Cette adaptation consiste à reconfigurer la (ou les) technique(s) appliquée(s) en trouvant les meilleures valeurs pour les seuils, les coefficients, etc. utilisés en phase d'exécution. Certains travaux, comme ceux décrits dans [Lee et al., 2007], proposent des approches de reconfiguration d'une technique d'alignement en phase de conception. Pour une adaptation aux spécificités des ontologies alignées, il doit toutefois être possible de reconfigurer une technique au moment de son exécution. C'est un challenge auquel les travaux de [Ehrig et al., 2005] et de [Sayyadian et al., 2005] répondent. Nous les décrivons dans ce qui suit.

L'outil APFEL (Alignment Process Feature Estimation and Learning) [Ehrig et al., 2005] utilise pour celà des techniques d'apprentissage automatique basées sur la validation des alignements par les utilisateurs, pour configurer automatiquement les paramètres (les poids et les seuils) d'une technique d'alignement.

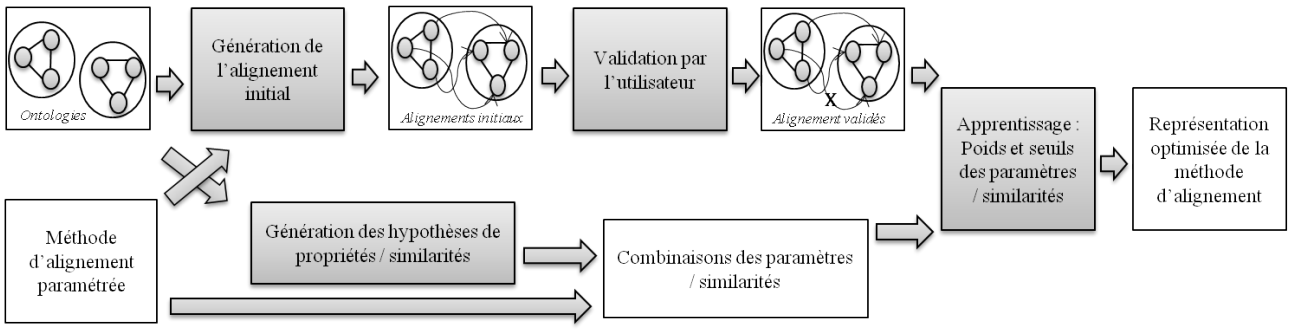

FIG. 1.7 – Architecture du système d'alignement APFEL [Ehrig et al., 2005]

APFEL utilise des représentations déclaratives pour décrire le processus d'alignement mis en œuvre (cf. Fig. 1.7) :

– la nature des données traitées (les labels des concepts, leur structure, les types de relations,

la présence d'instances et les caractéristiques du domaine d'application).

- le mode de calcul des mesures de similarité entre concepts.
- les poids utilisés pour l'agrégation des différentes mesures.
- la stratégie d'interprétation des mesures de similarité conduisant ou non à la proposition de mappings.

Il comprend une interface (PAM : Parametrizable Alignment Methods) qui permet à l'utilisateur de définir les différents outils d'alignement à utiliser, de façon à prendre en compte toutes les étapes du processus d'alignement. L'outil d'alignement RiMOM peut, par exemple, être choisi. Un premier alignement est généré et porté à la connaissance des utilisateurs pour qu'ils le valident. L'analyse de l'alignement validé couplée à l'utilisation d'une technique d'apprentissage automatique (par exemple : réseaux de neurones, arbres de décision, machines à vecteurs de support) implémentées dans l'environnement d'apprentissage automatique WEKA [2], permet d'affiner les paramètres utilisés dans RiMOM.

eTuner [Sayyadian et al., 2005] permet également le réglage automatique des paramètres utilisés par un outil d'alignement. L'alignement, dans ces travaux, porte sur des schémas de bases de données. Les mappings générés sont de type 1:1. Dans eTuner, l'outil d'alignement est vu comme un modèle M = <L, G, K> avec :

- L : une bibliothèque qui contient des composantes de l'outil d'alignement :
  - cinq algorithmes d'alignement : Les deux premiers comparent les noms d'attributs (n-gram et TF/IDF) [Do and Rahm, 2002][Doan et al., 2001] et les trois autres exploitent les instances [Doan et al., 2001].
  - une composante qui combine les similarités générées par l'outil d'alignement (par exemple, faire la moyenne entre la valeur maximale et minimale des différentes mesures calculées).
  - les contraintes ou les heuristiques pré-spécifiées du domaine (par exemple, "Adresse postale" ne peut être aligné qu'avec un seul élément).
  - Une composante qui génère les mappings, par exemple par application d'un seuil sur les mesures de similarité.
- G : un graphe orienté, dont les nœuds représentent les éléments de M et les arêtes, les flux d'exécution entre les composantes. Le graphe est constitué de plusieurs niveaux et doit être bien formé : (a) les composantes de plus bas niveau doivent être les algorithmes d'alignement qui prennent comme entrée les schémas à aligner, (b) la composante de plus haut niveau doit être celle qui génère les mappings en interprétant les valeurs de similarités calculées entre les éléments, et (c) toutes les entrées des composantes doivent être définies.
- K : un ensemble de boutons à régler, appelés boutons de configuration.

2. `http://www.cs.waikato.ac.nz/~ml/weka/`

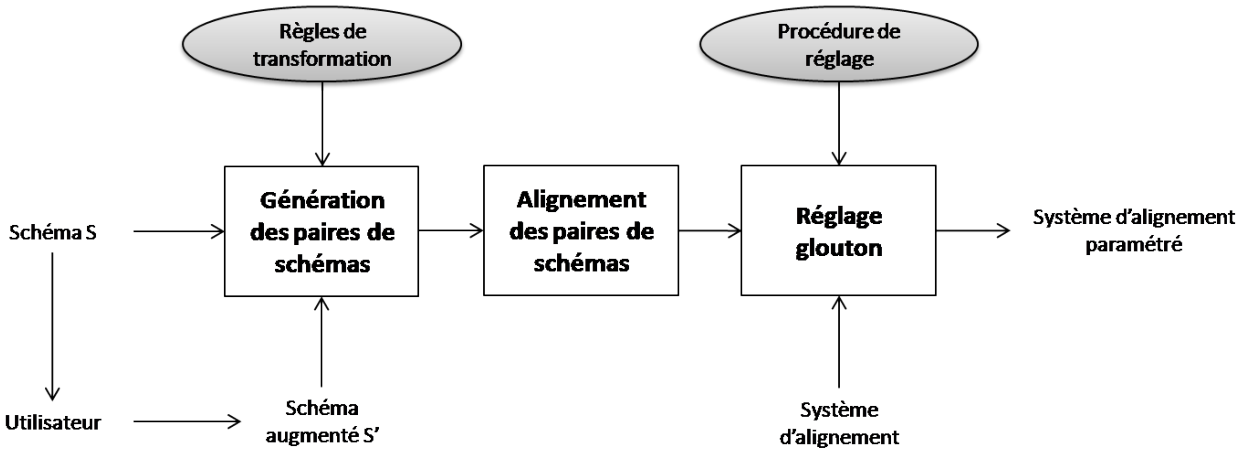

FIG. 1.8 – Architecture du système d'alignement eTuner (d'après [Sayyadian et al., 2005])

L'outil eTuner fonctionne en trois phases (cf. Fig. 1.8). Pendant la première phase, ce système génère un ensemble de jeux de tests constitués de paires de schémas de base de données à comparer. Ces schémas sont générés à partir d'un schéma initial S (il s'agit, par exemple, de considérer deux schémas de relation au départ identiques dont des noms d'attributs ont été remplacés dans l'un des deux par des synonymes). Ces paires sont ensuite mises en correspondance. L'alignement généré est analysé de façon à proposer, dans un troisième temps, des modifications des valeurs des paramètres des techniques d'alignement appliquées.

Comme l'espace des configurations des paramètres peut être grand, le processus de réglage des paramètres repose sur une approche gloutonne. Des propositions de modification des valeurs des paramètres sont faites pour chaque technique considérée isolément puis le système traite des valeurs utilisées dans le processus de combinaison.

Approches générant des mappings complexes

La prise en compte des spécificités des ontologies alignées peut aussi consister à s'apercevoir que le niveau de granularité des ontologies à aligner n'est pas équivalent et qu'il est, de ce fait, nécessaire de pouvoir générer des mappings dits complexes. Ce problème existe aussi en base de données [Doan and Halevy, 2005]. Par exemple, l'attribut nom dans un schéma est équivalent à la concaténation des attributs prénom et nom dans un autre schéma. C'est un problème complexe qui se pose aussi dans le domaine de l'alignement d'ontologies [Shvaiko and Euzenat, 2008]. La plupart des outils d'alignement qui existent actuellement génèrent seulement des correspondances simples entre deux éléments atomiques. Ces correspondances ne permettent pas d'aligner des ontologies dont les éléments sont décrits à des niveaux de granularité différents.

Les travaux portant sur les mappings complexes portent soit sur leur identification, soit sur leur représentation.

Les travaux décrits dans [Scharffe, 2009] décrivent les différentes sortes de mappings complexes que l'on peut rencontrer en alignement d'ontologies sous forme de patrons de correspondance. De tels patrons peuvent aussi être définis par les experts d'un domaine. Etant génériques,

ils sont ré-utilisables et permettent la génération de mappings complexes. En plus, comme [Ritze et al., 2009] le montre, les patrons de correspondance peuvent être exploités par des techniques d'alignement automatique.

Une autre approche d'identification de correspondances complexes, inspirée des travaux de [Scharffe, 2009], est proposée dans [Sváb-Zamazal and Svátek, 2009]. Cette approche, également à base de patrons, exploite la structure des ontologies. Après l'identification des patrons à l'aide d'un langage de requêtes et des méthodes terminologiques, les structures des ontologies alignées sont comparées et des correspondances entre entités complexes sont générées.

Enfin, des travaux identifient des correspondances complexes via des approches d'apprentissage automatique. Dans [Qin et al., 2007], l'identification des correspondances complexes entre les éléments des ontologies est basée sur l'étude des instances associées à ces éléments. Le problème est que des concepts équivalents de deux ontologies différentes peuvent être associés à des ensembles disjoints d'instances. Cette question est abordée dans [Ritze et al., 2009] qui propose d'utiliser un alignement de référence composé de relations simples d'équivalence et de subsomption pour identifier des correspondances complexes à l'aide de patrons.

1.2.2 Approches portant sur la prise en compte des spécificités des ontologies via des traitements portant sur un alignement préalablement généré

Les mappings obtenus par application des techniques d'alignement peuvent être modifiés pour prendre en compte les spécificités des ontologies alignées, non considérées jusqu'alors. Le processus d'alignement est alors suivi d'un processus de raffinement dont l'objectif est d'améliorer la précision des mappings. Nous présentons, dans un premiers temps, les approches de raffinement de mappings par un processus itératif et les approches basées sur la prise en compte des différents types d'utilisateurs. Nous présentons ensuite, les approches de raffinement de mappings basées sur l'analyse des données peuplant les ontologies. Enfin, nous présentons les approches se donnant pour objectif le débogage de mappings.

Approches de raffinement de mappings par un processus itératif

Parmi les travaux ayant adopté ce type d'approche, citons ceux décrits dans [Do and Rahm, 2002] [Do, 2005] concernant l'approche COMA++. COMA++ aligne des ontologies en combinant différentes techniques d'alignement et raffine ensuite, dans un second temps, les résultats obtenus considérés comme préliminaires. Le processus de raffinement est totalement automatique. Il s'agit de ré-appliquer le processus d'alignement sur des groupes d'éléments dont la proximité a été préalablement établie.
Comme les résultats générés par cette approche sont de type n :m, les mappings impliquant les mêmes éléments sont regroupés dans un même ensemble. Ce regroupement permet d'éviter le traitement d'un élément plusieurs fois. Par exemple, la correspondance établie entre l'élément "Adresse" et "Adresse de facturation" et regroupée avec celle établie entre "Adresse" et "Adresse de livraison". L'approche détermine ensuite les éléments de la source et de la cible pour chaque groupe de mappings. Dans l'exemple précédent, l'ensemble des éléments de la source contient "Adresse", et celui des éléments de la cible contient "Adresse de facturation" et "Adresse de livraison". Le processus d'alignement est appliqué ensuite sur les éléments ainsi regroupés afin

de générer de nouveaux mappings.

Approches de raffinement des mappings basées sur les utilisateurs

Tout utilisateur n'est pas forcément intéressé par les mêmes mappings. Par ailleurs, les relations de similarité acceptables peuvent dépendre du point de vue selon lequel un utilisateur se place ou de l'usage qui sera fait des mappings. Certaines approches prennent en compte cet état de fait.

Les travaux décrits dans [Zhdanova and Shvaiko, 2006] prennent en compte les différents types d'utilisateurs pouvant être intéressés par les mappings. Ils proposent une approche dite "publique", où un agent humain (qui peut correspondre à une communauté d'utilisateurs) ou logiciel peut aligner des ontologies et stocker les mappings de telle sorte qu'ils soient réutilisables par d'autres agents. Pour réaliser cet objectif, l'approche appelée "alignement d'ontologies guidé par une communauté" est proposée [Zhdanova and Shvaiko, 2006]. Il s'agit d'une extension du processus d'alignement d'ontologie traditionnel du fait de l'implication, dans le processus d'alignement même, des utilisateurs finaux, des ingénieurs de la connaissance et des communautés de développeurs. Toutes ces personnes peuvent, en effet, souhaiter réutiliser des mappings. L'approche se définit comme suit :

- Elle prend en entrée (1) les éléments (classes, propriétés, relations) des deux ontologies à aligner et (2) les informations (le contexte, les données personnelles) propres à un agent, par exemple un utilisateur final. L'approche exploite les informations d'entrée citées ci-dessus mais également des données provenant de l'analyse de réseaux sociaux ou du web.

- Le système qui met en œuvre cette approche génère des mappings annotés, qui sont propagés, à l'aide d'un processus de recommandation, aux communautés auxquelles appartient l'utilisateur qui a demandé l'alignement. Les utilisateurs qui appartiennent à ces communauté peuvent ensuite réutiliser ou modifier les mappings manuellement.
  Le processus de recommandation de mappings pour les utilisateurs et les communautés dépend de l'analyse du profils de ces derniers, par exemple, de leurs activités dans un domaine, l'histoire de leurs collaborations, leurs relations dans un réseau social.

Approches de raffinement de mappings basées sur les données

Une autre façon de raffiner des mappings peut être d'analyser les données peuplant les ontologies. Ainsi, Muse [Alexe et al., 2008], un outil établissant des correspondances entre schémas de bases de données utilise des échantillons de données pour aider un concepteur à comprendre des mappings et à les modifier. Muse comprend deux principales composantes, **Muse-G** et **Muse-D** :

- **Muse-G** permet au concepteur de regrouper les mappings selon une sémantique désirée. Il s'agira, par exemple, de regrouper les employés par nom et adresse ou seulement par nom [3]. Pour aider à réaliser ce regroupement, **Muse-G** génère automatiquement des exemples de données (d'instances) en se basant sur les mappings. La validation de ces exemples par

3. [Fuxman et al., 2006] ont démontré que le regroupement sémantique des mappings est important dans le cas de données imbriquées et dans le cas de schémas qui évoluent.

l'expert permet de déduire la sémantique du regroupement souhaitée.

- **Muse-D** est dédié au traitement des mappings ambigus (par exemple : les correspondances établies entre "nom du manager" et "superviseur" ou entre "nom du chef de projet" et "superviseur"). Un ensemble de données illustrant toutes les interprétations est généré et montré au concepteur. Les actions de ce dernier sur les données permettront à **Muse-D** de déterminer l'interprétation la plus pertinente.

Ces deux composantes peuvent être utilisées indépendamment pour raffiner des mappings générés manuellement ou automatiquement. Elles peuvent également être utilisées ensemble pour aider un concepteur à spécifier les mappings souhaités à partir de mappings préalablement générés et de données.

Débogage de mappings

Les approches se donnant pour objectif le débogage de mappings consistent à détecter les bugs au sein d'un mapping et à les réparer. Les techniques de débogage ne correspondent pas à de nouvelles approches d'alignement. Elles les complètent.

Dans [Wang and Xu, 2008a], une classification des bugs les plus communs rencontrés lors de la génération automatique de mappings entre deux ontologies, est présentée. Les bugs sont classés en quatre catégories :

- **Les mappings redondants :** ces mappings peuvent être inférés à partir d'autres mappings. Ils ne sont pas toujours inutiles, car s'ils sont stockés, cela évite de les recalculer à chaque fois qu'ils doivent être utilisés. Ceci peut être intéressant si le mécanisme d'inférence à appliquer est coûteux.

- **Les mappings imprécis :** ces mappings peuvent être produits dans le cas où, par exemple, l'algorithme d'alignement ne génère que des correspondances simples alors que des correspondances complexes, seraient parfois plus adaptées. Certains mappings simples générés peuvent, dans ce cas, être considérés comme approximatifs. Par exemple : les correspondances d'équivalences établies entre "organisme" et "organisme commercial", ou entre "organisme" et "organisme de recherche" sont imprécis, dans la mesure où les concepts "organisme commercial" et "organisme de recherche" sont disjoints. Ces deux correspondances peuvent être remplacées par un mappings complexe qui établit une relation entre "organisme" et ("organisme commercial" ou "organisme de recherche").

- **Les mappings incohérents :** ces mappings peuvent créer des incohérences par rapport à la structure et aux axiomes des ontologies. Par exemple considérons qu'une correspondance soit établie entre les éléments e_1 et e_2, s'il existe des mappings entre les frères de e_1 et les petits fils de e_2, ces derniers seront considérés comme incohérents par rapport à la structure des ontologies.

- **Les mappigns anormaux :** Ces mappings "suspects" n'appartiennent pas aux trois catégories ci-dessus. Le terme "suspect" est subjectif et dépend fortement de l'application qui va utiliser le mapping ou de l'utilisateur final. Des mappings peuvent être considérés

comme "suspects" ou anormaux quand ils relient des éléments proches les uns des autres dans une ontologie, avec des éléments éloignés les uns des autres dans l'autre ontologie.

Tous ces types de mappings sont signalés à l'utilisateur, soit comme des erreurs (mappings incohérents) soit comme des avertissements (mappings redondants, anormaux ou imprécis). Les mappings redondants, incohérents et anormaux ne peuvent être réparés que manuellement. Les mappings imprécis, qui sont souvent ambigus, peuvent être débogués à l'aide d'heuristiques générant des mappings complexes.

1.3 Positionnement de notre travail par rapport à l'état de l'art

Les approches d'alignement d'ontologies, décrites précédemment, ont pour objectif l'amélioration des résultats d'alignement en choisissant, par exemple, les techniques d'alignement les plus adaptées, la meilleure combinaison de ces techniques, et les paramètres (seuils, coefficients utilisés dans les formules, etc.) les plus appropriés. Deux types d'approches ont été distingués : les approches qui prennent en compte les différentes spécificités des ontologies à aligner au sein même du processus d'alignement et les approches qui prennent en compte les spécificités des ontologies à aligner via des traitements effectués sur des alignements préalablement générés.

Notre travail s'insère dans le deuxième type d'approche. Un de nos objectifs est le raffinement de mappings. Nous supposons, en effet, qu'un module d'alignement doit rester général. Nous ne souhaitons pas développer d'outil dédié à des ontologies particulières ni même à des ontologies appartenant à un domaine d'application particulier. De ce fait, nous considérons que le processus d'alignement lui-même n'est pas suffisant pour prendre en compte les spécificités des ontologies alignées. Il doit être complété. C'est pour cette raison que le raffinement de mappings est vu, dans notre proposition, comme un processus post-alignement. Toutefois, à la différence des travaux décrits dans l'état de l'art, nous ne nous limitons pas à cet objectif de raffinement de mappings. Notre ambition est d'aller au-delà. Nous souhaitons proposer une approche pour le raffinement de mappings qui soit ré-utilisable pour tous les traitements, quels qu'ils soient, reposant sur l'exploitation de mappings, comme l'enrichissement, la restructuration ou la fusion d'ontologies.

Concernant le processus de raffinement de mappings, une idée forte de notre travail est de considérer que le raffinement des mappings nécessite une très bonne connaissance des ontologies alignées. Nous faisons donc une place importante aux experts du domaine des ontologies alignées, des personnes qui ont une grande connaissance du sens des concepts représentés. Ces experts sont au centre du processus que nous proposons. Ils ont un rôle à jouer même si nous souhaitons, par ailleurs, les aider au maximum en automatisant le plus possible le processus. Pour que ces experts puissent intervenir efficacement dans le processus de raffinement, nous proposons une approche offrant un environnement qui leur permette de spécifier de manière déclarative des traitements portant sur des mappings. Certains travaux cités dans l'état de l'art s'appuient sur l'utilisateur mais aucun ne leur offre réellement un environnement d'aide à la définition des traitements de raffinement à exécuter.

L'approche de raffinement de mappings, ou de façon plus générale, d'exploitation de mappings, que nous proposons repose sur un alignement préalablement généré à l'aide d'une approche

d'alignement mise en œuvre dans TaxoMap. L'approche d'alignement implémentée dans cet outil fait également partie de nos contributions. Elle est caractérisée par sa modularité et son adaptabilité. En effet, même si un outil d'alignement ne peut pas s'adapter à 100% aux caractéristiques des ontologies traitées, il est important toutefois de pouvoir choisir les techniques appliquées, leurs paramètres (seuils, coefficients) et la façon de les exécuter. Cette flexibilité et l'aide fournie à l'utilisateur pour effectuer ses choix peuvent être rapprochées de celles offertes par certains systèmes d'alignement cités dans l'état de l'art comme COMA++ [Do, 2005] et AgreementMaker [Cruz et al., 2009]. En effet, tout comme ces systèmes, TaxoMap comporte également une bibliothèque de techniques. Il dispose, par ailleurs, d'une interface graphique permettant à l'utilisateur de choisir et de configurer les techniques d'alignement qu'il veut utiliser. Par comparaison aux systèmes précédemment cités, la bibliothèque de techniques accessible dans TaxoMap nous semble toutefois plus facile à enrichir du fait de l'indépendance des techniques les unes par rapport aux autres. Ces techniques sont, par ailleurs, toutes simples. Elles ne résultent pas d'une combinaison de plusieurs autres techniques. Ceci facilite leur compréhension.

Notre objectif est également d'être capable de gérer de grandes masses de données, en l'occurrence ici, de raffiner de grandes quantités de mappings. Aucune approche de raffinement citée dans l'état de l'art n'a cet objectif.

Conclusion

Ce chapitre a eu pour objectif de dresser un état de l'art sur les travaux effectués en alignement d'ontologies qui permettent d'adapter manuellement ou automatiquement, selon différents critères, le processus d'alignement. Nous avons ensuite positionné notre travail de thèse par rapport à cet existant en mettant en avant les problématiques centrales qui nous ont intéressés et auxquelles nous avons apporté des solutions. Les chapitres suivants décrivent nos différentes contributions.

Chapitre 2

TaxoMap : un module d'alignement générique, modulaire et paramétrable

Sommaire

Introduction

Nos travaux prennent appui sur l'outil d'alignement TaxoMap, développé dans l'équipe Leo/IASI du LRI. Une première version V_0 de TaxoMap a fait l'objet d'une partie de la thèse d'Hassen Kefi [Kefi, 2006] soutenue en mars 2006. Ces travaux avaient été motivés par le projet ANR RNTL eDot (Entrepôt de Données ouvert sur la Toile) dont l'objectif était le développement de technologies permettant la construction, la plus automatique possible, d'entrepôts de données thématiques alimentés en documents depuis le web. Le domaine d'application était le risque bactériologique de contamination d'aliments. Le travail de thèse d'Hassen Kefi avait consisté à permettre l'interrogation unifiée de deux bases de données couvrant le même domaine d'application à partir d'une unique interface. Ces bases de données avaient été développées indépendamment l'une de l'autre et possédaient chacune leur propre ontologie. Les utilisateurs étaient habitués à formuler leurs requêtes uniquement dans le vocabulaire de l'une d'elles. Une exigence du projet a été de conserver ce mode d'interrogation. L'ontologie offrant le vocabulaire d'interrogation a été appelée ontologie cible, par rapport à l'autre ontologie appelée ontologie source. TaxoMap est donc un outil d'alignement d'ontologies, orienté d'une ontologie source vers une ontologie cible.

Nous décrivons ci-dessous TaxoMap dans sa version courante V_2 en mettant en évidence les adaptations récentes réalisées au cours de cette thèse. Dans la seconde partie, nous montrons que l'approche mise en œuvre dans cette version V_2 de l'outil est une approche modulaire et paramétrable. Une interface graphique a été développée pour permettre à l'utilisateur de tirer parti de ces caractéristiques et l'aider à définir le processus d'alignement qu'il souhaite exécuter. Des copies d'écran provenant de cette interface illustrent notre propos.

2.1 Description de TaxoMap

TaxoMap est un module d'alignement générique qui s'applique à tout couple de taxonomies comprenant des concepts auxquels sont associés des labels et des relations entre concepts qui sont essentiellement des relations de subsomption.

L'approche générale mise en œuvre a été conçue par Hassen Kefi dans ses travaux de thèse. Des adaptations ont ensuite été régulièrement réalisées, d'une part lors de son intégration dans la plate-forme WebContent [Euzenat et al., 2008], d'autre part lors de notre participation chaque année depuis 2007 à la campagne d'évaluation OAEI et dans le cadre de travaux réalisés au sein du projet ANR GeOnto [Mustière et al., 2011]. Dans le cadre de travaux de cette thèse, nous avons eu la charge des adaptations nécessaires pour participer à OAEI ainsi que des travaux sur TaxoMap réalisés dans le cadre du projet GeOnto.

Nous donnons ci-dessous une description fonctionnelle de TaxoMap. Les sections suivantes font état des adaptations que nous avons réalisées : l'intégration de TreeTagger, la mise à jour de la mesure de similarité utilisée, l'adaptation et l'ajout de techniques d'alignement composant TaxoMap.

2.1.1 Description générale fonctionnelle de TaxoMap

Nous décrivons ci-dessous TaxoMap d'un point de vue fonctionnel (cf. Fig. 2.1).

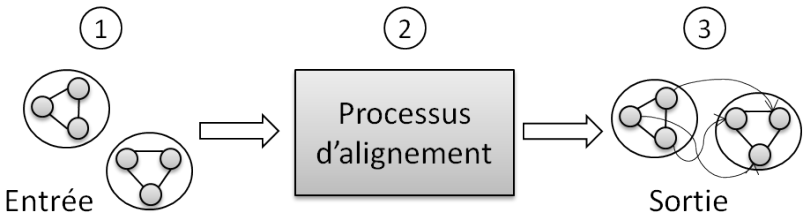

FIG. 2.1 – Les étapes du processus d'alignement

1. **Les entrées** : TaxoMap prend en entrée deux taxonomies. Les taxonomies sont des ontologies très sommaires avec des définitions de concepts très pauvres. Les concepts sont principalement définis par les labels qui leur sont associés (expressions qui peuvent être composées de plusieurs mots) et par les relations de subsomption qui le relient à d'autres concepts.

TaxoMap s'applique à tout couple de taxonomies correspondant aux caractéristiques suivantes :

– des taxonomies dissymétriques : une taxonomie pouvant être très peu structurée (voire pas du tout) dite "taxonomie source", alors que la seconde dite "taxonomie cible" l'est davantage. Cette spécificité de TaxoMap a été établie dans le cadre du projet ANR RNTL eDot, où les ontologies alignées n'étaient pas identiques du point de vue de leur structure.

– des taxonomies très spécifiques comportant des descriptions très fines du domaine d'application, ce qui se traduit au niveau des concepts par (1) des concepts appartenant à un même domaine circonscrit (par exemple, le domaine géographique), (2) des labels correspondant à des expressions composées de plusieurs mots (par exemple, le label "hôtel de montagne isolé situé dans le parc national"), (3) des labels de concepts généraux inclus dans les labels de concepts plus spécifiques (par exemple, "route" inclus dans "route départementale").

2. **Le processus d'alignement** : les étapes du processus d'alignement de TaxoMap sont les suivantes :

   – Normalisation des labels de concepts qui consiste à remplacer les signes de ponctuation et à reconnaître quelle est la racine d'un mot et sa forme canonique. Elle consiste aussi à identifier les mots importants (les noms) et les mots moins importants (par exemple, les adjectifs) dans un label (cf. 2a Fig. 2.2).

   – Calcul d'une mesure de similarité basée sur la mesure de Lin (cf. Chapitre 1) exploitant l'outil d'analyse morpho-syntaxique TreeTagger pour prendre en compte l'importance des mots dans un label (cf. 2b Fig. 2.2). Étant donné un concept de l'ontologie source, cette mesure permet d'identifier l'ensemble des concepts de l'ontologie cible candidats à un mapping avec ce concept.

   – Application séquentielle d'un ensemble de techniques exploitant la mesure de similarité (cf. 2c Fig. 2.2) précédemment calculée. Ces techniques d'alignement permettent de sélectionner le concept le plus pertinent pour établir un mapping, parmi l'ensemble des candidats. Les mappings proposés ne résultent de l'application que d'une technique donnée et d'une seule. Une première technique est appliquée sur l'ensemble des couples de concepts. Une seconde technique est appliquée sur l'ensemble des couples pour lesquels un alignement n'a pas été trouvé à l'aide de la première technique, etc. L'ordre d'application des techniques est, dans ce cas, très important. Dans TaxoMap, il s'agit d'un paramètre que l'utilisateur, à travers une interface graphique, peut modifier pour tenir compte des spécificités des ontologies à aligner.

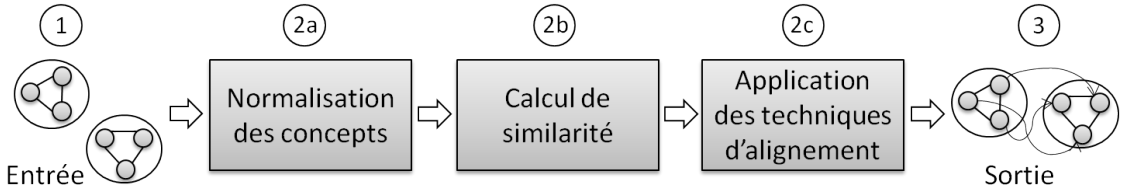

FIG. 2.2 – Les étapes du processus d'alignement

3. **Les sorties** : TaxoMap génère un alignement composé de relations de mise en correspondance, appelées mappings, de type 1:n (c'est-à-dire un concept de l'ontologie source n'est aligné qu'avec un seul concept de la cible). Les relations générées sont des relations d'équivalence, de spécialisation, de généralisation ou de proximité, auxquelles sont associées des mesures de similarité et le nom des techniques qui ont permis de les obtenir. Les mappings sont générés dans trois formats différents :
   - le format d'alignement [Euzenat, 2004] (Fig. 2.3) utilisé comme standard dans la compagne d'évaluation OAEI.

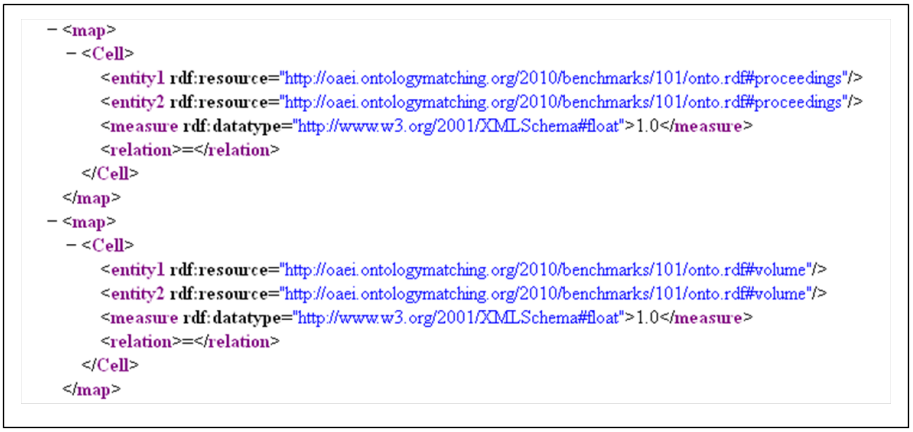

FIG. 2.3 – Format d'alignement OAEI

   - le format OAEI étendu (Fig. 2.4) dans lequel nous ajoutons l'information concernant les noms des techniques qui ont généré des mappings. L'objectif est de permettre à un expert de facilement identifier les mappings générés par les différentes techniques de façon à ce qu'il puisse très facilement concevoir les traitements à réaliser pour les raffiner si besoin (cf. Chapitre 3 de cette thèse).

FIG. 2.4 – Format d'alignement OAEI étendu

   - le format texte généré dans le but d'offrir à l'expert des sorties plus compréhensibles. Ce format désigne les concepts par leurs labels.

     Il est représenté de la façon suivante :

```
proceedings  isEq   proceedings (Sim: 1.0)
url          isEq   link        (Sim: 1.0)
```

Les mappings peuvent être acceptés ou refusés par l'expert lorsqu'il analyse les résultats générés mais il s'agit déjà de mappings. Ce n'est pas à l'expert, à partir d'une relation de mise en correspondance trouvée entre deux concepts et d'une relation de confiance, de dire si la relation correspond ou non à un mapping. Ce traitement est fait automatiquement. Pour étudier les mappings générés, l'expert peut utiliser l'outil AlignViz [4], un outil développé dans le cadre du projet GeOnto (cet outil est décrit en annexe B). AlignViz prend en entrées les deux ontologies rapprochées, ainsi que la liste des mappings produits par TaxoMap. Il permet de visualiser à la fois les deux ontologies, en présentant de chaque côté de l'écran les deux hiérarchies de classes correspondantes, et en bas de l'écran, la liste des mappings générés. Une interface graphique d'aide à la validation des mappings permet ensuite à l'expert de valider ou pas un mapping et éventuellement d'introduire un commentaire.

2.1.2 L'intégration de TreeTagger

Dans le processus d'alignement de TaxoMap, la phase dite "de normalisation de concepts" est importante pour les techniques d'alignement comparant des labels de concepts. Nous proposons une amélioration importante au niveau de cette phase en intégrant TreeTagger [5], un étiqueteur morpho-syntaxique, qui permet un paramétrage en fonction de la langue, une lemmatisation et une catégorisation des mots qui composent les labels.

Chacun des mots d'un label est ainsi étiqueté par sa catégorie morpho-syntaxique (nom, adjectif, adverbe, verbe, préposition, article, pronom, conjonction), puis lemmatisé, c'est-à-dire mis sous sa forme canonique, l'infinitif pour les verbes et le masculin singulier pour les noms et adjectifs. Ainsi, pour le label "hôtel de montagne isolé situé dans le parc national", l'analyse de TreeTagger donne :

hôtel	NOM	hôtel
de	PRP	de
montagne	NOM	montagne
isolé	ADJ	isolé
situé	VER :PPER	situer
dans	PRP	dans
le	DET :ART	le
parc	NOM	parc
national	ADJ	national

Dans la version V_0 de TaxoMap, la phase de lemmatisation, qui était basée sur l'utilisation du thésaurus WordNet, était inefficace voire impossible à effectuer lorsque les ontologies rapprochées étaient très volumineuses. En plus, la qualité des résultats n'était pas garantie (les mots pouvant ne pas se trouver dans WordNet). Avec l'intégration de l'étiqueteur morpho-syntaxique TreeTagger, ce traitement devient beaucoup plus rapide et les résultats plus exhaustifs et plus précis.

4. http://www.lri.fr/~hamdi/AlignViz/AlignViz.html
5. http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/

Une fois que l'ensemble des mots des labels des concepts d'une ontologie a été étiqueté par TreeTagger, nous avons introduit une étape d'étude automatique de la cohérence des différentes étiquettes. En effet, TreeTagger est un outil statistique et les labels des concepts d'une ontologie étant relativement courts, les erreurs d'étiquetage ne sont pas rares et peuvent perturber la suite du traitement. Par exemple, le même mot "rocher" peut avoir été étiqueté dans le label L_x comme la forme infinitive du verbe "rocher" et dans un autre label, comme la forme au singulier du nom commun "rocher". En cas de conflit de ce type, si un même mot apparaît dans différents labels avec des étiquettes distinctes, le principe est d'homogénéiser les étiquettes en donnant la priorité à la reconnaissance des noms. Ainsi, le "rocher" du label L_x sera finalement étiqueté lui aussi comme un nom.

La dernière étape consiste à répartir les différents mots d'un label en deux classes, *mot plein* ou *mot complémentaire*, en fonction de leur catégorie et de leur position relative dans le label. A priori, tous les noms sont des *mots pleins* sauf s'ils sont placés derrière une préposition et tous les autres mots sont classés comme *complémentaires*. Dans l'exemple "hôtel de montagne isolé situé dans le parc national", "hôtel" est le seul *mot plein*, puisque les deux autres noms du label, "montagne" et "parc", sont tous les deux placés derrière une préposition.
Dans le cas où tous les mots d'un label (un label français composé de un ou plusieurs mots) sont des *mots complémentaires*, le premier mot sera considéré comme un *mot plein*.

Cette répartition, entre *mot plein* et *mot complémentaire*, est ensuite utilisée dans le calcul de la similarité entre concepts.

2.1.3 La mesure de similarité utilisée dans TaxoMap et son adaptation

Dans TaxoMap, une similarité, $sim(c_s, c_t)$, est calculée entre les différents labels des concepts des deux ontologies pris deux à deux, où c_s appartient à l'ontologie source O_S et c_t appartient à l'ontologie cible O_T. Cette similarité s'appuie sur la mesure de Lin [Lin, 1998], qui mesure la similarité entre deux éléments, x et y, en se basant sur le nombre de tri-grammes partagés par les labels de ces deux éléments (cf. Chapitre 1).

TaxoMap adapte cette mesure pour prendre en compte l'importance relative des mots dans les labels et donner plus de poids aux *mots pleins*. Nous considérons que l'ensemble des tri-grammes communs aux concepts x et y se partitionnent en deux sous-ensembles, nommés respectivement dans la formule, $Inter$ et I'. L'ensemble $Inter$ contient les tri-grammes communs extraits à partir des mots considérés comme des *mots pleins* alors que l'ensemble I' comprend les autres. Un coefficient, dont il est possible de faire varier la valeur, permet dans la formule d'affecter à I' un poids plus faible que celui affecté à $Inter$. Si le poids de $Inter$ est 2 (coefficient existant dans la formule de Lin), celui de I' pourra être fixé à une valeur différente inférieure, par exemple 0.5, comme dans la formule ci-dessous.

$$Sim_{LinLike}(x, y) = \frac{2 * \sum_{t \in Inter} logP(t) + 0.5 * \sum_{t \in I'} logP(t)}{\sum_{t \in tri(x)} logP(t) + \sum_{t \in tri(y)} logP(t)}$$

Dans la version V_0 de TaxoMap, les mots *complémentaires* étaient tout d'abord identifiés par

l'expert du domaine. TaxoMap ajoutait ensuite une liste d'adjectifs générés automatiquement dans la phase de "normalisation de concepts", en repérant les mots se terminant par "ed". Ce traitement était très élémentaire et ne s'appliquait que dans un contexte très particulier (labels en anglais, présence d'adjectifs avec "ed" à la fin). En plus, la tâche d'identification manuelle était impossible dans le cas où le nombre de labels est très important.

Notons de plus que dans la version V_0 si un même mot n'était pas classé de la même façon dans deux labels, il n'était pas pris en compte par la mesure de similarité. Par exemple, le mot "parc" étant considéré comme un *mot plein* dans le label "parc national" et comme un *mot complémentaire* dans le label "hôtel de montagne isolé situé dans le parc national", les tri-grammes de ce mot, pourtant présents dans les deux labels, n'étaient pas comptabilisés.

L'outil TreeTagger, que nous utilisons dorénavant permet d'identifier totalement automatiquement les *mots pleins* et les *mots complémentaires* sont identifiés automatiquement à l'aide de leur catégorie morpho-syntaxique.

La répartition a priori des mots d'un label L_i en *mots pleins* et *mots complémentaires* doit être modulée en fonction des mots présents dans le label L_j auquel L_i est comparé, dans le calcul de similarité. Nous avons, pour cela, effectué les adaptations suivantes :

1. Lors du calcul de similarité de deux labels L_i et L_j, si l'intersection des deux labels contient déjà au moins un *mot plein*, les autres mots communs aux deux labels (ou les fractions de mots) qui partagent au moins leurs deux premiers tri-grammes sont considérés comme des *mots pleins* (ou fractions de *mots pleins*) même s'ils étaient a priori classés comme *complémentaires*.
   Exemple : Quand "voie ferrée de transit" est comparé à "voie ferrée", l'adjectif "ferrée" qui apparaît associé au même *mot plein* "voie" dans les deux labels, est considéré lui aussi comme un *mot plein*, alors qu'il serait considéré comme *complémentaire* dans une comparaison avec "ligne ferrée".
   De même, quand "hôtel régional" est comparé à "hôtel de région", les deux *mots complémentaires* "régional" et "région" sont considérés comme des *mots pleins* car le mot "hôtel" est un *mot plein* dans les deux labels, et "régional" et "région" partagent les mêmes premiers tri-grammes. Cette répartition permet de donner une similarité plus forte à la comparaison de "hôtel régional" avec "hôtel de région" plutôt qu'avec "hôtel".

2. Les *mots pleins* d'un premier label L_i, qui ne partagent leur premier tri-gramme avec aucun autre *mot plein* d'un deuxième label L_j, sont pénalisés et réciproquement.

   Soient : $L_i = \{mot_1\_plein, mot_2\_complementaire, mot_3\_plein, ...\}$
   $L_j = \{mot'_1\_plein, mot'_2\_plein, mot'_3\_complementaire, ...\}$
   $Inter = mots\_pleins(L_i) \cap^* mots\_pleins(L_j)$
   où $mots\_pleins$ est une fonction qui extrait les *mots pleins* d'un label et $\cap^*$ calcule l'ensemble des *mots pleins* qui partagent le même premier tri-gramme
   $\forall x \in mots\_pleins(L_i)$, si $x \notin Inter \Rightarrow x.type = mot\_complementaire$
   $\forall x \in mots\_pleins(L_j)$, si $x \notin Inter \Rightarrow x.type = mot\_complementaire$

   La modulation de la répartition des mots d'un label en fonction des mots présents dans le

label auquel il est comparé, a deux avantages :

- En pénalisant les similarités entre des *mots pleins* qui ne partagent pas leur premier trigramme, elle permet d'augmenter la précision de l'alignement en éliminant des mappings faux. Exemple : "carrière" et "barrière", "gave" et "aven".

- Le déclassement des *mots pleins* en *mots complémentaires* s'ils n'ont pas de tri-grammes partagés peut faire apparaître des similarités intéressantes même si elles sont très faibles. Exemple : le cas de "douane" et "poste de douane". Sans la modulation, la présence commune du mot "douane" n'est pas détectée puisqu'il est un *mot plein* dans le premier label et un *mot complémentaire* dans le second, et la similarité entre ces deux labels est égale à zéro. Après le passage du *mot plein* "douane" vers l'ensemble des *mots complémentaires*, l'augmentation de la valeur de similarité permettra d'identifier un mapping.

La version V_0 de TaxoMap supposait qu'un concept n'était associé qu'à un seul label. Nous avons proposé une adaptation de la mesure de similarité pour prendre en considération les multi-labels et gérer, ainsi, la synonymie. Un concept peut donc être caractérisé par plusieurs labels qui peuvent être des synonymes ou des variantes du même terme. Ainsi, le calcul de la similarité s'effectue entre tous les labels des concepts, ce qui améliore par la suite la précision des résultats obtenus.

2.1.4 Les techniques d'alignement composant TaxoMap

La version V_1 de TaxoMap (version intégrant l'analyseur morpho-syntaxique TreeTagger) comportait 8 techniques d'alignement. Des techniques exploitant la ressource complémentaire WordNet ont été supprimées au cours des premières adaptations, car leur utilisation s'est avérée inefficace voir impossible dans le cas d'ontologies volumineuses. En plus la qualité des résultats de mappings dépendait de la présence des labels comparés dans WordNet.

Nous avons rajouté à cette version de TaxoMap deux techniques d'alignement (les techniques T_9 et T_{10}) permettant d'identifier de nouveaux mappings. Ces deux techniques sont basées sur les mappings d'équivalence préalablement identifiés par la technique T_1 et sur la structure des deux ontologies alignées.

Nous présentons dans cette partie, les techniques d'alignement appliquées dans la version courante V_2 de TaxoMap. Les modes de sélection et de configuration de ces techniques sont détaillés dans la partie suivante. Les techniques utilisées dans TaxoMap sont des techniques terminologiques et structurelles. Nous avons décrit ces deux catégories dans le chapitre relatif à l'état de l'art. Nous présentons dans un premier temps les techniques terminologiques, puis les techniques structurelles.

Soient c_s le concept de l'ontologie source O_S pour lequel on recherche une correspondance et c_{tmax}, c_{t2} et c_{t3} les trois concepts de l'ontologie cible O_T qui ont les meilleures similarités avec c_s, par ordre décroissant. Les différentes techniques d'extraction de correspondances s'appliquent séquentiellement et s'appuient sur des valeurs de seuils de similarité qui peuvent être paramétrées suivant les applications.

Techniques terminologiques basées sur les labels (T_1, T_2, T_3, T_4, T_5, T_6 et T_7)

TaxoMap a été conçu en supposant que les ontologies à aligner étaient des ontologies dites légères, i.e. ne comportant pas ou peu de relations autres que celles de subsomption, car c'est le cas de la très grande majorité des ontologies existantes. De ce fait, nous avons supposé que les concepts étaient seulement définis par leurs labels et les relations de subsomption qu'ils entretiennent avec les autres concepts de leur hiérarchie. Pour expliciter une description fine d'un domaine qui peut nécessiter plusieurs milliers de concepts, les concepteurs de ce type d'ontologie ont le plus souvent recours à des labels de concepts expressifs et étendus, correspondant à des expressions composées de plusieurs mots. Les labels des concepts les plus spécifiques reprennent souvent les labels des concepts plus généraux en les spécialisant et plusieurs spécialisations d'un même concept peuvent ne se distinguent les unes des autres que par de très légères différences. TaxoMap met donc en œuvre des techniques permettant d'exploiter la richesse de ces labels, de rechercher les éventuelles inclusions de labels traduisant des relations de subsomption et d'exprimer dans les mappings d'autres relations entre concepts que la simple relation d'équivalence.

Ainsi, la technique T_1 s'applique quand les labels des concepts sont identiques ou définis comme synonymes. Les techniques T_2, T_3, T_4 et T_7, recherchent s'il existe des labels qui soient inclus dans d'autres. Enfin les techniques T_5 et T_6 s'appuient sur des similarités relatives entre les concepts proches. Dans ce qui suit, nous décrivons en détail ces techniques.

Technique de recherche des relations d'équivalence (T_1) Une relation d'équivalence (c_s $isEq$ c_{tmax}) est proposée lorsque la similarité d'un des labels de c_s avec un des labels de c_{tmax} est supérieure ou égale à un seuil (*Equiv.threshold*). Cette technique est utilisée pour identifier, comme le montre la figure 2.5, une relation d'équivalence entre deux concepts qui ont un ou plusieurs labels identiques ou synonymes. Le seuil de cette technique doit être proche de la valeur de 1.

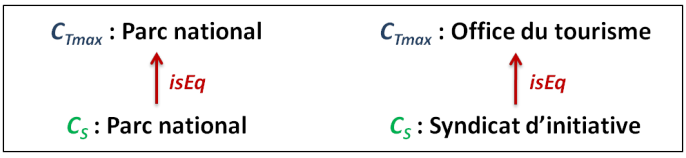

FIG. 2.5 – Exemple illustrant la technique T_1

Remarquons que cette technique n'est efficace que si les ontologies alignées décrivent le même domaine d'application : le risque d'homonymie est alors faible et les concepts qui ont le même label peuvent être considérés comme équivalents.

Techniques de recherche d'inclusions strictes (T_2, T_3) Selon T_2, une relation (c_s isA c_{tmax}) est générée si un des labels de c_{tmax} est inclus dans un des labels de c_s et si tous les mots du label inclus sont des *mots pleins* dans les deux labels (cf. Fig. 2.6). Inversement, selon T_3, une relation (c_s $isMoreGnl$ c_{tmax}) est générée si un des labels de c_s est inclus dans un des labels de c_{tmax} (cf. Fig. 2.7).

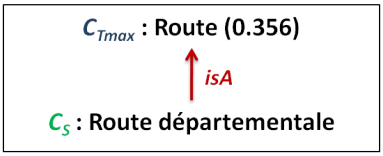

FIG. 2.6 – Exemple illustrant la technique T_2

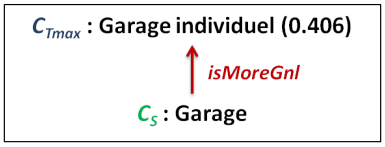

FIG. 2.7 – Exemple illustrant la technique T_3

Ces deux techniques permettent d'exprimer des relations de spécialisation (isA) et de généralisation ($isMoreGnl$) entre deux concepts. En effet, le fait qu'un des labels d'un concept soit inclus dans les mots pleins d'un des labels d'un autre concept permet de dire que les deux concepts sont liés et que le concept dont le label est inclus dans le label de l'autre représente une notion plus générale que l'autre.

Technique de recherche d'inclusions relâchées (T_4) Une relation (c_s $isClose$ c_{tmax}) est générée selon T_4 s'il existe une inclusion de labels entre c_s et c_{tmax}, (dans un sens ou dans l'autre), en tenant compte des mots complémentaires, i.e. apparaissant derrière un déterminant (cf. Fig. 2.8). On parle alors d'inclusions relâchées par opposition à l'inclusion stricte qui ne prend en compte que les mots pleins.

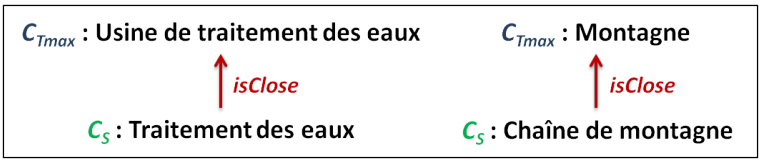

FIG. 2.8 – Exemple illustrant la technique T_4

Ces mappings de proximité ($isClose$) sont construits car, même si la similarité est très faible puisque les mots inclus n'ont pas le même statut dans leur label respectif, c_{tmax} apparaît comme le concept le plus proche de la cible auquel c_s puisse être lié. En revanche, les mots communs des deux labels n'étant pas tous des mots pleins, aucune relation de subsomption ne peut être établie a priori entre les deux concepts, ni dans un sens ni dans l'autre. La technique permet juste de dire que le concept le plus proche de c_s est c_{tmax}.

Techniques basées sur la similarité relative (T_5, T_6) La similarité relative est définie comme étant le rapport entre la similarité de c_{t2} sur celle de c_{tmax} (les deux concepts terminologiquement parlant les plus proches de c_s) et elle doit être inférieure ou égale à un seuil ($isA.threshold$) pour que l'une des deux techniques T_5 ou T_6 s'applique. L'idée est que plus la différence des similarités de c_{tmax} et c_{t2} à c_s est importante, moins il y a de doute sur le fait que

c_{tmax} soit le bon candidat pour le mapping.

Une relation (c_s $isClose$ c_{tmax}) est générée selon T_5 si la similarité de c_{tmax} est supérieure ou égale à un certain seuil ($isClose.thresholdMax$) (cf. Fig. 2.9).

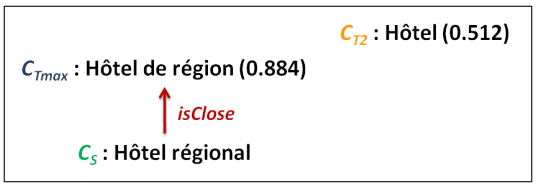

FIG. 2.9 – Exemple illustrant la technique T_5 avec $isClose.thresholdMax = 0.7$

Une relation (c_s isA $père(c_{tmax})$) est générée selon T_6 si la valeur de similarité de c_{tmax} est inférieure au seuil $isClose.thresholdMax$ mais supérieure ou égale à un deuxième seuil ($isA.thresholdMax$) (cf. Fig. 2.10).

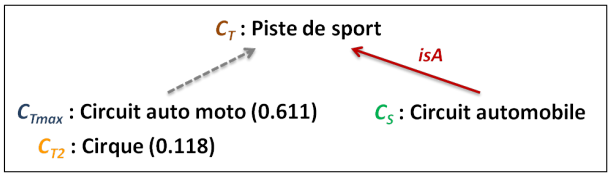

FIG. 2.10 – Exemple illustrant la technique T_6 avec $isClose.thresholdMax = 0.7$ et $isA.thresholdMax = 0.6$

Dans le cas de T_5, le fait que la similarité du label de c_{tmax} soit à la fois significativement plus importante que celle de c_{t2} et supérieure à un seuil lui-même élevé, permet de choisir une relation de proximité. Cette technique est conçue pour tirer parti de la supposée grande richesse des labels pour considérer c_s et c_{tmax} comme presque équivalents, si leur label sont très similaires. Dans le cas de T_6, la similarité de c_{tmax} n'est pas jugée assez élevée et il est considéré comme un frère de c_s, i.e. comme une spécialisation du même père.

Technique de recherche des inclusions cachées (T_7) Une relation (c_s isA c_t) est générée selon la technique T_7 si un des labels de c_t est inclus dans un des labels de c_s et que ce label, bien que n'étant pas celui qui a la similarité maximale, a une similarité supérieure ou égale à un seuil ($HiddenInc.thresholdSim$), le label de similarité maximale devant avoir une similarité supérieure à un autre seuil ($HiddenInc.thresholdMax$) (cf. Fig. 2.11).

FIG. 2.11 – Exemple illustrant la technique T_7

Quand aucune relation ne peut être établie avec le concept de la cible qui a la meilleure valeur de similarité avec aucune des techniques précédentes, la technique T_7 est utilisée pour vérifier s'il existe une inclusion de labels avec le concept de la cible qui a la deuxième meilleure valeur. Cette technique extrait des inclusions dites inclusions cachées.

Techniques structurelles (T_8, T_9 et T_{10})

Parmi les techniques structurelles, les deux techniques T_9 et T_{10} peuvent être qualifiées de structurelles pures car elles n'utilisent pas du tout les labels des concept c_s à aligner pour construire les mappings, mais uniquement la position relative de ces concepts dans leur ontologie par rapport à des ancres, i.e. des concepts précédemment identifiés par la technique T_1 comme équivalents à des concepts de la cible. Elles vont permettre d'aligner des concepts dont les labels n'ont pas ou peu de similarité avec ceux des concepts de la cible. A l'inverse, la technique T_8 s'appuie d'abord sur des similarités de labels pour identifier les concepts de la cible les plus proches d'un concept source c_s et n'utilise qu'ensuite la structure de la cible pour aligner c_s avec le père commun de ces concepts proches, si ce père commun existe.

Une relation (c_s *isA père*$Commun$) est générée selon la technique T_8 (cf. Fig. 2.12) si c_{tmax}, c_{t2} et c_{t3} ont une similarité supérieure ou égale à un seuil qui peut être assez bas ($Struct.threshold$), et un père commun (partagé par au moins deux concepts). En effet, si deux au moins des trois concepts de la cible les plus proches de c_s, sont frères dans la cible, le concept source peut être considéré comme leur frère et aligné comme une spécialisation de leur père.

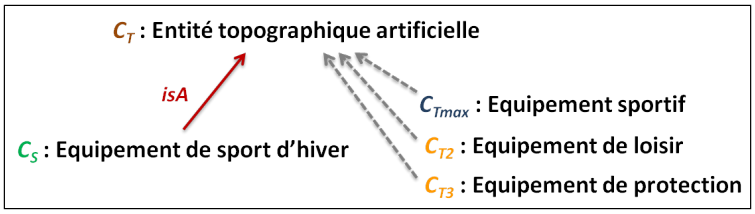

FIG. 2.12 – Exemple illustrant la technique T_8 avec $Struct.threshold = 0.4$

La technique T_9 (cf. Fig. 2.13) s'appuie sur la présence d'ancres, les mappings d'équivalence (X *isEq* c_t) identifiés en premier et considérés comme sûrs, pour rapatrier tous les fils directs, non encore alignés, d'une de ces ancres de la source, comme des spécialisations directes du concept associé à l'ancre dans la cible : une relation (c_s *isA* c_t) est ainsi générée selon T_9 pour tous les c_s tels qu'une relation de subsomption (c_s *isA* X) apparaît dans O_S.

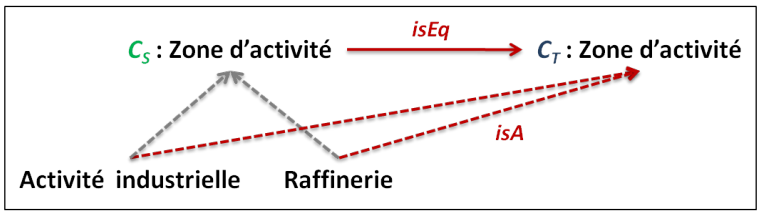

FIG. 2.13 – Exemple illustrant la technique T_9

La technique T_{10} (cf. Fig. 2.14) est basée sur la comparaison du voisinage (les descendants) des concepts comparés. Elle permet de relier par une relation de proximité $isClose$, un concept

c_s de la source à un concept c_t de la cible, si c_t est le concept de la cible qui a au moins parmi ses descendants deux ancres qui sont aussi des descendants de c_s et si c_t maximise le rapport nombre de descendants communs entre c_s et c_t sur le nombre total de descendants des deux concepts. Cette technique est basée sur la mesure de similarité *Semantic Cotopy* proposée par [Maedche and Staab, 2002].

***Semantic Cotopy* :** est une mesure basée sur la sémantique intentionnelle d'un concept c dans une ontologie O. $SC(c, O)$ est définie comme l'ensemble de concepts qui sont des ancêtres et des descendants de c dans O. La couverture taxonomique (TO) entre une ontologie O_1 et une ontologie O_2 pour un concept c (notée $TO(c, O_1, O_2)$) est définie comme le ratio entre le nombre d'éléments communs aux deux ensembles $SC(c, O_1)$ et $SC(c, O_2)$, et le nombre total d'éléments appartenant à l'union de ces ensembles. Si un concept c est dans O_1 mais pas dans O_2, une approximation optimiste de $TO(c, O_1, O_2)$ est définie comme la valeur maximale de couverture possible entre $SC(c, O_1)$ et les SC de tous les concepts de O_2.

Dans la technique T_{10}, nous utilisons SC_D qui ne comprend que le concept et ses descendants, sans prendre en compte ses généralisants. Nous définissons la mesure de "Semantic Descendant" que nous appelons M_{SC_D} comme suit :

Définition 5 (La mesure M_{SC_D}) *Soit l'ensemble $SD(c, O)$ composé du concept c et de tous ses sous-concepts dans O et soit $Equivalent(SD(c_s, O_S), SD(c_t, O_T))$ le nombre de relations d'équivalence vérifiées dans la table des mappings entre les concepts de $SD(c_s, O_S)$ et de $SD(c_t, O_T)$.*

$$M_{SC_D}(c_s, O_S, c_t, O_T) = \frac{Equivalent(SD(c_s, O_S), SD(c_t, O_T))}{|SD(c_s, O_S) \cup SD(c_t, O_T)|}$$

Si un concept c_s est dans O_S mais pas dans O_T, nous proposons comme candidat au mapping du concept c_s, le concept c_t de O_T qui maximise M_{SC_D}, si c_s et c_t ont au moins deux descendants en commun.

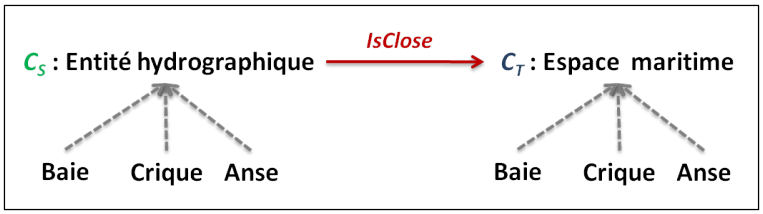

FIG. 2.14 – Exemple illustrant la technique T_{10}

La technique T_{10} permet de rapprocher des concepts qui sont des nœuds non feuilles dans leur ontologie respective, sans s'appuyer sur des similarités lexicales qui échouent souvent pour ces nœuds.

2.2 Une approche d'alignement modulaire et paramétrable

Initialement, TaxoMap était composé de 8 techniques appliquées séquentiellement dans un ordre fixe ayant été déterminé et optimisé par les expérimentations faites au sein du projet eDot.

Les techniques terminologiques étaient appliquées en priorité pour générer des mappings dits probables considérés comme relativement sûrs. Une seconde série de techniques, structurelles et sémantiques, conduisaient à des mappings supplémentaires potentiellement vrais, suggérés à l'expert lorsque des mappings probables n'avaient pu être trouvés. Les huit techniques composant TaxoMap étaient systématiquement exécutées. Les paramètres des techniques n'étaient pas facilement modifiables, leur modification nécessitant l'accès au code.

Ce mode d'utilisation de l'outil a été revu afin d'offrir plus de flexibilité et une utilisation plus facile. TaxoMap est aujourd'hui un outil mettant en œuvre une approche d'alignement modulaire et paramétrable.

2.2.1 Une approche modulaire

Les techniques composant TaxoMap sont aujourd'hui totalement indépendantes les unes des autres. Elles correspondent à des modules de codes indépendants. Cela facilite leur utilisation comme nous le décrivons dans le paragraphe suivant mais également l'ajout de nouvelles techniques puisqu'il n'est pas utile de faire interagir les nouvelles techniques développées avec celles pré-existantes.

Les différents modules communiquent par l'intermédiaire d'une base de données relationnelle (tableau 2.1) stockant, les données relatives aux ontologies alignées (par exemple, les descriptions des concepts et les relations de subsomption) et l'ensemble des mappings générés. L'utilisation d'une base de données permet aux techniques d'alignement implémentées dans TaxoMap d'accéder plus facilement et plus rapidement aux données utiles. Ceci améliore de manière considérable les performances de l'approche en terme de temps de calcul (cf. résultats d'alignement des ontologies OAEI présentés dans le chapitre 6 de cette thèse).

Tables	**Contenu**
Ontology (num, URI, type, language)	Données relatives aux ontologies alignées
Concept (URI, ontoURI)	Les concepts des deux ontologies
ISARelation (URISubsumed, URISubsumer)	Les relations de subsomption entre les concepts
Label (labelkey, label, normalizedLabel, conceptURI)	Les labels des concepts et leur forme normalisée
Word (word, position, category, occurrence, labelkey)	Les mots constituant les labels et leur catégorie morpho-syntaxique générée par TreeTagger
Mapping (URISource, URITarget, similarity)	Les mappings générés et leur valeur de similarité

TABLE 2.1 – Les tables de la base de données utilisée dans TaxoMap

Les mappings stockés dans la base de données sont utilisés en entrée des techniques exé-

cutées. En effet, chaque technique ne recherche des mappings qu'entre les couples de concepts pour lesquels aucun mapping n'a encore été trouvé par application des techniques déjà exécutées.

Le SGBD que nous utilisons dans notre approche est SQLite, qui est un moteur de base de données relationnelles accessible par le langage SQL. Contrairement aux serveurs de bases de données classiques, comme MySQL ou PostgreSQL, la particularité de SQLite est de ne pas reproduire le schéma habituel client-serveur mais d'être directement intégré aux programmes. L'intégralité de la base de données (déclarations, tables, index et données) est stockée dans un fichier indépendant de la plate-forme. Cette caractéristique nous permet d'améliorer, en temps de calcul, le processus d'alignement de TaxoMap et d'avoir une approche générique utilisable sur n'importe quelle plate-forme.

2.2.2 Une approche flexible car paramétrable

Du fait de leur implémentation sous forme de modules indépendants, les techniques à exécuter lors d'une session d'alignement peuvent aujourd'hui être choisies par l'utilisateur. Ce choix se fait parmi l'ensemble des techniques composant TaxoMap. Il a nécessité la conception d'une interface graphique adaptée.

Le choix porte sur les techniques proprement dites à exécuter : toutes ou un sous-ensemble. Ceci permet à l'utilisateur de choisir, à travers une interface graphique (cf. 1 Fig. 2.15) les techniques selon les spécificités des ontologies qu'il veut aligner. Par exemple, lorsque les ontologies ne sont pas bien structurées, l'utilisateur peut ne pas utiliser les techniques d'alignement T_8, T_9 et T_{10} basées sur la structure des ontologies.

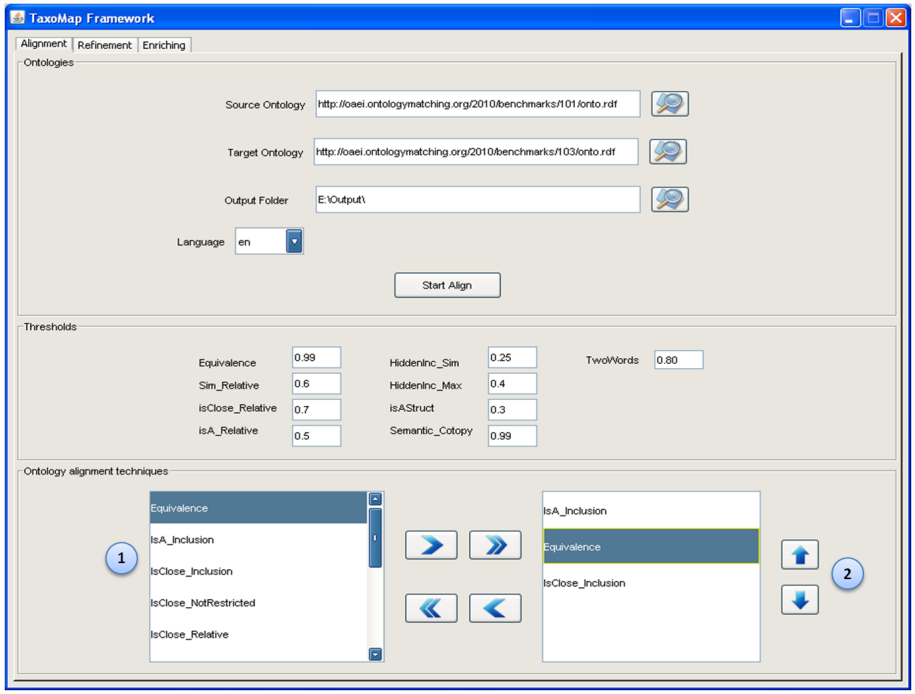

FIG. 2.15 – Interface graphique de TaxoMap

Le choix porte également sur l'ordre d'exécution des techniques sélectionnées. Une fois que l'utilisateur a choisi les techniques qu'il veut appliquer, il peut modifier l'ordre d'application des techniques à travers l'interface graphique (cf. 2 Fig. 2.15). Par exemple, l'utilisateur peut avoir constaté que, pour les ontologies qu'il veut aligner, la technique T_3 génère plus de mappings valides que la technique T_2. Dans ce cas, il peut choisir d'appliquer la technique T_3 avant la technique T_2.

Grâce à l'interface graphique, l'utilisateur peut aussi modifier les paramètres (les seuils) utilisés par les techniques. Par exemple, il peut choisir de diminuer le seuil de la technique T_1 lorsque il veut générer davantage de mappings d'équivalence.

Notons que l'utilisateur ne peut pas faire le choix d'une technique construite à partir de plusieurs autres techniques proposées. Toutes les techniques de TaxoMap restent des techniques simples ne résultant pas de la combinaison de plusieurs techniques de nature différente. Ces techniques sont facilement compréhensibles, ce qui est important lorsqu'il faut faire un choix.

Conclusion

Dans ce chapitre, nous avons décrit l'outil d'alignement TaxoMap, tel qu'il existe dans sa version actuelle V_2, ainsi que les adaptations variées du processus d'alignement qui ont été développées. TaxoMap est aujourd'hui dans une version stable. Toutes les adaptations que nous avons jugées utiles de faire, car pertinentes pour toutes les ontologies alignées, l'ont été. Ceci nous a permis de participer depuis 2008 à la compétition internationale OAEI et d'obtenir de bons résultats (présentés dans le chapitre 6 de cette thèse), ce qui montre que les adaptations réalisées ne dépendent pas du domaine d'application des ontologies alignées. Notre participation à la compétition 2010 a par ailleurs montré l'intérêt de disposer d'une approche flexible, qui a permis de nous adapter sans problème aux contraintes (l'utilisation du service web SEAL [6]) de la compétition.

Tous les développements décrits ont été complétés par le développement de TaxoMap sous forme de service web de façon à permettre à moyen terme à quiconque un accès via le web, sans avoir à télécharger le logiciel TaxoMap. L'accès est déjà possible en interne [7]. Ce service est décrit en annexe A. Il a fait l'objet d'une mission doctorale en 2010-2011 que j'ai encadrée.

6. `http://www.seals-project.eu`
7. `http://taxomap.lri.fr/`

Chapitre 3

Vers un alignement adapté aux spécificités des ontologies alignées

Sommaire

Introduction

De nombreux outils d'alignement d'ontologies ont été développés ces dernières années mais, comme le montrent les résultats des compétitions OAEI[8] (Ontology Alignment Evaluation Initiative) organisées chaque année depuis 2004, au niveau international, dans le domaine de l'alignement d'ontologies [Caracciolo et al., 2008] [Euzenat et al., 2009], aucun outil n'atteint une précision et un rappel de 100%, même si les résultats obtenus par certains de ces outils sont très bons.

Ce constat concerne également TaxoMap. Nous l'avons observé au travers des résultats obtenus lors de cette compétition ces dernières années [Hamdi et al., 2008, Hamdi et al., 2009a] mais également dans le cadre de notre participation au projet ANR GeOnto[9] qui vise la construction d'une ontologie topographique à partir de différents documents du domaine géographique, et en s'appuyant sur l'application de techniques d'alignement.

8. http://oaei.ontologymatching.org/
9. http://geonto.lri.fr/

Ainsi, nous proposons dans ce chapitre une approche pour spécifier des traitements de raffinements (cf. Section 3.1) mise en œuvre dans l'environnement TaxoMap Framework. Le module de raffinement est précisément décrit en section 3.2.

3.1 L'approche Taxomap Framework

L'approche TaxoMap Framework [Hamdi et al., 2010c, Hamdi et al., 2010a, Hamdi et al., 2010b, Hamdi et al., 2010d] a été conçue pour répondre aux objectifs décrits en section 3.1.1. Nous décrivons ensuite l'approche et un schéma représentant l'architecture de cet environnement respectivement en section 3.1.2 et 3.1.3. Cet environnement permet la spécification de traitements de raffinement à partir de primitives prédéfinies.

3.1.1 Objectifs de l'approche

La conception de l'approche TaxoMap Framework a été motivée, entre autres, par les travaux que nous avons développés dans le cadre du projet GeOnto, comme dit précédemment. En effet, les tests effectués sur les taxonomies mises à disposition par le COGIT de l'IGN, partenaire de ce projet, ont montré que l'outil d'alignement utilisé, TaxoMap, fournissait de très bons résultats (précision de $92,6\%$) mais que ces derniers pouvaient encore être améliorés.

Une étude des améliorations souhaitées par les experts a montré que celles-ci étaient souvent spécifiques aux ontologies alignées. L'environnement que nous proposons répond à ces besoins. L'objectif est de ne pas faire de TaxoMap un outil uniquement dédié à l'alignement de taxonomies topographiques car la qualité des résultats ne serait absolument pas garantie lors de l'alignement d'autres ontologies. Nous proposons alors de compléter l'utilisation de notre outil d'alignement par l'usage d'un environnement permettant aux experts du domaine de spécifier et d'effectuer des traitements de raffinement sur des alignements obtenus antérieurement.

L'environnement de raffinement de mappings satisfait deux objectifs principaux :

Premièrement, il fournit aux experts du domaine un outil leur permettant de détecter et de proposer des corrections pour des mappings invalides. La tâche de validation est parfois très difficile car le nombre de mappings générés peut être énorme lorsque les ontologies sont très grandes. L'expert peut avoir des difficultés à parcourir tous les mappings et à avoir une vision globale de la base de mappings pour proposer les bonnes modifications. En conséquence, il peut demander à modifier certains mappings sans se rendre compte que les modifications demandées ont un impact indésirable sur les autres mappings. Les observations des conséquences des modifications souhaitées peuvent être un moyen pour l'expert de spécifier avec une meilleure précision les traitements de raffinement à effectuer.

Deuxièmement, grâce au processus itératif de validation/correction, un tel environnement aide l'ingénieur à spécifier les traitements correctement. La phase de validation effectuée par l'expert permet de vérifier si la spécification d'un traitement destiné à être appliqué à un ensemble donné de mappings est correcte ou pas, c'est-à-dire si elle ne génère pas également des mappings indésirables.

3.1.2 Présentation de l'approche de raffinement

L'approche permet une spécification déclarative de traitements basés sur des résultats d'alignement particuliers et concernant des ontologies particulières, à l'aide d'un ensemble de primitives de base génériques et prédéfinies.

Elle permet de raffiner des résultats d'alignement. Elle a été conçue en prenant appui sur l'outil d'alignement TaxoMap mais pourrait être adaptée à d'autres outils. Via cette approche, il doit être possible, par exemple, de spécifier que le mapping "isA" généré entre "Chemin et sentier côtier" et "Sentier", conformément à la figure 3.1, doit être remplacé par un mapping de même type mais entre "Chemin et sentier côtier" et "Chemin". En effet, "Sentier" est défini comme une sorte de "Chemin" dans O_T et le terme "Chemin" est lui-même utilisé dans le label de "Chemin et sentier côtier". L'expert peut préférer établir une mise en correspondance directement entre "Chemin et sentier côtier" et "Chemin".

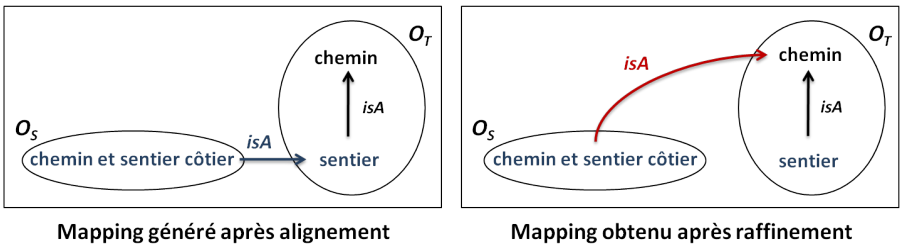

FIG. 3.1 – Exemple de raffinement souhaité

L'approche permet de spécifier des traitements de façon la plus générique possible. Ainsi, la spécification du traitement illustré figure 3.1 ne devra pas faire référence directement aux concepts dénotés par "Chemin", "Sentier" et "Chemin et sentier côtier".

Elle permet d'aider l'expert à expliciter les conditions d'application des traitements qu'il souhaite mettre en œuvre. Un ensemble de primitives génériques prédéfinies est ainsi mis à sa disposition. Ces primitives représentent les différentes conditions qui peuvent être testées sur les concepts intervenant dans un mapping construit par l'outil d'alignement utilisé, en l'occurrence, dans notre cas, TaxoMap.

L'analyse des résultats d'un alignement permet à un expert d'identifier des "familles" de mappings nécessitant un même raffinement. L'approche de raffinement que nous proposons lui permet de spécifier le traitement qu'il souhaite appliquer à chaque ensemble de cas identifié de façon générique. Cette spécification pourra ensuite être instanciée sur tous les résultats de l'alignement et les ontologies concernées pour exécuter les traitements correspondants, non seulement sur les mappings identifiés mais également sur tous les autres mappings correspondant à des cas similaires.

L'approche proposée permet de spécifier des traitements en fonction des caractéristiques des ontologies concernées et de la tâche visée. Les tâches visées sont variées. Il peut s'agir de raffinement de mappings, mais également d'enrichissement, de fusion ou encore de restructuration d'ontologies. L'approche proposée répond à ces différents besoins en étant modulaire. A chaque tâche correspond un module de spécifications différent, ayant son propre ensemble de primitives

de spécification.

Enfin, l'approche proposée, de fait de sa modularité, est facilement extensible. Elle est a priori applicable à tout traitement prenant appui sur les résultats d'un alignement, à condition que les primitives nécessaires aux traitements soient définies.

3.1.3 Architecture de Taxomap Framework

La figure 3.2 présente l'environnement de spécification mettant en œuvre l'approche Taxo-Map Framework. Cet environnement comporte trois parties : une partie "Contrôleur", une partie "Connaissance" et une partie "Traitement".

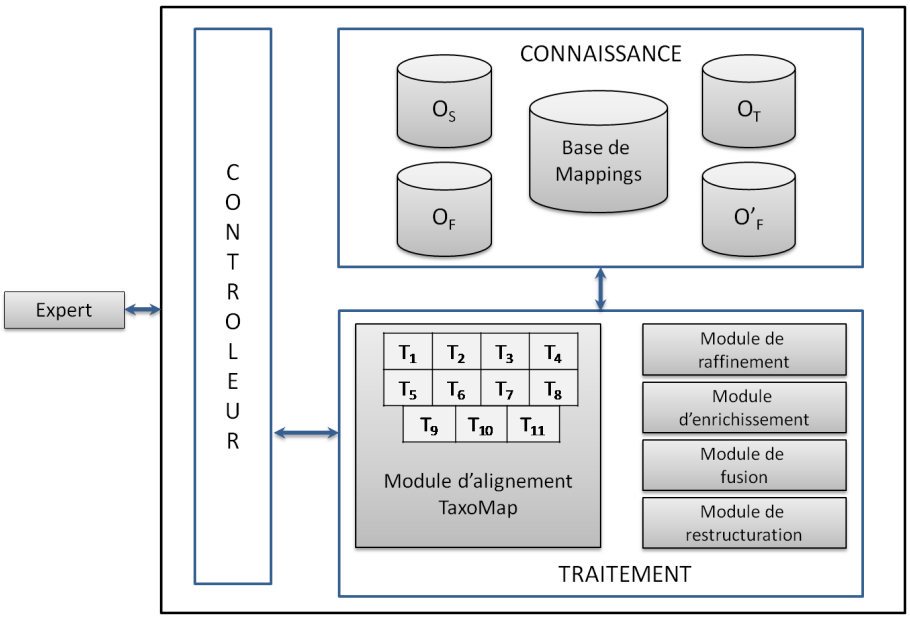

FIG. 3.2 – Architecture de TaxoMap Framework

La partie "Connaissance" regroupe l'ensemble des connaissances sur lesquelles les traitements à spécifier peuvent porter. Elle comprend ainsi les ontologies alignées, O_S et O_T, et l'alignement généré correspondant (Base de mappings). Selon les traitements effectués, on peut y trouver également l'ontologie O_F issue de la fusion entre O_S et O_T réalisée en exploitant la base de mappings ou l'ontologie O'_F correspondant à une version restructurée ou enrichie de O_F.

La partie "Traitement" regroupe l'outil d'alignement TaxoMap et l'ensemble des modules associés aux différentes tâches à réaliser. TaxoMap enchaîne a priori 10 techniques, qui peuvent être ou non mises en œuvre lors d'une session particulière et dont l'ordre d'exécution est paramétrable. Les modules associés aux tâches permettent de spécifier des traitements particuliers qu'un expert souhaite mettre en œuvre sur des ontologies particulières, mais également d'exécuter ces traitements. Des modules supplémentaires peuvent facilement être ajoutés à condition de leur associer des primitives de spécification adaptées, pouvant être reprises de primitives proposées dans d'autres modules.

Le "Contrôleur" permet de gérer l'ensemble des traitements possibles à l'aide de cet environnement, c'est-à-dire la spécification des traitements et leur exécution, l'accès aux données utiles et le stockage des résultats obtenus.

3.2 Présentation du module de raffinement

Dans cette partie, nous détaillons l'ensemble des étapes du processus de raffinement sous forme d'un workflow. Le langage utilisé pour représenter les patrons de raffinement fait l'objet de la section 3.2.2. Enfin, nous illustrons l'utilisation de l'approche mise en œuvre dans cet environnement de raffinement par la présentation de quelques patrons construits dans le cadre des travaux réalisés au sein du projet ANR GeOnto.

3.2.1 Le workflow de raffinement de mappings

La figure 3.3 présente le workflow du module de raffinement d'alignement implémenté dans l'environnement TaxoMap Framework.

Tout d'abord, TaxoMap est exécuté sur deux ontologies, une source et une cible (cf. 1 Fig. 3.3). Les résultats de l'alignement (les mappings) sont stockés dans une base de données (cf. 2 Fig. 3.3), prêts à être validés par l'expert du domaine ou un ingénieur (cf. 3 Fig. 3.3).

Lorsque l'expert/ingénieur examine de près l'alignement produit, il peut remarquer l'existence de mappings incorrects ou de mappings qui sont différents de ce qu'il aurait voulu. Ces mappings sont regroupés par l'ingénieur quand ils correspondent au même cas. Les exemples liés à un cas similaire sont généralisés (cf. 4 Fig. 3.3) et le patron correspondant est décrit (cf. 5 Fig. 3.3).

Les patrons sont ensuite appliqués à l'ensemble des mappings de la base de données, c'est-à-dire à la fois aux mappings cités par l'expert comme des exemples devant être raffinés, mais aussi à tous les autres mappings sur lesquels les patrons sont applicables même s'ils n'ont pas été cités par l'expert (cf. 6 Fig. 3.3).

Les résultats du processus de transformation de mappings doivent ensuite être validés (cf. 3 Fig. 3.3). La phase de validation permet de vérifier si un traitement génère des raffinements indésirables. Dans le cas où des mappings sont mis à jour alors qu'ils ne devraient pas l'être, ces mappings sont utilisés comme des contre-exemples pour trouver les bons traitements à effectuer. L'expert/ingénieur doit alors ré-écrire les patrons à appliquer pour prendre en compte non seulement les exemples qu'il avait identifiés mais également les contre-exemples.

Le processus de raffinement de mappings est ainsi un processus itératif de validation/correction.

La validation, la généralisation et la spécification des patrons sont des traitements manuels. La transformation des mappings basée sur l'utilisation des patrons est automatique.

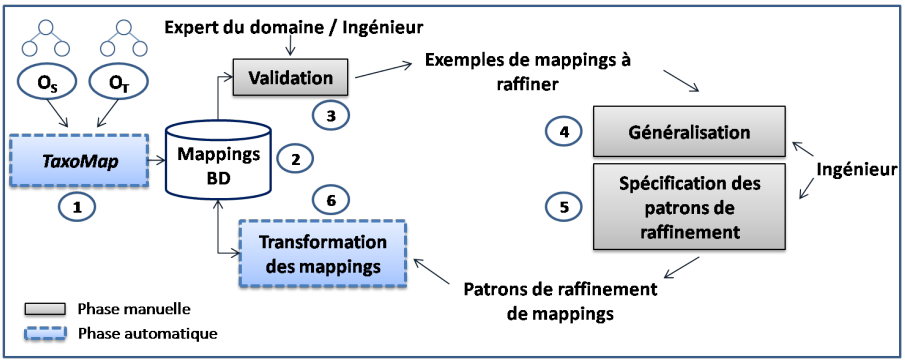

FIG. 3.3 – Workflow de raffinement de mappings

3.2.2 Le langage des patrons de raffinement MPL

Nous avons conçu le langage MPL (the Mapping Pattern Language) pour spécifier les patrons de raffinement de mappings dans TaxoMap Framework. Ce langage diffère de celui défini dans [Scharffe, 2009] en particulier parce qu'il inclut (1) des prédicats qui testent l'existence de mappings générés par une technique d'alignement spécifique, et (2) des prédicats qui exploitent la richesse des labels des concepts. Il diffère aussi du langage défini dans [Iannone et al., 2008] parce que ce dernier n'inclut pas de prédicats permettant de tester l'existence de mises en correspondance.

MPL est défini comme suit :

Le vocabulaire de MPL contient :

- *un ensemble de prédicats*. Trois catégories de prédicats sont distingués : les prédicats liés au type de techniques appliquées dans l'identification de mappings par TaxoMap, les prédicats exprimant des relations structurelles entre les concepts d'une même ontologie, et les prédicats exprimant des relations terminologiques entre les labels des concepts.

- *un ensemble de constantes individuelles* : $\{a, b, c, ...\}$

- *un ensemble de variables* : $\{x, y, z, ..., \_\}$ où _ est une variable sans nom utilisée pour représenter des paramètres qui n'ont pas besoin d'être précisés.

- *un ensemble de prédicats prédéfinis* : $\{Add\_Mapping, Delete\_Mapping\}$

- *un ensemble de symboles logiques* : $\{\exists, \wedge, \neg\}$

Le langage MPL permet la définition d'une **partie contexte** qui doit être satisfaite pour rendre possible l'exécution d'un patron, et d'une **partie solution** qui exprime le processus à réaliser lorsque la **partie contexte** est satisfaite. La partie contexte est une formule logique définie comme suit.

- *Les variables et les constantes sont des termes.*

- Si α et β sont des termes et que P est un symbole de prédicat avec deux paramètres alors $\mathrm{P}(\alpha, \beta)$ *est une formule.*

- Si α, β et γ sont des termes et que P est un symbole de prédicat avec trois paramètres alors $\mathrm{P}(\alpha, \beta, \gamma)$ *est une formule.*

- Si ϕ et ψ sont des formules alors $[\phi \wedge \psi]$ est une formule.

- Si ϕ est une formule alors $[\neg \phi]$ est une formule.

- Si ϕ est une formule et que v est une variable alors $\exists v \phi$ est une formule.

La **partie contexte** teste trois conditions :

1. la technique employée pour identifier le mapping considéré.

2. les contraintes structurelles portant sur les éléments mis en correspondance, par exemple, le fait qu'ils soient liés par une relation de subsomption à des concepts vérifiant ou pas certaines propriétés.

3. les contraintes terminologiques, par exemple le fait que des labels de concepts soient inclus dans d'autres labels de concepts.

Ces conditions sont représentées en utilisant les formules construites à partir des symboles de prédicat. Ainsi, on distingue trois types de formules en fonction du type de symboles de prédicat utilisé.

Prédicats relatifs aux techniques appliquées dans l'identification d'un mapping

Nous décrivons ci-dessous les prédicats correspondant aux techniques contenues dans TaxoMap. En testant l'existence dans la base de mappings d'une relation de correspondance particulière générée par une technique donnée, ces prédicats testent également implicitement l'ensemble des conditions d'application de cette technique. L'expert n'a donc pas besoin de les connaître précisément ni de les re-spécifier.

Ainsi la formule $isAStrictInclusion(x, y)$ teste l'existence d'un mapping *isA* généré entre deux concepts x et y par la technique t_2 de TaxoMap. Elle valide en même temps implicitement les conditions d'application de t_2, c'est-à-dire le fait que (1) le concept y est le concept de O_T qui a la plus forte similarité avec le concept x de O_S, (2) qu'un des labels de y est inclus dans un des labels de x, (3) tous les mots du label inclus sont des *mots pleins* dans les deux labels.

TaxoMap comprend plusieurs techniques d'alignement. Ainsi, plusieurs symboles de prédicat conduisant à des formules de ce genre sont nécessaires.

Plus formellement, soit :

$R_M = \{isEq, isA, isMoreGnl, isClose\}$, l'ensemble des relations de correspondance utilisées par TaxoMap.

$T = \{t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9, t_{10}\}$, l'ensemble des techniques.

T_M, la table stockant les mappings générés sous la forme de *4-tuple* (x, y, r, t) où $x \in C_S, y \in C_T, r \in R_M, t \in T$. Les couples de variables (x, y) qui pourront instancier ces prédicats prendront leurs valeurs dans l'ensemble $\{(x, y) \mid (x, y, r, t) \in T_M\}$.

Les symboles de prédicat nécessaires aux traitements de raffinement sont : $isEquivalent$, $isAStrictInclusion$, $isCloseStrictInclusion$, $isCloseNotStrictInclusion$, $isCloseRelativeSimilarity$, $isARelativeSimilarity$, $isAHiddenInclusion$, $isAStructural$, $isAafterEquivalence$, $isCloseCommonDescendant$ et $isCloseTwoWords$. Ils sont définis comme suit :

- $isEquivalent(x, y)$ est vrai ssi $\exists(x, y, isEq, t_1) \in T_M$
- $isAStrictInclusion(x, y)$ est vrai ssi $\exists(x, y, isA, t_2) \in T_M$
- $isCloseStrictInclusion(x, y)$ est vrai ssi $\exists(x, y, isMoreGnl, t_3) \in T_M$
- $isCloseNotStrictInclusion$ est vrai ssi $\exists(x, y, isClose, t_4) \in T_M$
- $isCloseRelativeSimilarity$ est vrai ssi $\exists(x, y, isClose, t_5) \in T_M$
- $isARelativeSimilarity$ est vrai ssi $\exists(x, y, isA, t_6) \in T_M$
- $isAHiddenInclusion$ est vrai ssi $\exists(x, y, isA, t_7) \in T_M$
- $isAStructural(x, y)$ est vrai ssi $\exists(x, y, isA, t_8) \in T_M$
- $isAafterEquivalence$ est vrai ssi $\exists(x, y, isA, t_9) \in T_M$
- $isCloseCommonDescendant(x, y)$ est vrai ssi $\exists(x, y, isClose, t_{10}) \in T_M$

Prédicats exprimant des relations structurelles

Ces prédicats permettent de décrire les liens existant entre les concepts x et y d'une même ontologie $O = (C, H)$ où C est un ensemble de concepts caractérisés chacun par un ensemble de labels et H est une hiérarchie de subsomption composée des relations isA entre les nœuds des concepts correspondant. Plus précisément, ils permettent de tester l'existence d'un lien de spécialisation ou de généralisation entre deux concepts, l'existence d'un père commun à deux concepts, ou si la distance entre un concept et la racine est inférieure à un certain nombre.

Les instances de variables intervenant dans ces prédicats sont contraintes, soit directement parce qu'elles instancient un prédicat relatif à une technique d'alignement (par exemple, vérifier s'il existe une relation de subsomption entre les deux concepts y et z dans O_T, sachant que y est relié avec le concept x de O_S par un mapping d'équivalence), soit indirectement par le fait de

devoir être en relation avec d'autres instances.

- $isSubClassOf(x, y, O)$ est vrai ssi $isA(x, y) \in H$

- $isParentOf(x, y, O)$ est vrai ssi $isA(y, x) \in H$

- $isSharedParent(p, y, x, O)$ est vrai ssi tous les y (au moins deux) qui partagent au moins une caractéristique (relation de mapping, inclusion de labels, etc.) avec x, ont le même parent p dans O. Ce prédicat doit être précédé par : (1) une formule qui définit la caractéristique partagée entre x et y, (2) la formule $isParentOf(p, y, O)$.

Algorithm 1 isSharedParent(p,y,x,O)

Require: $\{p, y, x\} \in C_S \cup C_T$
1: **if** $Features(x, y)$ **then**
2: **if** $isParentOf(p, y, O)$ **then**
3: **return** *true*
4: **end if**
5: **end if**

où $Features(x, y)$ est une fonction qui vérifie que x et y partagent au moins une caractéristique.

- $maxDepth(x, d, O)$ est vrai ssi la longueur du plus long chemin menant de x à la racine de H est inférieure ou égale à d.

Algorithm 2 maxDepth(x,d,O)

Require: $x \in C_S \cup C_T$
1: **if** $DistanceToRoot(x) \leq d$ **then**
2: **return** *true*
3: **end if**

où $DistanceToRoot(x)$ est une fonction qui calcule la distance du plus long chemin entre x et la racine de H.

Prédicats exprimant des relations terminologiques entre labels de concepts

Les prédicats de ce type implémentés à ce jour permettent de décrire si deux concepts ont des identifiants différents, de vérifier si une chaîne de caractères apparaît dans un label d'un concept ou encore de vérifier si un label (ou une partie du label) d'un concept est inclus dans un label d'un autre concept.

- $conceptsDifferent(x, y)$ est vrai ssi $ID(x) \neq ID(y)$ où $ID(x)$ est l'identifiant de x.

- $appearInLabel(c, y)$ est vrai ssi $\exists$ un label L_1 de y tel que $c \subset L_1$, où c est une chaîne de caractères et $y \in C_S \cup C_T$.

– $strictInclusionLabel(x, y)$ est défini comme suit :

Algorithm 3 strictInclusionLabel(x,y)

Require: $\{x, y\} \in C_S \cup C_T$
1: **for** each label L_1 of x and each label L_2 of y **do**
2: **if** $L_1 \subseteq FullWords(L_2, L_1)$ **then**
3: **return** *true*
4: **end if**
5: **end for**

où $FullWords(L_2, L_1)$ est une fonction qui calcule l'ensemble des termes de L_2 considérés comme des *mots pleins* dans sa comparaison avec L_1.

– $inclusionInLabel(x, c, y)$ est vrai $\Leftrightarrow$ $extractFromLabel(x, c, y, \_)$ est vrai.

Algorithm 4 extractFromLabel(x,c,y,r)

Require: $\{x, y\} \in C_S \cup C_T$ and $c \in \{"and", "or"\}$
1: **for** each label L_1 of x **do**
2: $SplitLabelPart(L_1, c, Part_1, Part_2)$
3: **if** one label of $y = Part1$ **then**
4: $r = Part_2$, **return** *true*
5: **else if** one label of $y = Part_2$ **then**
6: $r = Part_1$, **return** *true*
7: **else**
8: **return** *false*
9: **end if**
10: **end for**

où $SplitLabelPart(L_1, c, Part_1, Part_2)$ est une fonction qui extrait du label L_1 les deux composantes $Part_1$ et $Part_2$, constituées des mots apparaissant respectivement avant et après la chaîne de caractères c.

La **partie contexte** d'un patron est associée à une **partie solution** qui est un ensemble d'actions à exécuter dans la base de mappings (ajout, suppression de mappings). Cet ensemble d'actions est modélisé par une conjonction de prédicats prédéfinis exécutables dans une base de données.

Ces prédicats prédéfinis utilisés dans la partie solution des patrons sont définis comme suit :

– $Add\_Mapping(x, y, r)$ a pour effet d'ajouter un tuple à la table T_M qui devient $T_M \cup \{(x, y, r, t)\}$ où r et t sont fixées dans la partie condition du traitement, par l'instanciation du prédicat identifiant la technique considérée.

– $Delete\_Mapping(x, y, \_)$ a pour effet de supprimer un tuple dans la table T_M qui devient $T_M - \{(x, y, \_, \_)\}$.

3.2.3 Les patrons de raffinement de mappings

Dans cette section, nous présentons des patrons de raffinement de mappings conçus dans le cadre du projet ANR GeOnto à titre d'exemples. Nous avons utilisé l'environnement TaxoMap Framework pour spécifier ces patrons. La spécification a été faite par un expert de l'IGN assisté d'un ingénieur informaticien.

Notons que l'approche que nous proposons n'a pas pour objectif de créer une bibliothèque de patrons de raffinement. L'objectif est de fournir un environnement de spécification de patrons adapté à la spécification de traitements particuliers qu'un expert souhaite voir exécuter sur des mappings. Bien sûr, plus TaxoMap Framework sera utilisé, plus le nombre de patrons de sa bibliothèque seront nombreux. Peut-être l'expert trouvera t-il alors, parmi ces patrons, celui qui convient à la transformation qu'il veut opérer sur ses mappings. L'objectif ne consiste toutefois pas à fournir une bibliothèque la plus riche possible mais à offrir un environnement adapté à la spécification de patrons personnalisés.

Patron-1 :

Ce premier patron concerne les mappings générés par la technique t_2, reliant par une relation de subsomption "isA" un concept c_s de l'ontologie source O_S à un concept c_{tmax} de l'ontologie cible O_T, tels qu'un des labels de c_{tmax} est inclus dans un des labels de c_s.

Si un des labels du concept A qui subsume c_{tmax} dans O_T est aussi inclus dans le label de c_s, (cf. Fig. 3.4), l'expert préfère rattacher c_s à A, le concept le plus général de O_T.

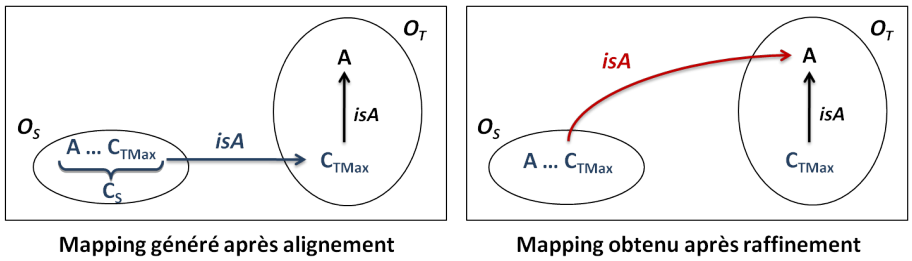

FIG. 3.4 – Illustration du Patron-1

Partie contexte du Patron-1 :
$\exists x \exists y\ (isAStrictInclusion(x, y)$
$\wedge\ \exists z\ (isSubClassOf(y, z, O_T) \wedge strictInclusionLabel(z, x)))$

Partie solution du Patron-1 :
$Delete\_Mapping(x, y, \_) \wedge Add\_Mapping(x, z, isA)$

Un exemple d'application de ce patron est donné Fig. 3.5.

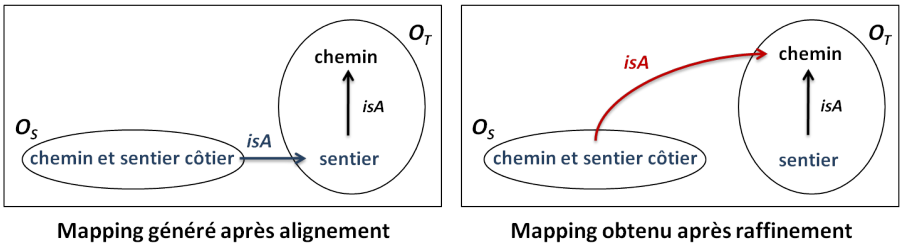

FIG. 3.5 – Exemple d'application du Patron-1

L'application de ce patron présenté Fig. 3.4 permet d'abord de sélectionner le mapping (id_1, id_2, isA, t_2) où un des labels de id_1 est "chemin et sentier côtier", un des labels de id_2 est "sentier" et la formule $isAStrictInclusion(id_1, id_2)$ est satisfaite dans la base de données des mappings. Les variables x et y sont instanciées par id_1 et id_2 respectivement.

L'utilisation de la formule $isSubClassOf(id_2, z, O_T)$ basée sur un symbole de prédicat structurel mène à (1) l'instanciation de la variable z par id_3, où un des labels de id_3 est "chemin", (2) et à la vérification de la formule $strictInclusionLabel(id_3, id_1)$.

Le mapping (id_1, id_2, isA, t_2) est ensuite supprimé de la base de données des mappings et remplacé par le mapping (id_1, id_3, isA, t_2).

Patron-2 :

Ce deuxième patron concerne aussi les mappings générés par la technique t_2. Si aucun des labels du concept A qui subsume c_{tmax} dans O_T n'est inclus dans les labels de c_s mais qu'au contraire celui-ci contient l'un des connecteurs "et" ou "ou", l'expert considère que c_s n'est pas une spécialisation de c_{tmax} mais plutôt un généralisant de celui-ci, ce que nous traduisons par la relation "$isMoreGnl$" (cf. Fig. 3.6).

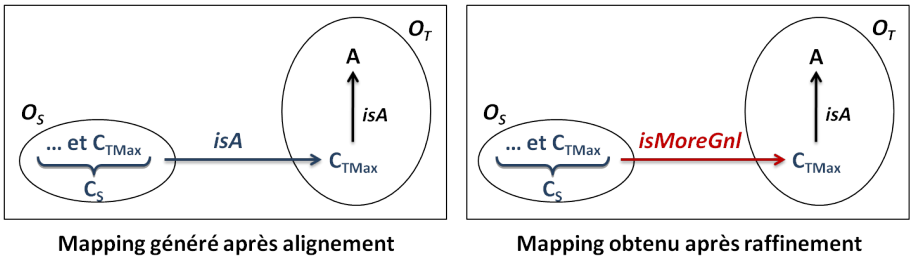

FIG. 3.6 – Illustration du Patron-2

Partie contexte du Patron-2 :

$\exists x \exists y\ (isAStrictInclusion(x, y) \wedge inclusionInLabel(x, "et", y)$
$\wedge\ \exists z\ (isSubClassOf(y, z, O_T) \wedge \neg strictInclusionLabel(z, x)))$

Partie solution du Patron-2 :

$Delete\_Mapping(x, y, \_) \wedge Add\_Mapping(x, y, isMoreGnl)$

Un exemple d'application de ce patron est donné Fig. 3.7.

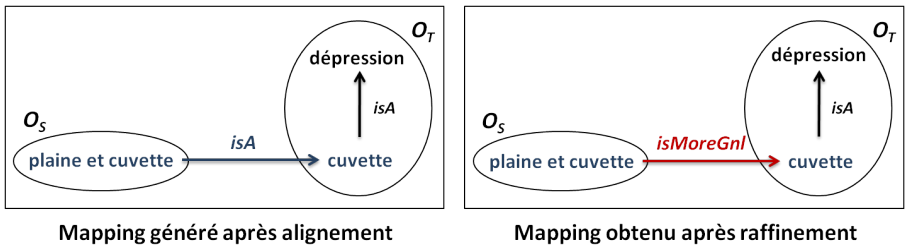

FIG. 3.7 – Exemple d'application du Patron-2

L'application de ce patron présenté Fig. 3.6 permet d'abord de sélectionner le mapping (id_1, id_2, isA, t_2) où un des labels de id_1 est "plaine et cuvette", et un des labels de id_2 est "cuvette" et les deux formules $isAStrictInclusion(id_1, id_2)$ et $inclusionInLabel(id_1, "et", id_2)$ sont satisfaites. Les variables x et y sont instanciées par id_1 et id_2 respectivement.

L'utilisation de la formule $isSubClassOf(id_2, z, O_T)$ mène à (1) l'instanciation de la variable z par id_3, où un des labels de id_3 est "dépression", (2) et à la vérification de la formule $\neg strictInclusionLabel(id_3, id_1)$.

Le mapping (id_1, id_2, isA, t_2) est ensuite supprimé de la base de données des mappings et remplacé par le mapping $(id_1, id_2, isMoreGnl, t_2)$.

Patron-3 :

Ce patron concerne les mappings reliant par une relation de généralité "$isMoreGnl$" un concept c_s de l'ontologie source O_S à un concept c_{tmax} de l'ontologie cible O_T, tels qu'un des labels de c_s est inclus dans un des labels de c_{tmax}. S'il existe un autre concept de O_T dont un des labels contient aussi c_s et que cet autre concept a le même père P dans O_T que c_{tmax}, l'expert préfère rattacher c_s à ce père P (cf. Fig. 3.8).

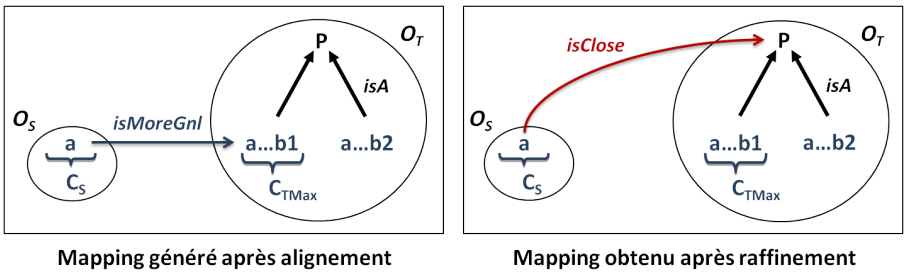

FIG. 3.8 – Illustration du Patron-3

Partie contexte du Patron-3 :

$\exists x \exists y\ (isCloseStrictInclusion(x, y) \wedge strictInclusionLabel(x, z)$
$\wedge\ \exists p\ (isParentOf(p, z, O_T) \wedge isSharedParent(p, z, x, O_T)))$

Partie solution du Patron-3 :

$Delete\_Mapping(x, y, \_) \wedge Add\_Mapping(x, p, isClose)$

Un exemple d'application de ce patron est donné Fig. 3.9.

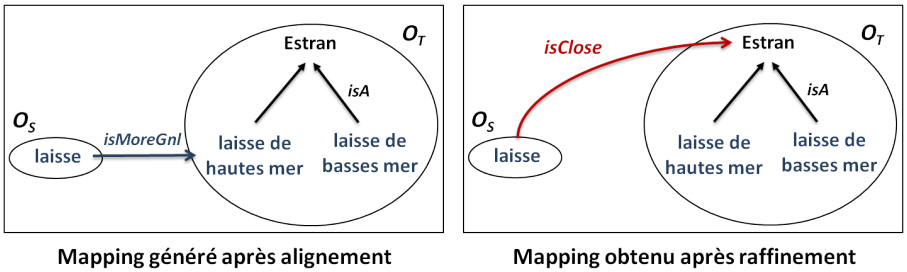

FIG. 3.9 – Exemple d'application du Patron-3

L'application de ce patron présenté Fig. 3.8 permet d'abord de (1) sélectionner le mapping $(id_1, id_2, isMoreGnl, t_3)$ où un des labels de id_1 est "laisse", et un des labels de id_2 est "laisse de hautes mer" et la formule $isCloseStrictInclusion(id_1, id_2)$ est satisfaite dans la base de données des mappings. Les variables x et y sont instanciées par id_1 et id_2 respectivement.

L'utilisation ensuite de la formule $strictInclusionLabel(id_1, z)$ permet l'instanciation de la variable z par id_3, où $strictInclusionLabel(id_1, id_3)$ est satisfaite. La formule $isParentOf(p, id_3, O_T)$ mène à (1) l'instanciation de la variable p par id_4, où un des labels de id_4 est "estran", (2) et à la vérification de la formule $isSharedParent(id_4, id_3, id_1, O_T)$.

Enfin, le mapping $(id_1, id_2, isClose, t_3)$ est supprimé de la base de données des mappings et remplacé par le mapping $(id_1, id_4, isMoreGnl, t_3)$.

Patron-4 :

Ce patron concerne les mappings identifiés par la technique t_8 reliant par une relation de subsomption "isA" un concept c_s de l'ontologie source O_S à un concept c_t de l'ontologie cible O_T, tel que c_t est le père dans O_T d'au moins deux des concepts de O_T qui ont la similarité la plus forte avec c_s. Si ce concept c_t est trop général, i.e. positionné trop haut dans la hiérarchie H_C, l'expert souhaite ne pas prendre en compte le mapping (cf. Fig. 3.10).

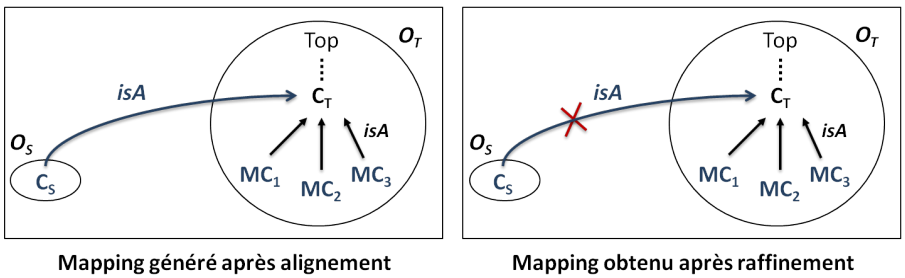

FIG. 3.10 – Illustration du Patron-4

Partie contexte du Patron-4 :
$\exists x \exists y\ (isAStructural(x, y) \wedge maxDepth(y, 2, O_T))$

Partie solution du Patron-4 :
$Delete\_Mapping(x, y, \_)$

Un exemple d'application de ce patron est donné Fig. 3.11.

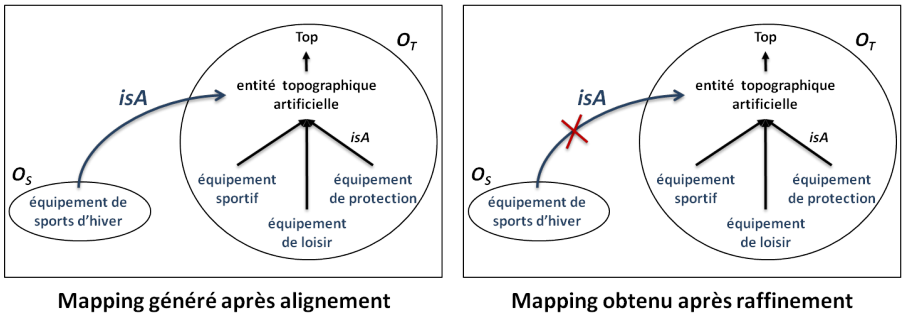

FIG. 3.11 – Exemple d'application du Patron-4

L'application de ce patron présenté Fig. 3.10 permet dans un premier temps de sélectionner le mapping (id_1, id_2, isA, t_8) où un des labels de id_1 est "équipement de sports d'hiver", et un des labels de id_2 est "entité topographique artificielle" et les deux formules $isAStructural(id_1, id_2)$ et $maxDepth(id_2, 2, O_T)$ sont satisfaites. Les variables x et y sont instanciées par id_1 et id_2 respectivement.

Le mapping (id_1, id_2, isA, t_2) est ensuite supprimé de la base de données des mappings.

Patron-5 :

Ce patron concerne les mappings identifiés par la technique t_{10} reliant par une relation de proximité *isClose*, un concept c_s de O_S à un concept c_t de O_T, si c_t est le concept dans O_T qui a au moins deux descendants en commun avec c_s et qui maximise M_{SD} pour c_s (cf. Chapitre 2).

Si la technique t_{10} a relié un concept c_s à un concept c_t tel qu'un mapping d'équivalence relie déjà à ce c_t un concept d de O_S qui soit un spécialisant de c_s dans O_S, le patron a pour effet de relier c_s au père de c_t dans O_T par une relation de subsomption. En effet, conserver le mapping initial reviendrait à relier un concept à l'équivalent d'un de ses fils. Une illustration est donnée Fig. 3.12.

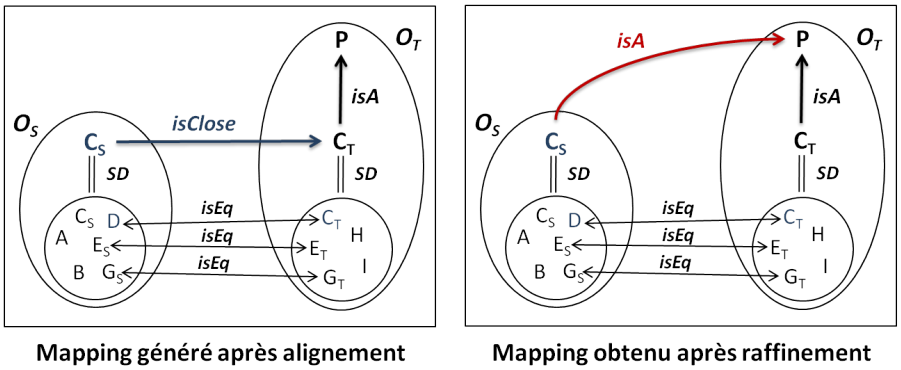

FIG. 3.12 – Illustration du Patron-5

Partie contexte du Patron-5 :
$\exists x \exists y\ (isCloseCommonDescendant(x, y) \wedge \exists d\ (isEquivalent(d, y)$
$\wedge\ isSubClassOf(d, x, O_S)) \wedge \exists p\ isParentOf(p, y, O_T))$

Partie solution du Patron-5 :
$Delete\_Mapping(x, y, \_) \wedge Add\_Mapping(x, p, isA)$

Un exemple d'application de ce patron est donné Fig. 3.13.

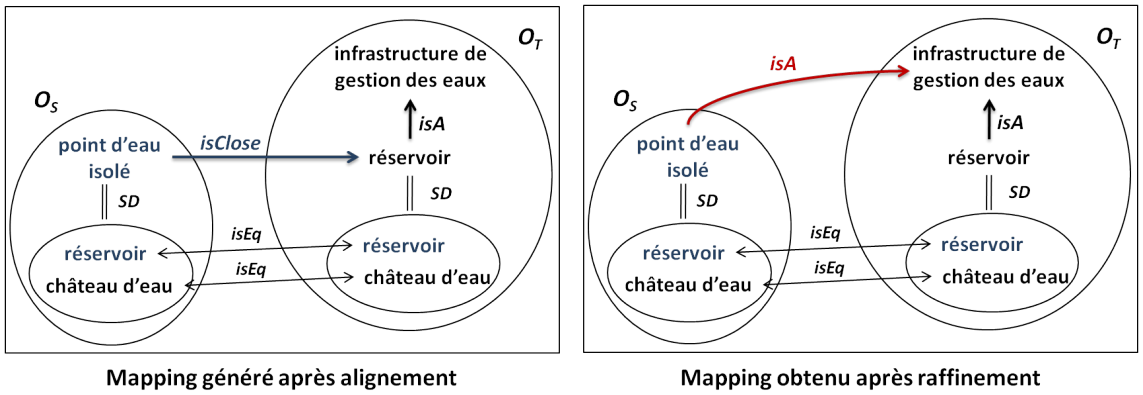

FIG. 3.13 – Exemple d'application du Patron-5

L'application de ce patron présenté Fig. 3.12 permet d'abord de (1) sélectionner le mapping $(id_1, id_2, isClose, t_1 0)$ où un des labels de id_1 est "point d'eau isolé", et un des labels de id_2 est "réservoir" et la formule $isCloseCommonDescendant(id_1, id_2)$ est satisfaite dans la base de données des mappings. Les variables x et y sont instanciées par id_1 et id_2 respectivement.

L'utilisation ensuite de la formule $isEquivalent(d, id_2)$ permet l'instanciation de la variable d par id_3, où les deux formules $isEquivalent(id_3, id_2)$ et $isSubClassOf(id_3, id_1, O_S)$ sont satisfaites. La formule $isParentOf(p, id_2, O_T)$ mène à l'instanciation de la variable p par id_4, où un des labels de id_4 est "infrastructure de gestion des eaux".

Enfin, le mapping $(id_1, id_2, isClose, t_1 0)$ est supprimé de la base de données des mappings et remplacé par le mapping $(id_1, id_4, isA, t_1 0)$.

Conclusion

TaxoMap Framework est un environnement de spécification de traitements qui s'appuie sur les résultats d'alignement générés par TaxoMap. Nous avons présenté l'approche mise en œuvre au sein de ce système, son architecture, le workflow de raffinement mappings et le langage MPL (Mapping Pattern Language) permettant à un expert du domaine de spécifier facilement les traitements qu'il souhaiterait appliquer sur un alignement. Notons que l'approche de raffinement repose sur l'utilisation de TaxoMap en tant qu'outil d'alignement mais qu'elle pourrait reposer sur un autre outil. Si les prédicats associés à cet autre outil ont été définis, la spécification des traitements de raffinement est simplifiée. Si ces prédicats n'ont pas été définis, il faut les ajouter.

La conception de TaxoMap Framework est adaptée à la spécification d'autres traitements tels que l'enrichissement, la fusion et la restructuration d'ontologies qui, comme le raffinement de mappings, exploitent un alignement. Nous présentons dans le chapitre 5 de cette thèse l'approche d'enrichissement d'ontologie comme exemple de traitement prenant appui sur les résultats d'un alignement et exploitant le langage de raffinement de mappings MPL.

Chapitre 4

Le partitionnement : un préalable à l'alignement de très grandes ontologies

Sommaire

Introduction

Le problème qui nous intéresse ici est celui du passage à l'échelle des méthodes d'alignement d'ontologies.

Les ontologies auxquelles nous nous intéressons sont des taxonomies ou ontologies "légères", comprenant un ensemble de concepts C modélisant un domaine d'application et une hiérarchie de subsomption entre concepts H_C.
Un concept est défini par son label et les relations de sous-classes qui le relient à d'autres concepts. Le label est un nom (chaîne de caractères) qui décrit des entités en langage naturel et qui peut être une expression composée de plusieurs mots. Les relations de sous-classes établissent des liens entre concepts. Il s'agit de l'unique association sémantique utilisée dans la classification. Une taxonomie est généralement représentée par un graphe acyclique dont les nœuds sont les concepts et les arcs correspondent aux liens de sous-classes.

Quand les ontologies sont de très grande taille, l'efficacité des méthodes d'alignement automatique diminue considérablement. La solution que nous proposons est de limiter la taille des ensembles de concepts en entrée de l'outil d'alignement, et pour cela de partitionner les deux

ontologies à aligner en plusieurs blocs, afin de n'avoir à traiter que des blocs de taille raisonnable. Cela signifie que les différents blocs obtenus après les partitions devront être ensuite alignés par paires, composées de deux blocs issus chacun d'une des deux ontologies initiales, et l'objectif consiste à minimiser le nombre de paires à aligner.

Nous proposons un algorithme de partitionnement adapté au contexte de l'alignement. Partitionner un ensemble E consiste alors à trouver des sous-ensembles $E_1, E_2, ..., E_n$, d'éléments sémantiquement proches c'est-à-dire liés par un ensemble de relations important. La réalisation de cet objectif consiste à maximiser les relations au sein d'un sous-ensemble et à minimiser les relations entre les différents sous-ensembles.

La qualité du résultat d'un partitionnement généré par notre algorithme, peut être apprécié selon différents critères :

- La taille des blocs générés : les blocs doivent avoir une taille raisonnable, i.e. inférieure au nombre d'éléments que peut traiter l'outil d'alignement.

- Le nombre de blocs générés : ce nombre doit être le plus faible possible pour limiter le nombre de paires de blocs à aligner.

- Le degré de dépendance entre les blocs : un bloc sera dit faiblement dépendant des autres si les relations (lexicales et structurelles) sont fortes à l'intérieur du bloc et faibles à l'extérieur. Ce degré groupe les éléments qui peuvent probablement s'aligner dans un nombre minimal de blocs et réduit ainsi le nombre de comparaisons à faire.

Notons que travaillant sur des ontologies légères, l'algorithme de partitionnement utilise seulement les relations de subsomption entre les concepts. Ceci permet d'envisager des traitements légers. Ainsi des ontologies de très grande taille peuvent être partitionnées. L'approche que nous proposons est donc une approche qui passe à l'échelle.

Nous présentons ci-dessous un état de l'art correspondant aux travaux sur le partitionnement. Dans la seconde partie, nous décrivons nos deux méthodes de partitionnement d'ontologies.

4.1 Etat de l'art sur le partitionnement d'ontologies

Dans les domaines d'applications réelles, les ontologies devenant de plus en plus volumineuses, de nombreux travaux [Stuckenschmidt and Klein, 2004], [Grau et al., 2005, Grau et al., 2006] et [Hu et al., 2006] se sont intéressés au problème de leur partitionnement.

Ainsi les travaux de [Stuckenschmidt and Klein, 2004] visent la décomposition d'une ontologie en sous-blocs (ou *îlots*) indépendants les uns des autres, de façon à faciliter en toute généralité différentes opérations sur les ontologies comme la maintenance, la visualisation, la validation ou le raisonnement. Les travaux de [Grau et al., 2005] s'intéressent plus particulièrement aux problèmes de raisonnement et cherchent à construire des modules centrés autour d'une sous-thématique qui soient cohérents et auto-suffisants pour raisonner. Seul [Hu et al., 2006] a pour objectif l'alignement d'ontologies mais nous verrons que sa méthode de décomposition ne prend pas complètement en compte toutes les contraintes imposées par cet objectif, en particulier le fait de travailler sur deux ontologies.

4.1.1 Des modules indépendants facilitant la gestion d'ontologies volumineuses

L'objectif de [Stuckenschmidt and Klein, 2004] est de décomposer une ontologie en blocs indépendants et cohérents. La méthode consiste à trouver des blocs (îlots) à partir d'un graphe de dépendance. Le processus de partitionnement passe par cinq étapes :

1. ***Création du graphe de dépendance***
   Le graphe de dépendance est extrait à partir du fichier source de l'ontologie. L'idée est que les éléments de l'ontologie (concepts, relations, instances) sont représentés par des nœuds du graphe. Les liens entre les nœuds sont introduits si les éléments correspondants sont liés dans l'ontologie.

2. ***Déterminer la force de dépendance***
   La force des dépendances entre les concepts est déterminée en utilisant des algorithmes d'analyse de réseau. Elle est basée sur le nombre de liens auxquels participe un nœud. L'idée est que moins un nœud est relié à d'autres nœuds, plus les liens avec les nœuds avec lesquels il est lié sont forts. Des poids peuvent être introduits pour représenter l'importance des différents types de liens. Ainsi on peut décider que les liens traduisant les relations de sous-classe sont plus importants que les autres relations du domaine.

3. ***Déterminer les modules***
   Un module ou îlot ("line island") est défini comme étant un ensemble de nœuds, de taille comprise entre des valeurs minimale et maximale données, tels que la force de chaque connexion interne à l'ensemble est supérieure à la force de chacune des connexions reliant un des nœuds de l'ensemble avec l'extérieur.

4. ***Attribuer les concepts isolés***
   Cette approche fixe une borne minimale à la taille des modules. Un mauvais choix de cette borne fait apparaître, d'après les auteurs, de très nombreux concepts isolés dont l'affectation doit être forcée dans le module avec lequel ils ont la plus forte connexion.

5. ***Fusion***
   L'algorithme a aussi tendance à construire beaucoup de petits blocs avec une très forte corrélation interne dont il faut là aussi forcer la fusion. Celle-ci est décidée si certains modules voisins sont assez fortement liés. Dans de nombreux cas, il n'existe qu'un module adjacent pour fusionner. Dans les cas où plus d'un module adjacent existe, la fusion est faite avec le voisin le plus proche, déterminé par la force des dépendances entre les modules.

Le processus de génération des modules impose une contrainte sur la taille minimale des modules générés conduisant à des regroupements pas forcément très pertinents sémantiquement. Il construit, par ailleurs, beaucoup de petits blocs. Ces deux raisons ne sont pas favorables à l'utilisation de ce processus dans un but d'alignement d'ontologies.

4.1.2 Des modules autonomes pour le raisonnement

Dans [Grau et al., 2005, Grau et al., 2006] la méthode de partitionnement vise à produire des modules autonomes dans le sens où toutes les inférences à l'intérieur d'un module peuvent être faites uniquement sur la base d'un raisonnement local [Grau et al., 2006]. La méthode comporte trois étapes de base :

1. ***Safety Check***
   Cette étape vérifie que l'ontologie est partitionnable. Une ontologie est partitionnable si elle ne contient pas d'axiomes d'inclusion dits "dangereux" (des axiomes d'inclusion qui imposent des contraintes sémantiques sur l'ensemble de l'ontologie) pour la préservation de la compatibilité sémantique de ses partitions.

2. ***Partitionnement***
   L'algorithme appliqué dans cette étape débute avec une seule partition. Il crée ensuite une nouvelle partition en y introduisant un concept ou un axiome quelconque puis tous les concepts ou axiomes dépendants, i.e. qui y font référence et peuvent être déplacés dans la nouvelle partition sans violer la condition de complétude.
3. ***Génération de module***
   Le partitionnement créé dans la deuxième étape est utilisé pour déterminer les modules. Cela se fait par la fusion des partitions au sein des modules qui se chevauchent. A ce stade, il est possible d'introduire de la redondance dans l'ontologie.

Cette méthode garantit que tous les concepts reliés par des liens de subsumption seront regroupés dans un seul module. Ceci est un inconvénient majeur pour l'alignement d'ontologies qui comportent des milliers de relations de subsomption (c'est le cas, par exemple, de AGROVOC et NALT). Les auteurs conviennent que cette méthode peut conduire à la création de très mauvaises répartitions de tailles des modules dans le cas d'ontologies "monolithiques", qui ne seront pas du tout adaptées pour l'alignement.

4.1.3 Des modules adaptés à l'alignement d'ontologies : la méthode PBM

A notre connaissance, la seule méthode adaptée à l'alignement d'ontologies volumineuse est la méthode PBM [Hu et al., 2006]. Cette méthode consiste à partitionner chaque ontologie en blocs en utilisant la méthode de clustering ROCK [Guha et al., 2000], puis à mesurer la proximité de chacun des blocs d'une ontologie avec chaque bloc de l'autre ontologie de façon à n'effectuer l'alignement qu'entre les concepts des paires de blocs les plus proches.

Pour effectuer la partition, alors que ROCK considère que les liens entre les concepts ont tous la même valeur, PBM introduit la notion de *liens pondérés* qui s'appuie sur deux mesures de similarité entre concepts, une mesure linguistique et une mesure structurelle.

Mesures de similarité

Similarité lexicale Soient d_i (resp. d_j) la chaîne de caractères correspondant à la description du concept c_i (resp. c_j) et $comm(d_i, d_j)$ (resp. $diff(d_i, d_j)$) le nombre de caractères communs

(resp. différents) dans les chaînes d_i et d_j.

La similarité lexicale entre deux concepts c_i et c_j, $Sim_L(c_i,c_j)$ se calcule comme suit :

$$sim_L(c_i, c_j) = comm(d_i, d_j) - diff(d_i, d_j) + Winkler(d_i, d_j)$$

où le terme $Winkler(d_i, d_j)$ est ajouté pour améliorer le résultat en utilisant la méthode de Winkler [Winkler, 1999] (méthode de comparaison de chaîne de caractères).

Les concepts c_i et c_j sont jugés similaires si $Sim_L(c_i, c_j) > 0.65$.

Similarité structurelle Soient c_i, c_j deux concepts d'une même ontologie O, c_{ij} leur plus petit ancêtre commun et $depthOf(c)$ la distance en nombre d'arcs entre le concept c et la racine de O. PBM mesure la similarité structurelle $aff_s(c_i, c_j)$ en utilisant la mesure de Wu et Palmer [Wu and Palmer, 1994] qui se calcule comme suit :

$$aff_s(c_i, c_j) = \frac{2 * depthOf(c_{ij})}{depthOf(c_i) + depthOf(c_j)}$$

Le calcul de similarité structurelle entre les concepts d'une ontologie de grande taille peut prendre beaucoup de temps. En considérant que seuls les concepts de profondeurs adjacentes auront des similarités élevées, PBM ne compare que les concepts qui satisfont la relation suivante :

$$|depthOf(c_i) - depthOf(c_j)| \leq 1$$

Les liens pondérés Le calcul du lien pondéré entre deux concepts, $link(c_i, c_j)$, s'effectue comme suit :

$$link(c_i, c_j) = \begin{cases} aff(c_i, c_j) & si\ aff(c_i, c_j) > \epsilon_1 \\ 0 & sinon \end{cases}$$

$$aff(c_i, c_j) = \alpha \cdot aff_s(c_i, c_j) + (1 - \alpha) \cdot Sim_L(c_i, c_j)$$

où ϵ_1 est un seuil donné tel que $\epsilon_1 \in [0, 1]$ et $\alpha \in [0, 1]$ permet à l'utilisateur de faire varier le poids relatif des mesures de similarité.

Algorithme de Partitionnement

L'algorithme permettant à PBM de partitionner une ontologie en blocs s'appuie sur deux notions essentielles : la *cohésion* au sein d'un bloc et le *couplage* entre deux blocs distincts.

La cohésion mesure la somme des poids des liens reliant les concepts appartenant à un même bloc et le couplage, la somme des poids des liens reliant les concepts appartenant à deux blocs différents.

Ces deux notions sont représentées au sein d'une même mesure dite *goodness* dont le sens varie suivant qu'elle s'applique sur un bloc unique B_i ou sur deux blocs distincts, B_i et B_j tels

que $B_i \neq B_j$:

$$Cohésion(B_i) = goodness(B_i, B_i),$$
$$Couplage(B_i, B_j) = goodness(B_i, B_j) \; avec \; B_i \neq B_j.$$

$$goodness(B_i, B_j) = \frac{\sum_{c_i \in B_i, c_j \in B_j} link(c_i, c_j)}{sizeOf(B_i) \cdot sizeOf(B_j)}$$

L'algorithme prend en entrée l'ensemble B des n blocs à partitionner, où chaque bloc est réduit au départ à un unique concept, et la taille maximale t_{max} à partir de laquelle les blocs ne doivent plus être fusionnés. PBM initialise tout d'abord de façon uniforme, la valeur de cohésion de chaque bloc ainsi que les valeurs de couplage. A chaque itération, l'algorithme choisit le bloc qui a la cohésion maximale et le bloc qui a la valeur de couplage maximale avec ce premier bloc. Il remplace ces deux blocs par celui résultant de leur fusion et met à jour les valeurs de couplage de tous les blocs avec ce nouveau bloc. L'algorithme s'arrête quand tous les blocs construits ont atteint la taille limite fixée au départ. Le pseudo-code de cet algorithme est présenté ci-dessous :

Algorithm 5 PBM(B, t_{max})

Require: $B = \{B_1, B_2, ...B_n\}$ l'ensemble des blocs à partitionner.

for each bloc B_i dans B **do**
  initialiser la valeur de la cohésion de B_i
  calculer la valeur de couplage de B_i avec tous les autres blocs
end for
while l'ensemble B is not *Empty* **do**
  choisir B_i le bloc qui a la valeur de cohésion maximale
  choisir B_j le bloc qui a la valeur de couplage maximale avec B_i
  if B_j existe **then**
    fusionner les blocs B_i et B_j dans B_p
    supprimer B_i et B_j de l'ensemble B
    if taille de $B_p > t_{max}$ / 2 **then**
      ajouter B_p à l'ensemble P
      supprimer B_p de l'ensemble B
    else
      mettre à jour les valeurs de cohésion et de couplage de B_p
      for each bloc B_k dans B sauf B_p **do**
        mettre à jour la valeur de couplage de B_k
      end for
    end if
  else
    ajouter B_i à l'ensemble P
    supprimer B_i de l'ensemble B
  end if
end while
return P l'ensemble des blocs.

Identification des paires de blocs à aligner

Une fois le partitionnement des deux ontologies réalisé, l'évaluation de la proximité des blocs s'effectue en s'appuyant sur des *ancres*, i.e. des appariements préalablement connus entre les termes des deux ontologies, définis par des techniques de comparaison de chaînes de caractères ou par un expert. Plus deux blocs contiennent d'ancres communes, plus ils sont jugés proches.

Soient k le nombre de blocs générés par la partition d'une ontologie O et B_i un de ces blocs, k' le nombre de blocs générés par la partition de la deuxième ontologie O' et B'_j un de ces blocs. Soit la fonction anchors(B_u, B'_v) qui calcule le nombre d'ancres prédéfinies partagées par deux blocs B_u et B'_v. Le nombre d'ancres contenues par un bloc B_i est donc calculé par la somme $\sum_{v=1}^{k'} anchors(B_i, B'_v)$.

La relation de *Proximité* entre deux blocs B_i et B'_j est finalement définie comme suit :

$$Proximity(B_i, B'_j) = \frac{2.anchors(B_i, B'_j)}{\sum_{u=1}^{k} anchors(B_u, B'_j) + \sum_{v=1}^{k'} anchors(B_i, B'_v)}$$

Les paires de blocs alignées sont toutes les paires ayant une proximité supérieure à un seuil $\epsilon_2 \in [0, 1]$. Un bloc pourra donc être aligné avec plusieurs blocs de l'autre ontologie ou avec aucun suivant la valeur choisie pour ce seuil.

Exemple

Nous avons appliqué l'algorithme PBM sur deux petites ontologies jouets pour visualiser son comportement et faciliter la comparaison ultérieure avec nos propres propositions.

Fig. 4.1 présente les deux ontologies O_S et O_T avant et après le processus de partitionnement en utilisant la méthode PBM. Pour effectuer ces partitionnements, nous avons utilisé une version de l'algorithme trouvée sur le web [10] qui ne se base que sur la similarité structurelle entre concepts. Les labels des concepts n'interviennent donc pas dans le traitement.

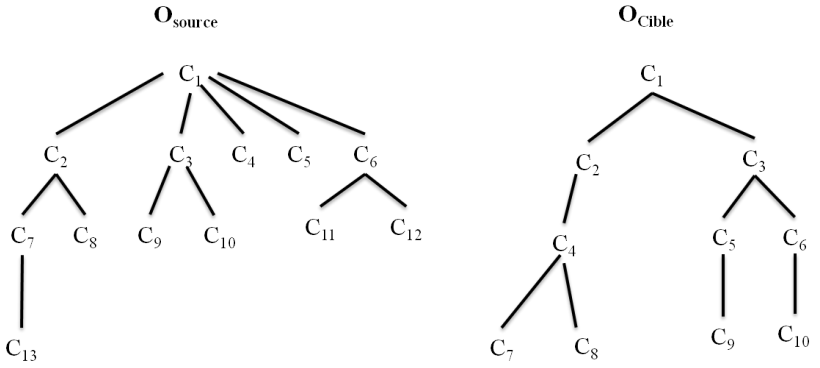

FIG. 4.1 – Les ontologies de l'expérimentation test

10. http://iws.seu.edu.cn/projects/matching/

Pour cette expérimentation, nous avons fixé cette taille limite à 4, ce qui signifie que l'algorithme a le droit de fusionner 2 blocs de taille 3 pour former au plus un bloc de taille 6. Nous sommes ainsi sûrs d'obtenir au moins 2 blocs dans O_T qui contient 10 concepts et 3 blocs dans O_S qui en contient 13.

Le partitionnement successif de O_T puis O_S fait effectivement apparaître 2 blocs pour O_T, B_{T1} et B_{T2} contenant 5 concepts chacun, et 3 blocs pour O_S, B_{S1}, B_{S2} et B_{S3} contenant respectivement 4, 3 et 6 concepts.

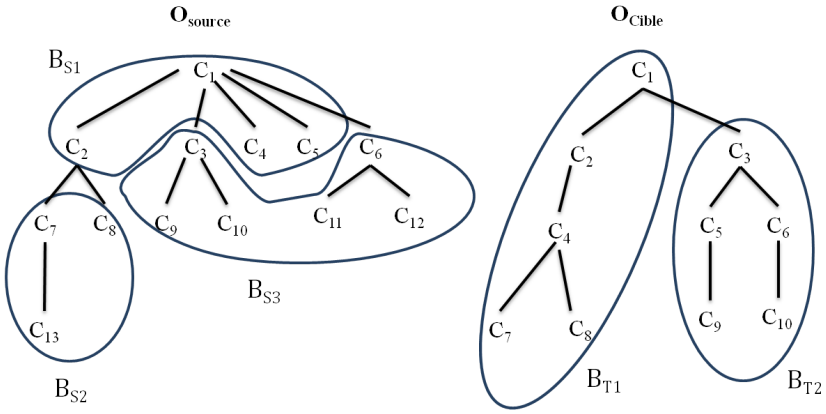

FIG. 4.2 – Les blocs construits avec la méthode PBM

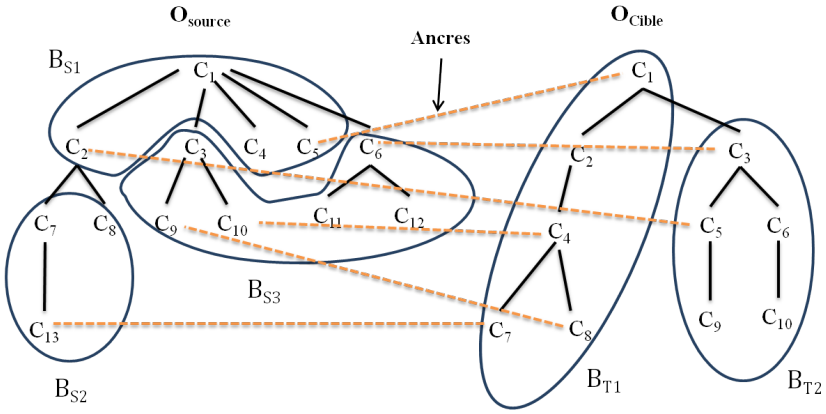

FIG. 4.3 – Les ancres partagées par les différents blocs

Le bloc B_{S1} contient 2 ancres, l'une partagée avec B_{T1}, l'autre avec B_{T2}. Le bloc B_{S2} n'en contient qu'une partagée avec B_{T1}, et le bloc B_{S3} en contient 3, 2 partagées avec B_{T1}, la troisième avec B_{T2}.

Le calcul de proximité fondé sur les ancres partagées par les deux ontologies doit être effectué sur toutes les paires de blocs possibles (ici, 6 paires).

Le nombre de paires effectivement alignées dépend de la valeur fixée pour le seuil : plus celui-ci est bas, plus on multiplie les alignements et les chances de retrouver des appariements, mais

plus on augmente aussi le temps d'exécution. Si le seuil est élevé, on passe moins de temps à aligner des blocs éloignés mais on perd peut être des appariements potentiels.

Par exemple, la paire (B_{S1},B_{T1}) ne partageant qu'une ancre alors que les blocs en contiennent respectivement 2 et 4, sa proximité est égale à 0.33. Si le seuil est fixé à 0.5, la paire n'est pas alignée et l'ancre partagée n'est pas retrouvée dans les appariements. Si le seuil est fixé plus bas, on retrouvera toutes les ancres dans les appariements, mais on devra aligner toutes les paires de blocs possibles à l'exception de la paire (B_{S2},B_{T2}) qui ne partage pas d'ancre.

Cette méthode permet de décomposer des ontologies volumineuses, mais cette décomposition est faite a priori, sans prendre en compte l'objectif d'alignement, en s'appliquant sur chaque ontologie indépendamment l'une de l'autre. Pour effectuer l'alignement, PBM doit d'abord identifier quels sont les blocs les plus proches qui doivent être alignés entre eux et pour cela calculer la proximité des différents blocs en s'appuyant sur les couples d'ancres préalablement connues. Le partitionnement ayant été fait au départ à l'aveugle, certaines ancres pourront ne pas se trouver dans des blocs finalement alignés et l'alignement résultant ne comprendra pas forcément tous les appariements possibles.

Enfin, le calcul des blocs pertinents à aligner est coûteux (en temps de traitement).

Malgré ces critiques, l'algorithme de décomposition de PBM est le plus adapté à la tâche d'alignement puisqu'il permet de contrôler la taille maximale des blocs construits.

La proposition que nous faisons prend appui sur cet algorithme. Elle peut être vue comme une adaptation qui modifie la façon de générer les blocs en prenant en considération, au plus tôt lors du partitionnement, les relations existant entre les concepts des deux ontologies.

4.2 Méthodes de partitionnement proposées

Nous proposons deux méthodes différentes [Hamdi et al., 2009c, Hamdi et al., 2009b] qui seront décrites dans ce qui suit. Pour prendre en compte, au plus tôt, l'objectif d'alignement, ces méthodes s'appuient sur deux éléments : d'une part les couples de concepts issus des deux ontologies qui ont exactement le même label et peuvent être reliés par une relation d'équivalence, d'autre part l'asymétrie structurelle possible des deux ontologies à aligner.

Même avec de grandes ontologies, il est possible d'identifier, avec une mesure de similarité stricte et peu coûteuse à calculer, des concepts qui ont un label en commun dans les ontologies. Comme dans PBM, nous appelons ces paires de concepts *ancres*. Nous allons les utiliser pour générer des partitions.

L'asymétrie structurelle des deux ontologies peut servir à ordonner les ontologies à partitionner. Si une ontologie est plus structurée que l'autre, il sera plus facile de la décomposer en blocs avec une forte cohésion interne. Ainsi sa décomposition peut servir comme un guide pour la décomposition de l'autre ontologie. C'est cet élément qui nous a conduit à faire deux propositions de méthodes d'alignement.

La première méthode que nous proposons, appelée PAP (Partition, Anchor, Partition), consiste d'abord à décomposer l'ontologie la plus structurée dite cible O_T, puis à utiliser les ancres identifiées, pour forcer le partitionnement de la deuxième ontologie dite source O_S à être en accord avec le partitionnement de O_T. Cette première méthode casse partiellement la structure de la source de O_S. Ceci n'est pas un problème quand l'ontologie source est mal structurée.

Lorsque les deux ontologies sont structurées selon le même point de vue, la méthode PAP est inadaptée. Nous proposons alors une autre méthode de partitionnement, appelée APP (Anchor, Partition, Partition), qui prend en compte la structure des *deux* ontologies. La méthode APP partitionne O_T en favorisant la fusion de blocs partageant des ancres avec O_S, et partitionne O_S en favorisant la fusion des blocs partageant des ancres avec le même bloc de O_T.

Les sections suivantes présentent successivement la mesure de similarité utilisée pour calculer les ancres à moindre coût, puis la description détaillée des deux méthodes. La validation de ces deux méthodes a été faite de façon expérimentale dans le chapitre 6 de cette thèse.

4.2.1 Mesure de similarité lexicale

Pour calculer les ancres, nous utilisons une mesure de similarité lexicale stricte. Nous disons que deux concepts sont équivalents seulement si leurs labels sont parfaitement identiques. Aucun prétraitement (filtrage, tokenisation, lemmatisation) n'est effectué sur ces labels. Nous définissons cette mesure comme suit :

$$sim_{light}(Label(c_i), Label(c_j)) = \begin{cases} 1 & si\ Label(c_i) = Label(c_j) \\ 0 & sinon \end{cases}$$

c_i, c_j : deux concepts tels que $c_i \in O_i$ et $c_j \in O_j$

4.2.2 Méthode PAP

La méthode PAP partitionne O_S en tenant compte du partitionnement de O_T. Elle comprend quatre étapes, en plus du calcul des ancres :

1. Partitionner l'ontologie cible O_T en plusieurs blocs B_{Ti} en utilisant l'algorithme PBM.

2. Identifier, dans chacun des blocs construits pour O_T, l'ensemble des ancres appartenant à ce bloc. Chacun de ces ensembles constituera le noyau ou *centre* CB_{Si} d'un futur bloc B_{Si} à générer dans l'ontologie source O_S.

3. Partitionner l'ontologie source autour des *centres* CB_{Si} identifiés dans l'étape précédente.

4. Aligner chaque bloc de O_S avec le bloc de O_T correspondant.

Déterminer les blocs de O_T

Pour déterminer les blocs de l'ontologie cible O_T, nous utilisons l'algorithme PBM tel qu'il est implémenté dans le code que nous avons trouvé sur le web [11]. La différence entre cet algorithme et celui décrit dans la présentation théorique faite en section 4.1.3 à partir de l'article [Hu et al., 2006], est que le lien pondéré calculé entre les concepts ne dépend que de la similarité structurelle ($\alpha = 1$ dans l'équation suivante). En effet, le calcul de la similarité linguistique prend beaucoup de temps, en plus, les concepts d'une même ontologie ont souvent des labels différents.

$$aff(c_i, c_j) = \alpha \cdot aff_s(c_i, c_j) + (1 - \alpha) \cdot Sim_L(c_i, c_j)$$

Déterminer les centres de O_S

Les centres des futurs blocs de l'ontologie source O_S sont déterminés en se basant sur deux critères : les couples d'ancres identifiés entre O_S et O_T, et les blocs B_{Ti} construits à partir de l'ontologie cible O_T. Pour chaque bloc B_{Ti} construit à l'étape précédente, nous regroupons les concepts de O_S qui ont des équivalents dans ce bloc en un paquet, CB_{Si}, en utilisant l'algorithme suivant :

Algorithm 6 PAP_Centre(T, E, S)

Require: $C = \{B_{T1}, B_{T2}, ...B_{Tn}\}$ l'ensemble des blocs de l'ontologie cible O_T.
Require: $E = \{E_1, E_2, ...E_m\}$ l'ensemble des couples de labels identiques trouvés, tels que $E_i = (c_{Si}, c_{Tj})$, où c_{Si} est un concept de l'ontologie source O_S et c_{Tj} est un concept de l'ontologie cible O_T.

```
for each bloc B_Ti dans T do
    initialiser CB_Si = ∅
    for each concept c_Tk dans B_Ti do
        for each équivalence E_j dans E do
            if c_Tk ∈ E_j then
                CB_Si = CB_Si ∪ c_Sk
            end if
        end for
    end for
end for
return S = {CB_S1, CB_S2, ...CB_Sn} les centres des futurs blocs de l'ontologie source O_S.
```

Partition de O_S autour des centres CB_{Si}

Après l'identification des centres des futurs blocs de O_S, nous appliquons l'algorithme PBM avec la différence suivante. Au lieu d'introduire en entrée l'ensemble des m concepts de l'ontologie comme m blocs réduits chacun à un unique concept, nous introduisons les n centres identifiés à l'étape précédente, comme autant de blocs distincts mais regroupant plusieurs concepts. Les autres concepts de O_S qui n'ont pas d'équivalents dans O_T donnent chacun naissance à un bloc individuel. La cohésion des blocs représentant les centres de O_S est initialisée avec la valeur de

11. `http://iws.seu.edu.cn/projects/matching/`

cohésion maximale.

Identification des blocs à aligner

La phase d'identification des paires de blocs à aligner est ici immédiate et sans calcul. Chacun des blocs B_{Si} construits à partir d'un centre n'est aligné qu'avec le bloc B_{Ti} correspondant. L'algorithme peut mener à la constitution de blocs B_{Sj} indépendants des centres, i.e. ne contenant pas d'ancres et qui, dans l'état courant de notre implémentation, ne sont pas pris en compte dans le processus d'appariement. En effet, ce bloc ne contenant pas d'ancres, nous n'avons pas encore étudié d'heuristiques permettant de choisir les blocs B_{Ti} avec lesquels on pourrait essayer de le rapprocher.

Exemple

Nous avons appliqué la méthode PAP sur les ontologies de l'exemple présenté en Fig. 4.1. La génération des blocs de la cible s'effectue comme dans la méthode PBM (Fig. 4.4).

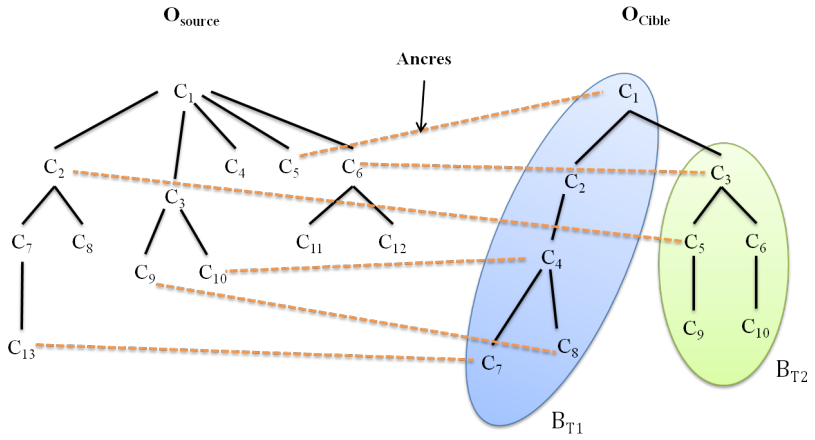

FIG. 4.4 – Génération des blocs de la cible identique à celle de PBM

A partir des blocs générés pour la cible B_{T1} et B_{T2}, l'algorithme identifie les centres des futurs blocs de l'ontologie source ; CB_{S1} : $\{c_5, c_9, c_{10}, c_{13}\}$ et CB_{S2} : $\{c_2, c_6\}$ (Fig. 4.5).

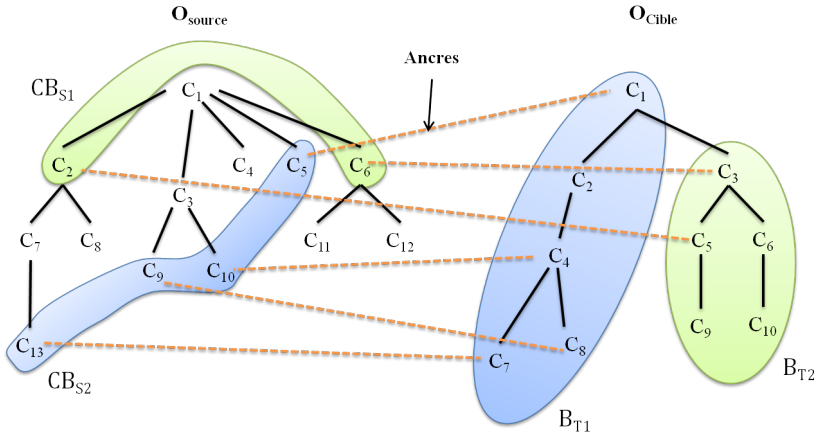

FIG. 4.5 – Identification des centres des blocs de l'ontologie source

La figure 4.6 montre les blocs construits après le partitionnement de O_S à partir de ces centres. Le test sur la taille maximale des blocs construits ne s'effectuant pour les centres qu'après l'exécution d'une première fusion, le bloc B_{S1} contiendra 7 concepts au lieu de 6, mais ne peut plus être fusionné avec un autre bloc. De même, après une première fusion, le bloc B_{S2} ne peut plus être fusionné puisque la taille atteinte (4 concepts) est la taille limite. Le partitionnement fait donc apparaître un bloc sans ancre, qui ne sera pas aligné. Les paires devant être alignées (B_{S1}, B_{T1}) et (B_{S2}, B_{T2}) sont immédiatement identifiables par construction.

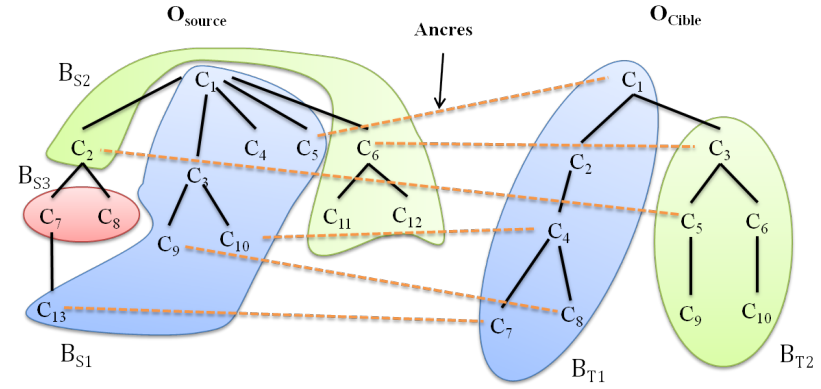

FIG. 4.6 – Génération des blocs de la source à partir des centres constitués précédemment

4.2.3 Méthode APP

L'idée de cette seconde méthode est de partitionner les deux ontologies en même temps, c.à.d. de faire du co-clustering. Le problème est que nous ne pouvons pas traiter réellement ces ontologies en parallèle du fait de leur grande taille. Pour simuler le parallélisme, nous partitionnons l'ontologie cible en nous fondant sur toutes les équivalences identifiées avec la source, et nous partitionnons ensuite la source en nous fondant sur une partie des équivalences trouvées dans les blocs générés pour la cible. Dans cette méthode les deux ontologies cible et source sont supposées être structurées selon le même point de vue.

Prendre en compte les relations d'équivalence identifiées entre les ontologies dès le partitionnement de O_T, doit permettre par la suite de faciliter la recherche des paires de blocs les plus proches et d'améliorer les résultats de l'alignement. Ainsi ce partitionnement, contrairement à celui de PBM ou à celui implémenté dans la méthode PAP, constitue, pour les deux ontologies, des blocs qui regroupent un maximum d'ancres et qui sont structurellement proches.

Cette deuxième méthode comprend trois étapes :

1. Partitionner la première ontologie O_T en utilisant l'algorithme PBM mais en prenant en compte l'ensemble des ancres lors de la génération des blocs.

2. Partitionner la deuxième ontologie O_S de la même manière mais en constituant des blocs contenant des ancres qui appartiennent à un même bloc de l'ontologie O_T.

3. Aligner les paires de blocs partageant le plus d'ancres.

Déterminer les blocs de O_T

Pour déterminer les blocs de l'ontologie cible O_T, nous utilisons l'algorithme PBM en modifiant la définition de la mesure de *goodness* pour prendre en compte les liens entre les deux ontologies. Nous lui ajoutons un coefficient qui représente la proportion d'ancres présentes dans un bloc de O_T relativement au nombre d'ancres total identifiées avec l'ontologie O_S.

De ce fait, au cours de la génération des blocs, le choix du bloc qui a la valeur maximale de cohésion ou de couplage ne dépend pas seulement des relations des concepts à l'intérieur ou à l'extérieur des blocs d'une même ontologie, mais aussi des relations d'équivalences identifiées avec l'autre ontologie.

L'équation de goodness devient :

$$goodness(B_i, B_j) = \alpha\left(\frac{\sum_{c_i \in B_i, c_j \in B_j} link(c_i, c_j)}{sizeOf(B_i) \cdot sizeOf(B_j)}\right) + (1-\alpha)\left(\frac{\sum_{c_j \in B_j, c_k \in O_S} sim_{light}(c_j, c_k)}{\sum_{c_n \in O_C, c_m \in O_S} sim_{light}(c_n, c_m)}\right)$$

où $\alpha \in [0,1]$, B_i et B_j sont deux blocs de O_T, $\sum_{c_j \in B_j, c_k \in O_S} sim_{light}(c_j, c_k)$ représente le nombre d'ancres présentes dans B_j et $\sum_{c_n \in O_T, c_m \in O_S} sim_{light}(c_n, c_m)$, le nombre d'ancres total.

Déterminer les blocs de O_S

Les blocs de l'ontologie source O_S sont générés en se basant sur les poids des liens entre les concepts de O_S, les relations d'équivalences entre les deux ontologies et les blocs de l'ontologie O_T.

Si nous considérons un bloc B_i dans O_S avec une valeur de cohésion maximale, le calcul de goodness pour trouver le bloc ayant la valeur de couplage maximale avec B_i se base non seulement sur les relations structurelles dans O_S, mais aussi sur les relations d'équivalences avec le bloc de O_T qui a le maximum d'équivalents avec B_i.

L'équation de goodness devient :

$$goodness(B_i, B_j) = \alpha(\frac{\sum_{c_i \in B_i, c_j \in B_j} link(c_i, c_j)}{sizeOf(B_i) \cdot sizeOf(B_j)}) + (1-\alpha)(\frac{\sum_{c_j \in B_j, c_k \in B_k} sim_{light}(c_j, c_k)}{\sum_{c_n \in O_C, c_m \in O_S} sim_{light}(c_n, c_m)})$$

où $\alpha \in [0, 1]$, B_i et B_j sont deux blocs distincts de O_S et où B_k est le bloc de O_T qui partage le plus d'ancres avec B_i.
Lors des opérations de fusion, nous conservons pour chaque bloc de O_S l'ensemble des ancres appartenant au bloc, ce qui facilitera lors de la recherche des blocs à aligner, l'identification des blocs partageant le plus d'ancres.

Identification des blocs à aligner

La conservation des ancres introduites dans chaque bloc facilite la phase de calcul des paires de blocs partageant le plus d'ancres, un bloc de O_S ne s'alignant qu'avec un seul bloc de O_T (cf. Algorithme 7). Une fois identifiés les blocs de O_S devant être alignés avec le même bloc de O_T nous pouvons éventuellement les fusionner si leur taille le permet, afin de diminuer le nombre de combinaisons à faire lors du processus d'alignement.

Algorithm 7 APP_Align(S, T, P)

Require: $C = \{B_{T1}, B_{T2}, ...B_{Tn}\}$ l'ensemble des blocs de l'ontologie cible O_T.
Require: $S = \{B_{S1}, B_{S2}, ...B_{Sn}\}$ l'ensemble des blocs de l'ontologie source O_S.

```
for each bloc B_Ci dans T do
    initialiser P_i = ∅
    for each bloc B_Sj dans S do
        if B_Ti est le bloc avec lequel B_Sj partage le plus d'ancres then
            P_i = P_i ∪ B_Sj
            Supprimer B_Sj de S
        end if
    end for
end for
return P = {(P_1, B_T1), (P_2, B_T2)...(P_m), B_Tm} où les P_i représentent l'ensemble des blocs
B_Sj devant être alignés avec un bloc B_Ti.
```

Exemple

Fig. 4.7 et Fig. 4.8 présentent également des résultats obtenus sur l'exemple précédent présenté en Fig. 4.1.

Fig. 4.7 montre les blocs construits à partir de O_T selon la méthode APP, favorisant le regroupement en tenant compte des ancres. Fig. 4.8 montre les blocs construits à partir de O_S, favorisant la constitution de blocs partageant des ancres avec ceux de O_T tout en prenant en compte la structure de O_S.

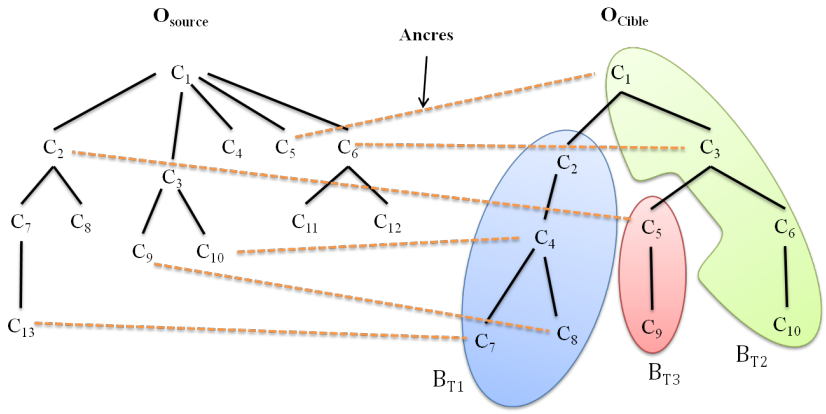

FIG. 4.7 – Génération des blocs de la cible

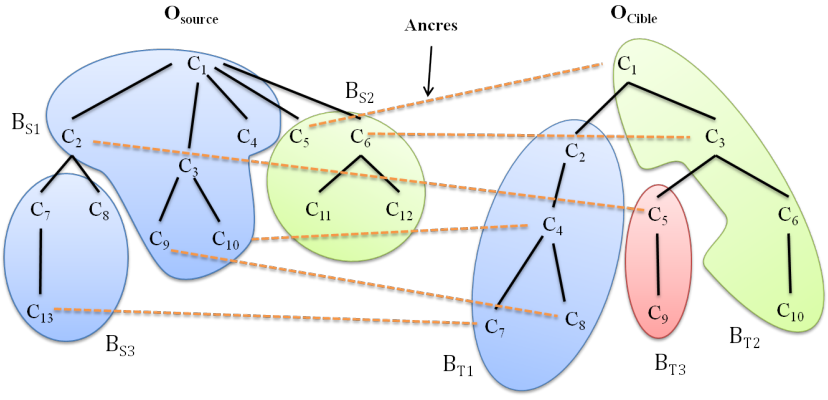

FIG. 4.8 – Génération des blocs de la source

L'alignement s'effectue entre les blocs qui partagent le plus d'ancres : B_{S2} et B_{S3} sont alignés avec B_{T1} et B_{S1} avec B_{T2}. B_{T3} ne participe pas à l'alignement puisque qu'il ne contient qu'une ancre partagée avec B_{S2}, qui partage davantage d'ancres avec B_{T2} qu'avec B_{T3}. Nous perdons donc ici l'appariement d'une ancre, (c_2, c_5), mais nous diminuons les temps d'alignement et avons construit, par co-clustering, des blocs prenant plus en considération les relations partagées entre les deux ontologies.

Conclusion

Comme les outils actuels d'alignement d'ontologies perdent leur efficacité sur de grandes ontologies, l'objectif de ce travail était d'étudier les techniques de partitionnement d'ontologies

orientées alignement.

Ce chapitre décrit deux méthodes utilisant l'algorithme de partitionnement d'ontologies PBM, développé pour le système d'alignement FALCON. Ces méthodes sont présentées par comparaison à la méthode PBM. Cette description est complétée, dans le chapitre 5, par la présentation de l'utilisation de la méthode de partitionnement PAP dans un contexte applicatif différent. Les expérimentations ainsi réalisées permettant de valider nos propositions en matière de partitionnement.

Au lieu d'appliquer l'algorithme, comme PBM, successivement et indépendamment sur chaque ontologie, nous prenons en compte dès que possible dans le processus de partitionnement le contexte de la tâche d'alignement. Nos méthodes sont appliquées sur les deux ontologies simultanément, et utilisent les données d'alignement. Ces données d'alignement sont faciles à extraire, même à partir de grandes ontologies. Elles comprennent des paires de concepts, un concept de chaque ontologie, qui ont le même label, et des informations structurelles sur les ontologies à aligner.

La méthode PAP est bien adaptée pour les ontologies qui ont une structure dissymétrique ou des structures correspondant à des décompositions selon des points de vue différents. Cette méthode commence par décomposer l'ontologie la mieux structurée ou celle dont le point de vue de décomposition est retenu pour la construction des partitions et ensuite force la décomposition de l'autre ontologie à suivre le même modèle. Nous utilisons cette méthode dans le chapitre 5 de cette thèse pour partitionner une ontologie de très grande taille, YAGO, dans le but d'enrichir une deuxième ontologie de plus petite taille.

La méthode APP peut être appliquée lorsque les deux ontologies sont bien structurées et que leur structuration est faite selon un même point de vue. Cette méthode favorise la génération de blocs de concepts comparables sémantiquement car contenant des éléments majoritairement liés par des liens d'équivalence.

Le fait que les algorithmes de partitionnement que nous proposons utilisent seulement des données faciles à obtenir via un traitement peu coûteux, permet à des ontologies de très grande taille d'être partitionnées. C'est donc une démarche qui passe à l'échelle. Nos méthodes ont été testées sur différents couples d'ontologies. Les résultats présentés dans le chapitre 6 de cette thèse, montrent, en particulier, que les deux méthodes proposées génèrent des partitions assez cohérentes sémantiquement tout en limitant le nombre de blocs générés et les concepts isolés.

Chapitre 5

L'alignement utilisable pour différentes tâches d'ingénierie ontologique : illustration sur la tâche d'enrichissement

Sommaire

Introduction

L'environnement TaxoMap Framework présenté au chapitre 3 nous permet de mettre en œuvre aisément des traitements basés sur des résultats d'alignement d'ontologies. Les traitements étant spécifiés de façon déclarative et générique en utilisant un langage de patrons MPL basé sur un vocabulaire prédéfini, l'environnement est extensible et a priori applicable à n'importe quel traitement basé sur des résultats d'alignement comme le raffinement de mappings, l'enrichissement, la restructuration ou la fusion d'ontologies. Nous nous intéresserons ici à la tâche d'enrichissement d'ontologies.

Une ontologie peut être enrichie par de nouveaux concepts, par des définitions et des propriétés formelles décrivant les concepts ou par l'introduction de nouvelles relations, ces ajouts pouvant se cumuler. Nous nous intéressons ici à l'enrichissement par introduction de nouveaux concepts dans une ontologie dite cible. Dans ce cas, la tâche d'enrichissement est alors souvent présentée comme composée de deux phases : l'une consiste en l'identification à partir de ressources externes des concepts pertinents à introduire et l'autre en leur placement, avec les bonnes relations, au sein de l'ontologie cible.

Pour réaliser ces deux phases, notre approche consiste à utiliser les résultats d'alignement entre une ressource externe (dite source) et l'ontologie à enrichir (dite ontologie cible). Les concepts de la source qui apparaissent dans l'alignement produit peuvent (pour certains d'entre eux) être considérés comme des concepts candidats pour l'enrichissement et les relations établies dans les mappings être pérennisées.

Ce chapitre est structuré de la façon suivante. En section 5.1 nous décrivons différents travaux portant sur l'enrichissement par de nouveaux concepts. En section 5.2, nous étudions l'apport des techniques d'alignement existant dans TaxoMap pour l'enrichissement et l'impact des caractéristiques de la ressource externe utilisée comme source d'enrichissement sur l'efficacité de ces techniques. Nous verrons ainsi que, quelles que soient les ressources externes utilisées comme source pour l'enrichissement, différents traitements complémentaires doivent être mis en œuvre, comme des phases de nettoyage des concepts candidats quand ceux-ci ont été construits par des méthodes automatiques, des phases de validation des relations d'équivalence établies entre les concepts ou encore des phases d'affinement des placements des concepts candidats dans l'ontologie cible.

Nous présentons ensuite quelques-uns des traitements complémentaires identifiés comme nécessaires suivant les différents contextes d'enrichissement expérimentés respectivement en 5.3 pour une ontologie de même domaine et de même taille que l'ontologie cible, en 5.4 pour une ontologie de taille réduite et issue d'une source généraliste et en 5.5 pour une ontologie généraliste très volumineuse.

5.1 Etat de l'art sur l'enrichissement d'ontologies

L'enrichissement d'ontologies est un processus qui consiste à étendre une ontologie par l'ajout de nouveaux éléments : des concepts, des propriétés, des relations, des axiomes. Ce processus est composé de deux phases principales : (1) l'identification des éléments pouvant enrichir une onto-

logie, (2) l'identification de leur place dans l'ontologie et leur insertion. Ainsi, l'état de l'art que nous présentons distingue les approches portant sur l'extraction d'éléments pertinents pour l'enrichissement (étape 1 du processus) des approches visant plus particulièrement leur représentation dans l'ontologie enrichie (étape 2 du processus). Cet état de l'art sera suivi d'un positionnement de notre approche d'enrichissement d'ontologie par rapport à l'ensemble des travaux cités.

5.1.1 Les approches d'extraction des éléments candidats à l'enrichissement

Le éléments (concepts, relations, propriétés, axiomes) candidats à l'enrichissement d'une ontologie peuvent être extraits de sources de données de nature différente allant des documents textuels non structurés à des sources de données plus structurées : des pages web, des documents XML, des ressources termino-ontologiques, des bases de données [Charlet et al., 2008] [Maedche and Staab, 2001]. Les textes sont des ressources très riches en connaissances. L'analyse de textes a toujours été très présente en ingénierie des connaissances. Avec le déploiement des ontologies, elle a trouvé un champ d'application privilégié. Nous décrirons, dans un premier temps, les approches d'extraction d'éléments par analyse de textes pour enrichir des ontologies. Dans un second temps, nous nous intéresserons aux approches adoptées en exploitant des sources plus structurées.

Les approches basées sur l'analyse de documents textuels

Les approches traitant l'extraction de termes candidats à partir de documents textuels sont basées soit sur des méthodes statistiques soit sur des méthodes dites linguistiques.

Les méthodes statistiques étudient les régularités, les redondances, les co-occurrences des termes en analysant, par exemple, leur distribution dans un corpus [Faatz and Steinmetz, 2002] ou en calculant leur probabilité d'occurrence [Velardi et al., 2001] [Xu et al., 2002] [Parekh et al., 2004]. L'idée sous-jacente est de repérer des mots ou des groupes de mots ayant des contextes linguistiques analogues dans un document pris dans sa globalité. Nous présentons, ci-dessous, quelques travaux illustrant différentes techniques, plus ou moins complexes, relevant des méthodes statistiques.

L'approche, présentée dans [Parekh et al., 2004], extrait pour chaque mot vedette d'un glossaire ou d'un dictionnaire, des termes qui apparaissent dans au moins deux tiers des définitions de ce mot vedette dans tout le glossaire/dictionnaire. L'utilisateur se voit ainsi retourner un ensemble présenté sous la forme : mot de la définition, mots apparaissant plus de trois fois dans la définition. Cet ensemble est utilisé par l'expert du domaine pour enrichir manuellement une ontologie du domaine du corpus : intégrer de nouveaux concepts, ajouter de nouveaux attributs aux concepts existants ou mettre à jour la hiérarchie de concepts.

Dans [Xu et al., 2002], l'approche proposée est basée sur une extension de la mesure TFIDF [Salton, 1991], appelée KFIDF, qui calcule l'importance d'un terme dans un document par rapport à l'ensemble des documents. La mesure KFIDF s'applique au corpus contenant des documents classifiés. Ainsi, un terme est considéré comme pertinent, dans une classe, s'il apparaît plus fréquemment que les autres termes dans cette classe, et moins fréquemment dans d'autres. Dans cette approche, comme seuls les adjectifs, les noms et les verbes sont considérés comme des

candidats potentiels, une phase d'étiquetage grammatical est exécutée en amont pour associer aux mots du corpus leur fonction grammaticale.
Après extraction des termes les plus pertinents de chaque classe, [Xu et al., 2002] identifie les co-occurrences entre ces termes. Pour cela, plusieurs mesures d'association (Mutual Information[Church and Hanks, 1989], Log-Likelihood et T-test [Daille, 1996], [Manning and Schütze, 1999]) sont utilisées.

L'approche [Velardi et al., 2001] s'intéresse au problème des termes qui ne sont pas détectés, car ils n'apparaissent que dans un seul document du domaine. Par exemple, alors que le terme "zone" est fréquemment mentionné dans de nombreux documents, des termes plus spécifiques comme "station essence" et "aire de repos" ont certes une fréquence élevée au sein de certains documents, mais sont complètement absents d'autres documents.
Pour considérer ces termes, [Velardi et al., 2001] propose une mesure dite de "Pertinence du domaine" qui extrait les termes propres à un domaine en prenant en compte, en plus de leur fréquence d'apparition dans le corpus global, le nombre de documents dans lesquels ils apparaissent.

Pour extraire, à partir d'un corpus, les termes apparaissant fréquemment dans la description des concepts d'une ontologie à enrichir, [Faatz and Steinmetz, 2002] utilise également la notion de co-occurrence. En calculant une distance entre les concepts de l'ontologie existante (une distance sémantique entre les concepts, basée sur les interconnexions relationnelles entre les concepts dans la hiérarchie de l'ontologie), la méthode tente de trouver de nouveaux concepts proches au sein d'un corpus de documents. L'approche ne porte que sur les concepts pouvant être reliés aux concepts existants de l'ontologie par une relation de généralisation/spécialisation.

Les méthodes dites linguistiques, quant à elles, étudient les formulations présentes dans les textes et détectent les groupes nominaux ou verbaux ayant une cohérence forte. Ces méthodes sont basées sur l'hypothèse que les dépendances grammaticales reflètent des dépendances sémantiques. Par exemple, le "verbe" d'une phrase peut correspondre à l'énoncé d'une relation entre le concept "sujet" et le concept "complément". Ceci permet d'identifier des relations entre concepts. Une cohérence forte entre des mots peut également correspondre à l'identification d'une expression désignant un concept. Là encore les techniques utilisées sont variées.

[Hearst, 1992] introduit l'idée d'expressions régulières syntaxiques afin d'extraire des relations sémantiques et taxonomiques. La méthode implique que le système comporte une liste exhaustive des expressions régulières intéressantes ; cette liste est constituée manuellement.

L'approche décrite dans [Maedche and Staab, 2000a] propose d'utiliser des patrons syntaxiques pour générer, à partir d'un dictionnaire, des relations taxonomiques entre concepts : le mot défini est associé à un concept, les termes de la définition sont associés à des concepts candidats. Les patrons génèrent ainsi des relations entre des concepts utilisées pour enrichir une ontologie. [Maedche and Staab, 2000b] suppose que toute dépendance grammaticale induit probablement une relation. Ainsi, tous les couples de termes liés par une fonction grammaticale seront extraits. Par exemple, à partir de la phrase "auberge de jeunesse", le couple (auberge, jeunesse) sera constitué, puisque le mot "de" induit une relation potentielle entre "auberge" et "jeunesse" et donc les concepts associés.

Notons en conclusion de ces types d'approches exploitant des documents textuels que la qualité du corpus de documents utilisé détermine grandement la qualité des résultats obtenus.

Construire un bon corpus associé au domaine d'étude est donc une étape importante à réaliser avec beaucoup de soin avant la phase d'extraction. Citons, à titre d'illustration de ces travaux, l'approche proposée dans [Agirre et al., 2000] dirigée par les concepts de l'ontologie qui doit être enrichie. L'ontologie est utilisée pour générer des requêtes. Les réponses aux requêtes sont des documents textuels pertinents par rapport au sens des mots désignant des concepts dans l'ontologie. Les documents récupérés sont organisés en collections, une collection par sens de mot. Pour chaque collection, les termes sont extraits, leurs fréquences sont calculées et comparées à celles des termes dans d'autres collections qui couvrent d'autres sens du mot. Les termes qui apparaissent très fréquemment dans une collection et peu dans d'autres constituent la signature thématique [Hovy and Lin, 1998] [Lin and Hovy, 2000] du sens du mot associé à la collection.

Les approches basées sur des sources semi-structurées ou structurées

L'extraction d'éléments candidats à l'enrichissement peut se faire également à partir de données plus structurées qui peuvent être, entre autres, des pages HTML, des schémas XML, des thésaurus ou des ontologies, des schémas de bases de données. Lorsqu'il s'agit de découvrir des connaissances à partir de données, en étudiant par exemple la présentation de données, des techniques de fouille de données ou de fouille du web peuvent être utilisées. En revanche, l'exploitation d'ontologies correspondant à des représentations sémantiques nécessite plutôt des techniques d'évaluation de la pertinence des éléments composant ces ontologies pour enrichir une autre ontologie.

Les travaux qui traitent des pages HTML exploitent la structure des fichiers HTML pour identifier des termes candidats à l'enrichissement. Ainsi, les polices, les couleurs, ou le niveau de titre (titre, sous-titre, etc.), sont des critères utilisés dans des algorithmes de regroupement. Les travaux présentés ci-dessous illustrent deux approches possibles.

L'approche proposée dans [Karoui et al., 2004] consiste à stocker des informations associées aux mots extraits : le label du mot, son type grammatical (nom, adjectif, etc.), son style (titre, gras, etc.), le nombre de fois que ce style apparaît dans le document HTML et le nombre de documents du corpus étudié contenant le mot. Une méthode de regroupement exploitant les critères précédemment cités est ensuite appliquée pour identifier les concepts candidats à l'enrichissement.

L'approche OntoMiner présentée dans [Davulcu et al., 2003] et [Davulcu et al., 2004] transforme la structure d'une page web HTML en un document semi-structuré XML à l'aide d'un algorithme de partitionnement hiérarchique. OntoMiner utilise ensuite, dans un second temps, un algorithme de fouille d'arbres pour identifier les concepts les plus importants du domaine, ainsi que les relations entre eux.

Ces approches tentent de découvrir des éléments sémantiques à partir d'indicateurs de format. Au contraire, les approches qui exploitent des ontologies disposent déjà d'éléments sémantiques. Le problème d'extraction d'éléments à partir des ontologies ne se pose donc pas de la même manière. Les éléments composant l'ontologie source utilisée pour l'enrichissement sont des éléments d'ontologie et, en tant que tels, sont directement intégrables dans une autre ontologie. Il n'y a pas de problèmes de découverte des éléments qui pourraient correspondre à des éléments d'ontologie. Ce sont déjà des composants d'une ontologie et donc, a priori, des candidats potentiels à l'enrichissement d'une autre ontologie.

Le challenge, dans le cas de l'exploitation d'ontologies, ne consiste donc pas à déterminer les éléments ontologiques présents dans une source externe mais à évaluer la pertinence d'intégrer les éléments d'une ontologie dans une autre. Ces éléments sont-ils redondants par rapport aux éléments déjà présents dans l'ontologie à enrichir ? Ces éléments génèrent-ils des incohérences ou sont-ils au contraire complémentaires des éléments de l'ontologie cible ? Ainsi, [Pammer et al., 2009] [Palmisano et al., 2008] étudient l'impact de l'ajout d'un élément à une ontologie et [Ji et al., 2009] étudie la cohérence dans une ontologie après un ajout. D'autres travaux, que nous présentons ci-dessous, évaluent la pertinence des éléments à ajouter par un rapport à un contexte applicatif.

Ainsi, dans l'approche proposée par [Zablith et al., 2010], l'ontologie est considérée comme un ensemble de triplets RDF manipulé en tant que graphe. [Zablith et al., 2010] évalue la pertinence de l'introduction d'un nouveau triplet RDF <sujet, relation, objet> en comparant son contexte (construit à partir des ontologies disponibles sur le web accessibles à l'aide de l'outil Scarlet [Sabou et al., 2008]) avec celui de l'ontologie à enrichir de façon à identifier les concepts partagés, i.e. de mêmes labels. Deux techniques sont utilisées : (1) dans la première technique, la pertinence est le rapport entre le nombre de concepts partagés et le nombre de concepts de l'ontologie cible, (2) dans la deuxième technique la calcul de la pertinence prend en compte non seulement les concepts partagés, mais aussi la façon dont ces concepts sont reliés les uns aux autres. La mesure de la pertinence est comparée ensuite à un seuil qui permet de décider si un triplet est pertinent ou pas pour l'enrichissement.

5.1.2 Les approches visant la représentation des éléments extraits dans l'ontologie enrichie

Les approches présentées précédemment permettent d'extraire, à partir de sources de données, des éléments qui peuvent être ensuite placés dans l'ontologie à enrichir. Le placement peut être effectué manuellement par l'expert. Il peut aussi être automatisé en indiquant à l'expert du domaine la partie de l'ontologie dans laquelle l'insertion doit se faire ou le concept proche, et en précisant, éventuellement, le type de relation reliant le concept extrait d'un concept de l'ontologie à enrichir. Nous décrivons ci-dessous quelques travaux portant sur l'automatisation du placement d'éléments extraits. Ils sont basés sur des méthodes de fouille de données, d'apprentissage automatique, sur l'application de règles d'association ou de patrons.

Nous distinguerons tout d'abord les approches visant à indiquer à l'expert le concept de l'ontologie à enrichir proche, à lui d'effectuer ensuite lui-même le placement.

L'approche proposée dans [Neshatian and Hejazi, 2004] consiste à classer des documents d'un corpus par catégorie puis à les analyser pour savoir s'il sont pertinents par rapport à des concepts d'une ontologie d'un domaine donné. Chaque document est, en effet, associé à un vecteur qui représente la fréquence d'apparition des termes dans ce document. Une distance entre ce vecteur et un concept de l'ontologie est calculée. Si la distance est faible, les termes sont considérés comme proches du concept étudié et devront, en conséquence, être placés près de lui.

L'approche [Parekh et al., 2004], que nous avons présentée dans la section précédente, utilise une technique de regroupement (PDDP [Boley, 1998]) des termes extraits, afin d'identifier des

groupes de termes qui ont les mêmes propriétés. Chaque regroupement est proposé à l'expert du domaine. Ses éléments sont des candidats à l'enrichissement d'un terme vedette qui décrit un concept de l'ontologie à enrichir.

D'autres approches cherchent à identifier les relations reliant les concepts extraits aux concepts de l'ontologie à enrichir.

Ainsi dans [Maedche and Staab, 2001], qui utilise une version modifiée de l'algorithme proposé par [Srikant and Agrawal, 1995], des règles d'association sont utilisées pour déduire les relations à représenter entre concepts proches. Ces règles d'association comparent la pertinence de différentes relations en exploitant la hiérarchie de l'ontologie. Les relations les plus pertinentes sont proposées à l'expert qui peut décider de les intégrer manuellement via une interface graphique.

Citons également l'approche de [Fernández-Breis et al., 2010] qui enrichit une ontologie du domaine biologique par application de patrons basés sur les labels des concepts. Il s'agira, par exemple, de relier tous les concepts dont les labels contiennent la chaîne de caractères "_reliant", avec le concept "fonction moléculaire" par une relation de subsomption.

5.1.3 Positionnement de notre approche d'enrichissement par rapport à l'état de l'art

Comme nous l'avons vu, les approches d'enrichissement nécessitent l'intervention de l'expert, pour sélectionner les éléments à introduire dans l'ontologie cible parmi tout un ensemble de candidats possibles, pour décider de la façon dont les éléments pertinents pour l'enrichissement peuvent être représentés dans l'ontologie cible (de façon à préserver la cohérence de l'ontologie cible et à ne pas introduire de redondances) ou pour valider des propositions faites par des processus automatiques. Il s'agit donc de méthodes proposant l'automatisation de certaines étapes et s'appuyant sur un expert humain pour d'autres. De la même façon, les travaux visant à enrichir une ontologie cible, que nous avons réalisés, n'ont pas non plus l'ambition d'automatiser complètement la tâche d'enrichissement. Nous sommes, au contraire, convaincus de la nécessité de l'intervention d'un expert humain à différents moments du processus.

Nos travaux ont pour objectif d'enrichir une ontologie cible à partir d'une autre ontologie. Notre problème n'a donc pas été un problème de découverte d'éléments de nature ontologique au sein de la source utilisée comme cela est fait dans les travaux d'extraction au sein de documents textuels.

Notre problème a été d'une part de délimiter les parties des ontologies source pertinentes pour l'enrichissement. Ces travaux peuvent être rapprochés, dans un certain sens, de ceux réalisés pour construire les corpus de documents lorsque les sources utilisées sont des sources composées de documents textuels.

D'autre part, notre travail a porté sur la détermination des éléments d'une ontologie source intéressants pour l'enrichissement. Plus précisément, notre approche extrait, à partir d'une ontologie du même domaine d'application ou généraliste, des sous-ensembles d'une hiérarchie de concepts proches du domaine de l'ontologie à enrichir. Ceci peut être rapproché des travaux exploitant des ontologies s'intéressant à définir les éléments pertinents pour l'enrichissement, no-

tamment ceux prenant en compte le contexte applicatif des éléments pouvant être introduits.

Enfin, nous nous sommes attachés à faire des propositions de placement des éléments intéressants pour l'enrichissement allant jusqu'à indiquer, dans certains cas, le concept de l'ontologie cible lié et le type de la relation le liant à ce concept. Peu de travaux de l'état de l'art fournissent, à notre connaissance, ce niveau de précision, la majorité indiquant seulement le concept de l'ontologie cible proche.

L'originalité de notre approche repose sur le fait qu'elle exploite les résultats d'alignement entre une ressource externe, (l'ontologie source), et l'ontologie à enrichir (l'ontologie cible), cet alignement étant composé de mappings qui ne sont pas uniquement des similarités mais peuvent correspondre à des relations plus précises, telles des relations de subsomption. Par ailleurs, il s'agit d'une approche flexible qui permet la définition de traitements qui peuvent être adaptés au contexte de l'enrichissement, c'est-à-dire aux spécificités des ontologies source et cible. Cette flexibilité est offerte via la définition de patrons d'enrichissement grâce auxquels il est possible de spécifier, entre autres, des traitements de (1) vérification de la compatibilité du domaine de l'ontologie source par rapport à celui de l'ontologie cible, (2) de sélection des concepts intéressants pour l'enrichissement par élimination de ceux jugés inintéressants, (3) d'enrichissement proprement dit en introduisant de nouveaux concepts et de nouvelles relations dans l'ontologie cible.

5.2 Apport des différentes techniques de TaxoMap pour l'enrichissement

Remarquons tout d'abord que, dans un contexte classique, l'alignement d'ontologies sert à identifier pour chaque concept d'une ontologie source s'il existe un concept de l'ontologie cible qui puisse être considéré comme similaire ou suffisamment proche pour que les deux concepts soient mis en relation. Ces résultats d'alignement peuvent, par exemple, être utilisés pour accéder à des documents indexés avec les concepts de la source en effectuant une interrogation à partir de ceux de la cible.

Dans ce contexte classique d'intégration d'information, si l'on souhaite faire une recherche de documents à partir d'un concept particulier de la cible, tous les mappings mettant en relation des concepts de la source avec ce concept cible seront importants. Ils permettront en effet de trouver au sein des documents indexés avec les concepts de la source, aussi bien ceux annotés précisément avec le concept de la source jugé équivalent au concept cible s'il existe, mais aussi d'élargir la recherche avec les documents annotés avec d'autres concepts considérés comme proches et que le mécanisme d'alignement aura permis d'identifier.

En revanche, pour le point de vue enrichissement, tous les concepts de la source mis en relation par un mapping avec un concept de la cible n'ont pas le même intérêt. L'objectif est d'enrichir la cible d'éléments qui, bien que proches de ceux déjà présents, doivent être différents, de façon à éviter les redondances.

Nous étudierons donc ici l'apport théorique des différentes techniques d'alignement développées dans TaxoMap pour la tâche d'enrichissement et l'impact des caractéristiques de la ressource

externe utilisée comme source d'enrichissement sur l'efficacité de ces techniques.

5.2.1 Un apport différent suivant les techniques mises en œuvre

Les mappings obtenus par les différentes techniques peuvent être regroupés en différentes catégories suivant les relations en jeu dans le mapping, la fiabilité des techniques employées et leur apport pour l'enrichissement.

- Mappings issus des techniques s'appuyant sur la forte similarité des labels pour établir des relations d'équivalence $<c_s\ isEq\ c_{tmax}>$ ou de proximité (forte) $<c_s\ isClose\ c_{tmax}>$. Ces techniques (t_1 et t_5) sont conçues pour tirer parti de la supposée grande richesse des labels de c_s et c_{tmax} pour considérer ces concepts comme équivalents ou presque, si leur label sont très similaires. Ces mappings ne vont pas permettre d'introduire de nouveaux concepts dans la cible. Eventuellement, dans le cas des mappings d'équivalence, si le concept de l'ontologie source possède différents labels ou propriétés, ceux-ci pourront être importés dans l'ontologie cible. Dans le cas des mappings de proximité, le label de c_s pourra être retenu comme un label alternatif à celui de c_{tmax}.

Exemple : le mapping < hôtel régional *isClose* hôtel de région> peut permettre d'introduire le label <hôtel régional> comme un label alternatif du concept de la cible <hôtel de région>.

Ces introductions doivent être dirigées par l'expert qui, seul, peut décider si un label jugé proche est pertinent pour un concept ou résulte d'une faute de frappe. Ainsi, dans les deux mappings suivants, <Pépinnière *isClose* (simRel) pépinière> et <terri *isClose* (simRel) terril>, la différence entre les labels des concepts sources et cibles résultent d'une faute de frappe et ces labels n'ont pas d'intérêt pour l'enrichissement. Dans le mapping <trunk skin *isClose* Skin of the Trunk>, la différence venant d'une réorganisation des termes et de l'absence des mots vides "of the" dans le label source, celui-ci n'a pas non plus d'intérêt. En revanche, dans le mapping <intrahepatic part of left hepatic duct *isClose* Intrahepatic Portion of the Left Hepatic Duct> la différence entre les labels des deux concepts se faisant sur un mot plein, "part" dans l'un, "portion" dans l'autre, le nouveau label pourrait être jugé intéressant, même chose pour < epididymal duct *isClose* Duct of the Epididymis>.

- Mappings s'appuyant sur des similarités de labels pour établir des relations de subsomption $<c_s\ isA\ c_t>$. Ces mappings sont susceptibles de conduire à de nombreux enrichissements. En effet, les techniques d'identification d'équivalences ou de fortes proximités, qui sont le plus souvent testées les premières, n'ayant pas été appliquées, le label du concept c_s est supposé être suffisamment différent de celui de c_t pour que les deux concepts soient considérés comme distincts.

Suivant les contextes et sous certaines conditions, c_s peut être considéré comme un nouveau concept pertinent intégrable à la cible et directement placé comme une spécialisation de c_t dans celle-ci. Mais sous d'autres conditions, il devra être reconnu comme redondant ou trop proche des concepts déjà existant et ne pas être pris en compte pour l'enrichissement.

Parmi les éléments à prendre en compte, le premier est la technique ayant conduit au mapping puisque toutes les techniques n'ont pas la même fiabilité. Ainsi la technique t_2 conduit à

des mappings a priori beaucoup plus sûrs que les techniques t_6, t_7 et t_8. En particulier, les deux dernières techniques sont utilisées quand plusieurs concepts de la cible sont candidats au mapping sans qu'aucun n'ait une similarité particulièrement significative. Le concept de la cible c_t intervenant dans le mapping n'est pas forcément celui qui a la plus forte similarité avec c_s et les mappings issus de ces techniques sont intéressants dans une optique d'alignement pure mais pas pour de l'enrichissement.

Les souhaits de l'expert sont bien sûr déterminants. Par exemple, la technique d'inclusion t_2 qui crée un mapping $<c_s\ isA\ c_{tmax}>$ si le label de c_{tmax}, le concept le plus similaire à c_s, est composé de mots pleins, tous inclus dans le label de c_s, permet d'identifier a priori les concepts qui sont des spécialisations de c_{tmax}. L'intérêt de ces spécialisations pour l'enrichissement dépend en grande partie de la sémantique ajoutée par la partie du label de c_s qui n'apparait pas dans c_{tmax}. Ainsi dans le mapping $<$*Bâtiment de type industriel isA Bâtiment industriel*$>$, la partie ajoutée "*de type*" peut être jugée sans intérêt par l'expert puisque le mot $<$*type*$>$ est un mot vide de sens dans le domaine.

On voit ainsi qu'une phase de tri et d'analyse va être nécessaire pour identifier les mappings qui ne seront pas utilisables pour l'enrichissement.

- Mappings établissant des relations de proximité (faible) $<c_s\ isClose\ c_{tmax}>$ à partir d'inclusions de labels tenant compte des mots complémentaires. Ces mappings sont construits car, même si la similarité est très faible puisque les mots inclus n'ont pas le même statut dans leur label respectif, c_{tmax} apparaît comme le concept le plus proche de la cible auquel c_s puisse être lié.

En revanche, les mots communs des deux labels n'ayant pas le même statut, aucune relation de subsomption ne peut être établie a priori entre les deux concepts, ni dans un sens ni dans l'autre, comme on le voit dans les exemples $<$ sortie d'autoroute *isClose* Autoroute $>$ ou $<$ Péage *isClose* Aire de péage $>$. Seul l'expert peut juger de l'apport potentiel du concept source et de son éventuel positionnement dans l'environnement de c_{tmax}.

- Mappings issus des techniques s'appuyant sur la structure des ontologies

Parmi les techniques structurelles, les deux techniques t_9 et t_{10} peuvent être qualifiées de structurelles pures car elles n'utilisent pas du tout les labels des concept c_s à aligner pour construire les mappings, mais uniquement la position relative de ces concepts dans leur ontologie par rapport à des ancres, i.e. des concepts précédemment identifiés comme équivalents à des concepts de la cible. Elles sont particulièrement intéressantes pour l'enrichissement car elles vont permettre d'aligner des concepts dont les labels n'ont pas ou peu de similarité avec ceux des concepts de la cible, et sont donc, a priori, des concepts différents de ceux déjà existants, donc pertinents pour l'enrichissement.

A l'inverse, la technique t_8 est moins intéressante puisque les concepts c_s alignés par cette technique sont peu différents des concepts existants dans la cible. En effet, cette technique s'appuie d'abord sur des similarités de labels pour identifier les concepts de la cible les plus proches d'un c_s et n'utilise qu'ensuite la structure de la cible pour aligner c_s avec le père commun de ces concepts proches, si ce père commun existe.

5.2.2 Un apport différent suivant les caractéristiques de l'ontologie source

Les caractéristiques de l'ontologie source ont elles-aussi un impact sur la fiabilité des mappings générés et leur intérêt pour l'enrichissement.

Les techniques basées sur la très grande similarité des labels ou sur l'existence d'ancres ne sont acceptables et efficaces que quand les ontologies alignées décrivent le même domaine d'application. Cette hypothèse permet de lever les problèmes d'ambiguïté dûs au phénomène de polysémie dans les labels. Quand les domaines d'application des ontologies à comparer sont proches et très focalisés, les risques d'homonymie sont très limités, les mots ont le même sens et les concepts qui ont le même label peuvent être considérés comme équivalents par des outils d'alignement. En revanche, dès qu'une au moins des deux ressources alignées est généraliste, les problèmes d'homonymie surgissent et nécessitent d'être attentif au contexte d'interprétation des concepts manipulés pour ne pas mêler des sens différents qui aboutiraient à faire des rapprochements erronés entre concepts.

Les éviter nécessite d'introduire une phase de vérification de la compatibilité des domaines des concepts manipulés pour ne pas mêler des sens différents.

De la même façon, les différences de structures ou de point de vue entre les deux ontologies entrent en ligne de compte et certaines techniques ne conduisent à des enrichissements intéressants que si les ontologies ont le même niveau de structuration et reflêtent les mêmes points de vue. Par exemple, la technique t_9 s'appuie sur la présence d'ancres, les mappings d'équivalence $<c_s\ isEq\ c_{tmax}>$ identifiés en premier, et rapatrie tous les fils directs non encore alignés d'une de ces ancres comme des spécialisations directes du concept associé dans la cible. Si l'ontologie source est peu structurée, i.e. présentant peu de profondeur et des nœuds internes possédant beaucoup de fils directs, les mappings produits par cette technique devront probablement au mieux être repositionnés au sein du sous-arbre issu de c_{tmax} dans la cible, comme des fils de ses fils plutot que comme des spécialisations directes. A l'inverse, si les ontologies sont toutes les deux très structurées, ce sont les techniques d'inclusion de labels qui seront prises en défaut puisqu'elles ne tiennent pas compte de la structure, et demanderont un repositionnement.

Vérifier la compatibilité des domaines des concepts manipulés ou qualifier les concepts candidats sont autant de traitements complémentaires qui s'appuient sur les résultats d'alignement. D'autres vont intervenir pour faciliter le placement des concepts dans la cible, comme la visualisation du sous-arbre issu du concept cible impliqué dans le mapping ou le regroupement des mappings liés à un même concept. D'autres traitements vont être nécessaires en préalable, en particulier quand l'ontologie exploitée pour l'enrichissement est de très grande taille, une phase préliminaire de partitionnement sera nécessaire. L'identification et la mise en œuvre de ces traitements dépendent du contexte et des ontologies considérées.

Dans les sections suivantes, nous présentons trois contextes d'enrichissement et les traitements associés. Ces contextes ont été identifiés dans le cadre du projet ANR GeOnto dont un des buts est de construire une ontologie de concepts topographiques par enrichissement d'une première taxonomie de concepts, élaborée manuellement, à partir de spécifications de base de données [Abadie and Mustière, 2010].

Nous avons cherché à enrichir l'ontologie cible de trois manières différentes :

1. L'enrichissement exploite successivement deux ontologies : l'une est une ontologie construite automatiquement [Aussenac-Gilles and Kamel, 2009] en appliquant des techniques de traitement du langage naturel sur les spécifications ayant permis la construction manuelle de la taxonomie à enrichir, l'autre est la première version manuelle de cette taxonomie. Le domaine et la taille de l'ontologie source considérée est donc le même que celui de la taxonomie cible et les problèmes de validation de thématique ne se posent pas.

2. L'enrichissement exploite un ensemble de petites ontologies, chacune formée par un ensemble de concepts extraits de façon semi-automatique d'un thésaurus généraliste de grande taille, RAMEAU. Le problème va être ici de vérifier que le domaine de chacune de ces petites ontologies est pertinent pour l'application avant de l'utiliser pour l'enrichissement.

3. L'enrichissement exploite une ontologie généraliste très volumineuse, YAGO. De ce fait, le problème est d'introduire un traitement spécifique préliminaire pour identifier des sous-parties thématiquement pertinentes pour l'alignement puis l'enrichissement.

5.3 Enrichissement à partir d'une ontologie aux caractéristiques proches (même domaine, même taille) de celles de l'ontologie cible

Dans ce premier contexte d'enrichissement [Hamdi et al., 2011], les ontologies considérées relèvent toutes du même domaine, la topographie, et ont des tailles comparables : la taxonomie cible manuelle comprend environ 700 concepts, l'ontologie construite automatiquement en comprend plus de 800 et la première taxonomie manuelle 600. De plus, ce contexte est vraiment très particulier car les trois ontologies ont été construites à partir de la même spécification. Elles sont donc vraiment très proches. L'alignement produit ainsi, pour les deux sources considérées, un très grand nombre de mappings d'équivalence ou de fortes proximités qui ne permettront pas d'introduire de nouveaux concepts mais sont intéressants car ils établissent des ancres sûres et utiles pour la mise en œuvre des techniques structurelles.

Nous allons donc nous focaliser ici sur les techniques permettant d'identifier des mappings entre des concepts de labels suffisamment différents pour que le concept de la source puisse être jugé non redondant avec ce qui existe déjà dans la cible et donc intéressant pour l'enrichissement. Comme nous l'avons vu précédemment, les mappings les plus intéressants dans ce contexte sont les mappings issus de la technique d'inclusion t_2 et des techniques structurelles t_9 et t_{10}.

Nous présentons plus particulièrement ici les traitements complémentaires nécessaires pour identifier les mappings issus de t_2 qui conduisent à des concepts redondants devant être rejetés et ceux qui permettraient de mieux positionner les mappings obtenus par la technique t_{10}. Nous présentons ensuite des propositions d'outils de visualisation qui permettraient à l'expert de juger des différents apports de la technique t_9 et des autres techniques puis de choisir les concepts qu'il souhaite conserver ou repositionner.

5.3.1 Identification des mappings issus de la technique t_2 non pertinents

La technique t_2, qui crée des mappings de la forme $<c_s\ isA\ c_{tmax}>$, permet a priori d'identifier les concepts qui devraient être des spécialisations de concepts de la cible puisque dans chaque mapping le label du concept cible c_{tmax} est intégralement inclus dans celui de la source c_s. L'intérêt pour l'enrichissement de ces concepts spécialisés dépend en grande partie de la sémantique ajoutée par la partie du label de c_s qui n'apparaît pas dans c_{tmax} et que nous appellerons "*remaining part*".

Ainsi, par exemple, lorsque l'ontologie source est l'ontologie construite automatiquement, de très nombreux mappings produits par t_2 concernent des concepts sources dont le label est constitué d'une conjonction de labels de concepts, tous déjà présents par ailleurs dans l'ontologie cible, comme le mapping $<$*Parc et Jardin isA Jardin*$>$ où les deux concepts *Parc* et *Jardin* existent dans la cible.

Comme nous l'avons dit précédemment, quand les résultats d'alignement sont utilisés pour accéder à des documents indexés avec les concepts de la source en effectuant une interrogation à partir de ceux de la cible, ce mapping est important puisqu'il permet aussi de trouver les documents annotés uniquement par le concept $<$*Parc et Jardin*$>$ quand on recherche les documents relatifs à *Jardin*. En revanche, dans un contexte d'enrichissement, le concept $<$*Parc et Jardin*$>$ bien que de label différent de celui des concepts déjà présents dans la cible n'apporte rien de plus, et ce faux nouveau concept ne doit pas être considéré.

Nous avons donc étudié l'ensemble des mappings produits par la technique t_2 et essayé d'identifier les caractéristiques vérifiées par les labels de certains des concepts sources qui permettraient de les détecter comme redondants et justifieraient leur rejet pour l'enrichissement.

Dans un premier temps, nous avons énuméré les caractéristiques syntaxiques dont l'identification serait relativement aisée à mettre en œuvre dans le langage de patrons MPL :

- Quand le label du concept source est composé de 2 labels de concepts déjà existant dans l'ontologie cible et reliés par des connecteurs comme "ou", "et" ou "/", le concept de la source ne sera pas importé.
  `Arène ou théâtre antique` *isA* `théâtre antique`
  `Plaine ou plateau` *isA* `plateau`

- Quand la *remaining part* du label du concept source contient principalement un mot *vide* pour le domaine de la cartographie comme *zone*, *aire*, *type*, aucun enrichissement ne sera réalisé.
  `Batiment de type industriel` *isA* `bâtiment industriel`

- Quand la *remaining part* du label du concept source est composée d'un adjectif entre parenthèses, le concept de la source ne sera pas importé mais l'adjectif pourra mener à la représentation d'une propriété de ce concept.
  `canal (large)` *isA* `canal`
  `atelier (grand)` *isA* `atelier`

- Quand le label du concept source est formé du label du concept de la cible puis d'un des

connecteurs "ou", "et" ou "/", puis du label d'un autre concept de la source non déjà présent dans la cible, et non aligné avec un concept de celle-ci, le concept source étudié ne sera pas pris en compte dans l'enrichissement. En revanche, l'autre concept source identifié dans son label sera retenu pour être introduit comme un frère du concept de la cible considéré.
Dans l'exemple ci-dessous, "radier" est le label d'un concept de la source non présent dans la cible et qui n'avait pas été aligné, il pourra être introduit dans l'ontologie cible, en tant que frère du concept "Gué". Même chose pour "Crémaillère" en tant que frère de "Funiculaire".

```
gué ou radier                isA   gué
Funiculaire ou crémaillère   isA   Funiculaire
```

L'élimination de ces mappings peut être réalisée par des patrons de même forme que le patron présenté ci-dessous et qui correspond au premier cas identifié. La partie contexte vérifie qu'il existe un mapping de subsomption construit par la technique t_2 entre x et y, et que le label de x peut être exactement décomposé en deux composantes dont l'une est bien sûr le label de y et l'autre est exactement le label d'un autre concept z de la cible.

Partie contexte du Patron :
$\exists x \exists y\ (isAStrictInclusion(x, y) \wedge inclusionInLabel(x, "and", y)$
$\wedge\ \exists z\ (extractFromLabel(x, "and", y, L) \wedge ConceptInclus(x, L, z)))$
Partie solution du Patron :
$Delete\_Mapping(x, y, \_)$

En revanche, il est beaucoup plus délicat d'identifier automatiquement les concepts auxquels sont associés des labels non signifiants et constitués d'une énumération de plusieurs labels existants par ailleurs dans la source, mais sans présence de séparateurs clairement identifiables ou quand ils correspondent à ce qui devraient a priori être compris comme des valeurs. L'utilisation d'une interface (cf. Section 5.3.3) permettant à l'expert de visualiser ensemble tous les concepts de la source alignés avec un même concept de la cible, devrait aider celui-ci à élaguer ces concepts erronés.

```
Parc municipal intercommunal départemental interdépartemental
                                  isA          parc municipal
Télécabine téléphérique téléski   isA          téléphérique
Industriel ou commercial          isA(struct)  bâtiment  industriel
```

5.3.2 Insertion des concepts sources alignés par la technique t_{10}

La technique t_{10} est basée sur la comparaison du voisinage (les descendants) des concepts comparés. Elle permet de relier par une relation de proximité *isClose*, un concept c_s de O_S à un concept c_t de O_T, si c_t est le concept dans O_T qui a au moins deux descendants en commun avec c_s et qui maximise la mesure M_{SD} pour c_s, i.e. le rapport *nombre de descendants communs avec* c_s sur le *nombre total de descendants des deux concepts.*

Cette technique est intéressante car elle permet de rapprocher des concepts qui sont des nœuds non feuilles dans leur ontologie respective, sans s'appuyer sur des similarités lexicales qui

échouent souvent pour ces nœuds. En effet, à la différence des feuilles des arbres ontologiques qui sont souvent des concepts du monde réel dont la définition et le nom sont clairement établis et partagés par tous, les nœuds plus généraux sont plus souvent identifiés et nommés par les concepteurs d'une ontologie pour expliciter des regroupements de propriétés ou des points de vue pertinents pour cette ontologie et ont donc souvent des labels très différents même s'ils recouvrent les mêmes concepts feuilles. Cette technique permet donc a priori d'identifier un label alternatif pour les concepts cible impliqués dans ces mappings.

Mais il se peut aussi que la technique ait relié le concept de la source à un concept de la cible tel que ce concept cible soit déjà relié par un mapping d'équivalence à un autre concept de la source qui soit un spécialisant du concept source de départ.

Par exemple, dans la figure ci-dessous (cf. Fig. 5.1), on voit que le concept "nœud du réseau routier" avait été relié par la technique t_{10} au concept "carrefour" avec lequel il partage 6 descendants communs, les 5 descendants de "carrefour" et le concept "carrefour" lui-même qui est également représenté dans la source comme un descendant de "nœud du réseau routier". "nœud du réseau routier" est donc relié par t_{10} à l'équivalent d'un de ses propres fils, "carrefour".

Pour éviter ce problème, un patron de raffinement remplace le mapping de proximité par une relation de subsomption entre le concept source et le père dans O_T du concept cible considéré. Ici, le concept "nœud du réseau routier" est finalement relié au concept "Infrastructure de transport routier".

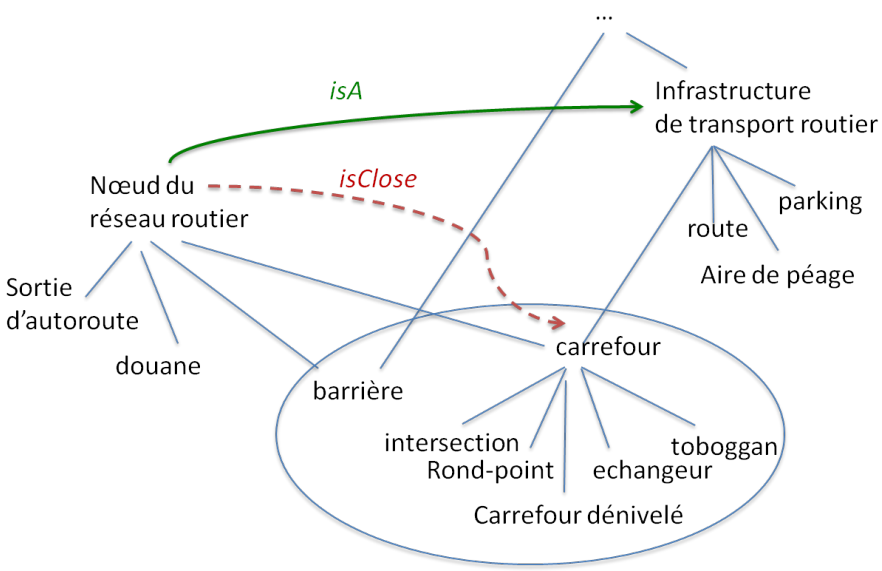

FIG. 5.1 – Exemple de relation identifiée par t_{10}

Dans ce genre de contexte, ainsi que dans ceux où le concept cible identifié par la technique est celui qui maximise la mesure M_{SD} pour le concept source, sans être celui qui partage le plus de descendants communs avec lui, on peut imaginer que le concept source puisse être proposé à l'expert comme un nouveau nœud intermédiaire. Il pourrait être positionné comme un père du premier concept cible impliqué au départ dans la relation, et on pourrait proposer de regrouper

sous ce nouveau nœud l'ensemble de ses fils dans la source.

Dans l'exemple présenté Fig. 5.1, le concept "nœud du réseau routier" serait proposé pour être introduit comme un fils du concept "Infrastructure de transport routier", regroupant le concept "carrefour" mais aussi "barrière" et ses autres fils dans la source comme "douane" et "Sortie d'autoroute".

Remarquons que cette technique d'enrichissement demande, pour être efficace, que les structurations des deux ontologies considérées ne soient pas trop éloignées.

5.3.3 Besoin d'interfaces visualisant les mappings ou des parties d'ontologies

L'ensemble des mappings obtenus par alignement est pour l'instant décomposé en paquets, en fonction des techniques utilisées pour les obtenir. Pour la phase d'enrichissement et la phase de nettoyage préalable, il est intéressant de les regrouper non plus en fonction de la technique utilisée mais en fonction du concept de la cible auquel ils font référence. L'exemple ci-dessous (cf. Fig. 5.2) présente l'ensemble des mappings reliant différents concepts de la source au même concept "Bâtiment industriel" de la cible, quelle que soit la technique utilisée et présentée entre parenthèses.

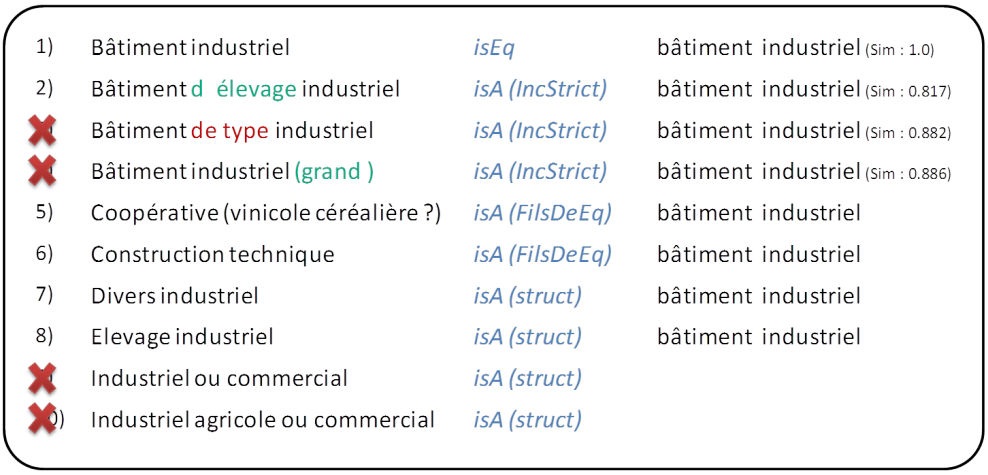

FIG. 5.2 – Ensemble des mappings établissant une relation avec "bâtiment industriel" de l'ontologie cible

Une partie des traitements identifiés dans le paragraphe 5.3.1 précédent pourrait être appliquée automatiquement. Ainsi, dans le mapping n°3, l'analyse de la composante supplémentaire au mot inclus devrait permettre de reconnaître qu'elle ne contient que le mot vide "de type" et que ce mapping peut être écarté. De même, dans le n°4 où l'adjectif entre parenthèses est facile à identifier et doit être considéré comme une propriété.

Les autres mappings sont moins simples à analyser automatiquement mais le fait de les regrouper par concept cible auquel ils font référence devrait permettre à l'expert de juger beaucoup plus rapidement de l'apport relatif de chaque concept. Ainsi l'existence du concept n°2, "Bâtiment d'élevage industriel" disqualifie immédiatement l'intérêt du n°8 "Elevage industriel", qui

peut être interprété soit comme redondant par rapport à ce concept (si l'expert considère que le terme "bâtiment" est sous entendu dans le label) soit comme intervenant dans un mapping erroné car si le concept "Elevage industriel" doit exister il devrait apparaître comme une spécialisation d'un "domaine d'activité" plutôt qu'une spécialisation d'un "bâtiment".

Des outils de visualisations complémentaires pourraient être proposés à l'expert, par exemple pour visualiser le positionnement actuel du concept cible en cours d'étude dans son ontologie ou ceux des concepts sources en cause.

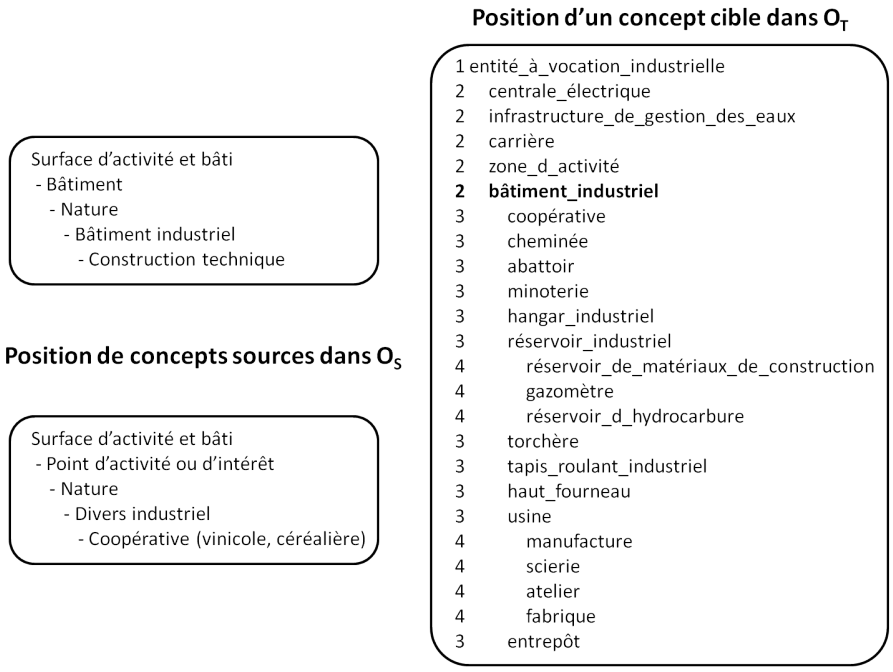

FIG. 5.3 – Exemple illustrant le positionnement de "bâtiment industriel" dans l'ontologie cible

Ainsi, en complément de la figure 5.2, la figure 5.3 ci-dessous positionne le concept "Bâtiment industriel" au sein de l'ontologie cible, en présentant son père "entité à vocation industrielle", ses frères et ses descendants. Le rapprochement de ces deux figures 5.2 et 5.3, dans deux fenêtres accessibles simultanément, devrait aider l'expert à affiner le positionnement d'un concept jugé pertinent pour l'enrichissement. Par exemple si l'expert juge que le concept n°5 de la figure 5.2, "Coopérative (vinicole, céréalière)", est pertinent pour l'enrichissement, il pourra se rendre compte que ce concept devrait être introduit non pas directement comme une spécialisation de "Bâtiment industriel" [12] mais plutôt comme une spécialisation d'un de ses fils existants, le concept "coopérative".

12. De fait, le mapping "Coopérative (vinicole, céréalière) isA (FilsdeEq) Bâtiment industriel" construit par TaxoMap et présenté dans la figure 5.2, est une erreur de l'outil. Le mapping correct "Coopérative (vinicole, céréalière) isA (IncStrict) Coopérative" aurait du être construit automatiquement par la technique d'inclusion stricte (t_2) qui aurait du reconnaître que le label "Coopérative" était inclus dans le label du concept source. Cette reconnaissance ayant échouée, probablement du fait d'un caractère invisible intervenant autour de la parenthèse "...(vinicole...)", c'est la technique t_9 qui a été appliquée ici, considérant tous les spécialisants du concept "Bâtiment industriel" de l'ontologie source non encore alignés, comme des spécialisants de son équivalent dans la cible.

Notre étude a également permis de pointer certains problèmes comme l'existence de labels ayant des sens différents qui nécessitent d'étudier les concepts ascendants avant toute décision d'enrichissement. Ce problème est illustré dans la figure 5.4 qui donne l'ensemble des concepts ayant pour label "Péage" dans l'alignement <Péage *isClose* Aire de péage>.

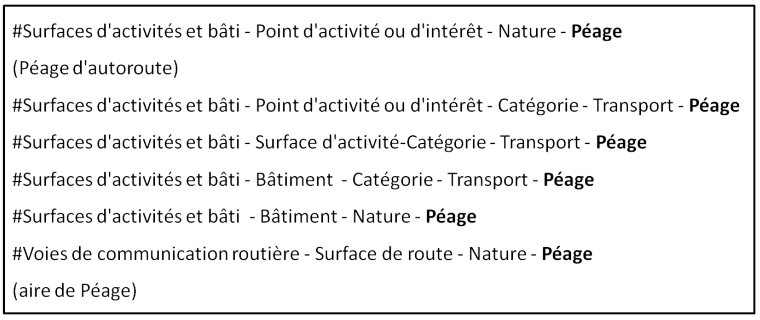

FIG. 5.4 – Différents ascendants de "Péage"

La décision d'enrichissement doit passer ici par l'étude du chemin menant à Aire de Péage dans l'ontologie cible (cf. Fig. 5.5).

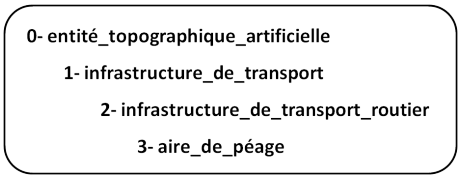

FIG. 5.5 – Chemin menant à Aire de péage dans l'ontologie cible

5.4 Enrichissement à partir d'ontologies de taille réduite extraites automatiquement d'une ontologie généraliste

Dans ce scénario [Hamdi et al., 2011], l'enrichissement exploite un ensemble de petites ontologies, chacune formée par un ensemble de concepts extraits de façon semi-automatique du thésaurus RAMEAU. Cette ressource généraliste est composée de 400 000 termes reliés entre eux par des relations qui peuvent en partie être traduites par des relations de subsomption. Ces ensembles ont été obtenus par nos partenaires dans le projet ANR GeOnto, en exploitant automatiquement des récits de voyage dans lesquels sont repérées des entités nommées spatiales ("le gave de Pau") constituées de termes ("le gave") associés à des toponymes ("Pau") [Kergosien et al., 2010]. L'idée est que les termes associés aux toponymes ont souvent un caractère topographique et pourraient enrichir la taxonomie s'ils n'y sont pas déjà.

Pour distinguer les termes topographiques comme "gave" dans "le gave de Pau", de ceux qui n'en sont pas, comme "maire" dans "le maire de Pau", il faut disposer d'informations supplémentaires. Nos partenaires ont donc lié chacun des termes considérés avec des informations extraites de RAMEAU et constitué un ensemble de petites sources, chacune associée à un des termes candidats.

Notre problème a donc été d'étudier chacune de ces sources et de vérifier que son domaine relèvait ou pas du domaine de l'ontologie cible, ici topographique, avant de l'utiliser pour l'enrichissement.

Nous présentons, dans un premiers temps, une étude des sources extraites de l'ontologie RAMEAU et des propositions d'enrichissement à partir de ces sources. Nous décrivons ensuite les tests conçus pour effectuer la vérification de la compatibilité du domaine d'une source avec celui de l'ontologie cible, puis nous présentons l'implémentation de ces tests sous la forme de deux patrons exprimés dans le langage MPL décrit en chapitre 3.

5.4.1 Etude des sorties extraites de RAMEAU

Dans RAMEAU, les parties extraites s'organisent autour d'un terme dit "vedette". Elles regroupent les termes génériques et spécifiques relatifs à cette vedette, ainsi qu'un ensemble de termes étiquetés "employé pour", i.e. les termes qui doivent être remplacés par la vedette, quand on fait de l'indexation.

Par exemple, le terme "abîme" n'appartient pas à la taxonomie mais existe dans le thésaurus RAMEAU sous la forme suivante :

Vedette :
Grottes
Employé pour :
« Abîmes, Antres, Avens, Cavernes, Cavernes préhistoriques, Gouffres, Grottes ornées.."
Termes génériques :
Habitat préhistorique, Relief, Zones souterraines

L'objectif défini dans le projet GeOnto était de dire que si la "vedette" de la composante extraite à partir d'un terme associé à un toponyme appartient à la taxonomie cible O_T, et que le terme appartient au champ "employé pour" de la vedette, ce terme peut être retenu comme un nouveau concept, spécialisant de la vedette. Ici, Grotte appartenant à O_T, le terme "Abîme" doit être introduit comme un fils de "grotte".

Nous avons proposé de le réaliser avec l'outil TaxoMap. La technique d'alignement t_9 de TaxoMap raisonne exactement avec le même principe. Si un concept c_s de O_S n'a pas pu être aligné après l'application de l'ensemble des techniques d'alignement de plus grande priorité testées, et si le père de c_s dans O_S, c_{ps}, est relié à un concept de l'ontologie cible c_t par un mapping d'équivalence, alors on introduit un mapping de subsomption entre le concept c_s et le concept de l'ontologie cible relié avec le père (c_s isA(FilsdeEq) c_t).

Mettre en œuvre cette proposition avec TaxoMap se fait donc naturellement si la composante extraite de RAMEAU est représentée sous la forme d'une ontologie OWL qui peut être alignée avec l'ontologie cible. D'autres techniques que t_9 pourraient être utilisées dans la foulée et mettre en évidence d'autres enrichissement. Encore faut-il s'assurer avant d'utiliser la technique t_9 (ou d'autres), que les mappings d'équivalence identifiés sont sûrs, i.e. que nous ne sommes pas en présence d'une fausse homonymie, et donc que le domaine des concepts extraits est compatible avec celui de la source.

Pour l'expression de la composante en OWL, les termes "employé pour" doivent être considérés comme des spécialisations de la vedette, au même titre que ses termes spécifiques et pas comme des labels alternatifs à cette vedette.

En effet, au moins sur cet exemple, la granularité des informations représentées dans RAMEAU n'est pas la même que celle exprimée dans l'ontologie. La "vedette" apparaît plutôt comme un terme général, générique pour tout un ensemble d'autres termes amalgamés qui, eux, apparaissent plus comme des "sortes de" cet élément vedette.

La traduction de la relation du champ "employé pour" par une relation de subsomption est, certes, une simplification, mais sa traduction par une relation d'équivalence, ce qui est le statut en OWL des labels alternatifs, serait une simplification encore plus forte. De plus, au moins sur cet exemple, le thésaurus regroupe dans le même champ "employé pour" de la vedette "Grottes", des expressions considérées dans l'ontologie comme les labels de concepts distincts de Grotte et reliés à la vedette par des relations de subsomption. C'est le cas de "Caverne" ou "Avens". De même, TaxoMap identifiera, lors de l'alignement, des relations de subsomption entre les termes comme "Caverne préhistorique" et "Caverne" ce qui n'a pas de sens si ces deux termes sont considérés comme équivalents dans RAMEAU. La traduction du champ "employé pour" comme des labels alternatifs de la vedette n'est donc pas adaptée ici et les termes de ce champ ont été systématiquement introduits comme des spécialisations du concept vedette.

5.4.2 Vérifier la compatibilité du domaine de la source avec celui de la cible

Etant donnée une ontologie source dont le contenu est potentiellement pertinent pour l'enrichissement d'une ontologie cible, le problème est ici d'expliciter un traitement permettant de vérifier que le domaine de cette source est compatible avec celui de la cible.

Les sources étant de très petite taille, entre 5 et 15 concepts, parfois moins, cette vérification peut être faite en s'assurant qu'il est possible d'aligner au moins deux concepts de la source avec ceux de l'ontologie cible. Les sources étant extraites d'une ressource généraliste, nous pouvons être confrontés à de fausses homonymies. Pour écarter ce risque de fausse homonymie, nous imposons que les deux concepts alignés de la source soient liés dans la cible à des concepts proches l'un de l'autre, la présence conjointe de deux faux homonymes étant peu réaliste.

Ainsi, chaque source issue de RAMEAU est alignée avec la cible et les mappings résultants sont analysés. Une source sera jugée valide pour l'enrichissement si on peut trouver dans l'alignement au moins deux mappings tels que l'un soit un mapping d'équivalence entre un c_s et un c_t, et l'autre, un mapping jugé fiable, entre un c_{s2} différent de c_s et un c_{t2} différent de c_t. De plus, les deux concepts de la cible c_t et c_{t2} doivent être en relation (fils-père ou fils d'un même père) dans O_T.

La notion de mapping *fiable* recouvre les mappings d'équivalence, d'inclusion par la technique t_2 ou de très grande proximité lexicale (t_5). Elle exclut en particulier les mappings issus de la technique t_9 puiqu'ils dépendent de mappings d'équivalence dont on a pas encore vérifié la fiablilité.

En accord avec l'expert, nous avons convenu de trois cas de sortie possibles du test de validation.

1. Si les différentes conditions explicitées plus haut sont réunies, le domaine de la source est qualifié de valide.

2. Si on ne trouve pas au moins deux mappings, l'un d'équivalence, l'autre fiable, la source est rejetée.

3. Si un mapping d'équivalence et un mapping fiable existe, mais que les deux concepts de la cible ne sont pas reliés dans l'ontologie cible, le domaine de la source est considéré comme restant à valider par l'expert.

Nous présentons, dans ce qui suit, des exemples correspondant aux trois cas identifiés.

Exemple cas 1

La figure 5.6 ci-dessous présente dans des encadrés les concepts identifiés comme équivalents lors de l'alignement entre la partie de RAMEAU extraite pour le terme "abîme" et l'ontologie cible. Puisqu'on peut trouver au moins deux mappings fiables tels que les concepts considérés dans les mappings soient en relation deux par deux dans leurs ontologies respectives (ici en fait, 4 mappings d'équivalence, relatifs à 4 concepts tous reliés entre eux), le domaine de la composante est jugé compatible avec celui de l'ontologie.

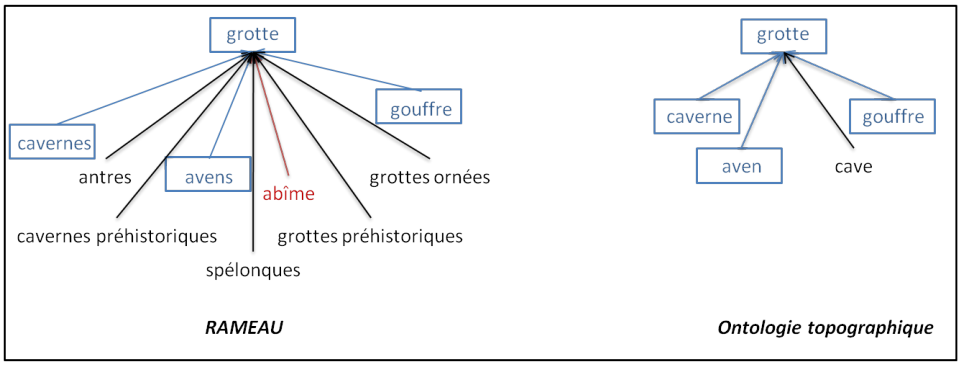

FIG. 5.6 – Exemple de source dont le domaine est validé

Les mappings trouvés sont alors utilisables pour l'enrichissement et peuvent être présentés à l'expert en les regroupant par concept cible considéré dans les mappings :

```
cavernes préhistoriques   isA(IncStrict)    caverne (Sim: 0.486)
grottes ornées            isClose(simRel)   grotte  (Sim: 0.724)
grottes préhistoriques    isA(IncStrict)    grotte  (Sim: 0.445)
abîmes                    isA(FilsdeEq)     grotte
antres                    isA(FilsdeEq)     grotte
```

```
cavités souterraines      isA(FilsdeEq)     grotte
spélonques                isA(FilsdeEq)     grotte
```

Exemple cas 2

Dans ce deuxième exemple (cf. Fig. 5.7), un mapping d'équivalence est trouvé sur le concept "commune". Mais les trois autres mappings trouvés n'étant pas jugés fiables pour la validation du domaine de la source, de celle-ci est rejetée pour l'enrichissement. En effet, parmi ces trois autres mappings possibles, l'un rattachait "propriété foncière" de label alternatif "fonds" comme une généralisation de "hauts fonds" et les deux autres considéraient "communaux" et "biens communaux" comme des spécialisation de "collectivité territoriale" puisqu'assez proches de "commune".

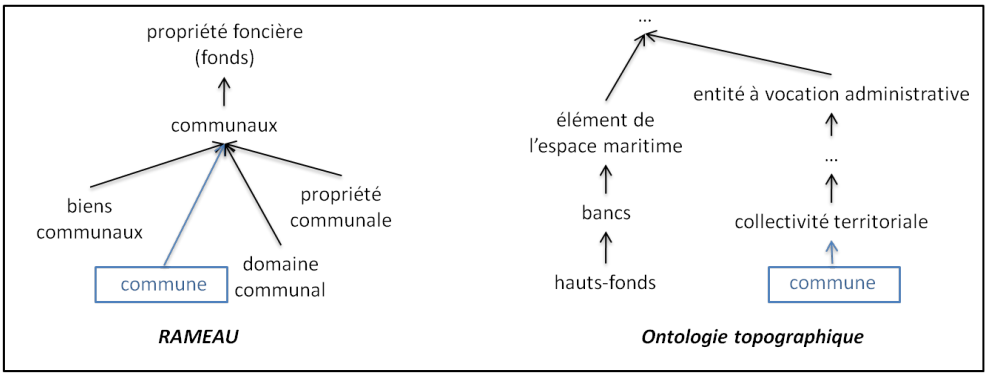

FIG. 5.7 – Exemple de source dont le domaine est rejeté

Exemple cas 3

La figure 5.8 présente une source dont le domaine reste à valider par l'expert. Dans cet exemple l'alignement identifie deux mappings d'équivalence mais les concepts cibles impliqués dans ces mappings, "métro" et "tunnel", ne sont pas en relation dans la cible et la validité du domaine de la source ne peut pas être affirmée automatiquement.

L'existence de ce statut intermédiaire, domaine ni validé, ni rejeté, a été décidé avec l'expert. En effet, à la présentation de cet exemple à l'expert, celui-ci a considéré qu'un certain nombre de concepts présents dans la composante pouvaient éventuellement être jugés intéressants pour enrichir la cible, et qu'il préférait pouvoir juger lui-même.

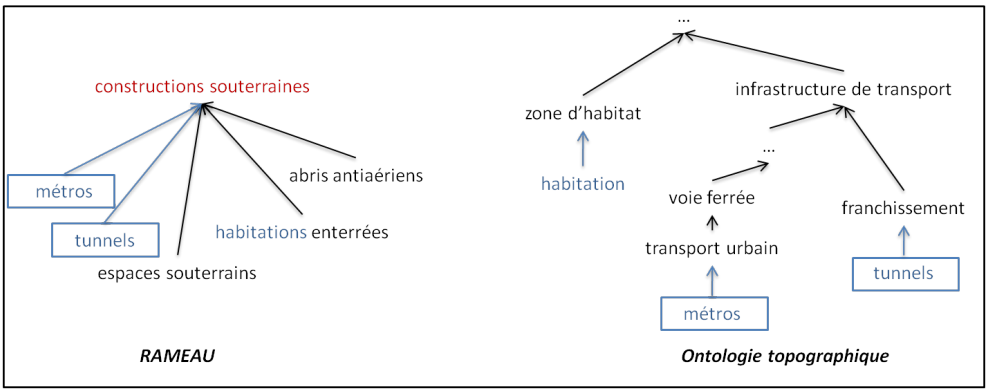

FIG. 5.8 – Exemple de source dont le domaine doit être évalué manuellement

Remarquons que même si le domaine de la source est qualifié de valide, tous les mappings trouvés ne sont pas forcément pertinents même s'ils l'ont été avec des techniques dites fiables. Ceci est dû à la granularité des informations représentées dans RAMEAU et par le fait qu'il s'agit d'un thésaurus et non d'une ontologie. Dans l'exemple "Glaciers" présenté fig 5.9, le domaine est validé puisqu'on trouve bien deux mappings fiables entre des concepts reliés dans leurs ontologies "Glacier" et "Moraine". Mais les concepts apparaissant sous le concept "Glacier" dans la composante RAMEAU n'étant pas tous des spécialisations topographiques de ce concept, les mappings construits ne sont pas tous pertinents pour l'enrichissement.

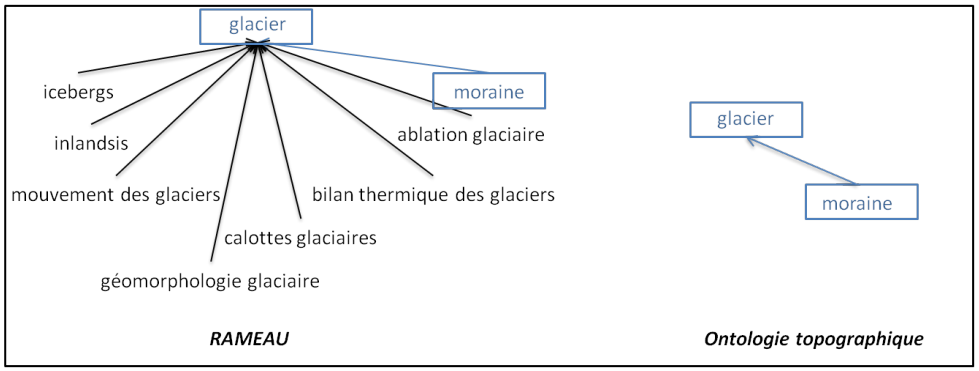

FIG. 5.9 – Exemple de source de domaine valide sans que tous les mappings soient pertinents

Les trois mappings présentés ci-dessous sont produits par alignement après validation du domaine de la source mais sont clairement non pertinents pour une ontologie topographique :

```
Géomorphologie glaciaire      isA(FilsdeEq)  Glacier
Mouvement des glaciers        isA(FilsdeEq)  Glacier
Bilan thermique des glaciers  isA(FilsdeEq)  Glacier
```

De ce fait, nous avons donc convenu avec l'expert que les propositions d'enrichissement basés sur les mappings produits devront donc toujours lui être soumis afin qu'il les valide. Ce n'est qu'à l'issue de ce processus de validation qu'elles viendront compléter l'ontologie à enrichir.

Exemple d'enrichissements attendus

Les mappings trouvés précédemment et validés par l'expert seront utilisés pour enrichir l'ontologie cible en s'appuyant sur les relations établies dans les mappings. Reprenons par exemple les mappings identifiés dans le cas 1 présenté plus haut dans ce chapitre. Supposons que l'expert a validé les 5 mappings suivants, tout en souhaitant regrouper les deux concepts "grottes ornées" et "grottes préhistoriques" en un seul concept possédant deux labels :

```
cavernes préhistoriques   isA(IncStrict)     caverne (Sim: 0.486)
grottes ornées            isClose(simRel)    grotte  (Sim: 0.724)
grottes préhistoriques    isA(IncStrict)     grotte  (Sim: 0.445)
abîmes                    isA(FilsdeEq)      grotte
antres                    isA(FilsdeEq)      grotte
```

Effectuer l'enrichissement en s'appuyant sur cette validation consistera à introduire les nouveaux concepts, "grottes ornées", "abimes", "antres" comme de nouvelles spécialisations du concept "grotte" et le concept "cavernes préhistorique" comme une spécialisation de "caverne" comme présenté en Fig. 5.10.

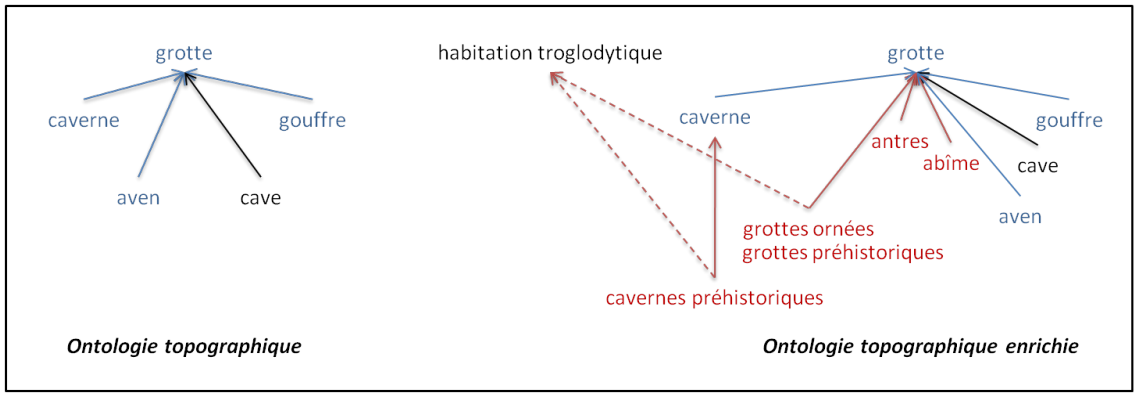

FIG. 5.10 – Exemple d'enrichissement à partir de résultats d'alignement validés

Nous avons aussi supposé en 5.10, que l'expert avait décidé de plus d'introduire des liens supplémentaires pour ces nouveaux concepts, en reliant par exemple les concepts "grottes ornées" et "cavernes préhistoriques" comme des spécialisations du concept "habitation troglodytique" présent par ailleurs dans l'ontologie. L'introduction des nouveaux concepts est faite pour l'instant manuellement, même si une interface dédiée à la fois à la validation des mappings et à leur introduction automatique dans l'ontologie est parfaitement envisageable.

L'enrichissement peut aussi s'appuyer sur les labels alternatifs des concepts d'une source validée, l'expert pouvant décider de rapatrier ou pas l'ensemble ou certains de ces labels soit comme des labels alternatifs du concept auquel ils sont associés, soit comme des concepts distincts. Par exemple dans un mapping, le concept "habitations" de la source est aligné avec le concept de même label de la cible. Ce concept possède 9 autres labels dans la source (exemple ci-dessous) parmi lesquels l'expert pourra choisir ceux qu'il souhaite retenir.

`Habitations` *isEq* `Habitations`

`Les labels du concept habitation : habitation, habitat humain, demeures, maisons, bâtiments d'habitation, logements, résidences, maisons d'habitation, locaux d'habitation, logis`

5.4.3 Patrons de validation de domaine

Dans le chapitre 3, nous avons présenté le langage MPL et décrit son utilisation pour le raffinement de mappings. Dans cette partie, nous montrons comment ce langage peut être utilisé pour automatiser le traitement de validation du domaine d'une source par rapport à celui de la cible.

Pour l'enrichissement à partir de fractions d'une ontologie généraliste, extraites automatiquement, nous proposons deux patrons de validation de domaine. Le premier patron valide le domaine de l'ontologie source O_S si le résultat de l'alignement entre O_S et O_T contient au moins un mapping d'équivalence et un mapping fiable, i.e. d'équivalence (t_1), d'inclusion stricte (t_2) ou de forte proximité relative (t_5), que les concepts intervenant dans ces mappings sont différents et qu'ils sont en relation dans leur ontologie respective, i.e. en relation fils-père ou frère. Le deuxième patron conclut sur une "validité à vérifier" si les deux mappings, d'équivalence et fiable, existent mais que les concepts de la cible concernés par ces mappings ne sont pas en relation dans celle-ci. Si les deux patrons précédemment décrits échouent, le domaine de la source est rejeté.

Les nouvelles primitives introduites sont les suivantes :

- $isReliableMapping(x, y)$ est vrai ssi $\exists(x, y, isEq, t_1) \in T_M$ ou $\exists(x, y, isA, t_2) \in T_M$ ou $\exists(x, y, isClose, t_5) \in T_M$
- $isRelated(x, y, O)$ est vrai ssi $isA(x, y) \in H$ ou si $isA(y, x) \in H$ ou ($isA(x, p) \in H$ et $isA(y, p) \in H$)

Les patrons de validation de domaine sont :

Partie contexte du patron :
$\exists x_1 \exists y_1\ \exists x_2 \exists y_2\ (isEquivalent(x_1, y_1) \wedge isReliableMapping(x_2, y_2) \wedge differentConcept(x_1, x_2)$
$\wedge\ differentConcept(y_1, y_2) \wedge isRelated(x_1, x_2, O_S) \wedge isRelated(y_1, y_2, O_T)\ )$
Partie solution du patron :
$domaineValide(x_1, y_1, x_2, y_2)$

Partie contexte du patron :
$\exists x_1 \exists y_1\ \exists x_2 \exists y_2\ (isEquivalent(x_1, y_1) \wedge isReliableMapping(x_2, y_2)$
$\wedge\ differentConcept(x_1, x_2) \wedge differentConcept(y_1, y_2) \wedge isRelated(x_1, x_2, O_S))$
Partie solution du patron :
$domaineValide(x_1, y_1, x_2, y_2)$

5.5 Enrichissement à partir d'une ontologie généraliste très volumineuse

L'ontologie YAGO (Yet Another Great Ontology) est une énorme base de connaissances décrivant plus de 2 millions d'entités, des personnes, des organisations, des villes et contenant plus de 20 millions de faits sur ces entités. YAGO a été développée à l'Institut d'informatique Max Planck de Sarrebruck. Les données de YAGO proviennent de Wikipedia et sont structurées à l'aide de WordNet. YAGO étant très volumineuse, elle ne peut pas être alignée directement dans sa totalité et doit être au préalable partitionnée.

Pour effectuer cette partition, nous allons utiliser l'une des deux méthodes de partitionnement précédemment présentées au chapitre 4, la méthode PAP. Nous choisissons cette méthode car elle permet de mieux maîtriser le mécanisme de partitionnement, en particulier celui de l'ontologie source, qui sera ici YAGO. En effet, la méthode PAP (Partition, Anchor, Partition) consiste à d'abord partitionner l'ontologie cible, puis à forcer le partitionnement de la source à suivre celui réalisé pour la cible, en s'appuyant sur les *ancres*, i.e. les concepts des deux ontologies qui ont les mêmes labels. Si l'ontologie cible est bien décomposée au départ, la décomposition de la source devrait suivre.

De plus, si le nombre de blocs cible est connue au départ, la méthode PAP permet, par construction, de centraliser dans le même nombre de blocs sources, les concepts de la source proches de ceux présents dans chacun des blocs cible. Si la taille des blocs sources est limitée, cela permettra donc de n'extraire de l'ontologie cible, ici la volumineuse source YAGO, que les sous-parties dont les thématiques sont proches de celles exprimées dans les différents blocs de la source et de laisser le reste de l'ontologie se décomposer dans des blocs qui ne seront pas examinés.

Pour mettre en œuvre la méthode PAP dans ce contexte, deux adaptations sont nécessaires pour s'assurer, d'une part, de la bonne décomposition de l'ontologie cible, et d'autre part, de la validité des ancres identifiées.

5.5.1 Adaptations nécessaires de la méthode de partitionnement PAP

La première adaptation souhaitable dans notre contexte découle de l'objectif d'enrichissement. Cet objectif impose de s'assurer que le partitionnement de la première ontologie conduit à des blocs les plus cohérents possibles, puisque cette première décomposition doit servir de guide à la suivante. Pour faire simple, et notre ontologie topographique étant de taille réduite (600 concepts), nous avons procédé à une décomposition manuelle de cette ontologie en 10 blocs de tailles variées mais de thématiques ciblées. Cette démarche manuelle peut aussi se justifier par le fait que l'ontologie cible à décomposer est bien connue de ses concepteurs et que cette décomposition sera éventuellement réutilisée, i.e. chaque fois qu'on voudra enrichir l'ontologie avec les concepts d'une nouvelle source potentielle.

La deuxième adaptation est due au fait que YAGO est une ontologie généraliste. L'adaptation porte donc sur la phase de calcul d'ancres, i.e. l'identification des paires de concepts qui peuvent être considérés comme équivalents. Ce calcul compare les labels des concepts de deux ontologies et n'est basé que sur une mesure de similarité lexicale, stricte et peu coûteuse. Quand les ontologies à partitionner sont supposées représenter le même domaine, le problème d'ambi-

guïté ne se pose pas : si les concepts ont des labels équivalents, ils peuvent effectivement être considérés comme équivalents. Cependant si une des ontologies à partitionner, en l'occurrence YAGO, est une ontologie généraliste, i.e. une ontologie qui peut représenter plusieurs domaines à la fois, la phase de calcul d'ancres est perturbée du fait de la présence dans la même ontologie de concepts qui ont les mêmes labels mais des sémantiques différentes (problème d'ambiguïté).

L'exemple de la figure 5.11 illustre ce problème d'ambiguïté : le concept "lock" de l'ontologie topographique a le même label que deux concepts de YAGO mais ne partage le sens que du deuxième concept.

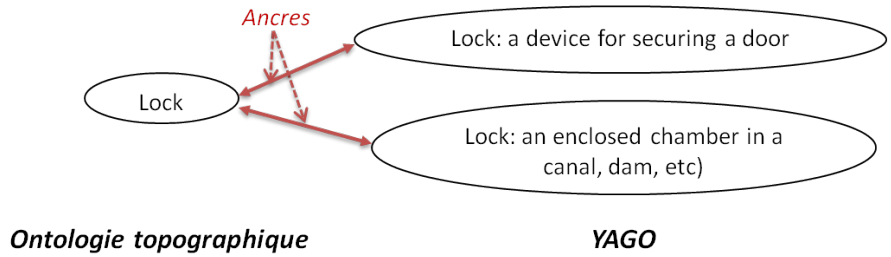

FIG. 5.11 – Concepts de même label mais de sens distincts

Le problème est donc d'écarter les faux homonymes et de choisir la bonne ancre, si elle existe. L'idée principale va consister à prendre en considération, pour chaque label partagé par des concepts dans les deux ontologies, le voisinage de chacun des concepts dans son ontologie respective et de comparer ces voisinages. Pour écarter les faux homonymes, un concept de la source ne pourra être reconnu comme une *ancre plausible* d'un concept de la cible que si son voisinage partage au moins une autre ancre avec le voisinage du concept cible. Si un concept de la cible a plusieurs ancres plausibles, on choisira l'ancre dont le voisinage est le plus proche de celui de la cible.

5.5.2 Algorithme de validation des ancres

Dans un premier temps, on recherche pour chaque concept de chacun des blocs cibles, si un ou plusieurs concepts de YAGO ont un label équivalent. On construit ainsi un ensemble de paires d'*ancres potentielles* où chaque paire associe un concept de la cible aux concepts de YAGO de même label que lui. L'exemple de la figure précédente sera ainsi représenté par la paire (Lock, {Lock_1, Lock_2 }). Puis on identifie le voisinage de chacun des concepts intervenant dans une même paire et on les compare.

Voisinage d'un concept

L'identification du voisinage d'un concept reprend une idée proposée par Maedche et Staab [2] (section 4 du Chapitre 2), qui définissent la sémantique d'un concept au sein d'une ontologie par l'ensemble de ses généralisants et de ses spécialisants. Dans notre contexte, le voisinage d'un

concept contiendra son père et son grand-père, ses fils et petits-fils, et ses frères.

Le choix d'introduire les frères dans le voisinage se justifie par les deux considérations suivantes.

Nous avons tout d'abord remarqué que les concepts qui avaient exactement les mêmes labels dans deux ontologies distinctes étaient, assez souvent, des concepts feuilles de ces ontologies, i.e. des concepts sans descendants. Une explication de ce phénomène peut être qu'ils correspondent souvent à des concepts du monde réel dont la définition et le nom sont clairement établis et partagés par tous, alors que les concepts plus généraux sont plus souvent identifiés et nommés par les concepteurs d'une ontologie pour expliciter des regroupements de propriétés ou des points de vue pertinents pour cette ontologie et sont donc moins universels. Si on ne prend pas en compte les frères, le voisinage d'un concept feuille se limiterait à son père et son grand père, i.e. un voisinage assez pauvre et qui ne permettrait pas de rendre compte assez précisément du contexte d'interprétation du concept considéré dans son ontologie.

La deuxième considération tient au fait que deux ontologies peuvent ne pas être exactement construites de la même façon et qu'un même concept peut être considéré dans l'une comme le père d'un concept et dans l'autre, comme son frère. L'exemple ci-dessous montre que dans ontologie topographique, une église est un bâtiment religieux chrétien au même titre qu'une cathédrale ou une abbaye, alors que ces concepts sont considérés comme ses spécialisations dans YAGO. En prenant en compte les frères d'un concept dans son voisinage, on évite ainsi de pénaliser la comparaison des voisinages lorsque les ontologies présentent de petites différences de structuration.

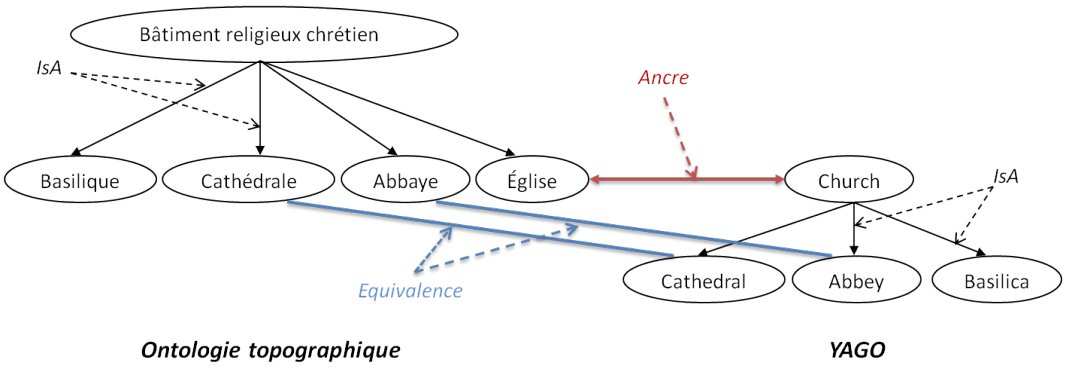

FIG. 5.12 – Exemple de structurations différentes

Comparaison des voisinages

Celle-ci s'effectue en plusieurs passes. Dans la première passe (cf. Algorithme 8), on ne retient comme "ancres plausibles" que les concepts de la source dont le voisinage partage au moins une deuxième paire de concepts de même label avec le voisinage du concept de la cible avec lequel on les compare. En effet, si les voisinages de deux concepts de même label partagent au moins un autre concept de même nom, il est très probable que le contexte d'interprétation des deux concepts étudiés soit le même.

Algorithm 8 anchorStructFilter(anchorList, nbCommon, thresholds_Sub)

1: $anchorMap = <, >$, $anchorList = \{\}$, $anchor = <, >$
2: $anchorMap = extractAnchorMap(anchorList)$ {la liste des mappings ambigues}
3: $trueAnchor = false$, $commonNeighbor = 0$
4: **for** each $concept1$ in $anchorMap$ **do**
5: **for** each $concept2$ in $getAnchors(anchorMap, concept1)$ **do**
6: **for** each $neighborConcept1$ in $calculateNeighbor(concept1)$ **do**
7: **for** each $neighborConcept2$ in $calculateNeighbor(concept2)$ **do**
8: $sim\_Sub = sub\_Similarity(neighborConcept1, neighborConcept2)$ {calcul de la similarité entre les labels}
9: **if** $(sim\_Sub = 1)$ **then**
10: $trueAnchor = true$ {au moins une vraie ancre dans le voisinage}
11: **end if**
12: **if** $(sim\_Sub >= thresholds\_Sub)$ **then**
13: $commonNeighbor = commonNeighbor + 1$
14: **end if**
15: **end for**
16: **end for**
17: $simSC = semanticCotopy\_Similarity(commonNeighbor\ ,\ |calculateSc(concept1)|, |calculateSc(concept2)|)$ {calcul de la sémantic Cotopy}
18: **if** $(simSC > previous\_simSC)$ and $(trueAnchor = true)$ **then**
19: $previous\_simSC = simSC$, $maxCommon = commonNeighbor$
20: $anchor =< concept1, concept2 >$
21: **end if**
22: trueAnchor = false, commonNeighbor = 0
23: **end for**
24: **if** $(maxCommon >= nbCommon)$ **then**
25: $anchorList = anchorlist \cup anchor$
26: **end if**
27: **end for**
28: **return** *anchorList*

Si une seule ancre est plausible pour un concept cible, elle est automatiquement considérée comme valide. Si plusieurs ancres sont plausibles pour un même concept cible, nous calculons, pour chaque ancre plausible, la similarité de chacun des labels des concepts présents dans son voisinage avec tout ceux du voisinage du concept de la cible, en excluant la paire d'ancres en cours de validation, en utilisant la mesure I-Sub [Stoilos et al., 2005] (utilisée par FALCON, Chapitre 4).

On décompte ensuite, dans chaque voisinage, le nombre de "concepts proches" d'un concept du voisinage du concept cible, i.e. ayant une similarité supérieure à un seuil, et on retient comme "ancre valide" celle dont le voisinage contient le plus de "concepts proches".

Le fait d'exclure la paire d'ancres en cours de validation de l'ensemble des concepts dont les labels sont comparés avec la mesure de Levenshtein, permet d'éviter de donner trop de poids aux concepts dont les labels incluent ceux des ancres puisqu'ils peuvent être rapprochés de l'ancre même si leur sens est très différent. Cela perturberait de manière considérable la résolution de l'ambiguïté. Dans l'exemple précédent, Fig. 5.13, le concept Lock (écluse) de l'ontologie topo-

graphique serait relié au concept Lock_1 de l'ontologie YAGO car le voisinage de ce concept contiendrait trois "concepts proches" qui en réalité n'en sont pas. En effet, le sens correct avec lequel on veut relier le concept de l'ontologie cible est le deuxième qui est "an enclosed chamber in a canal, dam, ...etc".

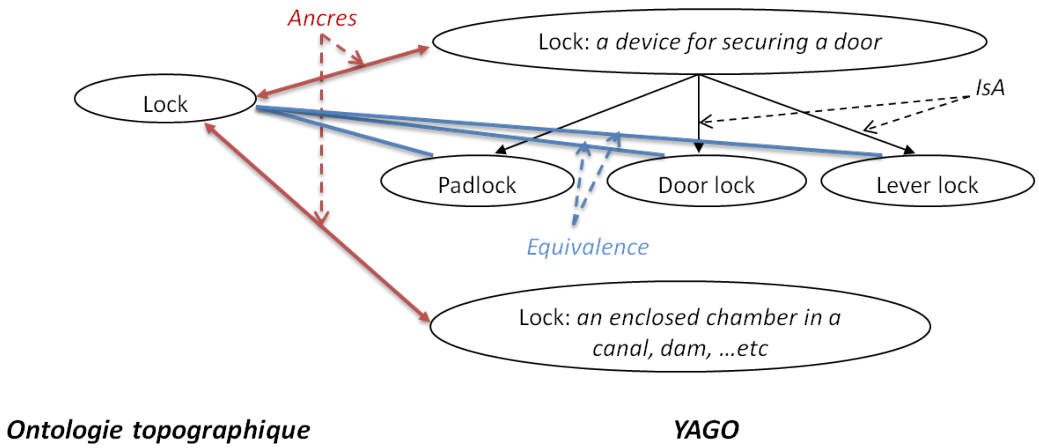

FIG. 5.13 – Exemple de concepts "faussement proches" d'une ancre

La deuxième passe (cf. Algorithme 9) s'effectue sur les concepts pour lesquels on n'a pas pu retenir dans la première passe une unique "ancre valide" soit parce que plusieurs "ancres plausibles" avaient le même nombre de "concepts proches" ou soit qu'aucune ancre n'était "plausible" i.e. aucun des voisinages ne partageait une deuxième ancre, y compris dans le cas où un seul concept de la source avait le même label que celui de la cible.

Dans cette deuxième passe, on essaye de tirer profit de l'information plus large donnée par les blocs. Au lieu de comparer le voisinage d'une ancre potentielle de la source avec le seul voisinage du concept de la cible correspondant, on le compare à l'ensemble des concepts de la cible qui apparaissent dans les paires d'ancres potentielles initiales du même bloc. L'idée est que si les deux ontologies n'ont pas du tout la même structure mais que les blocs de la cible ont une thématique suffisamment ciblée, le fait de retrouver dans le voisinage d'une "ancre potentielle" de la source des concepts de la cible peut suffire à lever l'ambiguité sur cette ancre, même si les concepts trouvés ne sont pas présents dans le voisinage du concept cible auquel on le compare. Le nombre de concepts de la cible trouvés dans le voisinage d'un concept source permettra de choisir entre deux ancres plausibles qui partageaient le même nombre de concepts proches ou permettra de valider une ancre qui ne l'était pas dans la passe précédente.

Exemple :

Pour la paire d'ancre potentielle ("Island", {"wordnet_island_103587318", "wordnet_island_109316454" }), le concept cible "Island" a pour voisinage dans la cible, les concepts suivants : (`seacoast, sea, navigable channel, ocean, reef, bank, bay, maritime feature, islet, natural topographic entity`)

Les voisinages respectifs de ses deux correspondants sont dans YAGO,

pour "wordnet_island_103587318", (`zone, topographic_point, traffic_island, kitchen_island`)

et pour "wordnet_island_109316454", (`oxbow, peninsula, floor, foreland, landmass, mainland, archipelago, coastal_plain, slash, plain, isthmus, neck, cape, forest, wonderland, beachfront, land, object, Aegean_island, barrier_island, South_Sea_Islands`)

Les voisinages de ces deux concepts ne contenant pas une autre paire d'ancres partagée avec le voisinage du concept "Island" de la cible, on les compare aux autres ancres du bloc cible auquel appartient "Island" et on retiendra "wordnet_island_109316454" comme ancre valide car son voisinage contient les deux ancres "peninsula" et "isthmus".

Algorithm 9 anchorStructFilter(anchorList, nbCommon)

```
1: commonNeighbor = 0, anchor =<, >
2: for each concept1 in anchorMap do
3:   if not (anchorListcontains < concept1, _ >) then
4:     for each concept2 in getAnchors(anchorMap, concept1) do
5:       for each neighborConcept2 in calculateNeighbor(concept2) do
6:         if anchorInSameBlock(concept1) contains neighborConcept2 then
7:           {anchorInSameBlock retourne les ancres qui sont dans le même bloc que
             concept1}
8:           commonNeighbor = commonNeighbor + 1
9:         end if
10:      end for
11:      simSC = semanticCotopy_Similarity(commonNeighbor , |calculateSc(concept1)|,
         |calculateSc(concept2)|)
12:      if (simSC > previous_simSC) then
13:        previous_simSC = simSC, maxCommon = commonNeighbor
14:        anchor =< concept1, concept2 >
15:      end if
16:      commonNeighbor = 0
17:    end for
18:    if (maxCommon >= nbCommon) then
19:      anchorList = anchorlist ∪ anchor
20:    end if
21:  end if
22: end for
23: return anchorList
```

Une fois identifiées les ancres valides des différents blocs de la source, le partitionnement de l'ontologie s'effectue comme décrit dans le chapitre 4. La dernière étape consiste à aligner les différents blocs et à utiliser les alignements pour l'enrichissement comme nous l'avons présenté dans les précédentes sections de ce chapitre.

Conclusion

Nous avons montré dans ce chapitre comment l'environnement TaxoMap Framework associé à notre outil d'alignement permet d'aider l'expert suivant le contexte considéré à faire le tri parmi les différentes sources qui pourraient être utilisées pour l'enrichissement, en rejetant celles dont la thématique ne correspond pas à celle de son ontologie particulière et, si les thématiques des deux ontologies sont compatibles, à l'aider à éliminer les mappings peu pertinents pour l'enrichissement. Nous avons montré aussi comment TaxoMap Framework associé à la méthode de partitionnement permet l'enrichissement à partir d'ontologies volumineuses telle que YAGO.

Les patrons d'enrichissement présentés ont pu être construits facilement en réutilisant les primitives déjà écrites pour la tâche de raffinement en ne nécessitant d'introduire que deux nouvelles primitives, ce qui prouve l'extensibilité de l'approche.

Chapitre 6

Développements et validations expérimentales

Sommaire

Introduction

L'environnement TaxoMap Framework a fait l'objet de développements sur lesquels nous nous somme appuyés pour faire des expérimentations. Ainsi, dans ce chapitre, nous présentons, dans un premier temps, les outils développés du point de vue technique.

Nous présentons ensuite certaines expérimentations réalisées dans le cadre de travail de thèse. Nous présentons, tout d'abord, les tests réalisés sur le module d'alignement TaxoMap. Nous présentons ensuite, les tests sur l'approche de raffinement de mappings et sur les méthodes de partitionnement d'ontologies de très grande taille. Enfin, nous présentons les tests sur l'approche d'enrichissement d'ontologies proposée. Nous détaillons, pour chaque test, la démarche expérimentale suivie et l'analyse des résultats obtenus.

6.1 Réalisations logicielles

Dans cette partie, nous présentons, dans un premier temps, les différentes architectures et interfaces graphiques implémentées dans l'environnement TaxoMap Framework. Nous présentons ensuite la façon d'installer et d'utiliser cet environnement.

6.1.1 Implémentation de l'environnement TaxoMap Framework

L'environnement TaxoMap Framework, que nous avons développé, intègre les différents modules qui implémentent les propositions de cette thèse. En particulier, il intègre la nouvelle version V_2 du module d'alignement TaxoMap (cf. Chapitre 2), les deux modules de raffinement et d'enrichissement (cf. Chapitre 3 et 5) et le module de partitionnement TaxoPart (cf. Chapitre 4).

Dans les sections suivantes, nous présentons, pour chaque module, son architecture et son interface graphique.

Architecture du module d'alignement TaxoMap

Le module d'alignement TaxoMap est composé de (cf. Fig. 6.1) :

- *un parseur OWL :* le parseur extrait, à partir d'une ontologie source et d'une ontologie cible, les identifiants et les labels des concepts ainsi que les relations de subsomption. Les informations extraites sont ensuite stockées dans une base de données SQLite. Le parseur OWL utilisé est Jena [13].

- *un module de lemmatisation :* ce module intègre l'analyseur morpho-syntaxique TreeTagger. TreeTagger est exécuté sur les labels des concepts stockés dans la base de données. Les résultats de lemmatisation sont stockés dans la base de données.

- *un module d'alignement :* ce module calcule, dans un premier temps, la similarité entre les concepts des deux ontologies alignées. Les techniques d'alignement sont ensuite appliquées pour identifier des correspondances entre les concepts. Les mappings générés sont stockés dans la base de données.

13. `http://jena.sourceforge.net/`

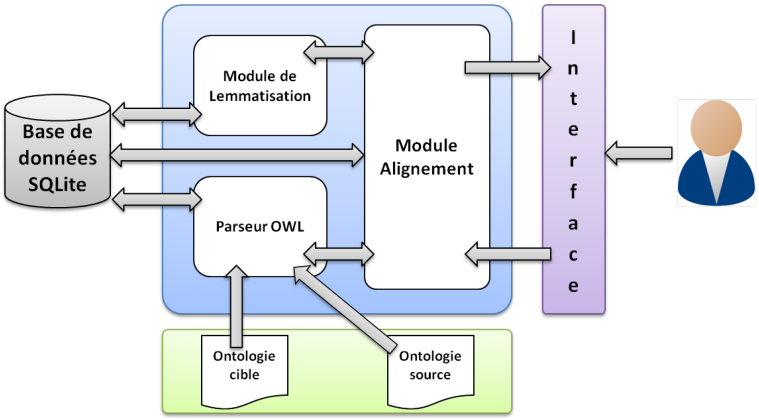

FIG. 6.1 – Architecture du module d'alignement TaxoMap

Interaction avec le module d'alignement TaxoMap

Dans l'interface graphique du module d'alignement (cf. Fig. 6.2), l'utilisateur indique, dans un premier temps, les ontologies source et cible à aligner, la langue dans laquelle le vocabulaire de ces ontologies est exprimée et le répertoire de sortie pour stocker les résultats. Dans un second temps, l'utilisateur choisit les techniques d'alignement qu'il veut utiliser, leur seuil, et leur ordre d'exécution. Notons que des valeurs de seuils par défaut sont proposées suite à de nombreuses expériences.

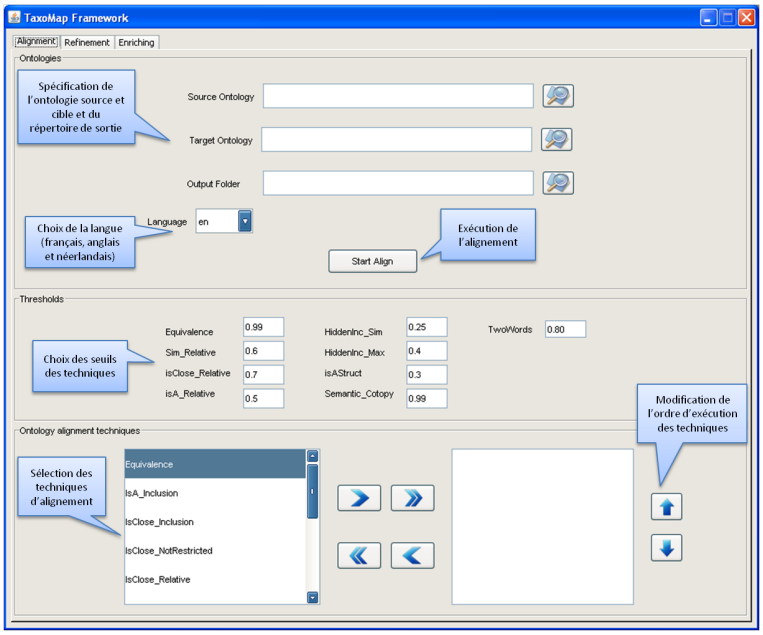

FIG. 6.2 – Interface du module d'alignement TaxoMap

Architecture des modules de raffinement et d'enrichissement

Les modules d'enrichissement et de raffinement font partie intégrante de l'environnement TaxoMap Framework (cf. Fig. 6.3). Leur description ne peut être dissociée de celle de cet envi-

ronnement dont les principaux composants sont :

- *un module de raffinement :* ce module extrait de la base de données les informations concernant les concepts (labels, relations) et les mappings générés par le module d'alignement. Les mises à jour effectuées (suppression, ajout) sur les mappings suite à l'exécution des traitements de raffinement (effectués par le moteur d'inférence) sont sauvegardées dans la base données.

- *un module d'enrichissement :* ce module extrait, comme le module de raffinement, de la base de données les informations concernant les concepts et les mappings. Après application de patrons, de nouveaux concepts/relations sont ajoutés à l'ontologie cible.

- *un moteur d'inférence :* le moteur d'inférence applique les patrons choisis par l'expert sur les mappings (dans le cas du raffinement) et sur les ontologies (dans le cas de l'enrichissement).

- *une bibliothèque de patrons :* cette bibliothèque est composée de patrons appliqués par le moteur d'inférence.

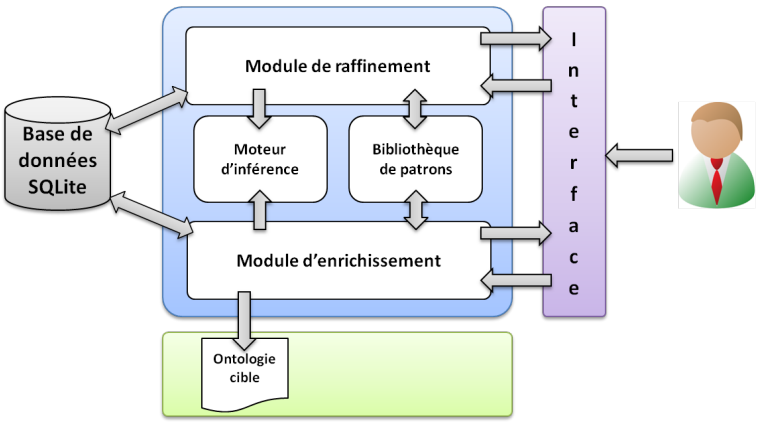

FIG. 6.3 – Architecture de l'environnement TaxoMap Framework

Interaction avec les modules de raffinement et d'enrichissement

L'interface graphique du module de raffinement (cf. Fig. 6.4) permet à l'expert, soit de créer un nouveau patron de raffinement, soit de choisir un patron qui existe dans la bibliothèque des patrons. Pour créer un nouveau patron, l'expert doit spécifier, dans un premier temps, pour chaque ontologie, les variables qu'il veut utiliser. Dans un second temps, il doit choisir les primitives à utiliser dans la partie contexte et les primitives à utiliser dans la partie solution. La validation d'un patron l'ajoute automatiquement à la bibliothèque des patrons. L'expert peut alors le choisir et l'exécuter.

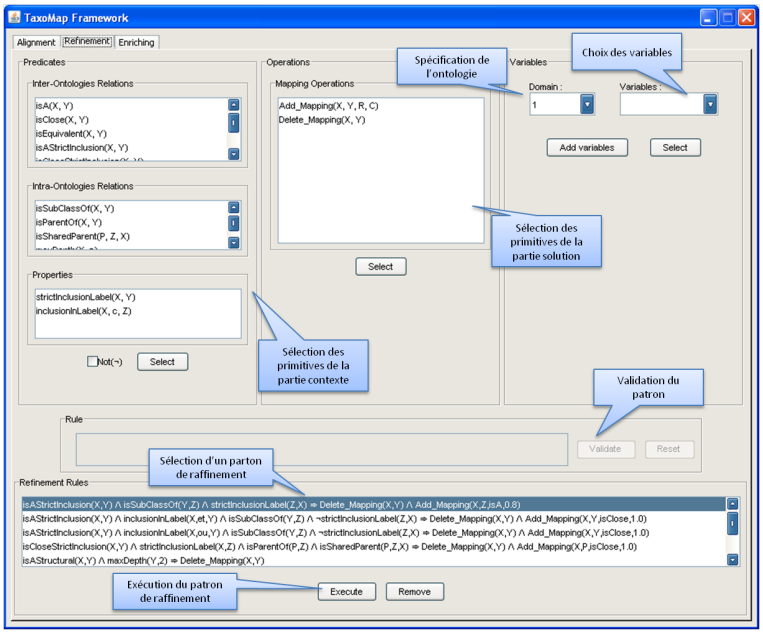

FIG. 6.4 – Interface du module de raffinement

L'interface graphique du module d'enrichissement (cf. Fig. 6.5) est semblable à celle du raffinement. La seule différence réside dans les primitives utilisées dans la partie contexte et la partie solution, et dans la bibliothèques des patrons (seuls les patrons d'enrichissement sont présentés à l'expert).

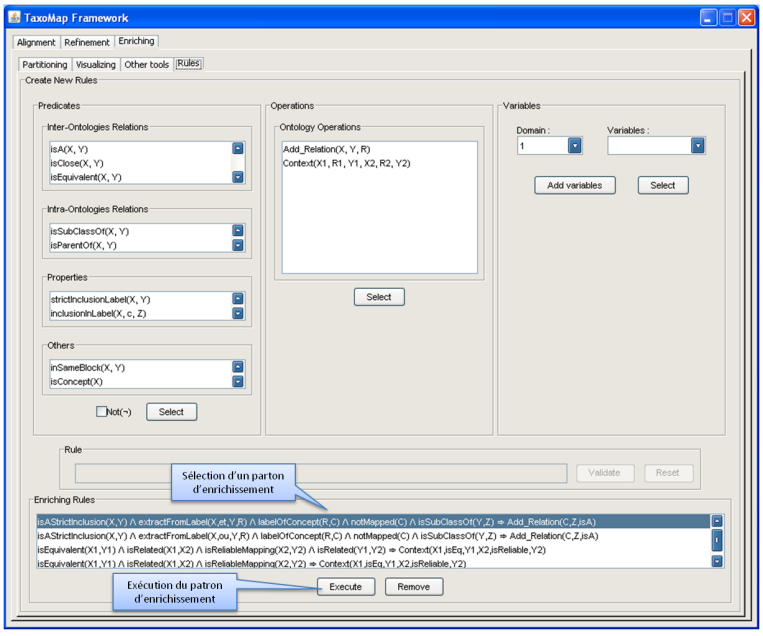

FIG. 6.5 – Interface du module d'enrichissement

Architecture du module de partitionnement TaxoPart

Le module de partitionnement (cf. Fig. 6.6) implémente les deux méthodes de partitionnement PAP et APP (cf. Chapitre 4). Il prend en entrée les deux ontologies à partitionner, et génère en sortie les blocs créés. Ce module peut également prendre en entrée un alignement de référence. Les mappings de cet alignement sont alors considérés comme des ancres. Il peut aussi prendre en entrée des blocs déjà construits de l'ontologie cible.

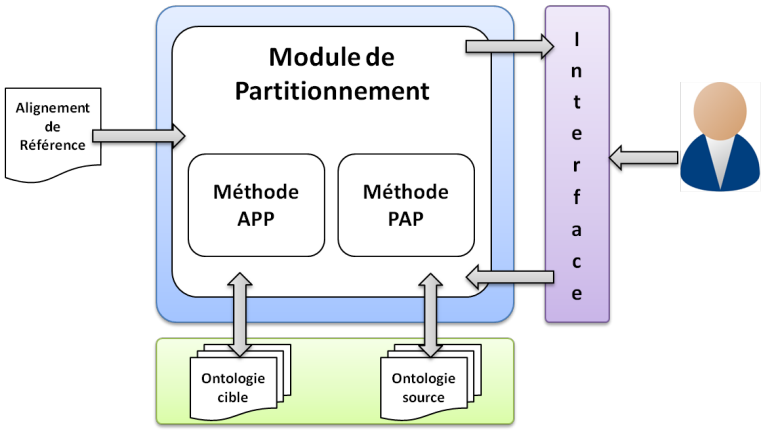

FIG. 6.6 – Architecture du module de partitionnement

Interaction avec le module partitionnement TaxoPart

L'interface graphique du module de partitionnement TaxoPart (cf. Fig. 6.7) permet à l'utilisateur de choisir les différents paramètres nécessaires au processus de partitionnement.

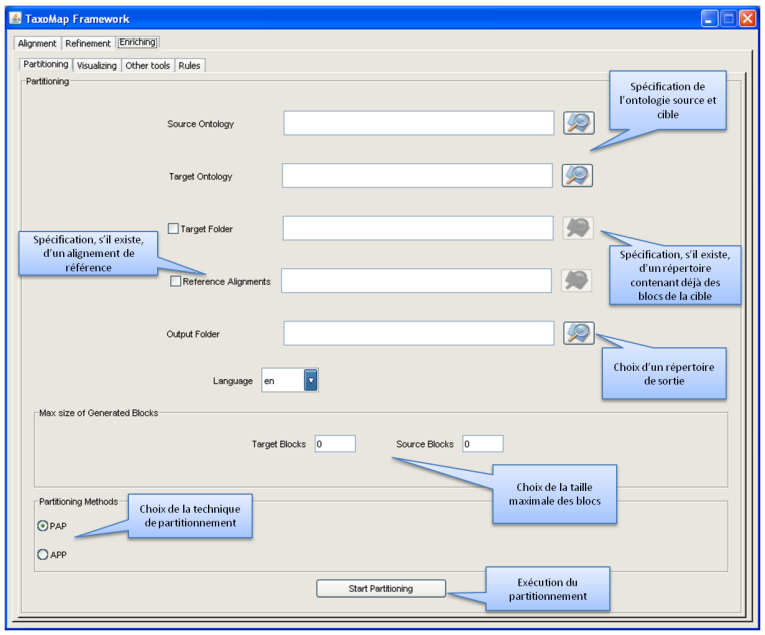

FIG. 6.7 – Interface de l'environnement TaxoMap Framework

L'utilisateur doit spécifier, dans un premier temps, l'emplacement des ontologies qu'il veut partitionner, l'emplacement du répertoire de sortie, l'emplacement de l'alignement de référence et du répertoire contenant déjà des blocs construits de l'ontologie cible (dans le cas où ces deux derniers existent). Dans un second temps, l'utilisateur doit choisir la méthode de partitionnement qu'il veut voir appliquer et la taille maximale des blocs à construire.

6.1.2 Installation de l'environnement TaxoMap Framework

Les différents modules de l'environnement TaxoMap Framework ont été implémentés dans le langage Java [14]. La dernière version de cet environnement est téléchargeable à partir de l'adresse web : `http://www.lri.fr/~hamdi/TaxoMap/TaxoMap.html`.

Concernant le module d'alignement TaxoMap, la base de données SQLite et l'analyseur morpho-syntaxique TreeTagger ont été ajoutés au paquetage de l'environnement TaxoMap Framework. Ainsi, l'installation de l'environnement TaxoMap Framework ne nécessite que Java dans le cas de machines linux et mac, et de Perl [15] (nécessaire pour utiliser TreeTagger) et Java dans le cas de machines windows.

Pour lancer l'environnement TaxoMap Framework, il faut double-cliquer sur le .jar, ou taper en ligne de commande :

```
$> tar xzf TaxoMapFrame-3.6.tar.gz
$> cd TaxoMapFrame-3.6
$> java -jar TaxoMapFrame3_6.jar
```

6.2 Tests du module d'alignement TaxoMap

Afin d'évaluer les différentes adaptations de TaxoMap, nous avons réalisé des expérimentations avec les versions suivantes du logiciel (cf. Chapitre 2) :

Version V_0 : il s'agit de la version initiale de TaxoMap développé en 2006 dans le cadre du projet ANR RNTL eDot.

Version V_1 : cette version correspond à une extension de la version V_0 intègrant l'analyseur morpho-syntaxique TreeTagger.

Version V_2 : cette version intègre non seulement TreeTagger mais prend aussi en compte les adaptations de la mesure de similarité, la normalisation des concepts, les nouvelles techniques proposées décrites au chapitre 2. Elle est, par ailleurs, utilisable via une interface.

Ces expérimentations ont été d'abord réalisées sur des ontologies du domaine géographique, fournies par le COGIT [16] dans le cadre du projet ANR GeOnto. Ces tests ont été très intéressants car les mappings générés ont pu être validés par des experts du domaine. Ils ont permis

14. `http://www.oracle.com/technetwork/java/index.html`
15. http ://www.perl.org/
16. Le laboratoire COGIT (Conception Objet et Généralisation de l'Information Topographique), Institut Géographique National, Saint-Mandé

d'évaluer la qualité de l'alignement généré. D'autres expérimentations ont ensuite été faites sur des ontologies du domaine de l'anatomie, afin d'évaluer les performances (temps d'exécution) de l'outil TaxoMap. Ces ontologies avaient été proposées dans le test "Anatomy" de la campagne d'évaluation des outils d'alignement OAEI [17] (2005-2011).

6.2.1 Tests portant sur la qualité de l'alignement produit

Les deux ontologies mises en jeu dans cette expérimentation sont les suivantes :

L'ontologie BDTopo : cette ontologie a été construite par le COGIT à partir des spécifications textuelles associées à la base de données géographiques BDTopo. Le processus de construction a été semi-automatique. Cette ontologie est composée de 612 concepts reliés par des liens de subsomption au sein d'une hiérarchie qui compte 7 niveaux de profondeur. La hiérarchie est compacte, sa racine ne comporte que 2 fils directs de profondeur 1, eux-mêmes pères directs d'un nombre relativement limité de nœuds (3 et 17).

L'ontologie BDCarto : cette ontologie a été construite par le COGIT à partir des spécifications textuelles associées à la base de données géographiques BDCarto. Le processus de construction a été semi-automatique. Cette ontologie comprend 504 concepts dans une hiérarchie de profondeur 4, beaucoup plus dispersée : la racine est reliée à 65 fils directs dont 30 sont des concepts isolés ou ne sont reliés qu'à un unique autre concept.

Nous avons effectué des tests avec la version V_1 de TaxoMap. Les résultats sont présentés dans le tableau 6.1 :

Techniques appliquées	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	Total	Reste
Mappings identifiés	197	70	22	3	6	8	17	19	342	162
Mappings invalides	1	14	4	0	1	3	1	12	36	

TABLE 6.1 – Nombre de mappings générés par technique par la version V_1 de TaxoMap

342 correspondances ont été identifiées dont 197 relations d'équivalence, 114 relations de spécialisation, 22 relations de généralisation et 9 relations de proximité. 162 concepts de BDCarto (sur 504) restent à apparier. Ces mappings ont été validés manuellement par le COGIT, ce qui a permis de déceler un certain nombre de mappings générés invalides. Les mappings invalides ont été analysés. Ils sont à la base des adaptations réalisées dans la version V_2 décrites dans le chapitre 2.

– La meilleure prise en compte du statut des mots d'un label (mots *pleins* ou *complémentaires*) dans la mesure de similarité a permis des améliorations pour les cas suivants :

17. `http://oaei.ontologymatching.org/`

1. Quand au moins un *mot plein* est commun aux deux labels comparés, l'évaluation des fractions de *mots complémentaires* communes a permis de générer de nouveaux mappings corrects non retrouvés sinon. Par exemple, dans la comparaison de **hôtel département** et de **hôtel départemental**, **hôtel** est un mot plein commun, l'évaluation de **département** et de **départemental** permet de générer le mapping :

   `hôtel de département` *isClose* `hôtel départemental`

2. Nous avons, par ailleurs, évité des mappings incorrects en exigeant que la racine des *mots pleins* soit commune. Voici, à titre d'exemple, quelques mappings incorrects évités :

   `prairie` *isA* `bâtiment administratif`
   (car précédemment rapproché de `mairie`, lui-même fils de `bâtiment administratif`)

   `gave` *isA* `grotte`
   (car précédemment rapproché de `aven`, lui-même fils de `grotte`)

– Concernant les techniques elles-mêmes :

1. Le réglage du seuil de la technique d'inclusion relâchée (t_4) a permis d'obtenir 13 nouveaux mappings, comme :

   `crémaillère` *isClose* `voie ferrée à crémaillère (sim 0.204)`
   `douane` *isClose* `poste de douane (sim 0.191)`

   En effet, quand le label inclus est un mot complémentaire dans c_{tmax}, la similarité est très faible. La version V_2 de TaxoMap génère quand même un mapping quand cette similarité est supérieure à 0.15 et qu'aucun autre candidat au mapping n'a une similarité supérieure.

2. La technique t_9 qui s'appuie sur les alignements sûrs déjà trouvés de la forme (A *isEq* B) pour déduire que les spécialisations de A sont aussi des spécialisations de B a permis d'obtenir 63 nouveaux mappings.

   Cette technique a permis, par exemple, de déduire, à partir du mapping sûr **Carrefour** *isEq* **carrefour**, les mappings suivants :

   `toboggan` *isA* `carrefour`
   `intersection` *isA* `carrefour`
   `tourne à gauche` *isA* `carrefour`

3. Enfin, la technique structurelle t_{10} qui essaye de prendre en compte la structure des deux ontologies en s'appuyant sur la notion de sémantique intentionnelle d'un concept, a permis d'identifier 14 nouveaux mappings.

Les résultats de l'alignement des deux ontologies BDCarto et BDTopo générés avec la version V_2 de TaxoMap sont présentés dans le tableau 6.2.

Techniques appliquées	T_1	T_2	T_3	T_4	T_5	T_6	T_9	T_7	T_8	T_{10}	Total	Reste
Mappings identifiés	197	87	13	13	5	4	63	0	10	14	406	98
Mappings invalides	1	13	5	0	0	0	3	0	6	2	30	

TABLE 6.2 – Nombre de mappings générés par la version V_2 TaxoMap

Nous avons calculé l'une des deux mesures classiquement utilisées pour comparer l'efficacité des différentes adaptations du processus d'alignement de TaxoMap, la *précision* (le nombre de mappings corrects identifiés par rapport au nombre total de mappings trouvés). Nous n'avons pas calculé le *rappel* (le nombre de mappings corrects identifiés par rapport au nombre total de mappings de référence) car nous ne disposons pas d'alignement de référence, mais nous présentons le nombre de mappings identifiés, qui, analysé en regard de la précision, est aussi une mesure intéressante. Les résultats sont présentés dans le tableau 6.3. Ils montrent l'intérêt des adaptations réalisées dans la version V_2 du logiciel.

Versions	Nb de Mappings trouvés	Précision
Version V_1	342	0.894
Version V_2	406	0.926

TABLE 6.3 – Nombre de Mappings trouvés et Précision par version

En comparant les deux tableaux, nous remarquons qu'avec la version V_2 par rapport à la version V_1 de TaxoMap, le nombre de mappings identifiés augmente de $18,7\%$ et la précision des résultats de $3,5\%$. Ceci montre que les adaptations portant sur la mesure de similarité, la normalisation des concepts, et l'utilisation de nouvelles techniques d'alignement améliorent bien les résultats.

6.2.2 Tests portant sur le temps d'exécution et la modularité des techniques

Nous avons testé TaxoMap sur les ontologies NCI et Mouse issues du domaine de l'anatomie. Cette paire d'ontologies a été utilisée comme test dans la campagne d'évaluation OAEI (2005-2011).

Les ontologies du test Anatomy sont (1) le thésaurus NCI décrivant l'anatomie humaine, publié par NCI [18] (the National Cancer Institute), (2) le dictionnaire de l'anatomie de la souris adulte [19], développé dans le cadre du projet de base de données d'expression du gène de souris. Les deux ressources font partie de OBO (the Open Biomedical Ontologies).

Le test OAEI consiste à aligner le dictionnaire de l'anatomie de la souris adulte (désignée par "Mouse") et le thésaurus NCI décrivant l'anatomie humaine (désigné par "Human"). Pour ce test, les organisateurs ont élaboré un alignement de référence, non accessible aux participants,

18. `http://www.cancer.gov/cancerinfo/terminologyresources/`
19. `http://www.informatics.jax.org/searches/AMA_form.shtml`

et quatre tâches sont proposées :

- tâche 1 : l'alignement généré doit être une solution optimale par rapport au rappel et à la précision.
- tâche 2 : l'alignement généré doit optimiser la précision.
- tâche 3 : l'alignement généré doit optimiser le rappel.
- tâche 4 : utiliser une fraction d'un alignement de référence pour générer les mappings.

L'ontologie "Mouse" contient 2 744 concepts, tandis que l'ontologie "Human" contient 3 304 concepts. Dans nos expérimentations, nous avons considéré l'ontologie "Human" comme ontologie cible car elle est bien structurée et contient un plus grand nombre de concepts que "Mouse".

La précision, le rappel et la F-mesure (une combinaison pondérée du rappel et de la précision), ainsi que le nombre de mappings proposés, sont présentés dans le tableau 6.4, selon les différentes versions de TaxoMap utilisées dans les campagnes d'évaluation de 2007, 2008, 2009 et 2010 pour le tâche 1 du test Anatomy.

Versions	Année	Temps d'exécution	Nb de mappings soumis	Précision	Rappel	F-mesure
Version V_0	2007	5 h	1 820	0.596	0.732	0.657
Version V_1	2008	25 min	2 533	0.460	0.764	0.574
Version V_2	2009	8 min	1 274	**0.870**	**0.678**	**0.762**
Version V_2	2010	12 min	1 223	**0.924**	**0.743**	**0.824**

TABLE 6.4 – Précision, rappel et F-mesure des résultats générés par les différentes versions de TaxoMap pour la tâche 1 du test Anatomy de OAEI.

Avec la version V_0 de TaxoMap, l'alignement des deux ontologies nécessitait 5 heures de temps d'exécution, et la qualité des résultats obtenus était très moyenne (F-mesure 65%). Ceci est dû au fait que TaxoMap utilisait un processus très compliqué et très coûteux pour normaliser les concepts, qui consistait à vérifier l'existence des mots constituant les labels dans WordNet. Avec l'intégration de TreeTagger en 2008, le temps d'exécution est devenu plus raisonnable. Il est passé à 25 minutes au lieu de 5 heures, mais la qualité des résultats, hormi le rappel, ne s'est pas améliorée : cela tient au fait que toutes les techniques développées à l'époque devaient être systématiquement enchaînées alors que certaines se sont avérées peu pertinentes ou fiables dans le contexte.

En 2009 et 2010, les adaptations concernant le calcul de similarité, la normalisation des concepts, l'introduction de nouvelles techniques et la possiblilité de choisir les techniques utilisées lors d'une session ainsi que leur ordre d'application, ont permis une amélioration considérable de la qualité de l'alignement produit.

Ainsi, l'alignement de référence utilisé pour le test Anatomy ne prenant en compte que les relations d'équivalence, nous avons cessé d'utiliser pour les sessions d'OAEI, les deux principales techniques basées sur les inclusions de labels et menant à des relations de subsomption. Le nombre de mappings soumis a considérablement diminué, le rappel faiblement, mais la précision

a sérieusement augmenté. En 2009, avec la version V_2 de TaxoMap, on obtient une augmentation de la F-mesure de $32,7\%$ par rapport à l'année 2008. En 2010 [Hamdi et al., 2010e], la prise en compte d'une phase de raffinement de mapping permet encore d'améliorer les résultats.

Le temps d'exécution a aussi beaucoup décru (12 minutes au lieu de 25 minutes) grâce à l'utilisation d'une base de données relationnelle qui a permis aux techniques d'alignement d'accéder plus facilement et plus rapidement aux données utiles. Enfin, la possibilité d'ajuster les paramètres d'alignement via une interface a également été un facteur d'amélioration des résultats qu'il ne faut pas négliger.

6.3 Expérimentations illustrant la spécification de traitements de raffinement de mappings

Nous avons réalisé des expérimentations utilisant l'environnement TaxoMap Framework pour le raffinement de mappings. Nous avons, tout d'abord, réalisé des expérimentations montrant comment la spécificité des ontologies alignées, notamment l'écriture des labels des concepts, pouvait être prise en compte. Ces tests ont été réalisés sur les ontologies géographiques BD-Carto et BD-Topo. Nous avons ensuite effectué des expérimentations montrant la prise en considération du type d'alignement souhaité. Cette deuxième série de tests a porté sur les ontologies du test "Anatomy" de la campagne OAEI, ce qui permet également d'illustrer l'applicabilité de notre approche sur un domaine autre que la géographie. Les ontologies utilisées ont été présentées dans la partie précédente, nous ne les re-décrivons pas.

6.3.1 Expérimentation illustrant la spécification de traitements de raffinement de mappings adaptés aux ontologies alignées

Cette expérimentation illustre le workflow de raffinement de mappings présenté dans le chapitre 3 de cette thèse. Elle montre comment l'expert/ingénieur peut interagir avec l'environnement TaxoMap Framework pour améliorer les résultats d'un alignement, ces raffinements étant basés sur les spécificités des ontologies alignées, plus particulièrement la façon dont les labels des concepts sont construits.

L'expérimentation que nous détaillons dans cette partie correspond à celle menant au patron-2 du chapitre 3 (après application du patron-1), où l'expert évalue les mappings produits par la technique t_2 (cf. Tableau 6.5).

Cette technique génère un mapping de type "isA" entre c_s et c_{tmax} si : (1) le concept c_{tmax} est le concept de O_T qui a la plus forte similarité avec le concept c_s de O_S, (2) un des labels de c_{tmax} est inclus dans un des labels de c_s, (3) tous les mots du labels inclus sont des mots pleins dans les deux labels.

Le prédicat qui correspond à la technique t_2 est *isAStrictInclusion*. Les trois itérations nécessaires pour spécifier le bon patron sont décrites ci-dessous. Rappelons que pour faciliter la tâche de validation, les mappings produits au départ par TaxoMap sont présentés à l'expert, par paquet, en fonction de la technique utilisée pour les obtenir.

concept source	**relation**	**concept cible**
aérodrome privé	*isA*	aérodrome
aire de repos et aire de service	*isA*	aire de service
autoroute ou voie route à chaussées séparées et carrefours dénivelés	*isA*	route à chaussées séparées et carrefours dénivelés
barrage au fil de l'eau	*isA*	barrage
barrage de retenue	*isA*	barrage
bois et forêt	*isA*	forêt
bretelle d'accès	*isA*	bretelle
bretelle d'accès aux aires de repos et aires de service	*isA*	bretelle
bretelle d'échangeur	*isA*	bretelle
campanile et beffroi non attenant	*isA*	beffroi
canal d'alimentation ou de restitution	*isA*	canal
canal de décharge	*isA*	canal
canal de dérivation	*isA*	canal
canal de navigation	*isA*	canal
canal d'irrigation	*isA*	canal
canal navigable	*isA*	canal
carrefour dénivelé	*isA*	carrefour
chemin de halage	*isA*	chemin
col routier	*isA*	col
cours d'eau dans la zone d'estran	*isA*	cours d'eau
cours d'eau eau salée permanent	*isA*	cours d'eau
cours d'eau naturel	*isA*	cours d'eau
cours d'eau principal	*isA*	cours d'eau
digue de barrage	*isA*	digue
édifice religieux chrétien	*isA*	édifice religieux
embouchure et perte de cours d'eau	*isA*	embouchure
embouchure logique	*isA*	embouchure
escarpement rocheux	*isA*	escarpement
étang périodique	*isA*	étang
forêt domaniale	*isA*	forêt
gare de péage	*isA*	gare
hôpital local	*isA*	hôpital
hôpital militaire	*isA*	hôpital
hôtel situé en montagne	*isA*	hôtel
île émergée	*isA*	île
lac d'altitude	*isA*	lac
ligne de transport de l'énergie électrique	*isA*	ligne électrique
mairie d'arrondissement	*isA*	mairie
métro aérien	*isA*	métro
mine à ciel ouvert	*isA*	mine
musée classé et contrôlé par l'état	*isA*	musée
musée de ministère	*isA*	musée

concept source	**relation**	**concept cible**
musée privé ou local	*isA*	musée
nécropole nationale	*isA*	nécropole
parc aquatique	*isA*	parc
parc de voitures	*isA*	parc
piste principale d'aérodrome	*isA*	piste d'aérodrome
plaine et cuvette	*isA*	cuvette
pointe promontoire	*isA*	promontoire
port de plaisance	*isA*	port
port de plaisance maritime	*isA*	port
port maritime	*isA*	port
refuge gardé	*isA*	refuge
réservoir et château d'eau	*isA*	réservoir
retenue sur cours d'eau	*isA*	retenue d'eau
rocher d'escalade	*isA*	rocher
rochers et sable	*isA*	rocher
route appartenant au réseau vert	*isA*	route
route départementale	*isA*	route
route départementale et nationale	*isA*	route
route en construction	*isA*	route
route nationale	*isA*	route
route nationale importante	*isA*	route
route vicinale	*isA*	route
rue ou route	*isA*	route
ruine d'époque médiévale ou postérieure	*isA*	ruine
sentier côtier	*isA*	sentier
sentier de montagne	*isA*	sentier
sommet important	*isA*	sommet
source d'intérêt touristique	*isA*	source
stade d'importance nationale	*isA*	stade
station de pompage et de traitement des eaux	*isA*	station de pompage
table d'orientation accessible au public située sur le territoire national	*isA*	table d'orientation
tour de télécommunication	*isA*	tour
tunnel routier	*isA*	tunnel
voie ferrée de transit	*isA*	voie ferrée
voie ferrée en construction ou en projet	*isA*	voie ferrée
voie ferrée urbaine	*isA*	voie ferrée
voie industrielle ou de service	*isA*	voie de service
voie route à chaussées séparées et carrefours dénivelés	*isA*	route à chaussées séparées et carrefours dénivelés
zone de sable humide	*isA*	zone humide

TABLE 6.5: Les mappings obtenus avec la technique $T2$

Itération 1 :

Dans un premier temps, l'évaluation des mappings produits par la technique t_2 a conduit l'expert à identifier 3 mappings comme exemple de mappings à raffiner :

"Plaine et Cuvette *isA* Cuvette" doit devenir "Plaine et Cuvette *isMoreGnl* Cuvette".

"Bois et forêt *isA* Forêt" doit devenir "Bois et forêt *isMoreGnl* Forêt".

"Rue ou route *isA* Route" doit devenir "Rue ou route *isMoreGnl* Route".

Le raffinement souhaité par l'expert, illustré par ces 3 exemples, a été généralisé par l'ingénieur de la façon suivante : dans le contexte de cette technique d'appariement, quand le label du concept c_s de la source contient un connecteur "et/ou", l'expert souhaite que c_s ne soit pas considéré comme une spécialisation de c_{tmax} mais plutôt comme une généralisation de celui-ci (cf Fig. 6.8).

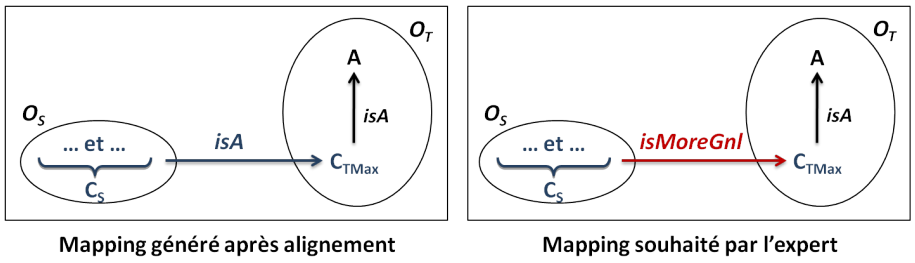

FIG. 6.8 – Illustration du raffinement souhaité

Ceci peut s'écrire ainsi :

Partie contexte :
$\exists x \exists y\ (isAStrictInclusion(x, y) \wedge appearInLabel(\text{"}et\text{"}, x)$
$\wedge\ \exists z\ (isSubClassOf(y, z, O_T) \wedge \neg strictInclusionLabel(z, x)))$
Partie solution :
$Delete\_Mapping(x, y, \_) \wedge Add\_Mapping(x, y, isMoreGnl)$

L'application de ce patron sur l'ensemble de la base de données des mappings a conduit à la modification de 20 mappings, dont les 3 proposés en exemple par l'expert et 17 autres mappings instanciant le patron. Les résultats sont présentés dans le tableau 6.6.

concept source	**relation**	**concept cible**
aire de repos et aire de service	*isMoreGnl*	aire de service
autoroute ou voie route à chaussées séparées et carrefours dénivelés	*isMoreGnl*	route à chaussées séparées et carrefours dénivelés
autoroute ou voie route à chaussées séparées et carrefours dénivelés	*isMoreGnl*	route à chaussées séparées et carrefours dénivelés
bois et forêt	*isMoreGnl*	***forêt***
bretelle d'accès aux aires de repos et aires de service	*isMoreGnl*	bretelle
campanile et beffroi non attenant	*isMoreGnl*	beffroi
canal d'alimentation ou de restitution	*isMoreGnl*	canal
embouchure et perte de cours d'eau	*isMoreGnl*	embouchure
musée classé et contrôlé par l'état	*isMoreGnl*	musée
musée privé ou local	*isMoreGnl*	musée
plaine et cuvette	*isMoreGnl*	***cuvette***
réservoir et château d'eau	*isMoreGnl*	réservoir
rochers et sable	*isMoreGnl*	rocher
route départementale et nationale	*isMoreGnl*	route
rue ou route	*isMoreGnl*	***route***
ruine d'époque médiévale ou postérieure	*isMoreGnl*	ruine
station de pompage et de traitement des eaux	*isMoreGnl*	station de pompage
voie ferrée en construction ou en projet	*isMoreGnl*	voie ferrée
voie industrielle ou de service	*isMoreGnl*	voie de service
voie route à chaussées séparées et carrefours dénivelés	*isMoreGnl*	route à chaussées séparées et carrefours dénivelés

TABLE 6.6 – Les mappings obtenus après application de la première formulation du patron

Itération 2 :

L'analyse des 17 mappings modifiés, alors qu'ils n'avaient pas été cités en exemple par l'expert, conduit à un nouveau cycle de raffinement de mappings.

Pour 5 mappings supplémentaires, les modifications respectent bien les souhaits de l'expert en identifiant dans le label de c_s une conjonction plus générale que le concept inclus c_{tmax} comme "rochers et sable *isMoreGnl* rocher".

En revanche, cette exécution révèle aussi des modifications a priori non souhaitables, en particulier quand une partie du label de c_s qui contient le label de c_{tmax} intervient sous une forme spécialisée de celui-ci. Ainsi, dans le label "musée privé ou local", "musée" (c_{tmax}) apparaît suivi du qualificatif "privé" dans l'expression "musée privé", ce qui correspond à une dénomination plus précise que "musée". Il n'apparaît donc pas souhaitable de considérer dans ce cas que c_s est un généralisant de c_{tmax}.

Il s'avère donc que la seule présence d'un connecteur "*et/ou*" ne suffit pas à garantir une

conjonction plus générale que le concept inclus dans la conjonction et qu'il faut de plus vérifier que le connecteur sépare effectivement le label exact de c_{tmax} et autre chose (que nous nommerons "*remaining part*"), sous la forme "P_1 *et/ou* P_2" où c_{tmax} est exactement soit P_1 soit P_2.

Ceci conduit l'ingénieur à modifier le patron précédent en utilisant au lieu du prédicat $appearInLabel("et", x)$", la formule $inclusionInLabel(x, c, y)$, qui permet de vérifier si une des deux parties du label de x connectées par le connecteur c est exactement le label de y.

Ainsi, *InclusionInLabel("station de pompage et de traitement des eaux", et, "station de pompage")* est vrai, alors que *InclusionInLabel ("musée privé ou local", ou, "musée")* et *InclusionInLabel ("campanile et beffroi non attenant", et, " beffroi")* sont faux.

L'écriture du patron devient :

Partie contexte :
$\exists x \exists y\ (isAStrictInclusion(x, y) \wedge inclusionInLabel(x, "et", y)$
$\wedge\ \exists z\ (isSubClassOf(y, z, O_T) \wedge \neg strictInclusionLabel(z, x)))$
Partie solution :
$Delete\_Mapping(x, y, \_) \wedge Add\_Mapping(x, y, isMoreGnl)$

L'application de cette deuxième formulation du patron sur l'ensemble initial des mappings conduit à la modification de 8 mappings, dont les 3 proposés en exemple par l'expert plus 5 autres seulement parmi les 17 mappings modifiés avec la première formulation. Les résultats sont présentés dans le tableau 6.7.

concept source	**relation**	**concept cible**
aire de repos et aire de service	*isMoreGnl*	aire de service
bois et forêt	*isMoreGnl*	***forêt***
embouchure et perte de cours d'eau	*isMoreGnl*	embouchure
plaine et cuvette	*isMoreGnl*	***cuvette***
réservoir et château d'eau	*isMoreGnl*	réservoir
rochers et sable	*isMoreGnl*	rocher
rue ou route	*isMoreGnl*	***route***
station de pompage et de traitement des eaux	*isMoreGnl*	station de pompage

TABLE 6.7 – Les mappings obtenus après application de la deuxième formulation du patron-2

Itération 3 :

L'analyse des nouveaux résultats générés est effectuée. L'expert valide les modifications des 5 mappings supplémentaires aux exemples qu'il avait identifiés, ainsi que la préservation de 10 des 12 mappings présentés en contre-exemples.

En revanche 2 mappings ne sont pas retenus par la nouvelle formulation du patron alors que l'expert aurait voulu les voir modifier. Il s'agit des mappings suivants :

"campanile et beffroi non attenant isA beffroi"

"autoroute ou voie route à chaussée séparée isA route à chaussée séparée".

L'analyse de ces 2 contre-exemples montre que dans les deux cas, le label de c_s est sous la forme "P_1 *et/ou* P_2" avec le label de c_{tmax} inclus dans P_2 sans être exactement équivalent. Toutefois, la chaîne de caractères P_1, que nous appelons "*remaining part*", est le label d'un concept du domaine ("Campanile" dans le premier exemple, et "autoroute" dans le second).

L'identification du concept serait simple à réaliser automatiquement dans le second exemple, car "autoroute" est un label d'un concept dans O_T. Il est plus difficile dans le premier exemple, puisque "Campanile" n'est pas le label d'un concept de O_T ni de O_S. Ainsi seulement un des deux nouveaux changements désirés peut être exécuté automatiquement par l'introduction d'un patron supplémentaire (ce nouveau patron n'a pas été implémenté dans le cadre de cette thèse).

Le patron précédemment défini n'a plus à être modifié. L'expert a validé les résultats générés. Le nouveau patron répond à un nouveau cas identifié par les experts lors de l'itération 3. Notons que les résultats ne dépendent pas de l'ordre d'application de ces deux patrons (le patron défini précédemment et le nouveau patron).

Le tableau 6.8 présente le résultat final obtenu après analyse des mappings générés par toutes les techniques de TaxoMap appliquées (T_2, T_3, T_8 et T_{10}) sur BD-Topo et BD-Carto. L'analyse de ces mappings a entraîné la définition de 5 patrons décrits dans le chapitre 3 (patron-1, patron-2, patron-2', patron-3, patron-4 et patron-5).

Technique	T_2	T_3	T_8	T_{10}	Total
mappings	87	13	8	14	122
mappings faux	13	5	6	2	26
mappings raffinés	14	1	5	2	22
mappings faux après raffinement	6	4	2	0	12

TABLE 6.8 – Le nombre initial de mappings générés, faux et raffinés par technique

Cette expérience dans le domaine de la topographie a conduit in fine à spécifier 6 patrons de raffinement liées à 4 techniques d'alignement. 22 mappings ont été modifiés. 20 satisfont les souhaits de l'expert. Deux améliorations sont incorrectes.

Cette expérimentation montre que le workflow de raffinement de mappings permet, grâce au processus itératif de validation/correction, à un ingénieur/expert du domaine de spécifier correctement un traitement destiné à être appliqué à un ensemble donné de mappings et d'aboutir via différentes itérations à une amélioration de la qualité d'un alignement. La précision des résultats a été augmentée de $11,47\%$.

6.3.2 Expérimentation illustant la spécification de traitements de raffinement de mappings adaptés au type d'alignement souhaité

Nous avons testé l'environnement TaxoMap Framework et son application pour le raffinement de mappings sur les alignements des ontologies NCI et Mouse du test "Anatomy" (tâche 1, tâche 2, tâche 3) de la campagne d'évaluation OAEI 2010 décrite précédemment. L'expérimentation réalisée montre que l'environnement TaxoMap Framework permet d'adapter les résultats obtenus au type d'alignement recherché, plus précisément il permet de sélectionner les types de mappings recherchés parmi tout un ensemble de mappings générés.

L'alignement de référence du test Anatomy ne contient que des mappings d'équivalence et ce, de type 1:1, alors que TaxoMap génère des mappings d'équivalence et de subsomption de type 1:n, puisque plusieurs concepts de l'ontologie source peuvent être liés à un même concept de l'ontologie cible, l'un par une relation d'équivalence, s'il existe, et les autres par des relations de subsomption. Pour les tests d'Anatomy nous avions minimisé l'usage des techniques menant à des subsomptions, et traduit les mappings de subsomption restants en mapping d'équivalence : nous pouvions donc produire, parmi les mappings soumis, plusieurs mappings d'équivalence relatifs à un même concept de la cible.

Pour mieux se plier aux contraintes de l'alignement de référence, TaxoMap Framework a alors été utilisé pour spécifier la suppression des mappings correspondant à des relations de subsomption, lorsque ces mappings portaient sur un concept de l'ontologie cible déjà relié à un autre concept de l'ontologie source par une relation d'équivalence. Ainsi, à la sortie de l'exécution du patron de raffinement correspondant, TaxoMap génère beaucoup moins de mappings de type 1:n.

Ceci nous a conduit à trouver moins de correspondances qu'en 2009 (cf. Tableau 6.4) mais la précision des résultats est meilleure.

La description du patron qui a été appliqué est la suivante :

Patron-6 : Ce patron concerne les mappings générés par la technique t_1, reliant par une relation d'équivalence un concept y de l'ontologie cible O_T avec un concept x de l'ontologie source O_S. Les mappings impliquant y obtenus avec d'autres techniques que t_1 sont supprimés (cf. Fig. 6.9).

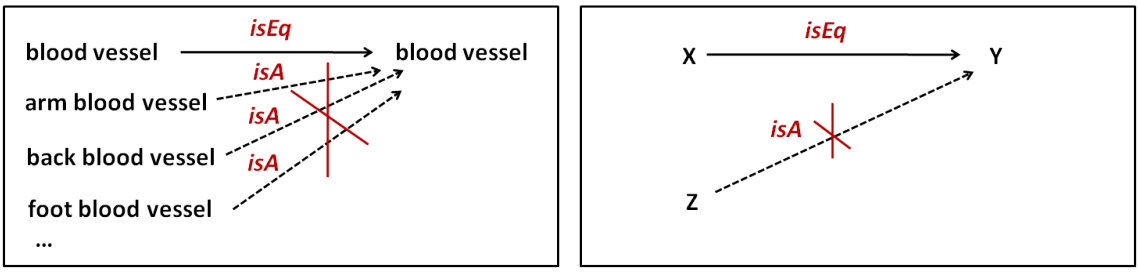

FIG. 6.9 – Illustration du Patron-6

Partie contexte du Patron-6 :
$\exists x \exists y$ $(isEquivalent(x, y)$
$\wedge \exists z$ $(mapping(z, y) \wedge differentConcept(z, x))$
Partie solution du Patron-6 :

$Delete\_Mapping(z, y, \_)$

	Tâche 1	Tâche 2	Tâche 3
Mappings générés	1 449	1 127	2 357
Précision sans raffinement	0.779	0.929	0.477
Mappings supprimés	226	32	945
Mappings soumis	1 223	1 095	1 412
Précision	0.924	0.956	0.838
Δ Précision avec raffinement	+ 0.145	+ 0.027	+ 0.361
Rappel	0.743	0.689	0.774
F-mesure	0.824	0.801	0.802

TABLE 6.9 – Résultats de TaxoMap pour les différents tâches du test Anatomy

Le tableau 6.9 montre que l'application du patron-6 permet d'améliorer la précision des mappings générés par TaxoMap quelle que soit la tâche. Pour la tâche 1, 226 faux mappings sont supprimés, 32 pour la tâche 2 et 945 pour la tâche 3.

Nous avons montré, au travers de ces deux expérimentations, comment TaxoMap Framework permet à un utilisateur (ingénieur/expert) de spécifier des traitements de raffinement de mappings, que ceux-ci soient dictés par des spécificités des ontologies alignées (par exemple, la façon dont les labels sont construits) ou par les types de mappings souhaités. Il s'agit de deux cas possibles de raffinement où TaxoMap Framework s'est révélé être très utile. Son utilisation dans d'autres contextes applicatifs devrait permettre, à l'avenir, de mettre en évidence d'autres scénarios intéressants d'utilisation.

6.4 Tests des méthodes de partitionnement pour l'alignement d'ontologies

Afin de comparer les méthodes de partitionnement présentées dans le chapitre 4 de cette thèse, et leur efficacité pour l'alignement, nous avons réalisé des expérimentations sur différentes ontologies. Les blocs constitués ont été alignés par paires à l'aide de TaxoMap.

Les expérimentations ont tout d'abord été réalisées sur les ontologies du projet ANR GeOnto, décrites précédemment. Ces ontologies sont de taille limitée, ce qui permet de les aligner directement avec TaxoMap, et nous les connaissons bien. Ceci nous a permis d'analyser la pertinence sémantique des blocs générés et d'utiliser les résultats des alignements directs (sans partitions) comme référence pour les résultats obtenus après le partitionnement. D'autres expérimentations ont ensuite été faites sur deux paires d'ontologies de grande taille, sur lesquelles notre outil d'alignement ne peut produire de résultats sans l'utilisation au préalable d'un outil de partitionnement.

6.4.1 Tests sur des ontologies de petite taille

Les résultats de l'alignement direct (sans partitions) effectué par TaxoMap et qui vont nous servir de référence pour les autres alignements sont présentés dans le tableau 6.10 ci-dessous. Ils décomptent les appariements obtenus suivant le type de relations établies entre les concepts : relation d'équivalence (isEq), de subsomption (isA) ou de proximité (isClose).

Ontologies	Taille Cible	Taille Source	isEq	isClose	isA	$\Sigma =$
BDTopo - BDCarto	612	505	197	13	95	305

TABLE 6.10 – Les relations identifiées par l'alignement de BDCarto vers BDTopo

Pour effectuer les partitions, nous avons fixé à 100 concepts la taille maximale des blocs fusionnables, ce qui veut dire qu'un bloc dépassant cette taille ne peut plus être fusionné. Les blocs générés contiennent ainsi au plus 200 concepts chacun. Le tableau 6.11 ci-dessous donne le nombre de blocs générés à partir des deux ontologies par les méthodes PBM, PAP et APP. Remarquons que le nombre d'ancres identifiées, 191, est plus faible que le nombre de concepts jugés équivalents dans l'alignement de référence, 197. En effet, la mesure de similarité utilisée pour calculer les ancres est plus stricte (mais plus rapide à calculer), que celle de TaxoMap.

		L'ontologie cible BDTopo			L'ontologie source BDCarto		
Méthodes	Ancres	Blocs générés	Concepts isolés	Plus grand bloc	Blocs générés	Concepts isolés	Plus grand bloc
PBM	191	5	0	151	25	22	105
PAP	191	5	0	151	10	16	143
APP	191	6	0	123	10	16	153

TABLE 6.11 – Partitionnement de BDTopo et BDCarto par méthode

L'ontologie cible BDTopo est l'ontologie de référence du COGIT, elle est donc bien construite, compacte et très structurée. La racine ne comporte que deux fils directs de profondeur 1, eux-mêmes pères directs d'un nombre limité de nœuds. L'ontologie cible est ainsi facile à partitionner en blocs sémantiquement pertinents que ce soit par la méthode PBM, méthode reprise dans la méthode PAP pour la décomposition de la cible et qui s'appuie essentiellement sur les relations structurelles entre concepts, ou par la méthode APP. Les deux décompositions proposées, composées respectivement de 5 et 6 blocs, ont toutes les deux leur cohérence.

A l'inverse, l'ontologie source BDCarto est moins structurée, très dispersée. La racine est reliée à presque une trentaine de fils directs, et de très nombreux sous-arbres ne contiennent pas plus d'une dizaine d'éléments. La décomposition de la source, plus dispersée, est plus délicate. PBM génère un nombre important de petits blocs ne comprenant pas plus de 5 ou 6 concepts, 19 blocs ne contiennent pas d'ancres, et 22 blocs contiennent seulement un concept isolé. En utilisant l'information sur les ancres partagées, nos méthodes permettent en revanche d'agréger dans des blocs plus importants plus de la moitié de ces petits blocs ainsi que de nombreux concepts isolés, tout en gardant la cohérence sémantique de ces derniers. La partition construite, moins dispersée, est plus lisible pour l'humain et plus efficace pour la phase suivante d'alignement des

blocs.

Le choix des paires de blocs à aligner diffère suivant les méthodes :

- **PBM** : parmi les 25 blocs générés, seulement 6 blocs de la source contiennent des ancres. Ces 6 blocs sont alignés avec les blocs de la cible pour lesquels le ratio d'ancres partagées sur la somme des ancres présentes dans les deux blocs est supérieur à un seuil donné ϵ, ici fixé à 0.1. Ce seuil est atteint par 9 paires de blocs, 9 alignements sont alors effectués.

- **PAP** : les 5 blocs de la source, construits à partir des 5 blocs de la cible qui contiennent des ancres, conduisent en tout à 5 alignements.

- **APP** : les 6 paires sélectionnées sont celles qui maximisent le nombre d'ancres partagées des 6 blocs de la source contenant des ancres et qui chacun ne participent qu'à un seul alignement.

Le tableau 6.12 montre le nombre de mappings que nous obtenons en alignant les différentes paires de blocs choisis par TaxoMap. Les résultats présentés montrent que même en appariant moins de paires de blocs que dans la méthode PBM, l'appariement des blocs générés par nos méthodes donne de meilleurs résultats en nombre de mappings identifiés.

Méthodes	Paires alignées	isEq	isClose	isA	Σ	Précision	Rappel
PBM	9	118	13	52	183	0.96	0.57
PAP	5	192	10	55	257	0.97	0.81
APP	6	147	11	61	219	0.97	0.69

TABLE 6.12 – Les relations identifiées par l'alignement des blocs générés par les différentes méthodes

Si nous analysons les résultats[20] des deux mesures pour comparer l'efficacité des méthodes, la *précision* et le *rappel*, nous voyons que nos méthodes ont un bien meilleur rappel. En effet, nos méthodes prennent en compte les relations d'équivalence entre les labels dans le processus de partitionnement. Ainsi, les concepts qui ont des relations entre eux sont regroupés dans des blocs qui seront par la suite considérés comme des paires à aligner, alors que la méthode PBM partitionne les ontologies indépendamment les unes des autres et effectue seulement un alignement a posteriori. La méthode PAP permet en particulier, par construction, de trouver tous les mappings correspondant aux ancres. Elle a ainsi un rappel supérieur.

Le fait que les différentes méthodes aient une précision inférieure à 1, signifie qu'elles trouvent toutes les trois des appariements ne faisant pas partie des mappings de référence. Bien que comptabilisés ici comme incorrects, ces mappings ne sont pas forcément faux. En effet, TaxoMap ne produit pour chaque concept de la source, qu'un unique appariement, celui qu'il juge le meilleur, même si plusieurs concepts de la cible pouvaient en être rapprochés. Si les deux concepts intervenant dans un appariement de référence ne sont plus comparés entre eux parce qu'ils sont répartis

20. Ces résultats ont été calculés automatiquement par l'API d'évaluation d'alignements disponible sur le Web, `http://oaei.ontologymatching.org/2008/align.html`, en fournissant en référence le fichier généré par l'alignement direct, par TaxoMap, sans partition.

dans des blocs non alignés, un autre appariement, qui ne sera pas forcément inintéressant, peut être trouvé pour le concept source.

Cette expérimentation montre que nos deux méthodes de partitionnement, PAP et APP, produisent des partitions moins dispersées que PBM en limitant le nombre de blocs générés et les concepts isolés. Elle montre également que nos deux méthodes trouvent davantage de mappings.

6.4.2 Tests sur des ontologies de grande taille

Nous avons testé nos deux méthodes de partitionnement sur deux paires d'ontologies volumineuses, Library et FAO. Ces ontologies ont été utilisées dans un test des campagnes d'évaluation OAEI 2007 et 2008. Ces tests on été réalisés pour montrer que nos deux méthodes de partitionnement, (1) permettent à TaxoMap de générer des mappings dans le cas des ontologies de très grande taille (le test Library), et (2) génèrent des blocs moins dispersés en limitant le nombre de concepts isolés (le test FAO).

Pour les deux tests (Library et FAO), la comparaison entre nos méthodes et la méthode PBM est complexe parce que le système FALCON a participé à la campagne OAEI en 2007 mais pas en 2008 et que, de notre côté, nous avons participé au test FAO de la campagne OAEI en 2008 mais pas en 2007. Par ailleurs, comme la paire d'ontologies FAO n'était pas un cas de test fourni lors de la campagne OAEI en 2008, nous n'avons pas pu accéder aux mappings de référence. Malgré cela, nous présentons dans cette section, deux types d'expériences. Premièrement, nous fournissons une comparaison entre nos résultats d'alignement et ceux obtenus par les participants (ne comprenant pas FALCON) ayant fait le test Library en 2008. Deuxièmement, nous utilisons le test FAO pour comparer le nombre de blocs générés par nos méthodes et la méthode PBM.

Le test Library

L'ensemble des tests de Library est constitué de deux thésaurus, GTT et Brinkman, en néerlandais. Ces deux thésaurus sont utilisés par la "Bibliothèque nationale des Pays-Bas" pour indexer des livres de deux grandes collections. Le thésaurus GTT contient 35 194 concepts et Brinkman contient 5 221 concepts. Chaque concept a (exactement) un label préféré, mais aussi des synonymes (961 pour Brinkman et 14 607 pour GTT).

Les organisateurs du test en 2007 ont montré que les deux thésaurus ont une couverture similaire (2 895 concepts ont exactement le même label) mais diffèrent dans leur granularité. Par ailleurs, les informations structurelles des thésaurus sont très pauvres. Le thésaurus GTT (respectivement Brinkman) ne contient que 15 746 (respectivement 4 572) liens hiérarchiques. Comme la structure du thésaurus GTT est particulièrement pauvre (il contient 19 752 concepts reliés à la racine), il a été considéré comme l'ontologie source dans nos expériences.

Comme les deux ontologies sont très déséquilibrées et que le nombre d'ancres identifiées est 3 535, ce qui n'est pas beaucoup par rapport à la taille de la source, nous n'avons effectué les expérimentations qu'avec la méthode PAP. La taille maximale d'un bloc à fusionner a été fixée à 500. Les résultats sont présentés dans le tableau 6.13.

		Le thésaurus cible BRINKMAN			Le thésaurus source GTT		
Méthodes	Ancres	Blocs générés	Concepts isolés	Plus grand bloc	Blocs générés	Concepts isolés	Plus grand bloc
PAP	3 535	227	0	703	2 041	16 265	517

TABLE 6.13 – Partitionnement de BRINKMAN et de GTT

La méthode PAP retourne 227 blocs pour le thésaurus BRINKMAN, dont le plus grand contient 703 concepts, et 2 041 blocs pour le thésaurus GTT, dont le plus grand contient 517 concepts. 16 265 concepts de GTT sont restés isolés.

Le tableau 6.14 présente les résultats d'alignement des blocs des deux ontologies. Comme plus de 1 800 blocs de GTT ne contiennent pas d'ancres, nous n'avons aligné que 212 paires de blocs. Ceci nous a conduit à identifier 3 217 mappings. Seuls 1 872 d'entre eux sont des relations d'équivalence.

Méthodes	Paires alignées	isEq	isMoreGnl	isClose	isA	Σ	Précision	Rappel
PAP	212	1 872	40	274	1 031	3 217	0.88	0.41

TABLE 6.14 – Les relations identifiées par l'alignement des blocs générés par la méthode PAP

La raison pour laquelle si peu de relations d'équivalence (1 872 relations) a été retournée, par rapport au nombre d'ancres identifiées (3 535 ancres), est que les deux thésaurus contiennent un grand nombre de synonymes.

Nous avons identifié 3 535 ancres, alors que seulement 2 895 concepts étaient censés avoir le même label. Ceci signifie qu'au moins 640 ancres concernent des concepts de la source qui avaient au moins deux labels considérés comme équivalents à 2 autres labels de concepts de la cible, sans que ces deux derniers labels ne soient nécessairement associés au même concept. Le problème ici est que, si un concept de la source est ancré à 2 concepts distincts de la cible, au mieux ces deux concepts de la cible appartiennent à un même bloc, et le concept cible n'est lié par une relation d'équivalence qu'à un seul de ces concepts. Dans le pire des cas, les deux concepts de la cible appartiennent à des blocs distincts et la méthode PAP ne permettant pas de déterminer à quel bloc le concept de la source doit être lié, elle le définit comme un concept isolé.

Notons que même si plusieurs ancres ont disparu, la précision et le rappel de l'alignement des ontologies du test Library, évalués uniquement sur les relations d'équivalence par les organisateurs, et présentés dans le tableau 6.15, placent TaxoMap exécutant la méthode PAP, dans une position raisonnable. Notons qu'uniquement 3 participants (dont TaxoMap) parmi 13 ont réussi à aligner les ontologies du test Library.

Participant	Mappings d'équivalence	Précision	Rappel
DSSim	2 930	0.93	0.68
TaxoMap	1 872	0.88	0.41
Lily	2 797	0.53	0.37

TABLE 6.15 – Les résultats des systèmes participant au test Library

DSSim [Nagy et al., 2008] obtient des résultats meilleurs que TaxoMap en ce qui concerne le nombre de mappings d'équivalence identifiés (27% de plus). La précision des résultats de TaxoMap est toutefois proche de celle de DSSim. Nous n'avons pas pu expliquer la différence de rappel entre TaxoMap et DSSim, car les auteurs de ce dernier disent qu'ils partitionnent les ontologies pour les aligner, mais n'expliquent pas comment.

Lily [Wang and Xu, 2008b] a obtenu de moins bons résultats que TaxoMap en ce qui concerne le nombre de mappings corrects identifiés (35% de moins). Nous n'avons pas pu non plus expliquer cette différence car les auteurs de Lily disent qu'ils traitent les ontologies selon une méthode qui n'est pas basée sur le partitionnement, en faisant référence à un article non publié.

Parmi les trois participants, TaxoMap est le seul à identifier des mappings autres que des mappings d'équivalence. TaxoMap génère 274 mappings de généralisation (*isMoreGnl*), 1 031 de spécialisation (*isA*) et 40 mappings de proximité (*isClose*).

Le test FAO

L'ensemble des tests de OAEI 2007 comprend deux ontologies, AGROVOC et NALT, qui contiennent respectivement, 28 439 et 42 326 concepts. AGROVOC est une ontologie multilingue créée par FAO (Food and Agriculture Organization). Elle couvre les domaines de l'agriculture, de la forêt, de la pêche, de l'environnement et de l'alimentation. NALT est le thésaurus de NAL (National Agricultural Library) sur le même sujet.

L'ontologie la plus importante, NALT, est utilisée comme cible et AGROVOC comme source. La taille maximale des blocs fusionnables a été fixée à 2 000 concepts.

		L'ontologie cible NALT			L'ontologie source AGROVOC		
Méthodes	Ancres	Blocs générés	Concepts isolés	Plus grand bloc	Blocs générés	Concepts isolés	Plus grand bloc
PBM	14 787	47	4	3 356	318	492	2 830
PAP	14 787	47	4	3 356	252	199	2 939
APP	14 787	47	4	3 118	95	199	3 534

TABLE 6.16 – Partitionnement de AGROVOC et de NALT

Malgré l'absence de mappings de référence qui permettraient d'analyser la qualité des alignements produits, nous avons néanmoins présenté les résultats de partitionnement dans le tableau 6.16, car ils nous semblent être pertinents. Le tableau 6.16 montre que dans cette expérience, comme dans la précédente, le partitionnement, selon nos méthodes, minimise le nombre

de concepts isolés (119 concepts isolés pour PAP et APP, contre 492 pour PBM). Il montre aussi que nos approches minimisent le nombre de blocs générés (252 blocs pour PAP et 95 blocs pour APP, contre 318 pour PBM), ce qui conduit à des partitions certainement moins dispersées.

En conclusion, ce test montre que nos deux méthodes de partitionnement, PAP et APP, permettent la génération de mappings dans le cas d'ontologies de très grande taille. Il montre aussi, comme dans l'expérimentation précédente, que nos deux méthodes limitent le nombre de blocs générés et le nombre de concepts isolés.

6.5 Tests de l'approche d'enrichissement d'ontologies à partir des résultats d'alignement

Un des objectifs du projet GeOnto est de construire une ontologie topographique à partir d'une première ontologie construite par l'IGN de manière semi-automatique, et appelée dans le cadre du projet TopoCarto_COGIT_enrichie.

Le processus d'enrichissement est réalisé en exploitant différentes sources. Dans un premier temps nous présentons la démarche générale d'enrichissement dans le cadre du projet GeOnto pour lequel nous avons identifié différents scénarios (cf. Chapitre 5). Nous présentons ensuite, deux expérimentations concernant l'enrichissement à partir des sources extraites du thésaurus RAMEAU et à partir de l'ontologie volumineuse YAGO. L'objectif de nos expérimentations est d'illustrer et de discuter une utilisation des résultats d'alignement pour l'enrichissement d'ontologies.

6.5.1 Démarche générale d'enrichissement dans le projet GeOnto

A partir d'une première taxonomie de termes décrivant des concepts spatialisés utilisés dans le domaine de la topographie, une première étape, réalisée avant le début du projet, a consisté à créer une ontologie topographique. Elle a été réalisée à partir des termes utilisés dans les spécifications des bases de données de l'IGN, par analyse semi-automatique de ces documents puis réorganisée interactivement.

Pour être plus intensément exploitée, cette ontologie mérite d'être enrichie, à la fois de nouveaux concepts, mais aussi de définitions et propriétés formelles décrivant ces concepts ainsi que leurs relations. L'objectif de l'approche d'enrichissement réalisée dans le cadre du projet GeOnto est la réalisation de tout ou partie de cet enrichissement. Nous présenterons la démarche globale suivie pour réaliser cet objectif.

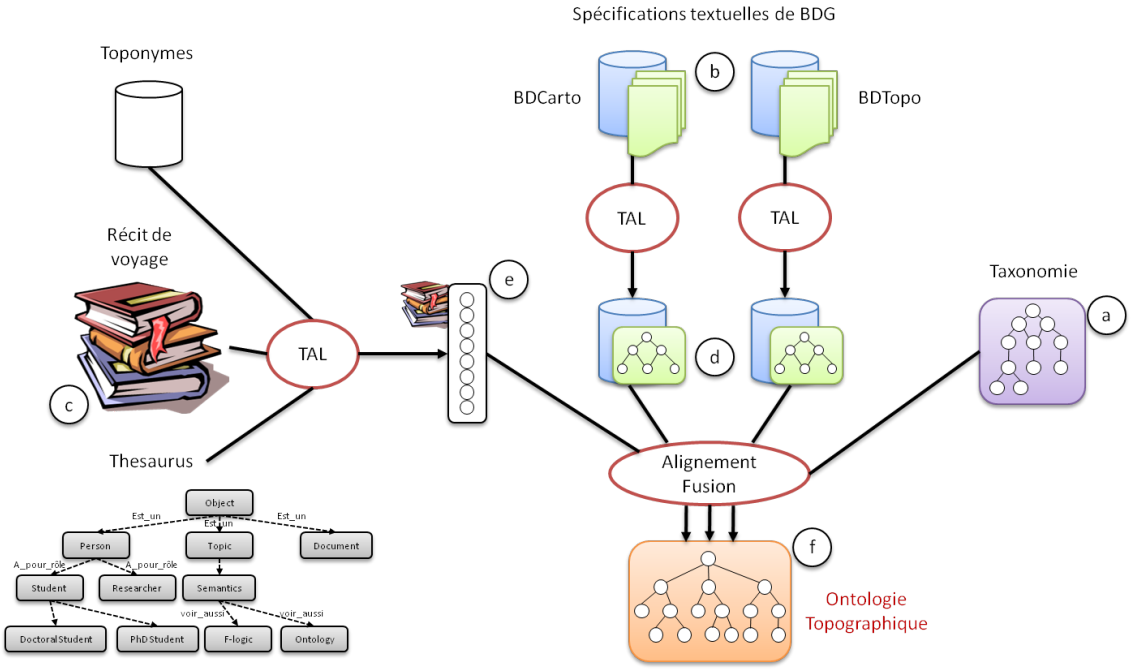

FIG. 6.10 – Projet GeOnto : Approche générale suivie pour la constitution de l'ontologie

L'idée principale du projet est d'utiliser les résultats d'alignement entre une ressource externe (dite source) et l'ontologie à enrichir (dite ontologie cible). Les concepts de la source qui apparaissent dans l'alignement produit peuvent (pour certains d'entre eux) être considérés comme les concepts candidats pour l'enrichissement et les relations établies dans les mappings entre ces concepts sources et ceux de la cible peuvent être pérennisées pour établir le placement des concepts dans l'ontologie cible.

Dans un premier temps, l'enrichissement dans le projet GeOnto a consisté à rapprocher l'ontologie Topo_IRIT construite à partir de l'exploitation des spécifications des bases de données topographiques de l'IGN (cf. b Fig. 6.10) avec la première taxonomie (cf. a Fig. 6.10). Dans un deuxième temps, une exploitation ciblée d'une ressource documentaire adéquate (des récits de voyage de la médiathèque de Pau) (cf. c et e Fig. 6.10), a été effectuée afin de prendre en compte un autre point de vue sur la topographie.

Des bases de toponymes (en l'occurrence la BD-NYME de l'IGN et le gazeteers Geo-names) ont été utilisées pour repérer des entités nommées spatiales ("le gave de Pau", "le maire de Pau") dans la ressource documentaire choisie, en s'appuyant sur l'idée que ces entités sont le plus souvent composées de concepts topographiques (rivière, rue, gave, ville, etc.) associés à des toponymes. Des techniques de traitement automatique du langage naturel ont ensuite été appliquées pour identifier les termes rattachés à ces toponymes ("le gave", "le maire"). Pour aider à distinguer parmi ces termes, ceux qui sont des concept topographiques ("le gave") de ceux qui n'en sont pas ("le maire"), les différents termes ont été confrontés au thésaurus RAMEAU, et pour chaque terme, la partie de RAMEAU relative à ce terme a été extraite puis traduite sous la forme d'une mini ontologie OWL.

Notre tâche a été ainsi d'élaborer un test pour vérifier si les concepts apparaissant dans une

des mini ontologies extraites de RAMEAU relevaient ou pas du domaine de la topographie. Ce test de validation de domaine étant basé sur les résultat de l'alignement de chaque ontologie extraite de RAMEAU avec l'ontologie cible, si le domaine de cette source est validé, les concepts alignés peuvent être considérés comme des concepts potentiellement intéressant pour l'enrichissement.

L'approche schématisée par la figure 6.10 a aussi été étendue par des travaux d'enrichissement basés sur l'exploitation d'une très grosse base de connaissances généraliste, YAGO. Nous avons présenté dans le chapitre 5 une approche permettant l'identification des parties de YAGO proches sémantiquement de celles de l'ontologie à enrichir.

La démarche présentée dans cette section doit permettre d'aboutir à une ontologie faisant le lien entre les concepts utilisés dans des ontologies généralistes et les éléments des bases de données qui y sont liés.

6.5.2 Expérimentations sur RAMEAU

Dans cette expérimentation, 118 composantes ont été extraites de RAMEAU et exprimées sous la forme d'ontologies sources OWL. Parmi ces 118 sources, 41 ne contiennent qu'un, deux ou 3 concepts et sont donc le plus souvent trop petites pour permettre un enrichissement puisque notre méthodologie de validation de domaine impose d'abord l'existence de deux mappings pour valider le domaine. L'enrichissement, qui ne peut porter que sur les concepts présents dans la source mais n'intervenant pas dans ces deux mappings de validation, ne pourra donc porter ici au plus que sur un unique nouveau concept. Ces sources n'ont donc pas été testées. Les 77 sources restantes, i.e. contenant plus de trois concepts, ont d'abord été évaluées manuellement, et réparties en deux groupes, 34 étant considérées comme "a priori à rejeter" et 43 comme "a priori intéressantes".

Parmi les sources jugées "a priori à rejeter", nous avons classé les sources qui a priori, ne devaient pas réussir le test de validation, comme les sources ne contenant aucun concept de même label qu'un concept de l'ontologie topologique. Par exemple, on trouve une source extraite à partir du terme "pied" qu'on imagine bien apparaissant dans une entité spatiale nommée comme "le pied du Mont-Blanc". Les différents concepts de cette source sont "pieds", "tendons", "tendons du pied" et "tendon d'Achille". Aucun de ces concepts ne s'alignant avec un des concepts de TopoCarto_COGIT, cette source sera rejetée par les tests de validation.

Nous avons aussi rangé dans cette catégorie les sources pour lesquelles les seuls mappings fiables identifiables concerneraient tous le même unique concept de la cible ayant le même label qu'un des concepts de la source. Ainsi, pour la source contenant les concepts "Montagne", "Montagne sacrée", "Montagne sainte" et "dieu des montagnes", on trouvera 3 mappings fiables, mais tous associés au même concept cible "Montagne". Aucun autre concept de la cible n'apparaissant dans un mapping fiable pour cette source, son domaine ne doit pas être validé. Faute de temps, les 34 sources considérées comme "a priori à rejeter" n'ont pas été effectivement testées avec les patrons, mais les résultats ne devraient pas apporter de surprise.

Pour les 43 sources jugées manuellement "a priori intéressantes", les résultats de l'application des patrons de validation donnent les résultats suivants : 19 sources ont été jugées valides, 9 de

domaines à valider et 15 ont été rejetées. Parmi les 15 sources rejetées, 8 l'ont été car l'alignement ne contenait finalement pas au moins un mapping d'équivalence, 4 car l'alignement ne contenait qu'un mapping d'équivalence et pas d'autres mappings fiables, 1 car tous les mappings fiables faisaient référence au même unique concept cible, et 2 car les mappings fiables faisaient intervenir des concepts de la source non directement reliées. Ces deux dernières sources rejetées reflètent peut-être des conditions trop restrictives dans les patrons.

Les résultats pour ces 43 sources ont été présentés à l'expert, en explicitant pour chaque source le résultat du test de validation de domaine, les concepts alignés et l'ensemble des éventuels concepts non alignés. Les résultats d'alignement des 28 sources jugées valides ou à valider apportaient environ 140 nouveaux concepts potentiels (pour en fait plus de mappings, mais certains concepts apparaissaient dans plusieurs sources différentes). Parmi ces 140 nouveaux concepts potentiels, l'expert n'en a retenu que 51.

Reprenons par exemple les mappings identifiés dans le cas 1 présenté dans le chapitre 5 concernant l'enrichissement avec les sources extraites de RAMEAU.

`cavernes préhistoriques`	*isA(IncStrict)*	`caverne (Sim: 0.486)`
`grottes ornées`	*isClose(simRel)*	`grotte (Sim: 0.724)`
`grottes préhistoriques`	*isA(IncStrict)*	`grotte (Sim: 0.445)`
`abîmes`	*isA(FilsdeEq)*	`grotte`
`antres`	*isA(FilsdeEq)*	`grotte`

Sur cet exemple, l'expert n'a effectivement validé pour l'enrichissement que les deux derniers mappings, `abîmes` *isA* `grotte` et `antres` *isA* `grotte`. En effet, l'expert a souhaité explicitement ne pas retenir pour l'enrichissement, les concepts exprimant une qualification de concepts déjà existants dans la cible, i.e. les cas où le concept est simplement précédé ou suivi d'un qualificatif qui n'en modifie pas le sens. Ainsi, les deux concepts `grottes ornées` et `grottes préhistoriques` sont considérées comme de simples qualifications du concept `grotte` et ne sont pas retenus. En revanche, dans d'autres contextes, l'ajout d'un adjectif traduit plus qu'une simple qualification et permet de créer effectivement un concept nouveau. C'est le cas par exemple, pour le concept `hôtel particulier`, où la présence de l'adjectif `particulier` introduit un nouveau sens et fait plus que qualifier le concept `hôtel`. Nos outils ne permettant pas de distinguer automatiquement ces différents contextes, nous laissons l'expert confirmer ou rejeter les propositions d'enrichissement.

L'expert a aussi retenu 12 concepts apparaissant dans 4 sources rejetées, dont 8 concepts non alignés dans des sources ne contenant pas au moins un mapping d'équivalence et 4 dans des sources ne contenant qu'un mapping d'équivalence. Les 51 nouveaux concepts retenus et leur placement dans l'ontologie cible sont présentés dans la tableau 6.17. Remarquons que la quasi-totalité de ces 51 nouveaux concepts retenus par l'expert sont issus de mappings construits par la technique t_9 qui s'appuie sur les mappings d'équivalence préalablement trouvés, pour aligner les fils d'un concept de la source comme des spécialisations de son équivalent dans la cible.

L'introduction des nouveaux concepts est faite pour l'instant manuellement, même si une interface dédiée à la fois à la validation des mappings et à leur introduction automatique dans l'ontologie est parfaitement envisageable.

Termes identifiés	**relation**	**concept**
abîme	*isA*	grotte
agglomération urbaine	*isA* *label*	agglomération zone urbaine
alpage	*isA*	paturage
amarrage	*isA*	quai
antre	*isA*	grotte
banlieue	*isA*	agglomération urbaine
bidonville	*isA*	ville
bourgade	*isA*	village
brêche	*isA*	roche
buffet d'eau	*isA*	fontaine
calotte glaciaires	*isA*	glacier
capitale	*isA*	ville
cavité	*isA*	grotte
cénote	*isA*	aven
chaumières	*isA*	habitation
conurbation	*isA*	agglomération urbaine
demeure	*isA*	habitation
dock	*isA*	port
embut	*isA*	aven
friche	*isA*	terrain vague
front de mer	*isA*	ville
galerie	*isA*	tunnel
herbage	*isA*	paturage
hôtel particulier	*isA*	habitation
iceberg	*isA*	glacier
igloo	*isA*	habitation
inlandsis	*isA*	glacier
inselberg	*isA*	montagne
localité	*isA*	village
logis	*label*	habitation
logement	*label*	habitation
maison	*isA*	habitation
mégalopole	*isA*	agglomération urbaine
mégapole	*isA*	agglomération urbaine
moraine d'ablation	*isPartOf*	glacier
pacage	*isA*	paturage
paturage	*isA*	végétation
pature	*isA*	végétation
pédiment	*isA*	montagne
pré	*isA*	végétation
puisard	*isA*	aven
reculée	*isA*	vallée
résidence	*isA*	habitation
spélonque	*isA*	grotte

Termes identifiés	**relation**	**concept**
spéléothème	*isPartOf*	grotte
stalactite	*isPartOf*	grotte
stalagmite	*isPartOf*	grotte
terminal pétrolier	*isA*	port
terrain vague	*isA*	entité à vocation résidentielle
vasque	*isA*	fontaine
zone urbaine	*isMoreGnl* *label*	ville agglomération urbaine

TABLE 6.17: Les concepts retenus et leur placement

Cette expérimentation montre que notre approche de validation de domaine permet à l'expert, dans le cas d'ontologie généraliste, de choisir parmi les sources extraites à partir de cette ontologie, celles dont la thématique correspond à son ontologie particulière. Elle montre également un exemple d'enrichissement que l'expert peut spécifier facilement à l'aide de notre environnement, TaxoMap Framework.

6.5.3 Expérimentations sur YAGO

Nous avons testé l'algorithme présenté dans le chapitre 5 sur l'ontologie cible BDTopo décomposée en 10 blocs.

Le tableau 6.18 ci-dessous présente pour chaque bloc, le nombre de concepts introduits dans le bloc, le nombre de concepts cible appartenant au départ à une paire d' "ancres potentielles", le nombre de concepts de YAGO ayant le même label qu'une ancre potentielle de la cible, le nombre d' "ancres valides" après la passe 1 (cf. Algorithme 8 chapitre 5), le nombre d' "ancres valides" après la passe 2 (cf. Algorithme 9 chapitre 5).

N° du bloc	0	1	2	3	4	5	6	7	8	9
Concepts cibles	102	62	60	24	68	23	69	110	69	30
Ancres cibles potentielles	51	33	38	11	37	13	39	90	43	24
Concepts YAGO ayant le label d'une ancre	120	97	171	44	117	27	142	371	107	73
Ancres valides (Passe 1)	19	9	17	9	21	4	18	47	18	14
Ancres valides (Passe 2)	22	11	23	9	21	4	18	60	24	15

TABLE 6.18 – Tests sur l'ontologie BDTopo

Les résultats de l'algorithme présentés dans le tableau 6.18 ont été précisément tracés pour le bloc n°7. Celui-ci correspond au bloc regroupant les "entités topographiques naturelles". L'intérêt de ce bloc étant qu'il est le plus important et qu'il contient le plus de concepts correspondant à des concepts du monde réel, i.e. dont les labels sont les plus clairement établis et donc a priori les plus faciles à traduire.

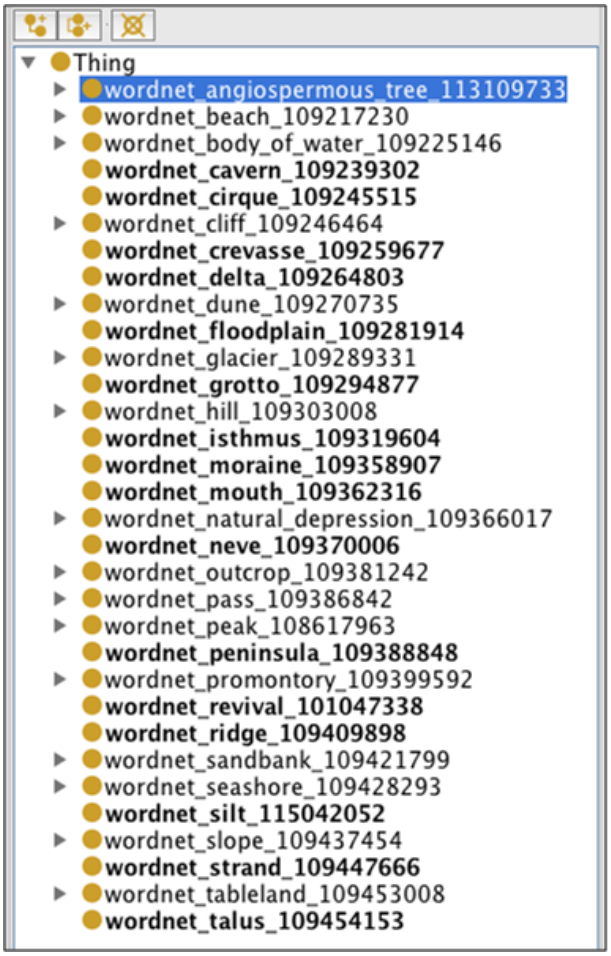

FIG. 6.11 – Expérimentation °1, Bloc n°7

Rappelons en effet que YAGO est une ressource de langue anglaise alors que notre ontologie est en français mais que chacun de ses labels a une ou plusieurs traductions anglaises. On voit ainsi que sur les 110 concepts du bloc cible n°7, 90 concepts peuvent être associés dans YAGO à au moins un concept de label équivalent. Par exemple, pour le concept "col" dont le label traduit en anglais est "pass", on trouve dans YAGO 11 concepts distincts ayant ce label. On trouve au total dans YAGO, 371 concepts ayant le même label qu'une des 90 "ancres cibles potentielles" du bloc cible n°7. Sur ces 90 "ancres potentielles", seules 60 ont finalement été retenues comme des ancres valides à la fin des deux passes et nous avons vérifié que chacun des voisinages des 60 concepts correspondants dans YAGO contenait effectivement des concepts topographiques, à une exception prés.

La qualité des ancres retenues s'évaluent de fait beaucoup plus rapidement en étudiant l'allure générale des blocs sources finalements construits par la méthode PAP à partir de ces ensembles d'ancres.

Pour valider les différentes étapes d'identification des ancres dans l'algorithme proposé, nous avons ainsi comparé les différents blocs cible n°7 qui pouvaient être construits, en fonction des différentes phases de l'algorithme mises en œuvre. Les différents blocs ont été construits avec la

même contrainte de taille limite, et donc, à chaque fois, la méthode PAP a essayé de mettre le plus de concepts possible dans le bloc. Pour comparer les blocs, nous présentons les copies d'écran effectuées à partir du logiciel *Protégé*, en ne visualisant que les concepts directement attachés à la racine du bloc. Classiquement dans *Protégé*, les nœuds ayant des descendants sont précédés d'un triangle, et les nœuds feuilles sont en gras.

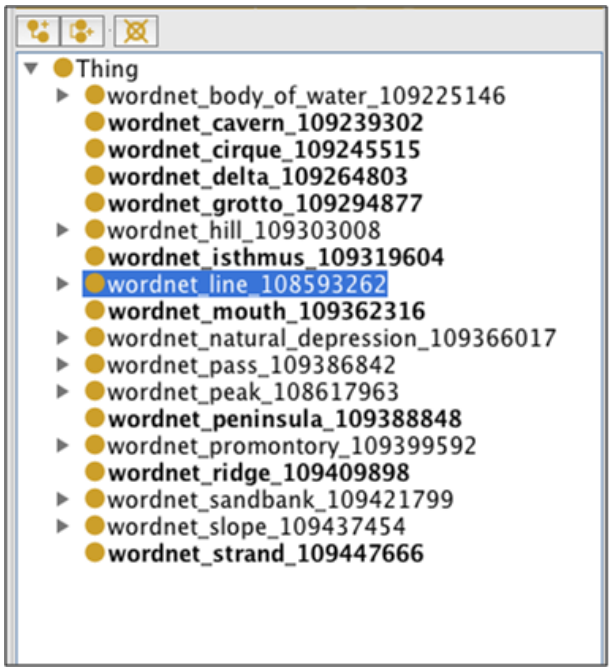

FIG. 6.12 – Expérimentation °2, Bloc n°7

Typiquement, dans notre contexte, moins il y a de feuilles apparaissant comme fils directs de la racine du bloc, plus le bloc construit est cohérent. En effet, par construction, les nœuds feuilles accrochés directement à la racine sont des ancres introduites initialement dans le bloc. Le fait qu'elles soient restées isolées sans s'agglomérer avec d'autres concepts de forte proximité structurelle dans YAGO et qu'elles auraient pu attirer dans le bloc est un élément à analyser. Cela peut signifier différentes choses : soit qu'elles sont de fausses ancres, soit qu'elles n'ont pas de descendant dans YAGO et ou que leur père dans YAGO a lui-même trop de fils trop éloignés du thème du bloc pour qu'ils puissent y être introduit.

Le bloc présenté dans la figure 6.11 a été construit sans l'utilisation de la passe 2 et en acceptant de valider directement les ancres potentielles uniques a priori sans ambiguité, i.e. quand un seul concept de la source a le même label que celui de la cible, sans vérifier la présence d'une autre paire d'ancres dans leur voisinage.

Cela a permis parfois de conserver des ancres correctes, et que l'on n'arrivera plus à valider dans les autres expérimentations plus restrictives sur la validation des ancres uniques, et éventuellement aussi de construire un sous-arbre correct dans le bloc, comme le sous-arbre autour de "glacier" que l'on ne retrouvera donc pas dans les autres blocs n°7. Cela a aussi favorisé à partir d'une seule ancre erronée (Mangrove), l'introduction d'un sous-arbre hors domaine, mais de forte densité et de grande taille dans le bloc, ici la "branche angiospermous-tree", surlignée

en bleu dans la figure.

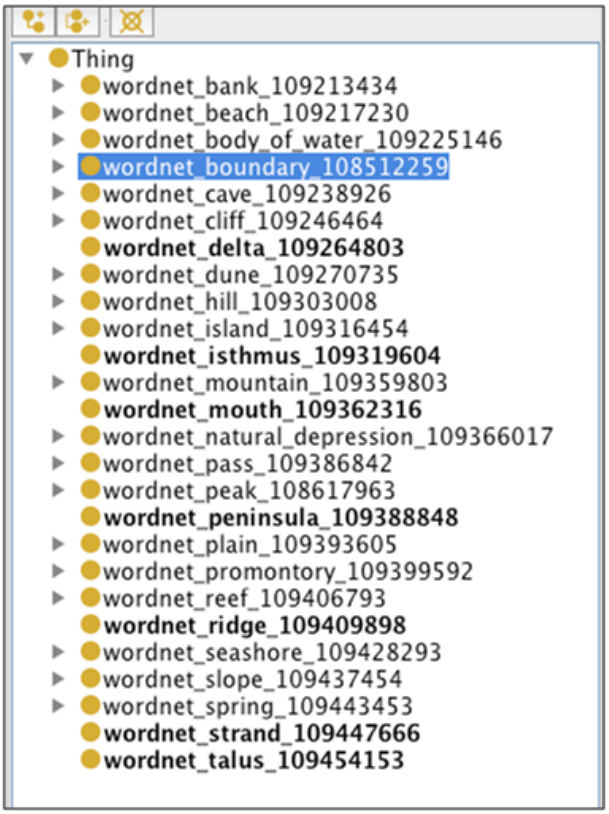

FIG. 6.13 – Expérimentation °3, Bloc n°7

Ce type de sous-arbre hors domaine peut facilement être supprimé du bloc, sous la supervision de l'expert, avant que le bloc ne soit utilisé pour l'enrichissement. Mais les blocs ayant une taille limitée, la présence d'une branche hors domaine occupe de la place qui ne pourra plus être utilisée par d'autres concepts plus pertinents qui auraient pu être ajoutés.

Le bloc présenté dans la figure 6.12 a été construit en utilisant une définition des voisinages plus limitée, ne prenant en considération les frères que si l'ancre considérée n'avait pas de fils directs. En revanche, le bloc de la figure 6.13 tient compte systématiquement des frères dans tous les voisinages. Dans ces deux blocs aussi, on remarque la présence d'une branche hors domaine, mais de taille moindre, et qui laisse plus place à des concepts pertinents. Dans la figure 6.13 par exemple, les ancres "cavern" et "grotto" n'ont pas disparu, mais ont été regroupées sous leur père commun "cave" et d'autres sous-arbres pertinents comme "plain", "reef", "spring" ont pu faire leur apparition.

L'expérimentation ayant mené au bloc de la figure 6.13 est celle qui a permis de valider le plus d'ancres, et, de facto, de ramener dans le bloc le plus de concepts différents des ancres mais appartenant à la thématique du bloc. C'est donc celle-ci qui permettra de proposer le plus d'enrichissements.

Les blocs finalement construits par extraction de YAGO des concepts proches des ancres va-

lides dépendent en plus d'un paramètre "taille limite de blocs" qui doit être fixé au départ. Nous avons fait différents tests autour des ancres valides du bloc n°7, pour trouver la taille permettant d'extraire le plus de concepts "nouveaux" possibles, et que ces concepts "nouveaux" extraits soient tous du domaine topographique. Cette taille limite optimale est très difficile à obtenir car elle dépend de plusieurs facteurs tel que le nombre d'ancres validées par l'algorithme servant à initialiser un bloc.

Cette expérimentation montre que notre approche permet d'aider un expert, dans le cas d'ontologies très volumineuses, à extraire des sous-parties des ces ontologies dont les thématiques peuvent être jugées proches de celle de l'ontologie cible afin de les utiliser pour l'enrichissement.

Conclusion

Dans ce chapitre nous avons présenté les expérimentations réalisées avec l'environnement TaxoMap Framework dans le cadre du projet ANR GeOnto avec deux ontologies réelles dans le domaine de la géographie et également sur des ontologies servant de test dans la campagne d'évaluation des outils d'alignement OAEI.

Dans les expérimentations que nous avons réalisées avec le module d'alignement, nous avons pu valider la nouvelle version adaptée de TaxoMap. Nous avons montré l'intérêt des adaptations et des nouvelles méthodes sur les performances du processus d'alignement et la qualité des résultats obtenus.

Dans les expérimentations que nous avons réalisées avec l'environnement TaxoMap Framework, nous avons pu valider deux modules : (1) le module de raffinement de mappings : nous avons montré l'intérêt du processus de validation/correction pour permettre à un expert de spécifier des traitements de raffinement de mappings, (2) le module de partitionnement : nous avons montré l'intérêt des algorithmes proposés générant des partitions qui sont moins dispersées tout en limitant le nombre de mappings perdus, et permettant à notre outil d'alignement d'aligner des ontologies de très grande taille.

Enfin, nous avons montré l'intérêt des traitements que nous proposons pour l'enrichissement d'ontologies, notamment des traitements permettant de valider des parties extraites considérées comme utiles à l'enrichissement, et des traitements délimitant des parties d'ontologies intéressantes pour l'enrichissement.

Conclusion et Perspectives

Nous présentons, dans ce chapitre, un résumé des principales contributions de cette thèse puis quelques perspectives.

Contributions de la thèse

L'adaptation du système d'alignement TaxoMap

Nous avons effectué des adaptations pour améliorer l'outil d'alignement TaxoMap et le rendre le plus modulaire et flexible possible.

Les améliorations apportées concernent les différentes phases du processus d'alignement. Nous avons amélioré la phase de normalisation des concepts, le calcul de la mesure de similarité, et nous avons ajouté de nouvelles techniques d'alignement.

Dans un second temps, nous avons revu l'architecture de TaxoMap afin de le rendre plus modulaire, plus flexible et plus facile à utiliser. TaxoMap est dorénavant conçu comme un ensemble de modules totalement indépendants les uns des autres. Chaque technique correspond à un module de codes séparé, ce qui facilite la maintenance du logiciel. L'ajout de nouvelles techniques s'en trouve aussi simplifié puisqu'il n'est pas utile de faire interagir les nouvelles techniques développées avec celles pré-existantes. Enfin l'utilisation de TaxoMap est facilitée par l'existence d'une interface graphique donnant la possibilité à l'utilisateur de paramétrer le logiciel, en choisissant les techniques à exécuter parmi les 10 techniques proposées, en fixant leurs paramètres (seuils) et l'ordre de leur exécution.

La proposition d'une approche de spécification de traitements de raffinement de mappings

Nous avons considéré que le processus de raffinement de mappings nécessitait une très bonne connaissance des ontologies alignées. Nous avons donc donné une place importante aux experts du domaine des ontologies alignées, des personnes qui ont une grande connaissance du sens des concepts représentés. Ces experts ont un rôle à jouer. L'approche de raffinement de mappings que nous avons proposée est donc semi-automatique. Certaines tâches, assistées au maximum par des processus automatiques, sont réalisées par des humains, experts ou ingénieurs.

L'approche de raffinement de mappings proposée est basée sur l'utilisation d'un environnement, TaxoMap Framework, qui permet à ces experts/ingénieurs de spécifier de manière déclarative des traitements de raffinement en s'appuyant sur des résultats d'alignement. Ceci nous a

amené à concevoir le langage MPL (the Mapping Pattern Language) qui offre un ensemble de primitives génériques prédéfinies. Ces primitives permettent d'exprimer (1) les différentes conditions qui peuvent être testées sur les concepts intervenant dans un mapping construit par l'outil d'alignement utilisé, en l'occurrence, dans notre cas, TaxoMap ainsi que (2) les actions de raffinement à exécuter.

L'approche proposée permet de spécifier des traitements de raffinement de mappings, mais également d'enrichissement, de fusion ou encore de restructuration d'ontologies. De par sa modularité, elle peut répondre à des besoins variés. A chaque tâche correspond un module de spécifications différent ayant son propre ensemble de primitives de spécification.

Enfin, notons que l'environnement TaxoMap Framework repose sur l'utilisation de TaxoMap, mais il pourrait exploiter des résultats d'alignement générés par un autre outil. Si les prédicats associés à cet autre outil sont déjà définis, la spécification des traitements de raffinement est simplifiée. Si ces prédicats n'ont pas déjà été définis, il faut les ajouter.

La proposition de deux algorithmes de partitionnement d'ontologies

L'objectif de nos propositions était de répondre au problème des outils actuels d'alignement d'ontologies inefficaces sur de grandes ontologies ou incapables de traiter des ontologies très volumineuses. Nous avons proposé deux méthodes de partitionnement d'ontologies basés sur l'algorithme PBM, développé pour le système d'alignement Falcon. Nous avons adapté cet algorithme de façon à prendre en compte l'objectif d'alignement, au plus tôt dans le processus de partitionnement.

Les deux méthodes que nous avons proposées, PAP et APP, sont appliquées sur les deux ontologies simultanément. Elles exploitent des informations faciles à découvrir, même en présence de grandes ontologies. Ces informations portent sur les concepts de chacune des ontologies qui ont le même label et sur la structure des ontologies à aligner.

La première méthode que nous avons proposée, PAP, est bien adaptée aux ontologies qui ont une structure dissymétrique ou des structures correspondant à des décompositions selon des points de vue différents. PAP commence par décomposer l'ontologie la mieux structurée ou celle dont le point de vue de décomposition est retenu pour la construction des partitions et ensuite force la décomposition de l'autre ontologie à suivre le même modèle de décomposition.

La deuxième méthode que nous avons proposée, APP, peut être appliquée lorsque les deux ontologies sont bien structurées et que leur structuration est faite selon un même point de vue. APP favorise la génération de blocs de concepts comparables sémantiquement car contenant des éléments majoritairement liés par des liens d'équivalence.

Les deux méthodes de partitionnement que nous avons proposées passent à l'échelle car elles exploitent des données faciles à obtenir, dont l'obtention est donc peu coûteuse en temps d'exécution. Ceci permet le partitionnement d'ontologies de très grande taille.

La proposition d'une approche d'enrichissement d'ontologies

Concernant le problème de l'enrichissement d'ontologies, nous avons proposé une approche qui utilise l'environnement TaxoMap Framework. Cette approche permet d'aider l'expert suivant le contexte considéré à faire le tri parmi les différentes sources qui pourraient être utilisées pour l'enrichissement, en rejetant celles dont la thématique ne correspond pas à celle de son ontologie particulière et, si les thématiques des deux ontologies sont compatibles, à l'aider à éliminer les mappings peu pertinents pour l'enrichissement.

Nous avons également proposé une approche qui utilise l'environnement TaxoMap Framework intégrant la méthode de partitionnement PAP pour permettre l'enrichissement à partir d'ontologies très volumineuses telles que YAGO.

Enfin, nous avons montré que les patrons ajoutés pour la tâche d'enrichissement ont pu être construits facilement en réutilisant les primitives déjà écrites pour la tâche de raffinement et en les complétant uniquement de deux nouvelles primitives. Ceci prouve l'extensibilité de l'approche proposée.

Une implémentation des approches proposées

TaxoMap et l'environnement TaxoMap Framework ont fait l'objet de développements.

Les différentes adaptations ont été implémentées dans l'outil TaxoMap qui est aujourd'hui muni d'une interface graphique. En plus l'outil TaxoMap a été implémenté sous forme de service web de façon à permettre à moyen terme à quiconque un accès via le web, sans avoir à télécharger le logiciel.

L'environnement TaxoMap Framework a été développé sous forme de modules : un module de raffinement de mapping, un module de partitionnement et un module d'enrichissement. Cet environnement comprend également une interface graphique qui permet à l'expert du domaine de créer des patrons et de choisir les différentes entrées nécessaires à la tâche de partitionnement d'ontologies.

Un ensemble d'expérimentations

Enfin, nous avons validé les différentes propositions de cette thèse grâce à l'environnement TaxoMap Framework.

La version V_2 de TaxoMap a été validée. Nous avons montré l'intérêt des adaptations réalisées et l'apport des nouvelles techniques d'alignement en termes de qualité des résultats générés et de temps d'exécution.

Nous avons testé les algorithmes de partitionnement proposés.

Nous avons illustré l'utilisation de l'environnement TaxoMap Framework pour le raffinement de mappings, et l'enrichissement d'ontologies.

Nous décrivons, dans la section suivante, quelques perspectives au travail réalisé.

Perspectives

Les perspectives que nous allons décrire se rapportent d'une part à l'approche mise en œuvre dans TaxoMap Framework, d'autre part à l'approche de partitionnement orientée alignement proposée.

L'approche mise en œuvre dans TaxoMap Framework est une approche extensible. Elle permet à un expert du domaine de spécifier, d'une part des traitements spécifiques à son domaine sur des mappings produits par un outil d'alignement dans le but de les raffiner, d'autre part des traitements basés sur des mappings pour ajouter de nouveaux concepts/relations à une ontologie qu'il veut enrichir. Pour réaliser ceci, nous avons mis à la disposition de l'expert un langage (MPL) avec des primitives lui permettant d'exprimer des traitements de raffinement et d'enrichissement. Ces primitives ont été conçues pour ces deux types de traitement mais elles pourraient être enrichies pour permettre l'expression d'autres types de traitement tels que la restructuration ou la fusion d'ontologies. Une perspective de notre travail serait donc de chercher à appliquer notre environnement TaxoMap Framework pour d'autres types de traitement et ainsi enrichir la bibliothèque de patrons. On pourrait concevoir d'utiliser TaxoMap Framework pour changer la structure d'une ontologie, par exemple modifier des relations lorsqu'on s'aperçoit que des relations de subsomption ont été utilisées à tort ou pour préciser des relations dont la sémantique n'est pas suffisamment précise. Ce travail correspond à de la restructuration d'ontologie. Il peut prendre appui sur des comparaisons entre l'ontologie à restructurer et une ontologie dite de référence, sur des comparaisons du voisinage de deux concepts équivalents, c'est-à-dire sur des résultats d'un alignement. De ce fait, les traitements de restructuration devraient pouvoir s'exprimer à l'aide de primitives définies dans TaxoMap Framework. De la même façon, un alignement entre deux ontologies peut être la base de traitements de fusion de ces ontologies. Ils pourraient permettre de repérer les parties similaires exprimées de façon identique, de repérer les parties similaires mais qui ne sont pas exprimées de la même façon, d'identifier les parties différentes et d'aider à les intégrer. Là encore les traitements utilisés pourraient s'exprimer sous forme de patrons dans l'environnement TaxoMap Framework.

Une utilisation intensive de TaxoMap Framework conduira à multiplier le nombre de patrons de la bibliothèque et également à spécifier de nouvelles primitives. Ceci nous amène donc à une seconde perspective de ce travail. En présence de primitives nombreuses et d'un nombre important de patrons, il devient nécessaire de concevoir une aide automatisée pour la sélection de ces éléments. Il faut, pour cela, associer à chaque patron et chaque primitive une description fonctionnelle, permettre à l'ingénieur/expert de préciser les éléments qu'il recherche et automatiser le rapprochement entre ce qui est recherché et les primitives/patrons déjà définis. On pourrait également compléter ce processus d'une aide à la construction de patrons lorsqu'il n'existe pas dans la bibliothèque de patron répondant à certains besoins. Cette aide pourrait, par exemple, être basée sur des exemples de traitement donnés en entrée d'un processus d'apprentissage automatique qui proposerait en sortie une généralisation des traitements à réaliser, ces traitements pouvant être éventuellement définis à partir de patrons déjà présents dans la bibliothèque.

Concernant nos travaux sur le partitionnement d'ontologies, une perspective intéressante serait de mettre en œuvre une méthode qui évalue automatiquement la qualité des blocs générés.

Ceci est un vrai problème. Pour l'instant, dans nos travaux, cette évaluation se fait manuellement, en vérifiant, pour chaque bloc, les concepts et les relations le constituant. Ceci est un point faible qu'il faut arriver à dépasser en l'automatisant ou l'outillant. La qualité peut se définir de différents façons. Il peut s'agir de vérifier qu'un bloc représente un certain sous-domaine et que chaque bloc correspond bien à un sous-domaine distinct. Un autre critère consiste à vérifier qu'un partitionnement génère un nombre important de mappings et que peu de mappings ne sont pas générés du fait du découpage en blocs réalisé.

Toujours concernant nos travaux sur le partitionnement, l'approche proposée oblige à déterminer une taille maximale pour les blocs construits. Dans les deux méthodes que nous avons proposées, la taille maximale est donnée par l'utilisateur. Ce paramètre dépend bien sûr de la taille maximale acceptable des données en entrée des outils d'alignement. Toutefois, ce n'est pas le seul critère à faire intervenir. Il faudrait pouvoir s'assurer que les blocs construits sont cohérents sémantiquement ou que les blocs générés pour l'ontologie source respectent au mieux la structuration des blocs de l'ontologie cible de façon à augmenter la qualité des résultats de l'alignement. Une perspective intéressante serait de trouver la taille optimale des blocs qui ne serait ni trop élevée, pour ne pas mêler dans un même bloc des concepts sémantiquement éloignés, ni trop limitée, pour permettre de rassembler tous les concepts sémantiquement proches. La détermination de cette taille optimale est une tâche difficile. Une étude approfondie de la façon dont il serait possible de prendre en compte la structuration des ontologies pour fixer ce paramètre serait nécessaire.

Enfin, une perspective de tout le travail réalisé dans cette thèse concerne la réutilisation de l'ensemble des modules développés dans de nouveaux contextes et/ou appliqués dans des domaines d'application nouveaux. Toutes les approches proposées sont des approches modulaires, flexibles, permettant une évolution facile. Leur ré-utilisation ou leur adaptation est ainsi facilitée. Les logiciels supportant ces approches sont facilement intégrables pour être utilisés en totalité ou partiellement, être éventuellement complétés d'autres modules et ainsi satisfaire des besoins nouveaux.

Annexe A

Présentation du service web TaxoMap

Dans le chapitre 2, nous avons décrit la nouvelle version du module d'alignement TaxoMap en intégrant les adaptations récentes réalisées. Nous décrivons, dans cette annexe, le service web TaxoMap correspondant à cette dernière version, TaxoMap web service, développé pour permettre aux utilisateurs un accès au logiciel via le Web, sans avoir à le télécharger.

Dans un premiers temps, nous présentons l'architecture de la plate-forme du service Web. Nous présentons ensuite les différentes parties de cette plate-forme. Enfin, nous présentons un exemple d'utilisation du service web TaxoMap.

Introduction

Le service web *TaxoMap Web Service* [Feliachi, 2011] implémente, sous forme de service web, le logiciel d'alignement d'ontologies TaxoMap. Son architecture est extensible de façon à faciliter l'intégration de nouvelles fonctionnalités. Ce service web a été développé pour faciliter l'utilisation de TaxoMap et son intégration dans d'autres plate-formes sans accéder à son code source.

A.1 La plate-forme *TaxoMap Web Service*

La plate-forme *TaxoMap Web Service* implémente une architecture orientée plugins. Cette architecture permettra l'évolution de la plate-forme en y intégrant de nouveaux plugins. Comme le montre la figure A.1, la plate-forme est composée essentiellement de trois parties : le noyau, les plugins et les interfaces.

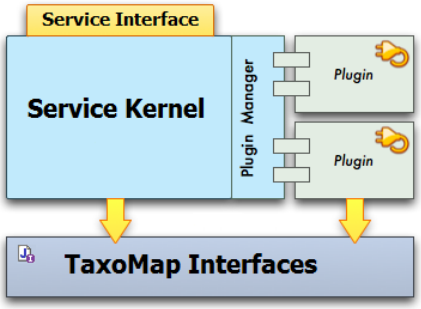

FIG. A.1 – L'architecture de la plate-forme *TaxoMap Web Service*

Le noyau constitue le coeur du service, il contrôle l'ensemble des traitements à exécuter. Les techniques d'alignement sont implémentées sous forme de plugins. Les interfaces assurent l'intégration des plugins dans le noyau et permettront aussi d'intégrer ce service dans d'autres systèmes. Les trois composantes de la plate-forme sont présentées en détail dans ce qui suit.

A.1.1 Le noyau

Le noyau est le coeur de la plate-forme. Il contient trois modules principaux, chaque module réalisant une tâche bien précise. Le module principal (cf. *Service Kernel* Fig. A.1) s'occupe de la tâche d'alignement, en exécutant les techniques implémentées dans les plugins. Le gestionnaire des plugins (cf. *Plugin Manager* Fig. A.1) s'occupe du chargement de ces derniers. L'interface du service Web (cf. *Service Interface* Fig. A.1) permet d'interagir avec le module principal.

Le module principal

```
1  public Matching[] ontologyAlignment(String OntoSource, String OntoCible, String lg,↩
       ArrayList listTechnics, HashMap params){
2      Matching[] matchings = null;
3
4      // Parsing source ontology
5      LiteParser4OWL2 parse2owl1 = new LiteParser4OWL2(OntoSource, "source", 1, lg);
6      Ontology ontoSrc = parse2owl1.parseOntology();
7
8      // Parsing target ontology
9      LiteParser4OWL2 parse2owl2 = new LiteParser4OWL2(OntoCible, "target", 2, lg);
10     Ontology ontoCbl = parse2owl2.parseOntology();
11
12     // Executing the lemmatization process
13     if (LiteParser4OWL2.file!=null){
14         ArrayList <Ontology> v = new ArrayList <Ontology>();
15         v.add(ontoSrc);
16         v.add(ontoCbl);
17
18         File f1 = LemmatizationProcess.preProcess(v);
19         LemmatizationProcess.setLanguage(lg);
20         File f2 = LemmatizationProcess.launchTreeTagger(f1);
21         LemmatizationProcess.saveLemmatization(f2, "$ $");
22
23         // Computing the similarity matrix
24         double[][] SimMatrice;
25         Alignement al = new Alignement("Align", ontoSrc, ontoCbl, 3, prop);
26         al.computeNgramFreq();
27         SimMatrice = al.computeSimilarityMat(...);
28
29         // Applying the available techniques
30         for(PluginInterface plg : listTechnics)
31         {
32             matchings = plg.Align(ontoSrc, ontoCbl, lg, matchings, SimMatrice, ↩
                   params);
33         }
34     }
35     return matchings;
36 }
```

FIG. A.2 – Le module principal

La fonction principale de ce module est d'analyser les ontologies puis d'appliquer les techniques d'alignement sur ces ontologies. La figure A.2 donne une vue globale sur cette fonction. Elle est appelée *ontologyAlignment* et accepte comme paramètres l'URI de l'ontologie source, l'URI de l'ontologie cible, la langue, les techniques à appliquer ainsi que leurs paramètres (seuils). La fonction commence par faire l'analyse des deux ontologies en créant des objets de type *Ontology*. Ces objets sont utilisés dans le processus de lemmatisation et pour calculer la matrice de similarité.

Finalement, les techniques d'alignement sont appliquées en respectant l'ordre donné en entrée et en utilisant les données précédemment calculées. Le résultat de l'application de chaque technique est une entrée de la technique suivante appliquée. Le résultat de la dernière technique est retourné comme résultat de l'alignement.

Le gestionnaire de plugins

Le gestionnaire des plugins s'occupe du chargement et de l'instanciation des plugins. L'ajout de nouveau plugins se fait par le chargement d'un fichier *zip* contenant la description du plugin et de ses sources. Le gestionnaire des plugins fournit les noms des plugins disponibles aux autres modules, il fournit aussi les instances au module principal en vue d'une éventuelle exécution. Les noms des plugins disponibles peuvent être récupérés par l'utilisateur à travers l'interface du service (cf. *Service Interface* Fig. A.1).

```
1  public class PluginClassLoader {
2      private void addFile(File file) {
3          // Using system ClassLoader to load class
4          Method addURL = URLClassLoader.class.getDeclaredMethod("addURL", new Class↩
               [] {URL.class});
5          addURL.setAccessible(true);
6          URLClassLoader cl = (URLClassLoader) ClassLoader.getSystemClassLoader();
7          addURL.invoke(cl, new Object[] { file.toURI().toURL() });
8      }
9
10     public void loadPlugin(File pluginDir) {
11         try {
12             // Loading source files
13             File classesDir = new File(pluginDir, "classes");
14             if (classesDir.exists())
15                 addFile(classesDir);
16             // Loading aaditional library (jar) files
17             File libDir = new File(pluginDir, "lib");
18             File[] jars = libDir.listFiles();
19             if (jars != null)
20                 for (File jar : jars)
21                     addFile(jar);
22         } catch (Exception ex) {
23             ex.printStackTrace();
24         }
25     }
26 }
```

FIG. A.3 – La classe *PluginClassLoader*

Pour l'instanciation des plugins, le gestionnaire contient une classe appelée *PluginClassLoader* (figure A.3) qui implémente le mécanisme d'introspection défini dans l'API *Reflection* de

Java. Ce mécanisme permet de charger de manière dynamique, au moment de l'exécution, les informations relatives à une classe Java. Il offre aussi la possibilité de créer dynamiquement des instances des classes chargées.

```
1  public PluginInterface getInstance(String name) {
2      String className="";
3      // Find the full class name
4      for(Plugin p : plugins){
5          if(name.equals(p.getName()))
6              className = p.getMain_class();
7      }
8      // Class not found
9      if(className == "")
10         return null;
11     // Create and return the instance
12     Class C = Class.forName(className);
13     PluginInterface plg = (PluginInterface) C.newInstance();
14     return plg;
15 }
```

FIG. A.4 – La méthode *getInstance*

La classe *PluginClassLoader* définit deux méthodes *addFile* et *loadPlugin*. La méthode *addFile* récupère une instance du chargeur de classes du système, puis lui demande d'ajouter le fichier à charger en utilisant la méthode *addURL*. La méthode *loadPlugin* s'occupe du chargement de tous les fichiers source et *jar* d'un plugin. Si un fichier a déjà été chargé, le chargeur de classe ne le charge pas une deuxième fois mais réutilise l'instance existante.

Une fois la classe chargée par le système, le gestionnaire des plugins offre la possibilité d'en créer des instances en utilisant la méthode *getInstance* (voir figure A.4).

L'interface du service Web

```
1  public class TaxoMapService {
2      private PluginManager manager = new PluginManager("...");
3
4      public String[] getLanguages() {
5          // List supported languages
6          ...
7      }
8
9      public String[] getHeuristics() {
10         // List available plugins
11         String[] plugs = new String[manager.getPlugins().size()];
12         for(int i=0; i< manager.getPlugins().size(); i++)
13             plugs[i] = manager.getPlugins().get(i);
14         return plugs;
15     }
16
17     public Threshold[] getThresholds() {
18         // List plugins thresholds
19         Threshold[] thresholds = new Threshold[thds.size()];
20         for(int i=0; i< manager.getThresholds().size(); i++)
21             thresholds[i] = new Threshold(...);
```

```
22          return thresholds;
23      }
24
25      public Matching[] alignment(String OntoSource, String OntoCible,
26              String lg, String[] listTechnics,
27              String[] thresholdsNames, Double[] thresholdsValues) {
28          ArrayList<Param> params = manager.getThresholds();
29          String name="", meth="";
30          // Loading thresholds values
31          for(int i=0; i<thresholdsNames.length; i++){
32              name = thresholdsNames[i].substring(...);
33              meth = thresholdsNames[i].substring(...);
34              for(Param p : params)
35                  if(p.getMethod(). == meth && p.getName() == name)
36                      p.setDefault_val(thresholdsValues[i]);
37          }
38          ArrayList techniques = new ArrayList();
39          HashMap realparams = new HashMap();
40          HashMap tmpar;
41          // Loading plugins instances
42          PluginInterface technique = null;
43          for (String s : listTechnics) {
44              if (manager.getPlugins().contains(s)) {
45                  technique = manager.getInstace(s);
46                  techniques.add(technique);
47                  tmpar = new HashMap<String, Param>();
48                  for(Param p : params)
49                      if(p.getMethod().equals(s))
50                          tmpar.put(p.getName(), p);
51                  realparams.put(technique, tmpar);
52              }
53          }
54          // Call the alignment method of the main module
55          AlignProcess process = new AlignProcess();
56          return process.ontologyAlignment(OntoSource, OntoCible, lg,    techniques, ↩
                realparams);
57      }
58  }
```

FIG. A.5 – L'interface du service Web

Le noyau dispose d'une interface utilisateur également implémentée sous forme d'interface de service Web. Cette interface interagit avec les autres composants du noyau en offrant quatre méthodes : *getLanguages*, *getHeuristics*, *getThresholds* et *alignment* (voir figure A.5). La méthode *getLanguages* consulte le fichier des paramètres et retourne la liste des langues supportées par le système. Les méthodes *getHeuristics* et *getThresholds* récupèrent les noms des plugins disponibles ainsi que leurs paramètres (seuils) pour les retourner à l'utilisateur. Ces informations sont données par le gestionnaire des plugins. La méthode *alignment* applique le processus d'alignement défini dans le module principal en lui fournissant les bons paramètres.

A.1.2 Les plugins

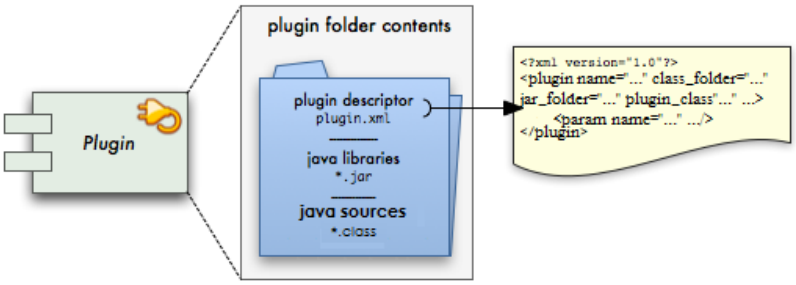

FIG. A.6 – Structure d'un plugin

Les plugins implémentent les techniques d'alignement. Pour ajouter une nouvelle technique, il faut donc l'implémenter sous forme de plugin et l'intégrer dans la plate-forme. La figure A.6 représente le contenu d'un dossier de plugin. Le dossier doit contenir au moins deux fichiers, un descripteur de plugin et une classe principale. Le descripteur de plugin est un fichier appelé *plugin.xml* qui décrit le plugin ainsi que ses paramètres. Le descripteur contient un élément principal *plugin* ayant les paramètres suivants :

- *name*, indiquant le nom de la technique,
- *class_folder*, le dossier qui contient les sources (*.class*),
- *jar_folder*, le dossier qui contient les librairies (*.jar*),
- *plugin_class*, contenant le nom de la classe principale implémentant le plugin et l'interface *PluginInterface*.

L'élément *plugin* peut contenir des éléments *param* décrivant les paramètres (seuils) d'une technique. Chaque paramètre est défini par un nom, un type et une valeur par défaut.

La classe principale du plugin est obligatoire, elle doit implémenter l'interface *PluginInterface*. Son nom est indiqué dans le paramètre *plugin_class*. Cette classe Java contient donc l'implémentation de la méthode *align* qui calcule l'alignement selon une technique donnée. Cette technique peut éventuellement utiliser des paramètres (seuils) définis dans le descripteur du plugin, et passés en paramètre par le noyau du service. La figure A.7 donne un exemple d'implémentation de la classe principale pour la technique *Equivalence*.

```
1  public class Equivalence implements PluginInterface {
2      public ArrayList<Matching> Align(OntologyInterface onto1, OntologyInterface ↩
           onto2, String lg, ArrayList<Matching> matchings, double[][] matSim, HashMap↩
           <String, Double> thresholds) {
3
4          Matching match = null;
5          ...
```

```
6
7           for (int i = 0; i < onto1.getNumberofConcepts(); i++) {
8               for (int j = 0; j < onto2.getNumberofConcepts(); j++) {
9                   ...
10              }
11              if (matSim[i][j] >= thresholds.get("Threshold")) {
12                  onto1.getListofConcepts().get(i).setMatch(true);
13                  match = new Matching(...);
14                  matchings.add(match);
15              }
16          }
17          return matchings;
18      }
19  }
```

FIG. A.7 – Le plugin *Equivalence*

Le plugin peut aussi contenir d'autres classes ou réutiliser d'autres librairies (*.jar*). Pour indiquer ceci au gestionnaire de plugins, il faut mettre toutes les classes dans un même dossier et tous les Jar dans un même dossier. Le nom du dossier qui contient les classes est indiqué dans le paramètre *class_folder* du descripteur de plugin. Le paramètre *jar_folder*, comme son nom l'indique, pointe sur le dossier des librairies Jar. Le plugin doit obligatoirement contenir au moins la classe principale (dans le dossier des classes) et le Jar des interfaces appelé *TaxoMapInterfaces.jar* (dans le dossier des librairies).

Une fois un plugin chargé dans la plate-forme, le contenu de son descripteur est ajouté à un plus grand fichier contenant les descripteurs des plugins disponibles. Ce fichier, appelé *plugins.xml*, est utilisé par le gestionnaire de plugins pour extraire les informations relatives aux plugins disponibles (nom, paramètres, ...). En plus du descripteur, tout le dossier du plugin est copié dans un dossier appelé *plugins* accessible au noyau (pour une éventuelle instanciation du plugin).

A.1.3 Les interfaces

```
1   public interface PluginInterface{
2       public ArrayList<Matching> Align(OntologyInterface OntoSource, ↩
            OntologyInterface OntoCible, String lg, ArrayList<Matching> matchings, ↩
            double[][][] SimMatrice, HashMap<String, Double> thresholds);
3   }
```

FIG. A.8 – L'interface de plugins

Les interfaces (cf. *TaxoMap Interfaces* Fig. A.1) assurent l'interopérabilité entre le service Web et les plugins et entre le service Web et les clients. L'interface la plus importante est la *PluginInterface*, qui contient la signature d'une seule méthode *align* (voir figure A.8). Cette méthode a six paramètres : l'ontologie source, l'ontologie cible, la langue, la liste des alignements précédemment générés, la matrice de similarité et les seuils des plugins.

Chaque implémentation de plugin doit impérativement implémenter cette interface pour permettre au gestionnaire des plugins de le charger correctement. Cette interface fait référence à

d'autres interfaces comme, par exemple, *OntologyInterface*, qui est une interface de manipulation des ontologies (voir figure A.9).

```
1  public interface OntologyInterface {
2      public int position (ConceptInterface c);
3
4      public void addLabel(LabelInterface l);
5
6      public void addLabels(Collection<LabelInterface> l);
7
8      public ArrayList<LabelInterface> getlistOfLabels();
9
10     public void addConcept(ConceptInterface c);
11
12     public void addConcepts(Collection<ConceptInterface> c);
13
14     public ArrayList<ConceptInterface> getListofConcepts();
15
16     public void computeNgramFreq(...);
17
18     public void clear();
19
20     public String getURI();
21
22     public int getNumberofConcepts();
23 }
```

FIG. A.9 – L'interface des ontologies

Les interfaces comportent également une classe importante pour l'utilisation du résultat de l'alignement par un client. Cette classe, appelée *Matching* (figure A.10), encode les mappings entre deux concepts. Un objet de type *Matching* contient les noms des concepts source et cible, les labels associés à ces concepts, le type de la relation établi entre ces concepts ainsi que le nom de la technique qui a généré le mapping.

```
1  public class Matching{
2      String sourceConcept, targetConcept, relation, heuristc;
3      ArrayList<String> sourcelabels, targetlabels;
4      double confidence;
5
6      public ArrayList<String> getSourceLabels() {return sourcelabels;}
7
8      public ArrayList<String> getTargetLabels() {return targetlabels;}
9
10     public void addSourceLabel(String l) {sourcelabels.add(l);}
11
12     public void addTargetLabel(String l) {targetlabels.add(l);}
13
14     public void addSourceLabels(Collection<String> labels) {
15         if(labels != null) this.sourcelabels.addAll(labels);
16     }
17
18     public void addTargetLabels(Collection<String> labels) {
19         if(labels != null) this.targetlabels.addAll(labels);
20     }
21
22     public String getRelation() {return relation;}
```

```
23
24      public void setRelation(String r) {relation = r;}
25
26      public String getTargetConcept() {return targetConcept;}
27
28      public void setTargetConcept(String tC) {targetConcept = tC;}
29
30      public String getSourceConcept() {return sourceConcept;}
31
32      public void setSourceConcept(String sC) {sourceConcept = sC;}
33
34      public double getConfidence() {return confidence;}
35
36      public void setConfidence(double c) {confidence = c;}
37
38      public String getHeuristc() {return heuristc;}
39
40      public void setHeuristc(String h) {heuristc = h;}
41
42      public Matching(String sourceConcept, String targetConcept, String relation, ↩
            String heuristc, double confidence) {
43          ...
44      }
45  }
```

FIG. A.10 – La classe *Matching*

A.2 Exemple d'utilisation

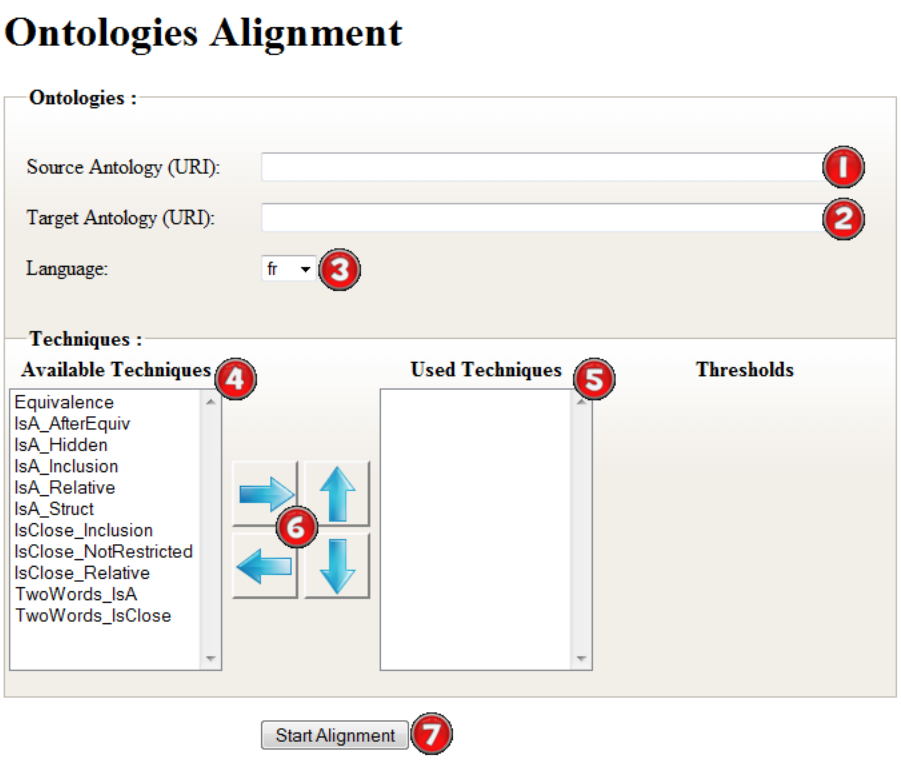

FIG. A.11 – L'interface du client Web

Un client Web a été développé pour illustrer l'utilisation du service web TaxoMap. L'interface du client est présentée dans la figure A.11. Les URIs des ontologies source et cible sont saisies respectivement dans (1) et (2). La liste des langues supportées est chargée à partir du service et est affichée dans (3). Dans (4) la liste des techniques (plugins) disponibles dans le système est affichée, permettant à l'utilisateur de choisir une ou plusieurs techniques à appliquer. Cette liste est aussi chargée à partir du service. Les techniques que l'on veut appliquer sont ajoutées à (5) afin d'être exécutées dans l'ordre indiqué. Pour choisir une technique, ou changer l'ordre d'exécution des techniques, les boutons (6) sont utilisés. Enfin, Le bouton (7) sert à lancer le processus d'alignement du service Web.

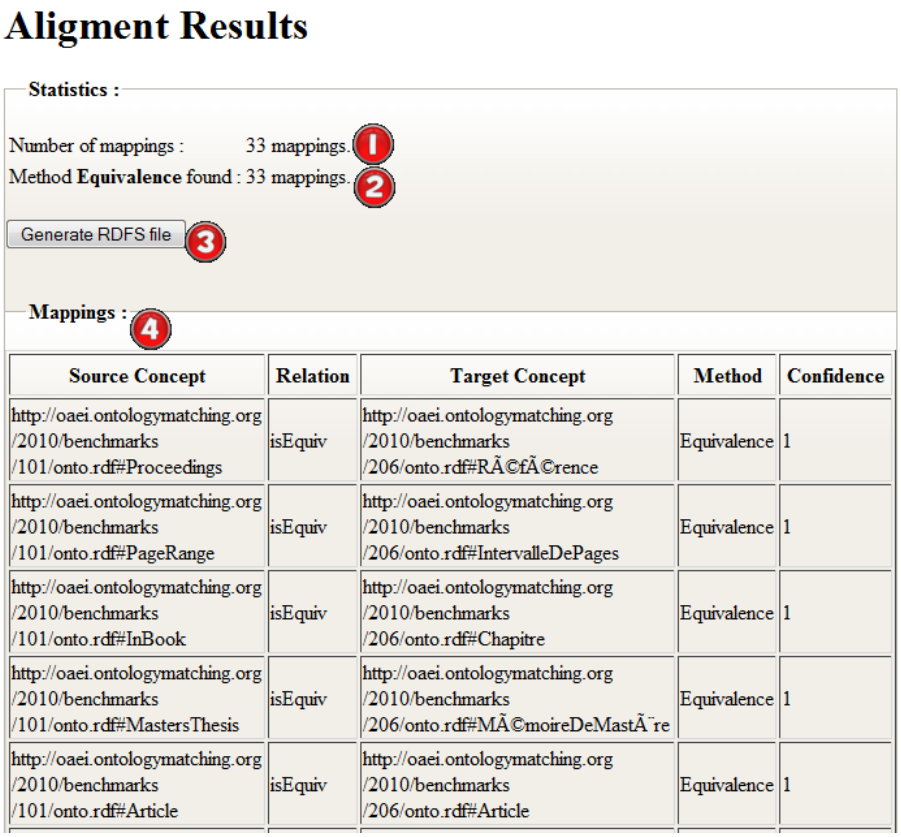

FIG. A.12 – Le résultat d'un alignement

Le résultat retourné par le service est présenté dans la figure A.12. Le nombre total de mappings trouvés est présenté dans (1). Le nombre de mappings trouvés par chaque technique est affiché dans (2). Le bouton (3) permet de générer un fichier *.rdfs* contenant le résultat de l'alignement. La liste de tous les mappings trouvés est détaillée dans (4).

Conclusion

Le web service *TaxoMap Web Service* est une implémentation de l'outil TaxoMap facilitant son évolution. De nouvelles techniques peuvent être ajoutées très facilement. Cette implémentation rend, par ailleurs, plus facile l'utilisation de TaxoMap et son intégration dans n'importe

quel autre système tout en préservant son code source.

Un autre aspect intéressant de l'implémentation réalisée vient du fait qu'il y a une claire séparation entre les aspects interfaces et les aspects traitements d'alignement. Ainsi, si d'autres calculs doivent compléter les traitements d'alignement actuels, ces calculs devront être définis dans la méthode *align* et l'interface n'aura pas à être modifiée.

Le service Web est disponible à l'adresse suivante :
`http://taxomap.lri.fr:8090/axis2/services/TaxoMapService?wsdl`

Le client est déployé à l'adresse : `http://taxomap.lri.fr/`

Annexe B

Présentation de l'outil de visualisation AlignViz

Il est possible d'aider l'expert dans son travail de validation des mappings générés par un outil d'alignement tel TaxoMap. Nous décrivons dans cette annexe, l'outil de visualisation AlignViz, qui permet à un expert du domaine de valider (ou non) un mapping et éventuellement d'introduire un commentaire.

Nous présentons, dans un premiers temps, l'interface qui permet l'accès à des fonctionnalités de visualisation des ontologies alignées et des mappings générés. Nous présentons ensuite, l'interface donnant accès aux fonctionnalités de validation des mappings.

Introduction

AlignViz est un outil de visualisation d'ontologies et de validation des mappings trouvés par un outil d'alignement [Alla, 2008]. Les ontologies sont représentées en OWL. Les mappings sont représentés en RDF/XML [21].

AlignViz a pour but de faciliter le travail de l'évaluation d'un alignement par un expert. Il essaie d'apporter les réponses aux besoins suivants :

- Un affichage ergonomique d'une ontologie avec diverses fonctionnalités de navigation telles qu'accéder à un type particulier de mappings ou à un concept particulier.

- Divers niveaux de visualisation, l'évaluateur doit pouvoir avoir une vision d'ensemble de ce qu'il a à évaluer avec la possibilité de zoomer sur les parties qui l'intéressent. Cette fonctionnalité est d'autant plus importante quand les ontologies sont volumineuses.

- Permettre d'évaluer les mappings en les validant ou en les supprimant ainsi que les sauvegarder. Cette fonctionnalité nécessite de gérer les sessions des utilisateurs. Il est aussi important que les actions de validation soient réversibles, ainsi l'expert peut à tout moment reconsidérer sa validation.

21. Selon le format décrit dans `http://alignapi.gforge.inria.fr/format.html`

Installation

AlignViz requiert l'installation de Graphviz (*Graph Visualization Software*). Graphviz contient un ensemble d'outils qui manipulent des graphes définis à l'aide de scripts suivant le langage DOT. L'outil est accessible à *http ://www.graphviz.org/Download.php* dans sa version 2.20 et plus.

Il faut disposer de Java à partir de 1.6.

Il faut copier le fichier .jar dans un répertoire.

Pour lancer ALignViz, il faut double-cliquer sur le .jar, ou taper en ligne de commande :

java -jar le_fichier_jar

B.1 Visualisation

AlignViz exploite l'outil Graphviz ainsi que les API Protégé-OWL et Jena ontology. Il permet de visualiser une ontologie, ainsi que deux ontologies simultanément afin de pouvoir visualiser les mappings retrouvés entre concepts par un outil d'alignement.

L'onglet *Visualisation* permet de visualiser une ontologie, l'onglet *Alignement* sert à visualiser deux ontologies ainsi que leur alignement.

B.1.1 Visualiser une ontologie

Il faut cliquer dans le menu *Fichier* sur *Charger un fichier OWL* (voir figure B.1).

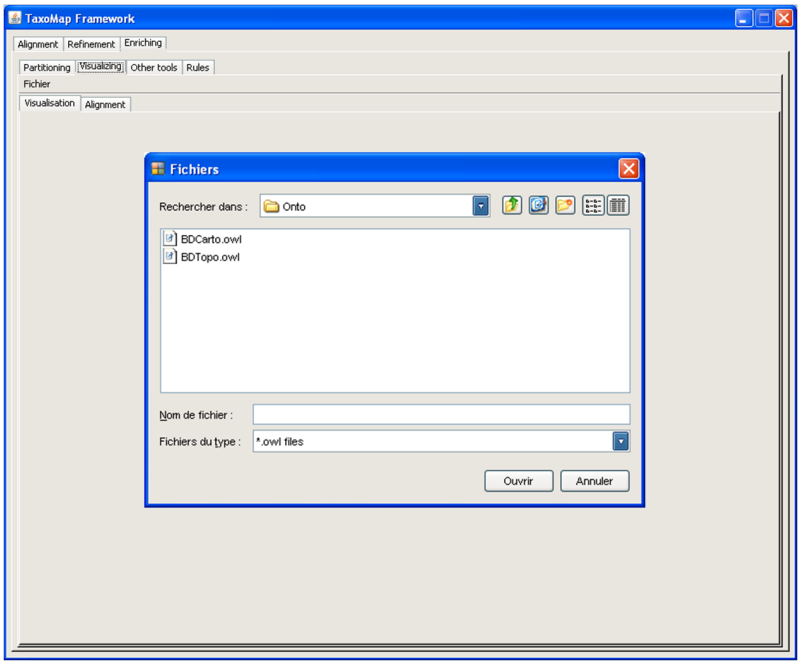

FIG. B.1 – Chargement d'une ontologie

Il est possible (1) de visualiser/cacher un concept avec ses ancêtres et ses fils, (2) de visualiser/cacher tous les concepts, (3) de zoomer, et (4) de rechercher un concept. Pour visualiser toute ou une partie d'une ontologie, il faut sélectionner le concept Top (*Thing*) et choisir ensuite la profondeur voulue. La figure B.2 présente les différentes fenêtres.

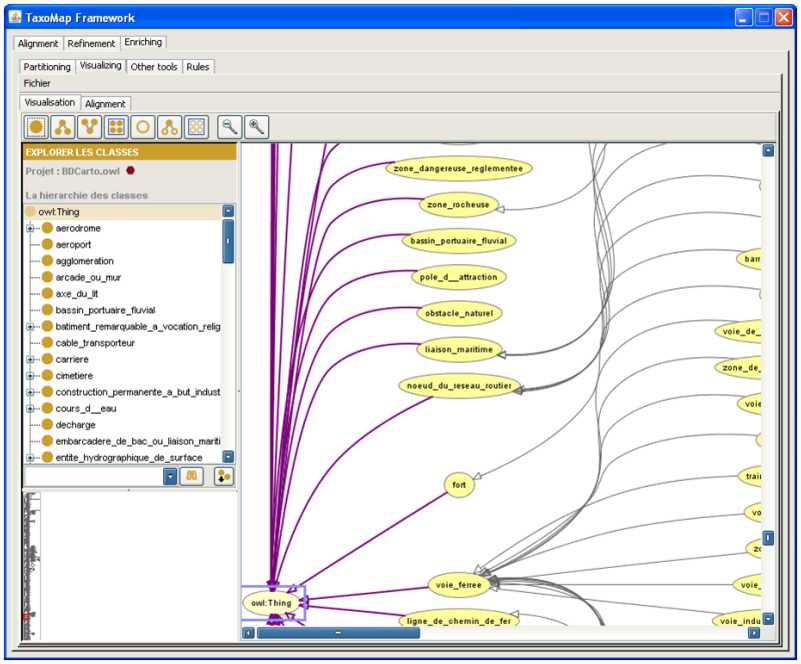

FIG. B.2 – Visualisation d'une ontologie

B.1.2 Visualiser deux ontologies simultanément

L'onglet *Alignement* permet de charger deux ontologies. Une fois les ontologies chargées, le bouton *Afficher* permet d'afficher simultanément les deux ontologies, avec les mêmes fonctionnalités que celles décrites dans la section précédente (voir figure B.3).

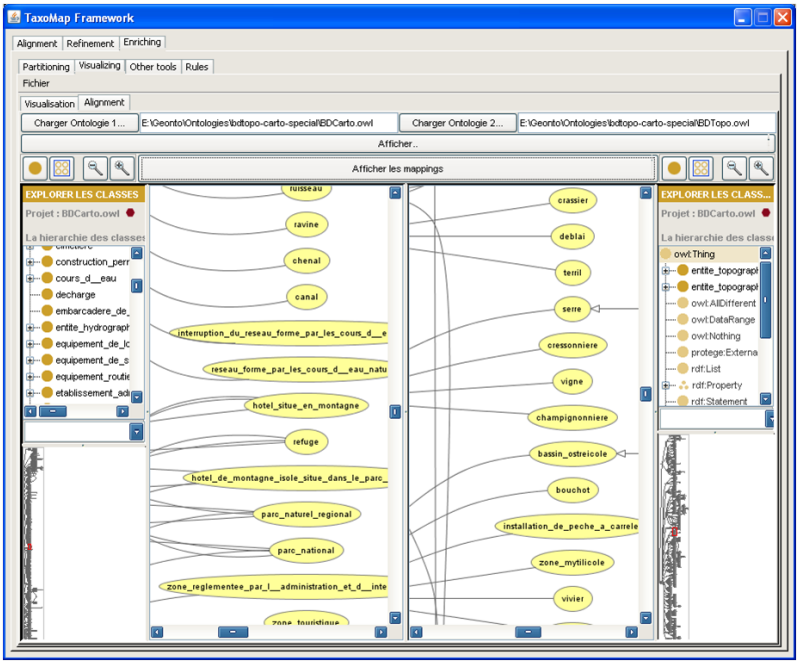

FIG. B.3 – Visualisation de deux ontologies

B.1.3 Visualiser un alignement

Le bouton *Afficher les mappings* permet de charger le fichier d'alignement des deux ontologies. Les mappings sont affichés dans un tableau qui comprend l'URI des concepts, les types des relations de mapping ainsi que leur état de validation. Ces aspects sont détaillés dans la section qui suit.

B.2 Validation des mappings

B.2.1 Affichage et tri

Les mappings peuvent être triés, en cliquant sur l'onglet correspondant au critère selon lequel on souhaite trier (par nom de concept, type de relation ou état de validation). Ainsi nous pouvons visualiser les mappings par type de relation (*isEq*, *isA*, *isMoreGnl*, *isClose*). En cliquant deux fois sur une ligne du tableau, les concepts concernés par le mapping sont marqués par un cadre dans leur ontologie respective (voir figure B.4).

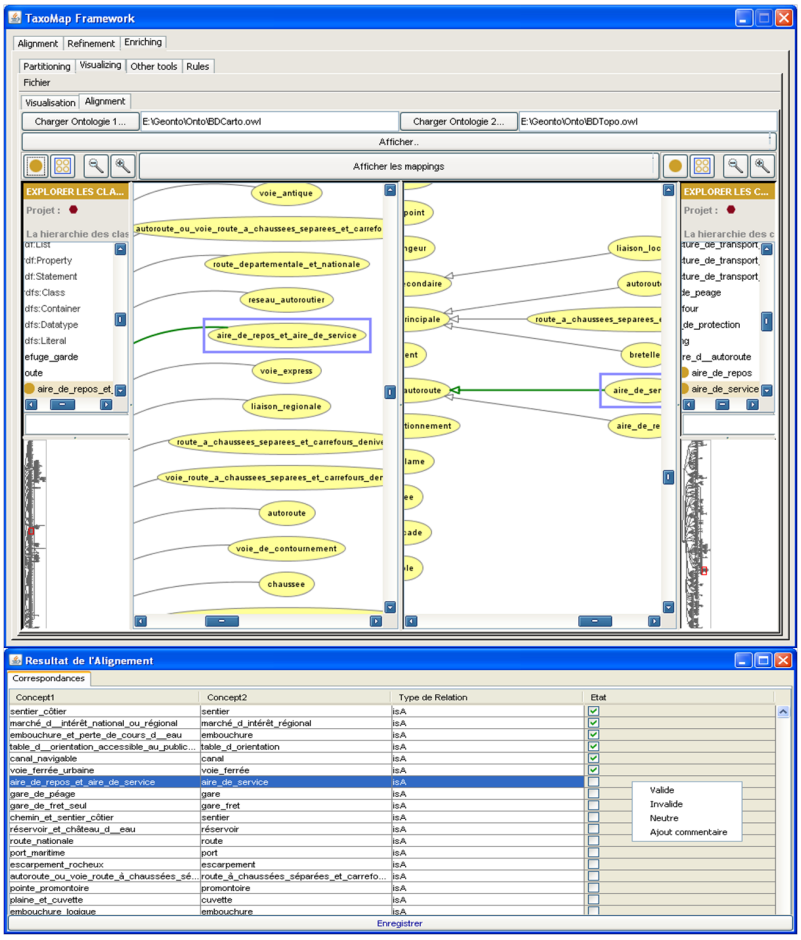

FIG. B.4 – Visualisation des mappings dans les deux ontologies

B.2.2 Notion de session

Lors d'un premier téléchargement, la case *Etat* des mappings qui n'ont pas encore été validés, n'est donc pas cochée. Elle le sera au fur et à mesure de la validation (voir section suivante). Les différentes validations peuvent être enregistrées à tout moment (en cliquant sur le bouton *Enregistrer*), un fichier *.viz* est donc crée et il est possible alors de reprendre la validation à l'état où on l'a laissée en téléchargeant le *.viz* à la place du *.rdf*. L'expert peut modifier son évaluation et re-enregistrer. Un autre expert peut valider le même alignement en repartant du *.rdf*.

B.2.3 Différents états de validation

Un clic droit sur une ligne du tableau permet de valider un mapping, les différents états sont :

– Valide : Quand le mapping est correct.

– Invalide : Quand le mapping est faux.

– Neutre : Quand l'expert n'arrive pas à se faire un avis.

Il est toujours possible de revenir sur ces mappings et modifier leur état.

Une fois évalué, la case *Etat* de le mapping est coché automatiquement. Les mappings qui n'ont pas été évalués sont marqués comme non analysés dans la sortie *.viz* (voir annexe).

Il est également possible d'ajouter un commentaire pour justifier l'évaluation du mapping. Un tel commentaire peut servir de base pour une discussion entre experts (voir figure B.5).

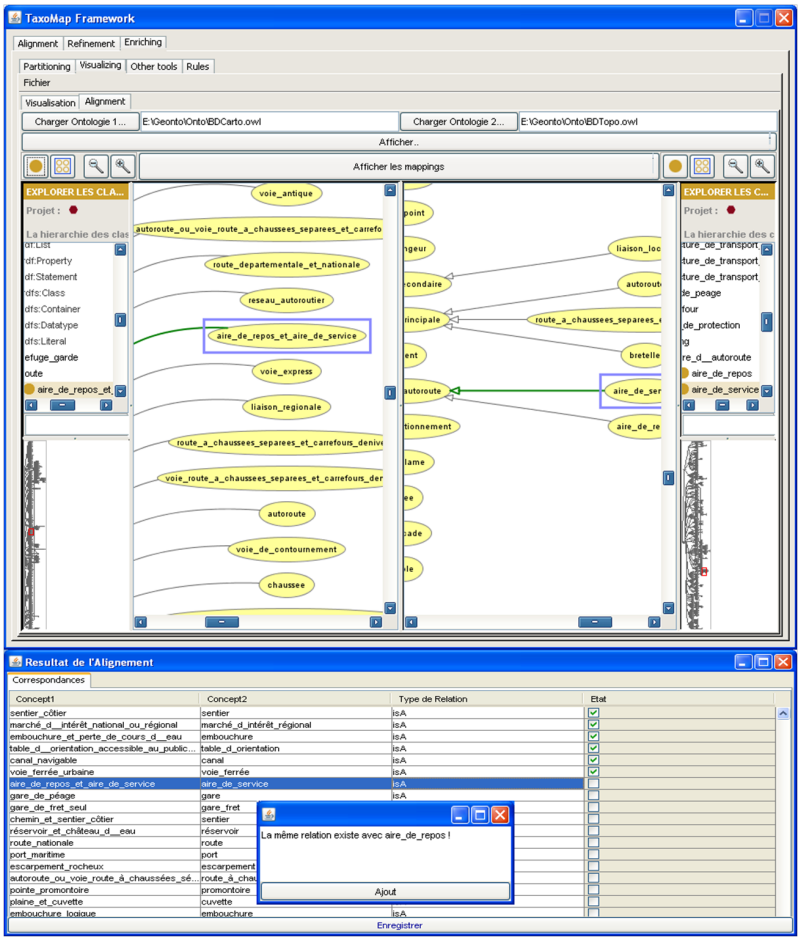

FIG. B.5 – Ajout d'un commentaire

B.2.4 Extrait d'une sortie d'alignement évalué (.viz)

```
...
<map>
    <Cell>
      <entity1 rdf:resource="file:/E:/BDCarto/BDCarto.owl#voie_ferrée_urbaine"/>
      <entity2 rdf:resource="file:/E:/BDTopo/BDTopo.owl#voie_ferré"/>
      <measure rdf:datatype="float">0.8</measure>
```

```
      <relation>isA</relation>
      <analyse>Valide</analyse>
      <comment/>
    </Cell>
  </map>
 <map>
    <Cell>
      <entity1 rdf:resource="file:/E:/BDCarto/BDCarto.owl#aire_de_repos_et_aire_de_service"/>
      <entity2 rdf:resource="file:/E:/BDTopo/BDTopo.owl#aire_de_service"/>
      <measure rdf:datatype="float">0.8</measure>
      <relation>isA</relation>
      <analyse>Valide</analyse>
      <comment>La même relation existe avec aire_de_repos!</comment>
   </Cell>
  </map>
<map>
    <Cell>
      <entity1 rdf:resource="file:/E:/BDCarto/BDCarto.owl#gare_de_péage"/>
      <entity2 rdf:file:/E:/BDTopo/BDTopo.owl#gare"/>
      <measure rdf:datatype="float">0.8</measure>
      <relation>isA</relation>
      <analyse>notAnalysed</analyse>
      <comment/>
    </Cell>
</map>
...
```

Conclusion

AlignViz est un outil permettant la visualisation de deux ontologies OWL et de leurs mappings. L'interface implémentée dans cet outil permet à un expert d'avoir une vision plus claire des mappings générés en repérant leur emplacement dans les ontologies alignées.

L'outil AlignViz est disponible à l'adresse suivante : `http://www.lri.fr/~hamdi/AlignViz/AlignViz.html`. Il a été intégré à l'environnement *TaxoMap FrameWork*.

Table des figures

Liste des tableaux

Bibliographie

[Abadie and Mustière, 2010] Nathalie Abadie and Sébastien Mustière 2010. Constitution et exploitation d'une taxonomie géographique à partir des spécifications de bases de données. Revue Internationale de Géomatique, 20(2) :145–174.

[Agirre et al., 2000] Eneko Agirre, Olatz Ansa, Eduard H. Hovy, and David Martínez 2000. Enriching very large ontologies using the WWW. In ECAI Workshop on Ontology Learning.

[Alexe et al., 2008] Bogdan Alexe, Laura Chiticariu, Renée J. Miller, and Wang chiew Tan 2008. Muse : Mapping Understanding and deSign by Example. In Proceedings of the 2008 IEEE 24th International Conference on Data Engineering, pages 10–19, Washington, DC, USA. IEEE Computer Society.

[Alla, 2008] Jamal Ait Alla 2008. Outil de visualisation d'ontologies et de validation d'alignement. Technical report, Ecole Mohammadia, Rabat.

[Aussenac-Gilles and Kamel, 2009] Nathalie Aussenac-Gilles and Mouna Kamel 2009. Ontology Learning by Analyzing XML Document Structure and Content. In KEOD 2009 - Proceedings of the International Conference on Know ledge Engineering and Ontology Development, Funchal - Madeira, Portugal, October 6-8, 2009, pages 159–165.

[Berners-Lee et al., 2001] Tim Berners-Lee, James Hendler, and Ora Lassila 2001. The Semantic Web. Scientific American, 284(5) :34–43.

[Boley, 1998] Daniel Boley 1998. Principal Direction Divisive Partitioning. Data Min. Knowl. Discov., 2 :325–344.

[Bonafonte and Mariño, 1996] Antonio Bonafonte and José B. Mariño 1996. Language modeling using x-grams. In The 4th International Conference on Spoken Language Processing, Philadelphia, PA, USA.

[Caracciolo et al., 2008] Caterina Caracciolo, Jérôme Euzenat, Laura Hollink, Ryutaro Ichise, Antoine Isaac, Véronique Malaisé, Christian Meilicke, Juan Pane, Pavel Shvaiko, Heiner Stuckenschmidt, Ondrej Sváb-Zamazal, and Vojtech Svátek 2008. Results of the Ontology Alignment Evaluation Initiative 2008. In Proceedings of the 3rd International Workshop on Ontology Matching (OM-2008) Collocated with the 7th International Semantic Web Conference (ISWC-2008), Karlsruhe, Germany, October 26, 2008.

[Castano et al., 2003] Silvana Castano, Alfio Ferrara, and Stefano Montanelli 2003. H-match : an algorithm for dynamically matching ontologies in peer-based systems. In In Proc. of the 1st Int. Workshop on Semantic Web and Databases (SWDB) at VLDB 2003, pages 231–250.

[Charlet et al., 2008] Jean Charlet, Sylvie Szulman, Guy Pierra, Nadia Nadah, H. V. Teguiak, Nathalie Aussenac-Gilles, and Adeline Nazarenko 2008. DAFOE : A Multimodel and Multimethod Platform for Building Domain Ontologies. In Djamal Benslimane (ed), 2^{emes}Journées Francophones sur les Ontologies, Lyon, France. ACM.

[Church and Hanks, 1989] Kenneth Ward Church and Patrick Hanks 1989. Word Association Norms, Mutual Information and Lexicography. In ACL, pages 76–83.

[Cimiano and Pinto, 2010] Philipp Cimiano and Helena Sofia Pinto (eds) 2010. Knowledge Engineering and Management by the Masses - 17th International Conference, EKAW 2010, Lisbon, Portugal, October 11-15, 2010. Proceedings, volume 6317 of *Lecture Notes in Computer Science*. Springer.

[Cruz et al., 2009] Isabel F. Cruz, Flavio Palandri Antonelli, and Cosmin Stroe 2009. AgreementMaker : efficient matching for large real-world schemas and ontologies. Proc. VLDB Endow., 2 :1586–1589.

[Daille, 1996] Béatrice Daille 1996. Study and Implementation of Combined Techniques for Automatic Extraction of Terminology. In Judith Klavans and Philip Resnik (eds), The Balancing Act : Combining Symbolic and Statistical Approaches to Language, pages 49–66. The MIT Press, Cambridge, Massachusetts.

[Davulcu et al., 2003] Hasan Davulcu, Srinivas Vadrevu, and Saravanakumar Nagarajan 2003. OntoMiner : Bootstrapping and Populating Ontologies from Domain Specific Web Sites. In Isabel F. Cruz, Vipul Kashyap, Stefan Decker, and Rainer Eckstein (eds), SWDB, pages 259–276.

[Davulcu et al., 2004] Hasan Davulcu, Srinivas Vadrevu, and Saravanakumar Nagarajan 2004. OntoMiner : bootstrapping ontologies from overlapping domain specific web sites. In Stuart I. Feldman, Mike Uretsky, Marc Najork, and Craig E. Wills (eds), WWW (Alternate Track Papers & Posters), pages 500–501. ACM.

[Dieng and Hug, 1998] Rose Dieng and Stefan Hug 1998. Comparison of Personal Ontologies Represented through Conceptual Graphs. In ECAI, pages 341–345.

[Do, 2005] Hong-Hai Do 2005. Schema matching and mapping-based data integration. PhD thesis, University of Leipzig, Leipzig (DE).

[Do and Rahm, 2002] Hong-Hai Do and Erhard Rahm 2002. COMA – A System for Flexible Combination of Schema Matching Approaches. In Proc. 28th International Conference on Very Large Data Bases (VLDB), pages 610–621, Hong Kong (CN).

[Do and Rahm, 2007] Hong-Hai Do and Erhard Rahm 2007. Matching large schemas : Approaches and evaluation. Information Systems, 32(6) :857–885.

[Doan et al., 2001] An-Hai Doan, Pedro Domingos, and Alon Halevy 2001. Reconciling Schemas of Disparate Data Sources : A Machine-Learning Approach. In Proc. 20th International Conference on Management of Data (SIGMOD), pages 509–520, Santa Barbara (CA US).

[Doan and Halevy, 2005] An-Hai Doan and Alon Halevy 2005. Semantic Integration Research in the Database Community : A Brief Survey. AI Magazine, 26(1) :83–94. Special issue on Semantic integration.

[Ehrig, 2007] Marc Ehrig 2007. Ontology Alignment : Bridging the Semantic Gap, volume 4 of *Semantic Web And Beyond Computing for Human Experience*. Springer.

[Ehrig et al., 2005] Marc Ehrig, Steffen Staab, and York Sure 2005. Bootstrapping Ontology Alignment Methods with APFEL. In Proc. 4th International Semantic Web Conference (ISWC), volume 3729 of *Lecture notes in computer science*, pages 186–200, Galway (IE).

[Euzenat, 2004] Jérôme Euzenat 2004. An API for Ontology Alignment. In The Semantic Web - ISWC 2004 : Third International Semantic Web Conference,Hiroshima, Japan, November 7-11, 2004. Proceedings, pages 698–712.

[Euzenat et al., 2009] Jérôme Euzenat, Alfio Ferrara, Laura Hollink, Antoine Isaac, Cliff Joslyn, Véronique Malaisé, Christian Meilicke, Andriy Nikolov, Juan Pane, Marta Sabou, François Scharffe, Pavel Shvaiko, Vassilis Spiliopoulos, Heiner Stuckenschmidt, Ondrej Sváb-Zamazal, Vojtech Svátek, Cássia Trojahn dos Santos, George A. Vouros, and Shenghui Wang 2009. Results of the Ontology Alignment Evaluation Initiative 2009. In Proceedings of the 4th International Workshop on Ontology Matching (OM-2009) collocated with the 8th International Semantic Web Conference (ISWC-2009) Chantilly, USA, October 25, 2009.

[Euzenat et al., 2007] Jérôme Euzenat, Antoine Isaac, Christian Meilicke, Pavel Shvaiko, Heiner Stuckenschmidt, Ondrej Sváb, Vojtech Svátek, Willem Robert van Hage, and Mikalai Yatskevich 2007. Results of the Ontology Alignment Evaluation Initiative 2007. In Proceedings of the 2nd International Workshop on Ontology Matching (OM-2007) Collocated with the 6th International Semantic Web Conference (ISWC-2007) and the 2nd Asian Semantic Web Conference (ASWC-2007), Busan, Korea, November 11, 2007.

[Euzenat et al., 2008] Jérôme Euzenat, Chantal Reynaud, Brigitte Safar, and Haïfa Zargayouna 2008. Intégration des prototypes d'alignement à la plate-forme WebContent. Rapport d'avancement webcontent, WC/Lot3/CEA-LIST/D3.13.

[Euzenat and Shvaiko, 2007] Jérôme Euzenat and Pavel Shvaiko 2007. Ontology matching. Springer.

[Faatz and Steinmetz, 2002] Andreas Faatz and Ralf Steinmetz 2002. Ontology Enrichment with Texts from the WWW. In Proceedings of ECML-Semantic Web Mining 2002, Helsinki.

[Feliachi, 2011] Abderrahmane Feliachi 2011. Des services web pour l'alignement d'ontologies. Mission doctorale, Laboratoire de Recherche en Informatique (LRI), Université Paris-Sud 11.

[Fernández-Breis et al., 2010] Jesualdo Tomás Fernández-Breis, Luigi Iannone, Ignazio Palmisano, Alan L. Rector, and Robert Stevens 2010. Enriching the Gene Ontology via the Dissection of Labels Using the Ontology Pre-processor Language. In [Cimiano and Pinto, 2010], pages 59–73.

[Fuxman et al., 2006] Ariel Fuxman, Mauricio A. Hernández, C. T. Howard Ho, Renée J. Miller, Paolo Papotti, and Lucian Popa 2006. Nested Mappings : Schema Mapping Reloaded. In Proceedings of the 32nd international conference on Very large data bases, VLDB '06, pages 67–78. VLDB Endowment.

[Garey and Johnson, 1979] Michael Garey and David Johnson 1979. Computers and intractability : a guide to the theory of NP-completeness. W. H. Freeman & Co., New York (NY US).

[Giunchiglia and Shvaiko, 2003] Fausto Giunchiglia and Pavel Shvaiko 2003. Semantic Matching. Technical report, DISI, University of Trento.

[Grau et al., 2005] Bernardo Cuenca Grau, Bijan Parsia, Evren Sirin, and Aditya Kalyanpur 2005. Automatic Partitioning of OWL Ontologie Using E-Connections. In DL 2005, Proceedings of 18th International Workshop on Description Logics, Edinburgh, UK.

[Grau et al., 2006] Bernardo Cuenca Grau, Bijan Parsia, Evren Sirin, and Aditya Kalyanpur 2006. Modularity and Web Ontologies. In Patrick Doherty, John Mylopoulos, and Christopher A. Welty (eds), Proceedings of KR2006 : the 20th International Conference on Principles of Knowledge Representation and Reasoning, Lake District, UK, June 2–5, 2006, pages 198–209.

[Guha et al., 2000] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim 2000. ROCK : A Robust Clustering Algorithm for Categorical Attributes. Information Systems, 25 :345–366.

[Hamdi et al., 2010a] Fayçal Hamdi, Chantal Reynaud, and Brigitte Safar 2010a. A framework for mapping refinement specification. In International Symposium on Matching and Meaning, Leicester, United Kingdom.

[Hamdi et al., 2010b] Fayçal Hamdi, Chantal Reynaud, and Brigitte Safar 2010b. L'approche TaxoMap Framework et son application au raffinement de mappings. In Reconnaissance des Formes et Intelligence Artificielle, Caen, France.

[Hamdi et al., 2010c] Fayçal Hamdi, Chantal Reynaud, and Brigitte Safar 2010c. Pattern-Based Mapping Refinement. In [Cimiano and Pinto, 2010], pages 1–15.

[Hamdi et al., 2010d] Fayçal Hamdi, Chantal Reynaud, and Brigitte Safar 2010d. TaxoMap Framework appliqué à l'alignement d'ontologies géographiques dans le projet GéOnto. In Atelier OntoGeo "Ontologies Géographiques" associé à la conférence SAGEO'10, pages 51–53, Toulouse, France.

[Hamdi et al., 2009a] Fayçal Hamdi, Brigitte Safar, Nobal B. Niraula, and Chantal Reynaud 2009a. TaxoMap in the OAEI 2009 Alignment Contest. In Proceedings of the 4th International Workshop on Ontology Matching (OM-2009) collocated with the 8th International Semantic Web Conference (ISWC-2009) Chantilly, USA, October 25, 2009.

[Hamdi et al., 2011] Fayçal Hamdi, Brigitte Safar, and Chantal Reynaud 2011. Utiliser des résultats d'alignement pour enrichir une ontologie. In Ali Khenchaf and Pascal Poncelet (eds), EGC, volume RNTI-E-20 of *Revue des Nouvelles Technologies de l'Information*, pages 407–412. Hermann-Éditions.

[Hamdi et al., 2010e] Fayçal Hamdi, Brigitte Safar, Chantal Reynaud, and Nobal B. Niraula 2010e. TaxoMap alignment and refinement modules : Results for OAEI 2010. In The Fifth International Workshop on Ontology Matching, Shanghai, Chine.

[Hamdi et al., 2009b] Fayçal Hamdi, Brigitte Safar, Chantal Reynaud, and Haïfa Zargayouna 2009b. Alignment-Based Partitioning of Large-Scale Ontologies. In Fabrice Guillet, Gilbert Ritschard, Djamel Abdelkader Zighed, and Henri Briand (eds), EGC (best of volume), volume 292 of *Studies in Computational Intelligence*, pages 251–269. Springer.

[Hamdi et al., 2009c] Fayçal Hamdi, Brigitte Safar, Haïfa Zargayouna, and Chantal Reynaud 2009c. Partitionnement d'ontologies pour le passage à l'échelle des techniques d'alignement. In Jean-Gabriel Ganascia and Pierre Gançarski (eds), EGC, volume RNTI-E-15 of *Revue des Nouvelles Technologies de l'Information*, pages 409–420. Cépaduès-Éditions.

[Hamdi et al., 2008] Fayçal Hamdi, Haïfa Zargayouna, Brigitte Safar, and Chantal Reynaud 2008. TaxoMap in the OAEI 2008 Alignment Contest. In Proceedings of the 3rd International Workshop on Ontology Matching (OM-2008) Collocated with the 7th International Semantic Web Conference (ISWC-2008), Karlsruhe, Germany, October 26, 2008.

[Hearst, 1992] Marti A. Hearst 1992. Automatic Acquisition of Hyponyms from Large Text Corpora. In COLING, pages 539–545.

[Hovy and Lin, 1998] Eduard Hovy and Chin-Yew Lin 1998. Automated text summarization and the SUMMARIST system. In Proceedings of a workshop on held at Baltimore, Maryland : October 13-15, 1998, TIPSTER '98, pages 197–214, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Hu et al., 2005] Wei Hu, Ningsheng Jian, Yuzhong Qu, and Qanbing Wang 2005. GMO : A Graph Matching for Ontologies. In Proc. K-CAP Workshop on Integrating Ontologies, pages 43–50, Banff (CA).

[Hu et al., 2006] Wei Hu, Yuanyuan Zhao, and Yuzhong Qu 2006. Partition-Based Block Matching of Large Class Hierarchies. In The Semantic Web - ASWC, pages 72–83.

[Huza et al., 2006] Mirella Huza, Mounira Harzallah, and Francky Trichet 2006. OntoMas : a tutoring system dedicated to ontology matching. In Proc. 1st ISWC International Workshop on Ontology Matching (OM), pages 228–323, Athens (GA US).

[Iannone et al., 2008] Luigi Iannone, Mikel Egaña Aranguren, Alan L. Rector, and Robert Stevens 2008. Augmenting the Expressivity of the Ontology Pre-Processor Language. In Proceedings of the Fifth OWLED Workshop on OWL : Experiences and Directions, collocated with the 7th International Semantic Web Conference (ISWC-2008), Karlsruhe, Germany.

[Jaccard, 1901] Paul Jaccard 1901. Étude comparative de la distribution florale dans une portion des Alpes et des Jura. Bulletin del la Société Vaudoise des Sciences Naturelles, 37 :547–579.

[Ji et al., 2009] Qiu Ji, Peter Haase, Guilin Qi, Pascal Hitzler, and Steffen Stadtmüller 2009. RaDON - Repair and Diagnosis in Ontology Networks. In Lora Aroyo, Paolo Traverso, Fabio Ciravegna, Philipp Cimiano, Tom Heath, Eero Hyvönen, Riichiro Mizoguchi, Eyal Oren, Marta Sabou, and Elena Paslaru Bontas Simperl (eds), ESWC, volume 5554 of *Lecture Notes in Computer Science*, pages 863–867. Springer.

[Jian et al., 2005] Ningsheng Jian, Wei Hu, Gong Cheng, and Yuzhong Qu 2005. Falcon-AO : Aligning Ontologies with Falcon. In Proc. K-CAP Workshop on Integrating Ontologies, pages 87–93, Banff (CA).

[Karoui et al., 2004] Lobna Karoui, Marie-Aude Aufaure, and Nacéra Bennacer 2004. Ontology Discovery from Web Pages : Application to Tourism. In Proceedings of the Workshop W6 on Knowledge Discovery and Ontologies, pages 115–120, Pisa, Italie.

[Kefi, 2006] Hassen Kefi 2006. Ontologies et aide à l'utilisateur pour l'interrogation de sources multiples et hétérogènes. PhD thesis, Université Paris-Sud 11.

[Kergosien et al., 2010] Eric Kergosien, Mouna Kamel, Christian Sallaberry, Marie-Noëlle Bessagnet, Nathalie Aussenac-Gilles, and Mauro Gaio 2010. Construction et enrichissement automatique d'ontologie à partir de ressources externes. CoRR, abs/1002.0239.

[Langdon, 2000] William B. Langdon 2000. Natural language text classification and filtering with trigrams and evolutionary nearest neighbour classifiers. Technical report.

[Lee et al., 2007] Yoonkyong Lee, Mayssam Sayyadian, AnHai Doan, and Arnon Rosenthal 2007. eTuner : tuning schema matching software using synthetic scenarios. The VLDB Journal, 16(1) :97–122.

[Levenshtein, 1966] Vladimir Levenshtein 1966. Binary codes capable of correcting deletions, insertions, and reversals. Technical Report 8.

[Li et al., 2009] Juanzi Li, Jie Tang, Yi Li, and Qiong Luo 2009. RiMOM : A Dynamic Multistrategy Ontology Alignment Framework. IEEE Trans. Knowl. Data Eng., 21(8) :1218–1232.

[Lin and Hovy, 2000] Chin-Yew Lin and Eduard Hovy 2000. The automated acquisition of topic signatures for text summarization. In Proceedings of the 18th conference on Computational linguistics - Volume 1, COLING '00, pages 495–501, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Lin, 1998] Dekang Lin 1998. An information-theoretic definition of similarity. In Proc. 15th International Conference of Machine Learning (ICML), pages 296–304, Madison (WI US).

[Maedche and Staab, 2000a] Alexander Maedche and Steffen Staab 2000a. Discovering Conceptual Relations from Text. In ECAI 2000, Proceedings of the 14th European Conference on Artificial Intelligence, Berlin, Germany, August 20-25, pages 321–325.

[Maedche and Staab, 2000b] Alexander Maedche and Steffen Staab 2000b. Mining Ontologies from Text. In Proceedings of the 12th European Workshop on Knowledge Acquisition, Modeling and Management, EKAW '00, pages 189–202, London, UK. Springer-Verlag.

[Maedche and Staab, 2001] Alexander Maedche and Steffen Staab 2001. Ontology Learning for the Semantic Web. IEEE Intelligent Systems, 16(2) :72–79.

[Maedche and Staab, 2002] Alexander Maedche and Steffen Staab 2002. Measuring Similarity between Ontologies. In Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web, EKAW '02, pages 251–263, London, UK. Springer-Verlag.

[Manning and Schütze, 1999] Christopher D. Manning and Hinrich Schütze 1999. Foundations of statistical natural language processing. MIT Press, Cambridge, MA, USA.

[Melnik, 2004] Sergey Melnik 2004. Generic Model Management Concepts and Algorithms. Springer, Heidelberg (DE).

[Mochol et al., 2006] Malgorzata Mochol, Anja Jentzsch, and Jérôme Euzenat 2006. Applying an Analytic Method for Matching Approach Selection. In Ontology Matching.

[Mustière et al., 2011] Sébastien Mustière, Nathalie Abadie, Nathalie Aussenac-Gilles, Marie-Noëlle Bessagnet, M. Kamel, Eric Kergosien, Chantal Reynaud, Brigitte Safar, and Christian Sallaberry 2011. Analyses linguistiques et techniques d'alignement pour créer et enrichir une ontologie topographique. Revue Internationale de Géomatique, 21(2) :155–179.

[Nagy et al., 2008] Miklos Nagy, Maria Vargas-Vera, Piotr Stolarski, and Enrico Motta 2008. DSSim Results for OAEI 2008. In Proceedings of the 3rd International Workshop on Ontology Matching (OM-2008) Collocated with the 7th International Semantic Web Conference (ISWC-2008), Karlsruhe, Germany, October 26.

[Neshatian and Hejazi, 2004] Kourosh Neshatian and Mahmoud R. Hejazi 2004. Text Categorization and Classification in Terms of Multi- Attribute Concepts for Enriching Existing Ontologies. In Proceedings of The Second Workshop on Information Technology and its Disciplines (WITID2004).

[Noy, 2004] Natalya Noy 2004. Tools for mapping and merging ontologies. In Steffen Staab and Rudi Studer (eds), Handbook on ontologies, chapter 18, pages 365–384. Springer Verlag, Berlin (DE).

[Palmisano et al., 2008] Ignazio Palmisano, Valentina A. M. Tamma, Luigi Iannone, Terry R. Payne, and Paul Doran 2008. Dynamic Change Evaluation for Ontology Evolution in the Semantic Web. In Web Intelligence, pages 34–40. IEEE.

[Pammer et al., 2009] Viktoria Pammer, Luciano Serafini, and Stefanie N. Lindstaedt 2009. Highlighting assertional effects of ontology editing activities in OWL. In Proceedings of the 3rd International Workshop on Ontology Dynamics, (IWOD 2009), collocated with the 8th International Semantic Web Conference (ISWC-2009).

[Parekh et al., 2004] Viral Parekh, Jack Gwo, and Timothy W. Finin 2004. Mining Domain Specific Texts and Glossaries to Evaluate and Enrich Domain Ontologies. In Proceedings of the International Conference on Information and Knowledge Engineering. IKE'04, June 21-24, 2004, Las Vegas, Nevada, USA, pages 533–540.

[Qin et al., 2007] Han Qin, Dejing Dou, and Paea Lependu 2007. Discovering executable semantic mappings between ontologies. In Proceedings of the 2007 OTM Confederated international conference on On the move to meaningful internet systems : CoopIS, DOA, ODBASE, GADA, and IS - Volume Part I, OTM'07, pages 832–849, Berlin, Heidelberg. Springer-Verlag.

[Rahm and Bernstein, 2001] Erhard Rahm and Philip A. Bernstein 2001. A survey of approaches to automatic schema matching. The VLDB Journal, 10 :334–350.

[Ritze et al., 2009] Dominique Ritze, Christian Meilicke, Ondrej Sváb-Zamazal, and Heiner Stuckenschmidt 2009. A Pattern-based Ontology Matching Approach for Detecting Complex Correspondences. In Proceedings of the 4th International Workshop on Ontology Matching (OM-2009) collocated with the 8th International Semantic Web Conference (ISWC-2009) Chantilly, USA, October 25, 2009.

[Sabou et al., 2008] Marta Sabou, Mathieu d'Aquin, and Enrico Motta 2008. Exploring the Semantic Web as Background Knowledge for Ontology Matching. J. Data Semantics, 11 :156–190.

[Salton, 1991] Gerard Salton 1991. Developments in automatic text retrieval. Science, 253 :974–979.

[Sayyadian et al., 2005] Mayssam Sayyadian, Yoonkyong Lee, An-Hai Doan, and Arnon Rosenthal 2005. Tuning Schema Matching Software using Synthetic Scenarios. In Proc. 31st International Conference on Very Large Data Bases (VLDB), pages 994–1005, Trondheim (NO).

[Scharffe, 2009] François Scharffe 2009. Correspondence Patterns Representation. PhD thesis, University of Innsbruck, Innsbruck (AT).

[Shannon, 1948] C. E. Shannon 1948. A mathematical theory of communication. Bell system technical journal, 27.

[Shvaiko and Euzenat, 2005] Pavel Shvaiko and Jérôme Euzenat 2005. A Survey of Schema-Based Matching Approaches. In Journal on Data Semantics IV, Lecture Notes in Computer Science, chapter 5, pages 146–171. Springer.

[Shvaiko and Euzenat, 2008] Pavel Shvaiko and Jérôme Euzenat 2008. Ten Challenges for Ontology Matching. In Proceedings of the OTM 2008 Confederated International Conferences, CoopIS, DOA, GADA, IS, and ODBASE 2008. Part II on On the Move to Meaningful Internet Systems, OTM '08, pages 1164–1182, Berlin, Heidelberg. Springer-Verlag.

[Siu and Ostendorf, 2000] Manhung Siu and Mari Ostendorf 2000. Variable n-gram and extensions for conversational speech language modeling. In IEEE Transactions on Speech and Audio Processing, volume 8, pages 63–75.

[Srikant and Agrawal, 1995] Ramakrishnan Srikant and Rakesh Agrawal 1995. Mining Generalized Association Rules. In Proceedings of the 21th International Conference on Very Large Data Bases, VLDB '95, pages 407–419, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[Stoilos et al., 2005] Giorgos Stoilos, Giorgos B. Stamou, and Stefanos D. Kollias 2005. A String Metric for Ontology Alignment. In Yolanda Gil, Enrico Motta, V. Richard Benjamins, and Mark A. Musen (eds), International Semantic Web Conference, volume 3729 of *Lecture Notes in Computer Science*, pages 624–637. Springer.

[Stuckenschmidt and Klein, 2004] Heiner Stuckenschmidt and Michel C. A. Klein 2004. Structure-Based Partitioning of Large Concept Hierarchies. In Sheila A. McIlraith, Dimitris Plexousakis, and Frank van Harmelen (eds), International Semantic Web Conference, volume 3298 of *Lecture Notes in Computer Science*, pages 289–303. Springer.

[Sváb-Zamazal and Svátek, 2009] Ondrej Sváb-Zamazal and Vojtech Svátek 2009. Towards Ontology Matching via Pattern-Based Detection of Semantic Structures in OWL Ontologies. In Proceedings of the Znalosti Czecho-Slovak Knowledge Technology conference.

[Tu and Yu, 2005] Kewei Tu and Yong Yu 2005. CMC : Combining Multiple Schema-Matching Strategies Based on Credibility Prediction. In Proc. 10th International Conference on Database Systems for Advanced Applications (DASFAA), volume 3453 of *Lecture notes in computer science*, pages 888–893, Beijing (CN).

[Velardi et al., 2001] Paola Velardi, Paolo Fabriani, and Michele Missikoff 2001. Using text processing techniques to automatically enrich a domain ontology. In FOIS, pages 270–284.

[Wang and Xu, 2008a] Peng Wang and Baowen Xu 2008a. Debugging Ontology Mappings : A Static Approach. Computing and Informatics, 27(1) :21–36.

[Wang and Xu, 2008b] Peng Wang and Baowen Xu 2008b. Lily : Ontology Alignment Results for OAEI 2008. In Proceedings of the 3rd International Workshop on Ontology Matching (OM-2008) Collocated with the 7th International Semantic Web Conference (ISWC-2008), Karlsruhe, Germany, October 26.

[Winkler, 1999] William E. Winkler 1999. The state of record linkage and current research problems. Technical Report RR/1999/04, Statistics Research Division, U.S. Bureau of the Census.

[Wu and Palmer, 1994] Zhibiao Wu and Martha Palmer 1994. Verb semantics and lexical selection. In Proc. 32nd Annual Meeting of the Association for Computational Linguistics (ACL), pages 133–138, Las Cruces (NM US).

[Xu et al., 2002] Feiyu Xu, Daniela Kurz, Jakub Piskorski, and Sven Schmeier 2002. An Domain Adaptive Approach to Automatic Acquisition of Domain Relevant Terms and their Relations with Bootstrapping. In Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC), Canary island, Spain.

[Zablith et al., 2010] Fouad Zablith, Mathieu d'Aquin, Marta Sabou, and Enrico Motta 2010. Using Ontological Contexts to Assess the Relevance of Statements in Ontology Evolution. In [Cimiano and Pinto, 2010], pages 226–240.

[Zhdanova and Shvaiko, 2006] Anna V. Zhdanova and Pavel Shvaiko 2006. Community-Driven Ontology Matching. In The Semantic Web : Research and Applications, 3rd European Semantic Web Conference, ESWC 2006, Budva, Montenegro, June 11-14, 2006, Proceedings, pages 34–49.