République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Saad Dahleb de Blida 1
Faculté des Sciences
Département d'informatique

Mémoire de fin d'étude

Pour l'obtention du diplôme de Master en *Informatique*

Option : *Ingénierie des Logiciels*

Thème : *Utilisation de la structure d'ontologies pour le passage à l'échelle du problème d'alignement des ontologies.*

Réalisé par :

BERRANI Mohamed Oussama et **BOUDANI Nacereddine**

Président : Mme CHIKHI
Examinateur : Mme BERRAMDANE
Promoteur : Mme FAREH

Soutenu le : **2019/ 2020**

Résumé

L'alignement d'ontologies est une tâche importante pour l'interopérabilité des systèmes puisqu'elle autorise la prise en compte conjointe de ressources décrites par des ontologies différentes, en identifiant des appariements entre concepts.

Avec l'apparition de très grandes ontologies dans des domaines comme la médecine ou l'agronomie, les techniques d'alignement, qui mettent souvent en œuvre beaucoup de calculs , se trouvent face à un défi : passer à l'échelle et faire face à des ontologies volumineuses.

Pour relever ce défi, dans ce mémoire nous proposons une méthode d'alignement qui se base sur un partitionnement d'ontologie en utilisant sa structure. Après l'extraction de différentes partitions des ontologies, le processus d'alignement est lancé pour trouver les concepts équivalents. Nous allons utiliser différentes mesures de similarité en proposant une combinaison de ces dernières pour obtenir une mesure sémantique utilisée dans les algorithmes de la phase d'alignement.

Nous avons montré l'efficacité de notre méthode à travers un ensemble de tests effectués sur différents couples d'ontologies de différentes tailles .

Mots clés : Ontologie, Alignement, partitionnement, Mesure de similarité, sémantique, structure d'ontologie.

Abstract

The alignment of ontologies is an important task in the interoperability of systems since it allows the joint consideration of resources described by different ontologies, which is done by identifying pairings between concepts.

With the emergence of large ontologies in fields such as medicine or agronomy, alignment techniques, which often involve complex calculations, are faced with a challenge to scale up and deal with voluminous ontologies.

To meet this challenge, we propose in this thesis an alignment method that is based on ontology partitioning. We will use an algorithm that relies on the structure of the ontology in the process of clustering. The algorithm will partition the ontology to smaller clusters then verifies if each element is in the right cluster.

The results of the tests carried out with our method on different pairs of ontologies of different sizes show a remarkable decrease in execution time while retaining the precision of the alignment, which indicates great efficiency

Key words : Ontology, Alignment, Partitioning, Similarity , Semantic , Ontology Structure.

ملخص

تعتبر المحاذاة عملية مهمة في وقتنا الحالي، لكن مع تقدم حجم البيانات ومنه ظهور أنطولوجيات ضخمة مثلما هو الحال في مجال الطب والبيولوجيا الخ. أصبحت هذه العملية صعبة بحيث أصبح عائق امامها وهو الكم الهائل من العمليات والبيانات التي يتم تحليلها.

لحل هذه المشكلة سنقدم في هذه الأطروحة طريقة لتطبيق عملية المحاذاة تعتمد هذه الطريقة أساسا على تقسيم الأنطولوجيا إلى مجموعات صغيرة باستعمال شكل هيكلها الذي يعتبر الحل الأساسي لهذه المشكلة بما أنه سيختصر علينا الكثير من الوقت الذي يعتبر عامل أساسي في المحاذاة، سنستعمل قياسات تشابه مختلفة للحصول على نتائج دقيقة بين المصطلحات.

في الأخير لقد أثبتنا فعالية نظامنا وذلك بعمل عدة اختبارات الدقة لعملية المحاذاة وتقسيم الأنطولوجيات المختلفة الأحجام.

الكلمات المفتاحية : الأنطولوجيات، المحاذاة، خوارزمية التجميع، مقاييس التشابه، علم الدلالة.

Remerciement

En tout premier lieu, nous remercions le bon Dieu, tout puissant, de nous avoir donné la force, le courage et la volonté nécessaire pour réaliser ce modeste travail, ainsi que l'audace pour dépasser toutes les difficultés.

Le travail présenté dans ce mémoire a été réalisé sous la direction de Madame **M.Fareh**, notre plus grande gratitude va à notre promotrice, pour sa disponibilité et la confiance qu'elle nous a accordée. Nous avons profité pendant longtemps du savoir et du savoir-faire dont nous avons pu bénéficier au cours de nombreuses discussions. Nous aimerions aussi la remercier pour l'autonomie qu'elle nous a accordée, et ses précieux conseils qui nous ont permis de mener à bien ce travail.

Nous tenons également à remercier **Mr. Riali**, qui a enrichi ce travail avec ses conseils et ses idées surtout concernant la partie d'implémentation.

Enfin nous tenons à exprimer toute notre reconnaissance à nos parents et nos deux familles pour leur soutien constant et leurs encouragements.

Table des matières

Table des figures

Liste des Algorithmes

Liste des tableaux

Introduction générale

Contexte de travail

Une ontologie est «une spécification explicite et formelle d'une conceptualisation partagée» [Gruber, 1993], c'est une représentation qui regroupe un ensemble de concepts et relations décrivant un domaine.

Les ontologies offrent le partage et la réutilisation des connaissances entre les utilisateurs, et cela permet d'avoir une compréhension commune de la connaissance.

Les ontologies se développent à toute vitesse dans différents domaines d'application réels, elles deviennent de plus en plus volumineuses en termes de nombre de concepts (milliers de concepts).Au niveau du passage à l'échelle, le partitionnement d'ontologies en plusieurs partitions est nécessaire pour faciliter l'application de différentes opérations qui peuvent être effectuées sur les ontologies comme le mapping, la fusion et l'intégration.

Le mapping d'ontologies est un paradigme dans le web sémantique où les ontologies sont utilisées pour la représentation de l'information en utilisant un langage de description tel que OWL (Ontology Web Language). Le mapping tente de découvrir les correspondances entre les entités de deux ontologies afin de les combiner et de produire des nouvelles informations.

Problématique

Quand les ontologies sont de très grandes taille, par exemple en agronomie et en médecine, des ontologies comportent plusieurs dizaines de milliers de concepts (AGROVOC : 28439, NALT : 42326, NCI : 27652), l'efficacité des méthodes classiques d'alignement des ontologies diminue considérablement que ce soit en terme de temps d'exécution, de taille mémoire utilisée ou de la précision des résultats obtenu du fait de l'augmentation du bruit.

Peu d'approches d'alignement d'ontologies sont intéressées à la réduction de la complexité d'alignement, et à la réduction de l'espace de recherche des correspondances entre les ontologies d'entrée qui

sont composées d'un grand nombre de concepts, donc comment mettre en œuvre un système d'alignement d'ontologies volumineuses ?

Le partitionnement d'une ontologie en plusieurs partitions sert à faciliter le processus de mapping d'ontologies, ce dernier pose des difficultés qui peuvent se résumer par :

— L'hétérogénéité des ontologies vues la manière dont elles sont développées, à savoir l'hétérogénéité, terminologique, structurelle et sémantique. En effet, l'hétérogénéité sémantique présente un défi majeur dans le processus d'alignement des ontologies, elle est due aux différentes interprétations des objets du monde réel.

— Leur nombre et leur taille qui ne caisse de croitre vu l'augmentation continue des sources d'informations sur le web, d'où l'efficacité des méthodes de mapping diminue considérablement, vu la complexité de ce processus.

La majorité des travaux existants ne traitent pas le passage à l'échelle, et le petit nombre de travaux qui proposent des approches d'alignement d'ontologies volumineuses utilisent le clustering comme une technique pour le passage à l'échelle, cette phase de clustering est basée sur le calcul de mesures de distance qui est représenté par les mesures de similarité, en effet, sur des ontologies volumineuses le calcul de similarité entre tous les concepts des deux ontologies est utilisé pour réaliser le partitionnement, à base de clustering, et d'une manière itérative. Une fois le clustering est réalisé, les mesures de similarité sont utilisées à nouveau pour trouver l'alignement, ceci est très couteux en terme de temps d'exécution et d'espace mémoire.

Donc la question qui se pose, comment réaliser un partitionnement sans calculer les mesures de similarité entre les concepts ?

Objectif

L'objectif de ce projet consiste à concevoir et de réaliser un système d'alignement d'ontologies, étant données deux ontologies représentées par le langage OWL (Ontology Web langage). En effet, le problème du passage à l'échelle a une très grande importance. Une solution possible pour résoudre ce problème est d'essayer de limiter la taille des ensembles de concepts en entrée, et pour cela de partitionner les ontologies en plusieurs blocs, et ce, en utilisant la structure des ontologies initiales, afin de n'avoir à traiter que des blocs de taille raisonnable.

Notre objectif est de partitionner l'ontologie en blocs sans utiliser les mesures de similarité, afin

d'obtenir que des blocs à aligner. L'alignement s'effectue en utilisant une combinaison des mesures de similarité afin de trouver les correspondances entre des deux ontologies.

Organisation

Ce mémoire est organisé en 4 chapitres :

Dans le premier chapitre, nous avons abordé une étude sur les ontologies, ses composants, les langages utilisés pour la représentation des ontologies, nous avons aussi parlé du probléme du mapping .

Le deuxième Chapitre, commence par l'introduction de partitionnement, nous présentons par la suite les différentes mesures de similarité, puis nous enchainons avec un état de l'art sur les méthodes d'alignement d'ontologies et une comparaison entre ces dernières.

Dans le chapitre 3, nous avons présenté notre solution proposée pour le problème de passage à l'échelle, avec une description détaillée des deux phases de partitionnement et d'alignement.

Le chapitre 4 est consacré à l'implémentation de notre solution proposée, nous présentons les différents outils utilisés, ensuite, nous abordons l'évaluation de notre système selon le paramètre de précision pour évaluer la qualité du mapping, et du paramètre de séparation pour évaluer le partitionnement.

La conclusion de ce mémoire synthétise les principales propositions dans notre système, et dégage quelques perspectives de ce travail.

Chapitre 1

Ontologies et Mapping des ontologies.

1.1 Introduction

Les ontologies proposent des représentations sémantiques des connaissances, susceptibles d'être manipulées par les machines. Elles participent dans les dimensions scientifiques et techniques du domaine de l'Intelligence Artificielle et en particulier dans la branche de l'ingénierie des connaissances.

Le champ d'application des ontologies ne cesse de s'étendre et couvre les systèmes conseillers (systèmes d'aide à la décision, e-learning, etc.), les systèmes de résolution de problèmes et les systèmes de gestion et la manipulation des connaissances (dans plusieurs domaines : agricole, médical et biologique). Le principal objectif des ontologies est de doter le web d'une couche sémantique assurant la recherche d'informations aussi bien sur le niveau syntaxique qu'au niveau sémantique. L'objectif est de créer un Web intelligent permettant de se rendre compte de la sémantique à travers la prise en charge du sens des informations par l'intermédiaire des ontologies.

Dans ce chapitre, nous introduisons d'abord la notion des ontologies, puis nous énonçons quelques principes fondamentaux d'ingénierie ontologique, classes d'ontologies et les langages de représentation les plus connus et les plus utilisés.

1.2 Notion d'Ontologie

1.2.1 Définition

Le terme ontologie est emprunté de la philosophie d'où il fait référence à la science qui « étudie l'être en tant qu'être ». Avec l'émergence de l'ingénierie des connaissances et du web sémantique, ce terme a pris une toute autre tournure pour désigner la problématique de représentation et de manipulation des connaissances dans un système informatique. D'après Gruber les ontologies sont définies

comme suit «Une ontologie est une spécification explicite et formelle d'une conceptualisation d'un domaine de connaissance» [Gruber, 1993].

1.2.2 Composants d'ontologie

Les ontologies rassemblent les connaissances propres à un domaine donné. En représentant des connaissances, ces ontologies existent sous la forme de concepts et de relations, et permettent d'en fixer la sémantique selon un degré de formalisme variable. Nous détaillons ci-dessous les différents composants de l'ontologie.

1. **Concepts** Un concept peut représenter un objet, une notion, une idée. Un concept peut être divisé en trois parties : un terme (ou plusieurs), une notion et un ensemble d'objets.

   (a) **Le terme :** Le terme est un élément lexical qui permet d'exprimer le concept en langue naturelle, il peut admettre des synonymes.

   (b) **L'intension :** La notion également appelée intension du concept, contient la sémantique du concept, exprimée en termes de propriétés et attributs, et de contraintes.

   (c) **L'extension :** L'extension du concept, regroupe les objets manipulés à travers le concept ; ces objets sont appelés instances du concept. Par exemple, le terme « table » renvoie à la fois à la notion de table comme objet de type « meuble » possédant un « plateau » et des « pieds », et à l'ensemble des objets de ce type.

2. **Relations** Certains liens conceptuels existant entre les concepts peuvent s'exprimer à l'aide de propriétés portées par les concepts, d'autres doivent être représentés à l'aide de relations autonomes. Une relation permet de lier des instances de concepts, ou des concepts génériques. Elles sont caractérisées par un terme (voire plusieurs) et une signature qui précise le nombre d'instances de concepts que la relation lie, leurs types et l'ordre des concepts, C'est–à–dire la façon dont la relation doit être lue. Par exemple, la relation « écrit » lie une instance du concept «personne» et une instance du concept «texte», dans cet ordre.

3. **Fonctions** Les fonctions constituent des cas particuliers de relations, dans laquelle un élément de la relation, le nième (extrant) est défini en fonction des n-1 éléments précédents (intrants).

4. **Axiomes** Les axiomes constituent des assertions acceptées comme vraies à propos des abstractions du domaine traduit par l'ontologie. Interviennent dans la définition des concepts ou des relations, dans l'inférence de nouvelles informations

5. **Instances** Les instances constituent la définition extensionnelle de l'ontologie ; ces objets véhiculent les connaissances (statiques, factuelles) à propos du domaine du problème.

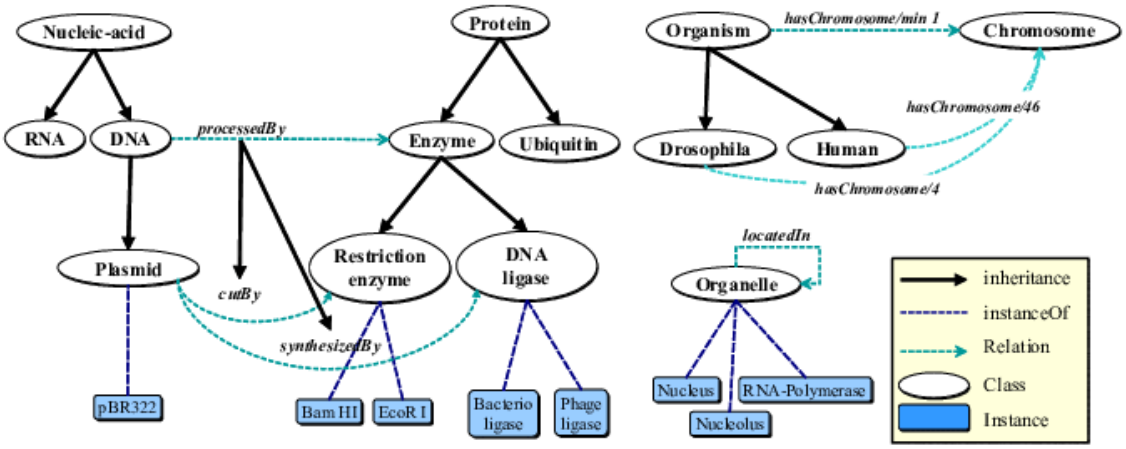

FIGURE 1.1 – Exemple d'une ontologie [Ki-Heon et al., 2010]

1.2.3 Principe d'ingénierie ontologique

Il existe un ensemble de critères et de principes déjà prouvés dans le développement des ontologies :

1. **Clarté et Objectivité** [Gruber, 1993] : L'ontologie doit fournir la signification des termes définis en fournissant des définitions « objectives » ainsi qu'une documentation en langage naturel.
2. **Cohérence** [Gruber, 1993] :Pour permettre des inférences conformes aux définitions.
3. **Maximiser l'extensibilité monotone** [Gruber, 1993] : L'ajout de nouveaux concepts à l'ontologie ne devrait pas entraîner la révision des définitions existantes.
4. **Diversification des hiérarchies** [Arpirez et al., 1998] pour augmenter la puissance fournie par les mécanismes d'héritage multiple.
5. **Minimiser la distance sémantique** [Arpirez et al., 1998] Minimiser la distance sémantique entre les concepts enfants de mêmes parents.

1.2.4 Classification des Ontologies

Nous présentons par la suite différentes classifications des ontologies selon plusieurs critères :

1. **Selon l'objet de conceptualisation :** On distingue quatre types d'ontologie :
   - **Ontologie Générique :** Elle est appelée également noyau ontologique ou méta-ontologie, modélise des connaissances assez générales néanmoins pour être réutilisées à travers différents domaines.
   - **Ontologie du domaine :** Cette ontologie exprime des conceptualisations spécifiques à un domaine, elle fournit les concepts et les relations permettant de couvrir les vocabulaires, activités et théories de domaines.
   - **Ontologie de Tâches :** Fournit un vocabulaire systématisé des termes employés pour résoudre des problèmes liés aux tâches.
   - **Ontologie d'application :** Elle contient des concepts dépendants d'un domaine et d'une tâche particulière, ces concepts correspondent souvent aux rôles joués par les entités du domaine lors de l'exécution d'une certaine activité.
2. **Selon le niveau de détail :** Cette classification est selon le niveau de détail des objets de la conceptualisation.
   - **Granularité fine :** Correspondant à des ontologies très détaillées, possédant ainsi un vocabulaire plus riche capable d'assurer une description détaillée des concepts pertinents d'un domaine ou d'une tâche [Furst, 2004].
   - **Granularité large :** Correspondant à un vocabulaire moins détaillé.
3. **Selon le niveau de Complétude :** Selon [Bachimont, 2000] ils existent trois niveaux de complétude :
   - **Niveau sémantique :**Tous les concepts, caractérisés par un terme/libellé, doivent respecter les quatre principes différentiels :
     - Communauté avec l'ancêtre .
     - Différence, spécification, par rapport à l'ancêtre .
     - Communauté avec les concepts frères, situés au même niveau .
     - Différence par rapport aux concepts frères.

     Ces principes correspondent à l'engagement sémantique et assurent que chaque concept aura un sens univoque et non contextuel associé. Deux concepts sont identiques si l'interprétation du terme/libellé à travers les quatre principes différentiels aboutis à un sens

équivalent.

— **Niveau-Référentiel :** Les concepts référentiels ou formels, se caractérisent par un terme/libellé dont la sémantique est définie par une extension d'objets. L'engagement ontologique spécifie les objets du domaine qui peuvent être associés au concept, conformément à sa signification formelle. Deux concepts formels seront identiques s'ils possèdent la même extension.

— **Niveau-Opérationnel :** Les concepts du niveau opérationnel ou computationnel sont caractérisés par les opérations qu'il est possible de leur appliquer pour générer des interfaces ou engagement computationnel.

1.2.5 Construction d'une ontologie

La construction d'ontologie est décomposée en 3 étapes : conceptualisation, ontologisation, et opérationnalisation, comme il montre la figure 1.1.

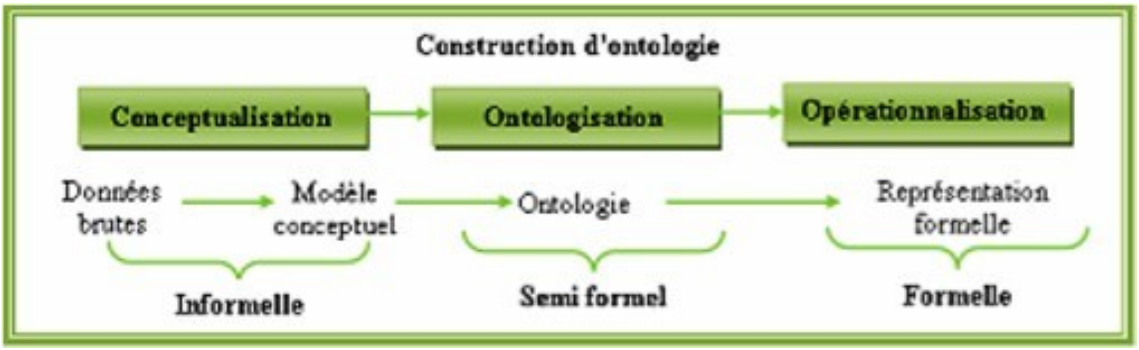

FIGURE 1.2 – Construction d'ontologie [Furst, 2004]

1. **Etape de conceptualisation** identification des connaissances contenues dans un corpus représentatif du domaine .
2. **Etape d'ontologisation** formalisation, autant que possible, du modèle conceptuel obtenu à l'étape précédente .
3. **Etape d'opérationnalisation** transcription de l'ontologie dans un langage formel et opérationnel de représentation de connaissances.

1.3 Langages d'ontologies

Différents langages ont étés conçues pour la construction des ontologies. Parmi ces langages, nous citons :

1.3.1 RDFS (Resource Description Framework Schema)

RDFS est un langage de spécification de schémas associé à RDF. RDFS a pour but d'organiser les objets du web sous forme d'une hiérarchie. Il était introduit comme étant une couche au-dessus de RDF.

1.3.2 OWL (Ontology Web Language)

OWL est une recommandation de W3C(World Wide Web Consortium) depuis février 2004.Il s'agit d'une extension du langage RDF et RDFS. OWL est un langage basé sur la syntaxe XML, très expressif et avec un vocabulaire très riche. Il permet de définir des ontologies de domaines complexes, par exemple, en spécifiant les relations hiérarchiques entre les classes et les restrictions et en plus il permet de faire du raisonnement pour déduire des informations qui ne sont pas explicitement présents dans l'ontologie. Avec OWL, nous pouvons définir :

- La subsomption entre classes et relations.
- Les domaines et les cardinalités des relations.
- Définition des concepts par énumération des instances.
- Définition des concepts par la combinaison des opérations : union, intersection et complément.
- Définition des propriétés et les cardinalités.
- Définition des contraintes en logique du 1er ordre.

1.4 L'alignement des ontologies

L'alignement d'ontologies consiste à chercher les concordances entre les concepts, les relations et les individus des diverses ontologies. Le but c'est de trouver les points de jonctions (les entités en commun.) Figure 1.3 qui permettront de concevoir des ponts entre ces ontologies. L'alignement est considéré comme une technique à base de toute autre opération de réconciliation entre les ontologies [Touzani, 2005].

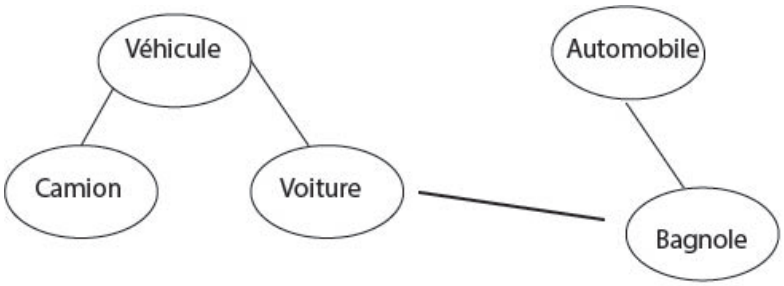

FIGURE 1.3 – Alignement d'ontologies

1.5 Le Mapping des ontologies

Le mapping d'ontologies est un nouveau paradigme dans le web sémantique où les ontologies sont utilisées pour la représentation d'information en utilisant un langage de description tel que OWL (Ontology Web Language). Le mapping d'ontologies peut être vu comme un processus qui relie sémantiquement deux vocabulaires différents [Euzenat et al., 2007]. Il tente de découvrir les correspondances sémantiques Figure 1.4 et les relations implicites qui peuvent exister entre deux ou plusieurs ontologies en appliquant des mesures de similarité.

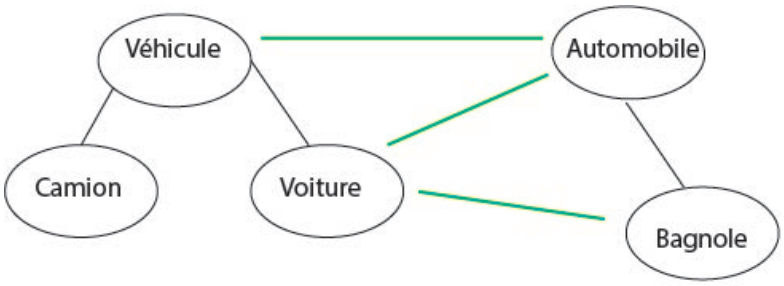

FIGURE 1.4 – Mapping d'ontologies

1.6 Dimensions de l'alignement

L'alignement regroupe trois dimensions : l'input, le processus d'alignement et l'output.

1. **L'input** Est constitué essentiellement des structures destinées à être alignées et qui peuvent être, comme énoncé précédemment, des schémas XML, des schémas relationnels, des ontologies décrites en OWL, RDFS...etc, ou des instances d'une ontologie.

   L'input peut être enrichi par un alignement en entrée (qui aurait besoin d'être complété par une nouvelle itération d'alignement).

2. **Le processus d'alignement**

Comme le montre Figure 1.5 le processus d'alignement peut être considérée comme une fonction f , qui donne un alignement A' entre une paire d'ontologie O et O' à partir de ces deux derniers, un alignement en entrée A (optionnel), un ensemble de paramètre p (ex : paramètres de pondération, seuils ...)et un ensemble de ressources externes r (ex : thésaurus, lexique, etc.), A'= f(O,O',A,p,r).

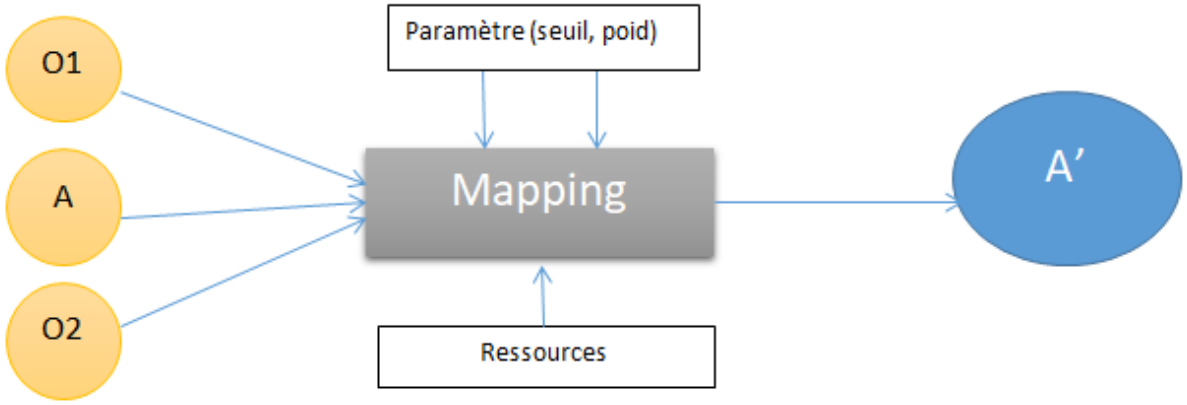

FIGURE 1.5 – Processus d'alignement [Euzenat et al., 2007]

3. **L'output** Est un ensemble d'alignement reliant les entités qui composent les deux ontologies. Un alignement est décrit par un ensemble de cinq éléments <id, e, é , r, n > telle que :
   - **id :** identifiant unique d'un Mapping .
   - **e :** une entité, à aligner, appartenant à O (class, propriété, contraint, instance) .
   - **é :** une entité, à aligner, appartenant à O' .
   - **r :** la relation qui relie e à é .
   - **n :** la mesure de confiance de la relation r est généralement une valeur réelle compris entre 0 et 1, Plus le n est proche du 1, plus la relation est considérée comme étant forte.

1.7 Problèmes de mapping

Dans les systèmes distribués et ouverts , comme le Web sémantique et beaucoup d'autres applications, le problème de l'hétérogénéité des sources d'information est inévitable. Différents concepteurs ont différents intérêts et vues, ils utilisent différents outils et connaissances et avec différents niveaux de details.

Le mapping d'ontologies permet de réduire l'hétérogénéité entre les ontologies. Cette hétérogénéité ne réside pas seulement sur l'objectif dont ces dernières sont conçues mais aussi sur le forma-

lisme et la terminologie utilisée pour les encoder. La variation des raisons d'hétérogénéité à ramener différentes formes d'hétérogénéité. Plusieurs classifications de l'hétérogénéité ont été proposées, parmi lesquelles nous citons :

1. **Hétérogénéité syntaxique** Elle se produit lorsque nous comparons deux ontologies écrites dans deux langages ou modélisées par des formalismes de représentation de connaissances différents [Euzenat, 2008].

   Pour pallier ce problème, des outils sont développés pour transformer une ontologie d'un formalisme vers un autre mais il existe toujours un risque de ne pas préserver le sens et le but de l'ontologie. Les différences que nous pouvons soulever au niveau du langage de représentation sont des différences dans les expressions et la sémantique des langages.

   D'après [kalfoglou et al., 2003], il est facile de comparer des concepts s'ils sont exprimés dans le même langage.

   Un autre exemple d'hétérogénéité est la représentation des instances [Tang, 2006]. Une information peut être présentée sous différentes formes, par exemple la date peut être définie sous le format 12/10/2010 ou sous le format Déc.12. 2010. Un nom peut aussi être représenté sous la forme " Jackson Michael " ou bien sous forme " Michael Jackson"

2. **Hétérogénéité terminologique** Le problème majeur dans le mapping est surtout celui de la terminologie car nous pouvons avoir une multitude de manières de représenter des données similaires et parfois des conflits sur les appellations. A titre d'exemple, l'utilisation de termes linguistiques identiques pour désigner différents concepts ou relation ou l'utilisation de différents termes pour désigner un même concept ou une même relation [Stephen et al., 2002].

   Il y a aussi le problème de polysémie où un concept peut avoir différents sens. Ce type d'hétérogénéité pose, par conséquence, une grande difficulté. Pour diminuer cette difficulté, des ressources externes sont utilisées comme les dictionnaires, les thésaurus etc.

3. **Hétérogénéité conceptuelle** Elle englobe les différences dans la modélisation du même domaine d'intérêt. D'après l'étude faite dans [Euzenat et al., 2007] trois types d'hétérogénéité conceptuelle peuvent se produire selon le degré de couverture de domaine et le but pour lequel l'ontologie est conçue :

   (a) Différence dans l'assurance : Plusieurs ontologies peuvent décrire le même domaine mais avec des manières différentes et avec le même degré de granularité.

(b) Différence dans la granularité : De multiple ontologies peuvent présenter le même domaine mais avec des niveaux de détails différents. Dans ce cas, nous pouvons trouver une ontologie O2 est définie comme une spécialisation d'une ontologie O1.

(c) Différence dans le but : Chaque ontologie a un objectif à atteindre et la différence se produit lorsque les ontologies décrivent le même domaine avec le même niveau de granularité mais pour des objectifs différents.

4. **Hétérogénéité sémantique** Elle concerne la façon dont les entités sont interprétées par les personnes. En effet, des entités qui ont exactement la même sémantique sont souvent interprétées différemment selon le contexte d'utilisation. Par exemple, la façon dont elles sont utilisées. Ce type d'hétérogénéité est très difficile à détecter par la machine et difficile à résoudre.

1.8 Conclusion

Dans ce premier chapitre, nous avons donné une vision générale des ontologies, leurs caractéristiques et leurs langages de représentations, nous avons également abordé l'alignement et le mapping des ontologies.

Dans le chapitre suivant nous allons présenter des méthodes d'alignement d'ontologies qui existent dans la littérature.

Chapitre 2

Méthodes existantes pour l'alignement d'ontologies

2.1 Introduction

Ce chapitre a pour premier objectif d'aborder les techniques d'alignement dans le domaine du web sémantique en présentant quelques critères sur lesquels une comparaison doit s'appuyer pour pouvoir les classifier.

Dans ce chapitre nous commençons par définir le partitionnement et les différentes mesures de similarité, ensuite nous présentons les différentes approches et méthodes qui existent pour l'alignement des ontologies. A la fin nous comparons et analysons ces systèmes.

2.2 Partitionnement d'ontologies

Le partitionnement d'ontologies peut être utilisé dans des applications telles que l'alignement d'ontologies, la fusion d'ontologies et la synthèse de texte basée sur l'ontologie.

Le partitionnement d'ontologies divise une ontologie en un ensemble de sous-ensembles, chaque sous-ensemble étant appelé une partition.

Le partitionnement d'ontologies est généralement applicable pour diviser de grandes ontologies et agir sur des sous-ontologies pour augmenter la qualité des partitions, dont le but est de rendre l'application de différentes opérations sur ces ontologies pratique et efficace.

La figure suivante représente un partitionnement avec k=2, les concepts de cette ontologie ont été partitionnés en 2 parties .

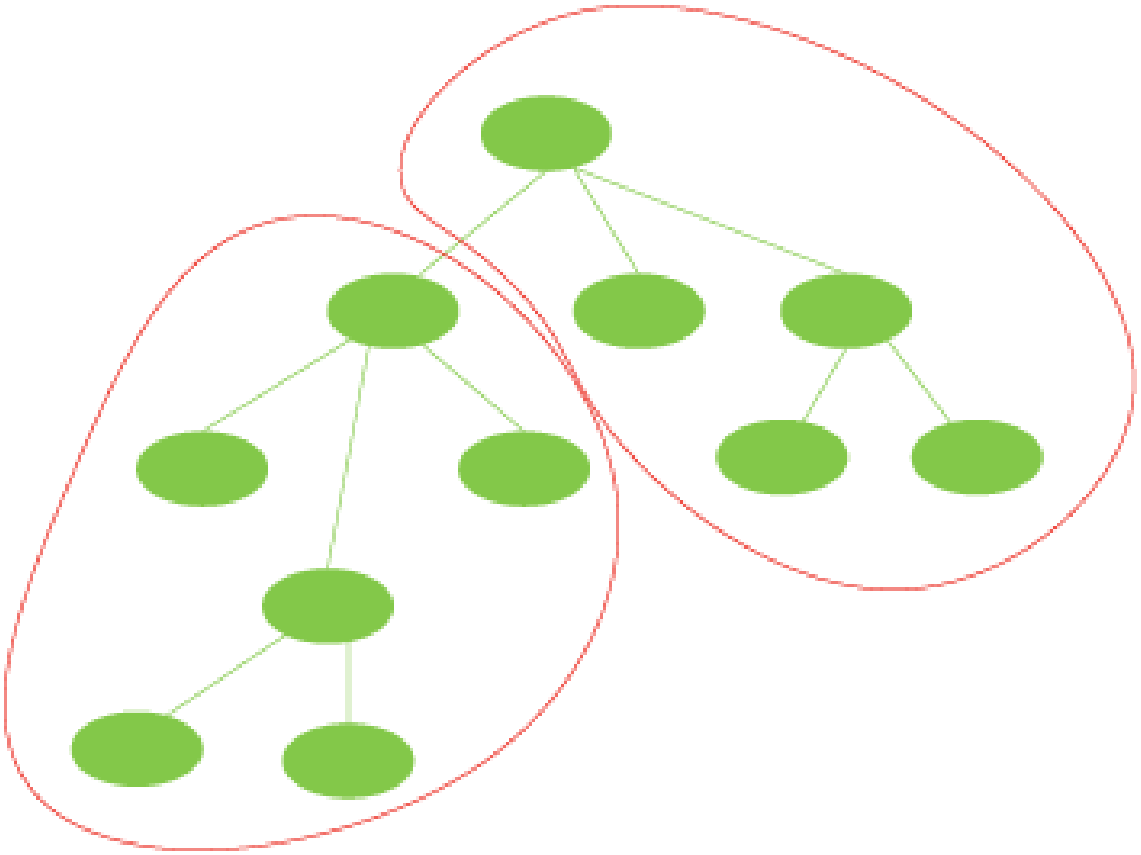

FIGURE 2.1 – Partitionnement d'une ontologie en 2 groupes

2.3 La similarité

La similarité est un concept important et largement utilisé, qui est lié à un domaine d'application particulier ou à une forme de représentation des connaissances, on peut trouver en psychologie ou en mathématiques. En psychologie sociale, la similarité se rapporte à comment les attitudes, les valeurs, les intérêts et la personnalité correspondent entre les personnes. En mathématiques, plusieurs relations d'équivalence sont appelées similarité, En topologie, la similarité est une fonction telle que sa valeur est plus grande quand deux points sont plus proches (contrairement à la distance, qui est une mesure de dissimilarité : plus les points sont proches, plus la distance est petite). Dans notre contexte, la notion de similarité sémantique est vue comme celle de la similarité topologique en mathématiques, où on l'associe à une fonction, appelée fonction de la similarité.

2.3.1 Mesures de similarité

La Figure 2.2 résume différentes mesures de similarité, catégorisées selon les techniques utilisées. Ce résumé est une synthèse des travaux présentés dans [Rahm et al, 2001] , [Euzenat et al., 2004] et [Shvaiko et al., 2005].

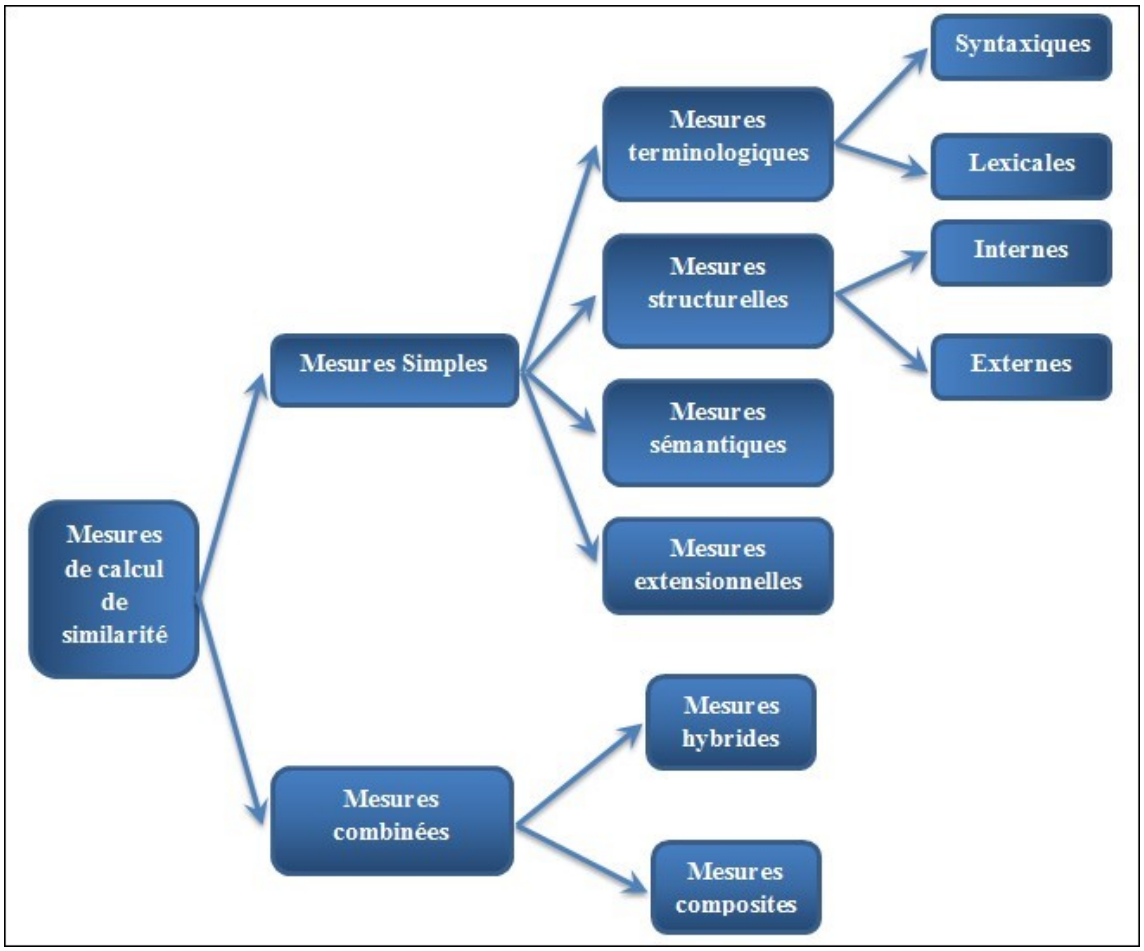

FIGURE 2.2 – Mesures de calcul de similarité [Euzenat et al., 2004]

Les différentes mesures de similarité utilisées dans le processus d'alignement sont organisées selon la classification suivante :

1. **Mesures Simples** Elle regroupe les mesures suivantes :

   (a) **Mesures terminologiques** Ces mesures reposent sur la comparaison des termes ou des chaînes de caractères ou bien les textes. Elles sont exploitées pour calculer la valeur de la similarité des entités textuelles, telles que des noms, des étiquettes, des commentaires, des descriptions, etc. Nous trouvons dans cette méthode deux approches : l'approche syntaxique et l'approche lexicale, appelée aussi linguistique [Monge et al., 1996].

   — **Approche syntaxique** approche basée sur les chaînes de caractères (texte) : comme leur nom l'indique, ces méthodes considèrent l'entité comme une séquence de lettres. Les résultats obtenus par ces méthodes sont utiles si les concepteurs utilisent des chaines de caractères similaires pour définir la même entité, en revanche, s'il y a des synonymes avec des structures différentes ces méthodes donnent une mauvaise esti-

mation de la similarité.

— **Approche lexicale ou linguistique** : les informations exploitées peuvent être celles intrinsèques (des propriétés linguistiques internes des termes telles que des propriétés morphologiques ou syntaxiques) ou celles extrinsèques en effectuant la correspondance à travers les relations lexicales (par exemple, synonymie, hyponymie, etc.) en faisant appel à des ressources auxiliaires telles que les dictionnaires de synonymes et hyponymes, mais aussi à des thésaurus et des ressources sémantiques ainsi qu'a des dictionnaires spécifiques aux domaines étudiés.

—

(b) **Mesures structurelles** Elles se focalisent sur la comparaison des structures des entités à mapper, en fonction de la nature interne ou externe des structures à comparer, on peut distinguer :

— **Méthodes structurelles internes** Ces méthodes utilisent la comparaison de structure interne d'une entité (ex : la comparaison des attributs, des noms, des types des attributs...) indépendamment des autres entités.

— **Méthodes structurelles externes** qui utilisent la comparaison externe en mettant par exemple en jeux la disposition des entités dans leur hiérarchie, le voisinage, ...

(c) **Mesures sémantiques** La similarité sémantique est une évaluation du lien sémantique entre deux concepts dont le but est d'estimer le degré par lequel les concepts sont proches dans leur sens. La similarité entre deux concepts est liée aux caractéristiques qu'ils ont en commun (plus ils ont de caractéristiques communes, plus les concepts sont similaires) et à leurs différences (plus deux concepts sont différents, moins ils sont similaires). La similarité maximale est obtenue lorsque deux concepts sont identiques [Mellal, 2007].

(d) **Mesures extensionnelles** Elles résultent de la similarité entre deux entités qui sont notamment des concepts ou des classes tout en analysant ainsi leurs extensions (leurs ensembles d'instances). Chaque instance peut être représentée par un vecteur de noms et/ou de valeurs. Des calculs de similarités entre vecteurs permettent de comparer les instances [Ziani et al., 2010].

On distingue deux approches pour comparer les ontologies à partir des instances associées aux concepts d'ontologies :

— Soit les deux ontologies à comparer référencent les mêmes instances et dans ce cas on

génère une similarité entre les concepts qui partagent les mêmes instances.

— Soit les deux ontologies à comparer ne référencent pas les mêmes instances et dans ce cas on fait des recherches par mots-clés dans les instances. La similarité est ensuite calculée entre les instances à l'aide de ces mots-clés.

2. **Mesures combinées** Ces mesures combinent plusieurs mesures lorsqu'une seule est insuffisante [Leacock et al., 1998].Il existe deux types de combinaison :

   (a) **La combinaison séquentielle (hybride)**La méthode la plus simple pour combiner les mesures est l'utilisation séquentielle de ces dernières en choisissant un ordre d'exécution. Par exemple, nous choisissons de lancer une mesure terminologique avant de lancer une autre mesure structurelle ou sémantique [Elbyed, 2009].

   (b) **La combinaison parallèle (composite)** Une autre manière de combiner les résultats des différentes mesures (c.-à-d. les valeurs de similarité) consiste tout d'abord à lancer parallèlement plusieurs mesures, puis par la suite à combiner leurs résultats [Elbyed, 2009].

2.3.2 Bilan sur les mesures de similarité

— Les techniques présentées sont exécutées dans le processus d'alignement et considérées comme des éléments clef pour l'alignement d'ontologies.

— Les résultats de ces méthodes sont composés ou agrégés selon la stratégie utilisée pour avoir l'ensemble de correspondances entre les ontologies en entrée.

— Elles peuvent être exécutées séquentiellement ou indépendamment les unes des autres. Cependant, quel que soit la méthode de mise en correspondance appliquée, elle doit exploiter les caractéristiques des ontologies en entrée.

2.4 Systèmes d'alignement existants

Les techniques d'alignement sont exploitées dans beaucoup de systèmes qu'ils revendiquent leur contribution à la résolution du problème de l'alignement d'ontologies. Les travaux de [Bach, 2006], [Euzenat, 2007], [David et al., 2006] et [Stephen et al., 2002] fournissent des bons états de l'art sur les systèmes d'alignement proposés avant 2006. Dès 2008, l'initiative d'évaluation ,OAEI donne chaque année une évaluation des meilleurs systèmes d'alignement. Nous présentons dans cette section quelques systèmes proposés dans la littérature tels que :

2.4.1 GLUE [Doan et al., 2002]

Est la version évoluée de LSD (Doan et al, 2000) dont le but est de trouver semi automatiquement des correspondances entre des schémas pour l'intégration de données. Comme LSD, GLUE utilise la technique d'apprentissage (telle que Naïve Bayes) pour trouver des correspondances entre deux ontologies. GLUE comprend plusieurs modules d'apprentissage (learners), qui sont entraînés par des instances des ontologies.

Ces modules emploient la technique d'apprentissage Bayes naïf (Domingos et al, 1997) en exploitant différentes caractéristiques des instances telles que les valeurs textuelles des instances, les noms des instances, les formats des valeurs.... Les prévisions de ces modules de mise en correspondance sont combinées par un méta module de mise en correspondance en employant la somme pondérée. Le résultat final des correspondances sera déduit à partir des valeurs de similarité agrégées en employant la technique d'optimisation de contraintes « relaxation labeling » (la technique permettant de résoudre le problème d'assignement des étiquettes aux nœuds d'un graphe en donnant un ensemble de contraintes).

Un inconvénient de cette approche est qu'elle se fonde principalement sur les instances des ontologies, qui ne sont pas toujours abondamment disponibles pour plusieurs ontologies. Un autre inconvénient est que l'ontologie est modélisée comme une taxonomie des concepts et que chaque concept a quelques attributs. Avec cette organisation, GLUE n'emploie pas des informations contenues dans la taxonomie (hiérarchie) des relations. GLUE fait également l'utilisation modeste des informations sur la taxonomie des concepts.

2.4.2 AROMA [David et al., 2006]

AROMA (Association Rule Ontology Matching Approach) est une approche d'alignement d'ontologies représentées en OWL. Elle permet de trouver les correspondances sémantiques de type « subsomption » ou « équivalence » entre deux entités des ontologies alignées. L'approche AROMA suit un processus d'alignement qui se déroule en trois étapes :

1- L'acquisition des termes contenus dans des descriptions et instances des entités à partir d'outils de traitement automatique du langage (TAL) et l'association d'un ensemble de termes représentatifs pour chaque entité.

2- La création des relations de subsomption entre les entités à partir des règles d'association, une règle d'association admissible, signifie que le vocabulaire associé à une entité source tend à être inclus

dans le vocabulaire de l'entité cible. Si une règle est également significative, le système retiendra une relation d'implication entre ces deux entités.

3- et enfin l'analyse de règles d'association afin de déduire des relations d'équivalence à partir de l'alignement implicatif, éliminer les incohérences, supprimer les relations redondantes, et sélectionner le meilleur alignement pour chaque entité. La version 1.16 de cette approche est évaluée dans OAEI. AROMA a l'avantage de tirer profit à la fois des descriptions intentionnelles et extensionnelles d'une hiérarchie afin d'extraire des alignements consistants et minimaux, et de détecter les relations d'implication.

2.4.3 Taxo Map [Kasri, 2010]

TaxoMap est un outil d'alignement qui a pour objectif de permettre un accès unifie via le Web aux documents d'un même domaine d'application. Il est adapté au traitement de taxonomies dont les structures sont hétérogènes et dissymétriques. L'objectif de TaxoMap est de mettre en correspondance les concepts de la taxonomie la moins structurée, la taxonomie source.

2.4.4 COMA++ [Massmann et al., 2011]

Un système dédié à l'alignement des graphes de schémas larges. La méthode proposée se base sur l'idée 'diviser pour régner'. Une ontologie est décomposée en des clusters (appelés fragments) qui seront, par la suite, mis en correspondance. Le clustering de l'ontologie est basé sur un ensemble de règles heuristiques prédéfinies pour la production d'un ensemble de clusters de taille réduite. Chaque cluster est identifié par un concept racine qui sera utilisé pour la comparaison des clusters dans la phase d'alignement de clusters. En effet, deux clusters dont les deux concepts racine sont sémantiquement similaires, ceci implique la similarité des deux clusters. L'alignement sera ensuite mené sur les entités appartenant aux clusters jugés similaires en utilisant la technique structurelle et terminologique.

2.4.5 Travail de [Kachroudi et al., 2013]

Les auteurs proposent également une méthode initiée par les concepts terminologique ment équivalents dans les deux ontologies à aligner. Le concept initié dans l'ontologie cible est considéré comme étant le centre d'un nouveau block. Ainsi, les concepts sémantiquement et structurellement similaires au centre seront rajoutés dans le cluster. A cette fin, les auteurs proposent d'utiliser la ressource externe WordNet pour examiner le voisinage de chaque concept.

Pour chaque concept, on identifie la liste des termes qui lui sont sémantiquement équivalents à partir de WordNet. De ce fait, chaque concept similaire sera assigné au cluster d'intérêt. Ceci permet d'assurer la cohésion des clusters produits. Le même traitement sera répété pour chaque concept rajouté au cluster. Le processus s'arrête lorsqu"il n'y aura aucun lien sémantique entre les concepts de cluster et les concepts restants ou lorsque la taille du cluster atteint une valeur maximale.

Une fois que le partitionnement de l'ontologie cible est achevé, l'auteur crée le même nombre de clusters dans l'ontologie source. La phase d'alignement finale consiste à aligner les entités des clusters correspondants aux ancres (les concepts des ontologies source et cible).

2.4.6 Travail de [Algergawy et al., 2014]

Le système d'alignement proposé est initié par la phase de clustering d'ontologies où chaque ontologie est clustérisée indépendamment en un ensemble des clusters disjoints.

La tâche de clustering se base sur une mesure de similarité structurelle afin d'identifier les éléments structurellement similaires. Chaque concept sera assigné à un cluster, ensuite un algorithme itératif sera déclenché. Ce dernier consiste à fusionner les clusters structurellement similaires. Le processus s'arrête lorsque la taille du cluster atteint une valeur maximale. La deuxième phase, consiste à identifier les clusters source et cible similaires.

A cette fin, chaque cluster est converti en un vecteur de termes. Une fois les clusters similaires sont identifiés, on procède à l'alignement des éléments correspondants.

2.5 Comparaison de méthodes d'alignement

Nous présentons un tableau récapitulatif (tableau 2.1) de notre analyse des méthodes d'alignement des ontologies en prenant comme critères les entrées du système, cardinalité, mesure et technique de similarité, automatisation et est-ce que cette méthode supporte le passage à l'échelle et la technique utilisée :

Méthode d'alignement	Entrée du système	Cardinalité	Mesure de similarité	Technique de similarité	Automatisation	Passage à l'échelle	Technique de passage à l'échelle
GLUE **[Doan et al., 2002]**	Schéma, bases de données relationnelles, Taxonomie Xml	(1,1)	-Terminologique -Extensionnelle -Structurelle	-Tokenisation -String-comparaison pour des tokens	Semi-Automatique	/	/
AROMA **[David et al., 2006]**	OWL,RDF	(0,n)	Terminologiques Structurelle	String-based Outils de TAL pour l'extraction des termes et règles d'associations.	Automatique	Non	/
TaxoMap **[Kasri, 2010]**	Ontologie OWL RDF (Taxonomies)	(1,n)	-Terminologique -Structurelle	Technique basée sur la mesure de similarité de Lin (SimLinLike)	Automatique	Oui	La méthode de clustering proposé dans Falcon
COMA++ **[Massmann et al., 2011]**	XSD Schéma graphe	(0,n)	-Syntaxique -Structurelle (descendants , ascendant)	-Fragment-based -Matching -Taxonomy -Matcher	Automatique	Oui	-Clustering basé sur la Mesure structurelle

[Kachroudi et al., 2013]	Reseau sémantique	(0,n)	-Terminologique -Sémantique	WordNet	Automatique	Oui	Clustering basé sur WordNet
[Algergawy et al., 2014]	Schéma XML, Ontologies	(0,n)	-Syntaxique -Structurelle	Levenstein distance ,N-gram distance	Automatique	Oui	Clustering basé sur la mesure structurelle

TABLE 2.1 – Comparaison des méthodes d'alignement

2.5.1 Analyse

Après une étude comparative des différents systèmes d'alignement présentés dans La table (Tableau 2.1) nous constatons ce qui suit :

— La plupart des systèmes comparés admettent en entrée deux ontologies OWL et pour certains (GLUE et Taxomap) un type particulier des ontologies OWL qui sont des taxonomies, (Algergawy) ont des schémas XML comme entrée.

— Parmi les différentes approches d'alignement citées auparavant, il y a des systèmes du Mapping qui sont complètement automatisés, dans le sens où les similarités entre les concepts des deux ontologies à faire correspondre sont identifiées sans aucune intervention de l'utilisateur. Cependant, l'approches (GLUE) nécessitent une intervention humaine pour accomplir la tâche du Mapping.

— Pour la découverte de la similarité entre les concepts des ontologies, la technique de calcul de similarité n'est pas la même, la plupart des algorithmes utilisent la mesure de similarité structurelle et terminologique, celle de (Kachroudi) utilise une mesure sémantique. La mesure structurelle est commune dans (Algergawy, AROMA, Taxomap, GLUE et COMA++), d'autre mesures dans (Algergawy,et COMA++ , GLUE) qui est la mesure syntaxique, et la mesure extensionnelle pour (GLUE).

— Les approches d'alignement des ontologies volumineuses utilisent le clustering comme une technique pour le passage à l'échelle, pour (Algergawy et COMA++) le clustering est basé sur la mesure structurelle.

— Les techniques de passage à l'échelle n'ont pas été présentées dans l'article du (GLUE).

— La phase de clustering est basée sur le calcul des mesures de distance pour partitionner l'ontologie en parties, ce qui facilite le passage à échelle pour l'alignement des ontologies volumineuse. Ces calculs sont répétés une deuxième fois pendant l'étape d'aligne- ment des ontologies.

2.6 Conclusion

Tout au long de ce chapitre, nous avons essayé d'éclaircir la notion du mapping en présentant différentes définitions des concepts liées à ce contexte. Nous avons également présenté plusieurs méthodes d'alignement d'ontologies proposées par la littérature, chacune d'elle repose sur le calcul des mesures de similarité.

A la fin du chapitre , nous avons présenté une comparaison entre les méthodes d'alignement des ontologies. Pour montrer les techniques de calcul de similarité et celles du passage à l'échelle existantes.

Après une étude sur des méthodes existantes, Nous avons constaté qu'un bon processus de mapping est celui qui permet de trouver les correspondances les plus pertinentes avec le moindre nombre de calculs possibles.

Dans le prochain chapitre nous allons présenter notre solution au problème posé, un algorithme du passage à l'échelle qui vise à traiter le problème du passage à l'échelle des méthodes d'alignement

Chapitre 3

Conception du système d'alignement d'ontologies

3.1 Introduction

Après avoir donné un aperçu général sur le domaine d'alignement d'ontologies. Dans ce chapitre nous proposons notre solution pour résoudre le problème de l'alignement des ontologies larges et éliminer la redondance de calculs.

Nous commençons ce chapitre, en présentant les caractéristiques de notre système par rapport aux systèmes d'alignement présentés dans le chapitre précédant, et les objectifs de notre système. Ensuite, nous allons donner une description détaillée de notre solution pour le partitionnement et l'alignement, en présentons les différents algorithmes et les mesures de similarité utilisées.

3.2 Caractéristiques de notre système par rapport aux systèmes existants

Comme nous avons vu dans le chapitre précédent, l'une des tech- niques de passage à l'échelle étudiée consiste à partitionner les ontologies en blocs avant de réaliser l'alignement, ce qui est considéré comme une bonne solution pour diminuer l'espace de recherche des correspondances, mais le problème avec cette approche c'est que les mesures de similarités sont utilisées deux fois, pour le clustering et également pour l'alignement.

Notre objectif principal est de proposer un système d'alignement sans passer deux fois par les mesures de similarité, notre système se caractérise par :

1. Réaliser le clustering sans passer par les mesures de similarité

— Dans les systèmes d'alignement proposés dans la littérature, la phase de clustering est basée sur le calcul des mesures de similarité pour construire des blocs, ces derniers facilitent le passage à échelle pour l'alignement des ontologies volumineuse. Mais le travail de calcul de similarité sera fait une deuxième fois pendant l'étape d'alignement des ontologies, dans ce chapitre on va proposer un système d'alignement sans cette redondance de calculs.

2. Utiliser la structure de l'ontologie pour effectuer le partitionnement
   — Pour utiliser la structure de l'ontologie on doit d'abord vérifier que l'ontologie est bien cohérente et consistante.
   — La cohérence et la consistance d'une ontologie sont deux caractéristiques fondamentales définissant sa qualité, ils indiquent que l'ontologie a été crée sans erreurs, "l'absence de contradiction entre les concepts d'ontologies" [Djedidi et al., 2010].
   — Quand l'ontologie est bien construite, les concepts proches dans son hiérarchie sont similaires, ce qui nous donne l'idée d'utiliser la structure de l'ontologie dans notre intérêt et grouper les concepts voisins en cluster.
3. Alignement : Notre système d'alignement est doté de plusieurs critères, nous citons :
   — Une structure d'ontologie OWL comme entrée.
   — Le système est Automatique et il ne nécessite aucune intervention d'un expert du domaine.
   — Traitement des ontologies volumineuse après avoir les partitionner en groupes
   — Plusieurs mesures de similarités sont combinées pour avoir la similarité sémantique, Terminologique : (Levenshtein, Wordnet) et Structurelle : Wu et palmer.

3.3 Système d'alignement proposé :

Dans cette partie du chapitre nous allons présenté notre proposition du système d'alignement. Notre système comprend deux principales étapes (Figure 3.1), le partitionnement des deux ontologies selon la structure de chacune, et l'étape d'alignement. Comme nous avons déjà mentionné, dans notre système nous éliminons la redondance des calculs de similarité. Pour faire cela , nous proposons un système d'alignement d'ontologies volumineuses avec un partitionnement basé sur la structure d'ontologie.

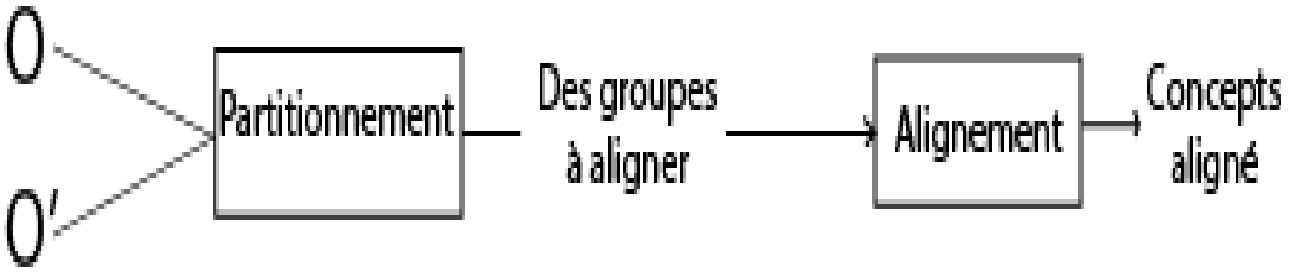

FIGURE 3.1 – Vue générale de notre système d'alignement

3.3.1 Phase I : L'étape du partitionnement

La phase de partitionnement proposé consiste à partitionner les deux ontologies à aligner en blocs sans passer par les mesures de similarité. La figure 3.3 illustre cette phase.

1. La Première étape de l'algorithme consiste à extraire les éléments de l'ontologie et les mettre dans une liste à part.
   - Extraire les éléments se fait en utilisant le parcours par profondeur.
   - Ensuite on met le résultat de ce parcours dans une liste .
   - Le parcours par profondeur (DFS, pour Depth-First Search) veillera à ce que les éléments voisins dans l'ontologie seront plus proche dans la liste, afin de respecter les principes différentiels pour la construction d'ontologie (une grande similarité entre les éléments proche , fils-père , fils-fils..).
2. En utilisant la liste déjà crée dans l'étape précédente, nous allons créer une liste de partitions pour séparer chaque partition.

   La première liste représente les groupes k (Nombre de partition voulu) , chaque élément de cette liste contient une liste des concepts d'une partition. Pour remplir les listes de partitions on va partitionner notre première liste déjà parcourus en profondeur en k parties et on ajoute chaque partie dans une liste.
3. A la fin de l'algorithme, on vérifiera chaque élément pour son appartenance à sa partition
   - Pour chaque élément nous allons calculer un poids (indice d'appartenance) pour chaque partition.
   - Nous allons ensuite mettre chaque élément dans la partition où il doit appartenir.

3.3.2 L'algorithme de partitionnement

Pour réaliser le partitionnement, on doit utiliser les algorithmes suivants :

— **ParcoursProfondeur** (Algorithme 1) :Parcourir et extraire les concepts de l'ontologie.

— **Construction Liste Des partitions** (Algorithme 2) : Pour faciliter le traitement des partitions, nous allons construire une liste des partitions.

— **Poids** (Algorithme 3) : Dans le but de mettre chaque élément dans sa partition on doit calculer le poids d'un élément dans la partition actuelle pour savoir s'il est dans la bonne partition ou non.

— **Vérification** (Algorithme 4) : Après avoir calculé le poids, le concept se déplace vers sa bonne partition.

3.3.2.1 Le parcours en profondeur

L'algorithme de « parcours en profondeur » consiste à parcourir les concepts de l'ontologie O en profondeur et les insérer dans une Liste L .

Liste des fonctions utilisées

— **Racine() :** retourne la racine d'une ontologie.

— **Depiler() :** récupérer un élément de la pile .

— **Empiler() :** ajouter un élément à la pile.

— **tailleO() :** retourne la taille d'une ontologie .

— **tailleL() :** retourne la taille d'une liste .

— **Ajouter() :** Ajouter un élément dans la liste.

— **GetListFils() :** Récupère les concepts fils du concept passé au paramètres .

Algorithme 1 : ParcourProfondeur

Entrées : Ontologie **O**

Sorties : Liste **L**

```
1  Variables : Pile p ; Liste L;
2  début
3      L<-∅ ;
4      R<-Racine(O) ;                                  // récupérer le concept racine
5      p.empiler(R) ;                                  // empiler le concept dans la pile
6      tant que tailleL(L)<tailleO(O) faire
7          S<-p.depiler() ;    // récupérer le concept qui se trouve au sommet dans la pile
8          L.ajouter(S) ;                              // insérer le concept dans la liste
9          Listfils<-getListFils(S) ;   // récupérer les concepts fils du concept en cours de
             traitement
10         pour chaque F_i ∈ Listfils faire
11             p.empiler(F_i) ;                        // empiler les concepts fils dans la pile
12         fin
13     fin
14     retourner L;
15 fin
```

3.3.2.2 Construire une liste de partitions

Pour bien gérer les partitions nous allons utiliser deux structures (voir figure 3.2) :

La première structure de « partition » est utilisée pour représenter la liste des partitions pour bien se déplacer d'une partition aux autres.

La deuxième de « concept » est utilisée pour la représentation de la liste de concepts d'une partition donnée

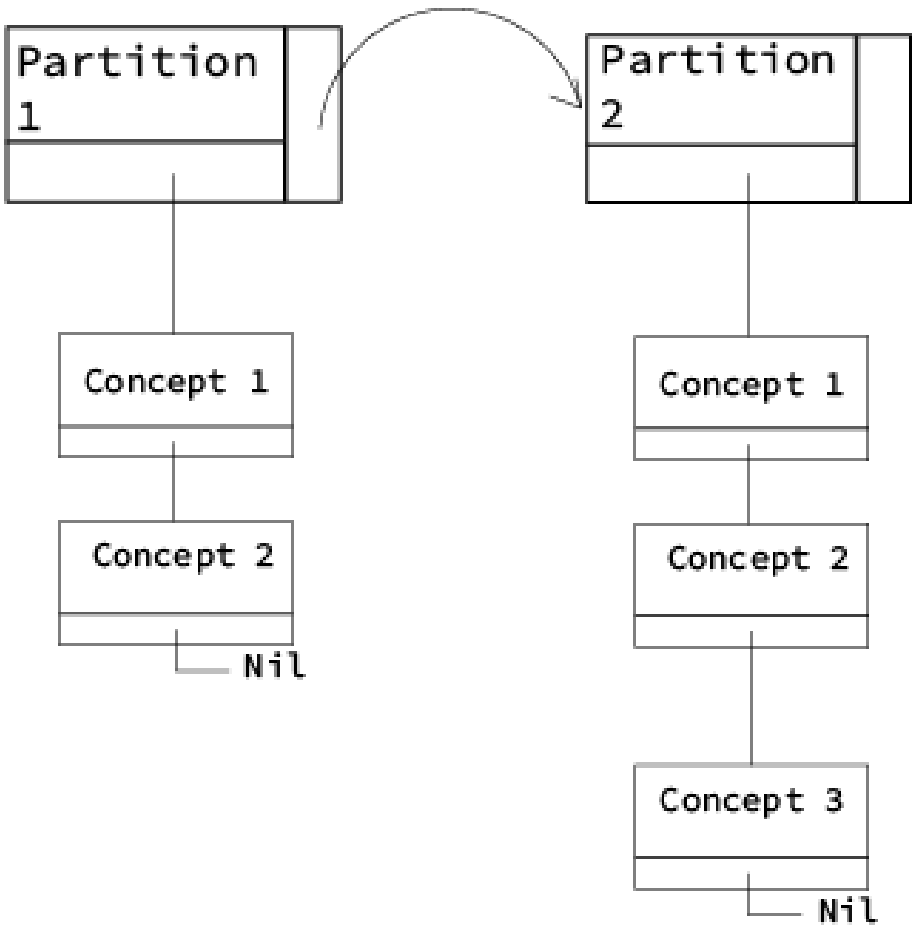

FIGURE 3.2 – La structure d'une liste de partitions

L'algorithme « ConstructionListeDesPartitions » consiste à partitionner la liste du parcours précédent en k groupe de taille n/k élément, ensuite nous allons mettre chaque groupe dans une liste de partition comme il est représenté ci-dessus ;

Liste des fonctions utilisées

— **tailleL() :** retourne la taille d'une liste

— **Ajouter() :** Ajouter un élément dans la liste

— **getConcept() :** retourne le concept

Algorithme 2 : ConstructionListeDesPartitions

Entrées : Liste l,Nombre de partitions k

Sorties : Liste des partitions

```
1  Variables : Liste de partitions Lp ; Sous-liste de partition Q;
2  début
3      TaillePart<- tailleL(L)/k ;                  // calculer la taille de chaque partition i
4      i<- 0;
5      np<- 1 ;                                                  // numéro de la partition
6      tant que  i < tailleL(L) faire
7          Q_np < −∅ ;                                            // créer la partition i
8          j<- i;
9          tant que  j < TaillePart*np ET j < taille(L) faire
10             s<-getConcpt(j);
11             Q_np.Ajouter(s) ;              // On ajoute s dans la liste des partitions i
12             j++;
13         fin
14         Lp.ajouter(Q_np) ;        // On insère la partition courante dans la liste globale
15         np++;
16         i=j;
17     fin
18     retourner Lp;
19 fin
```

3.3.2.3 Calcul du poids et Vérification

L'algorithme de « calcul de poids » a comme entrées un élément et une liste et il retourne un entier, cet entier est le poids de l'élément donné dans la liste, la façon de calculer le poids est la suivante : Le poids est égal au nombre des voisins de l'élément x dans cette liste ;

Liste des fonctions utilisées

- **EstFilsDe() :** vérifier si le concept passé en paramètre est un fils
- **EstParentDe() :** vérifier si le concept passé en paramètre est un parent
- **EstFréreDe() :** vérifier si le concept passé en paramètre est un frère

Algorithme 3 : Poids

Entrées : Concept m, ListePartition L

Sorties : Entier

```
1  début
2      p<-0;
3      pour chaque c_i ∈ L faire
4          si (c_i.EstFilsDe(m)) alors
5              p++;
6          fin
7          si (c_i.EstParentDe(m)) alors
8              p++;
9          fin
10         si (c_i.EstFréreDe(m)) alors
11             p++;
12         fin
13     fin
14     retourner p;
15 fin
```

L'algorithme « Vérification » consiste à vérifier si le terme est dans la bonne partition en utilisant l'algorithme 3 précédent (Poids), cet algorithme veillera à ce que chaque élément dans chaque partition est dans sa bonne partition. Si le concept appartient à sa partition actuelle on le garde, sinon d'après la façon d'extraire les concepts, notre concept est dans soit la partition précédente soit dans la partition suivante, à partir du calcul de son poids dans les 3 partitions actuelle, précédente, suivante, et s'il n'est pas dans sa partition on le déplace vers le nouveau groupe et on le supprime de l'ancien.

Liste des fonctions utilisées

- **P.Suivante :** retourne la partition suivante de P.
- **poids() :** retourne un entier qui représente le poids de l'élément donnée dans la liste.
- **Ajouter(P,c) :** Ajouter le concept c dans la partition P.
- **Supprimer(P,c) :** Supprimer un concept c de la partition P.

Algorithme 4 : Vérification

Entrées : Liste des partitions list

Sorties : Liste des partitions

```
1  début
2      p1<-0;
3      p2<-0;
4      PartitionPrecedente<-Nil;
5      pour chaque Partition P ∈list faire
6          PartitionSuivante<-P.Suivante;
7          pour chaque Concept c ∈P faire
8              si PartitionPrecedente ≠Nil alors
9                  p1<-Poid(c,PartitionPrecedente);
10             fin
11             si PartitionSuivante ≠Nil alors
12                 p2<-Poid(c,PartitionSuivante);
13             fin
14             p3<-Poid(c,P);
15             si p3<p1 alors
16                 Ajouter(PartitionPrecedente,c);
17                 Supprimer(P,c);
18             fin
19             si p3<p2 alors
20                 Ajouter(PartitionSuivante,c);
21                 Supprimer(P,c);
22             fin
23         fin
24     fin
25     retourner list;
26 fin
```

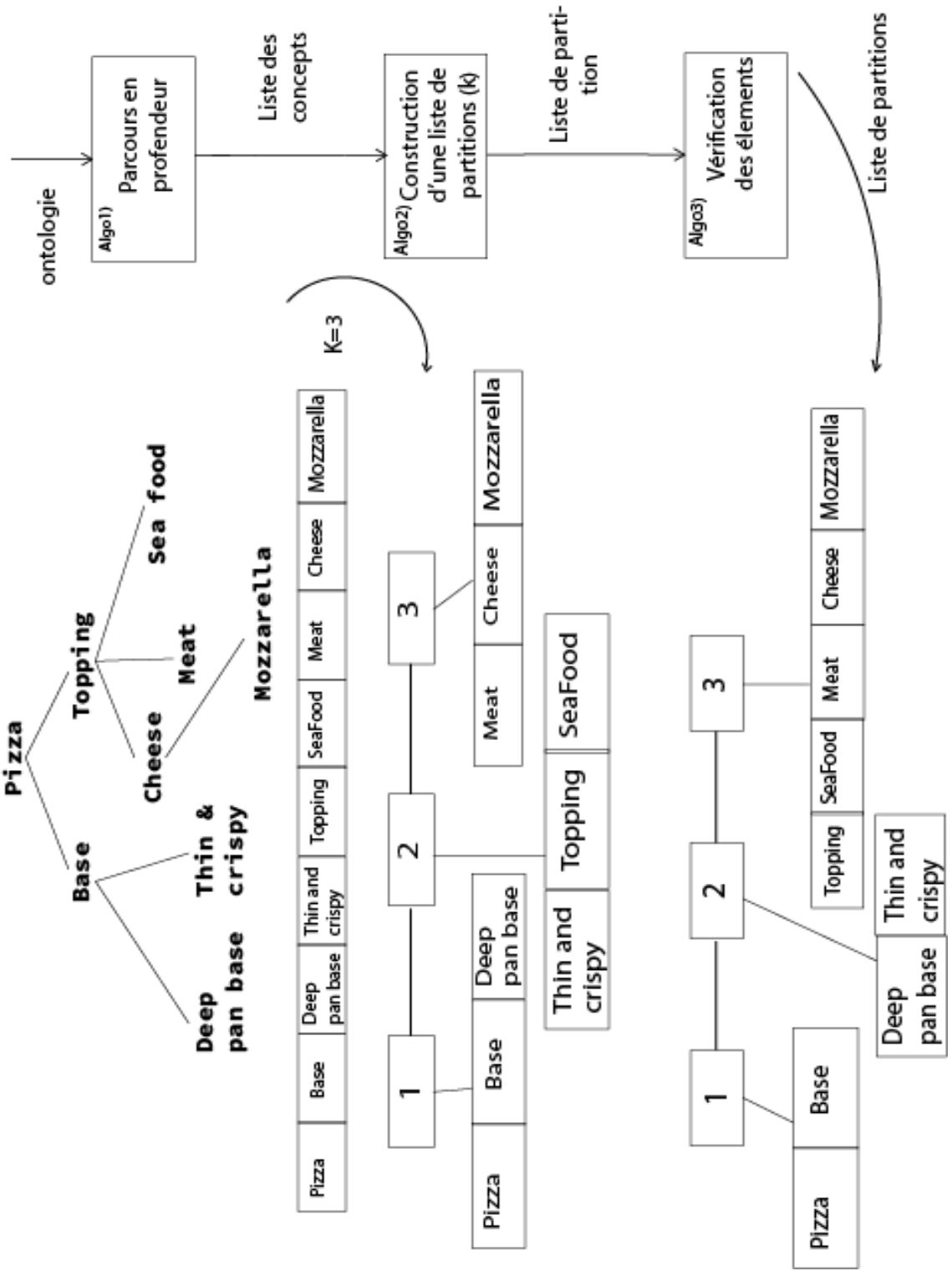

FIGURE 3.3 – Illustration de notre phase du partitionnement

3.3.3 Phase II : L'étape d'alignement

Pour atteindre notre objectif d'effectuer un alignement entre les deux ontologies, on va passer par plusieurs étapes, la première est de trouver un représentant pour chaque cluster de notre partitionnement.

3.3.3.1 Le représentant des clusters

Le représentant de chaque cluster est un élément du cluster, qui ressemble à la notion du médoide . Nous avons utilisé la formule de l'algorithme K-mediode [Vialle, 2017] pour trouver le medoide de chaque cluster.

Pour déterminer le représentant d'un cluster, on calcule la somme des similarités de chaque concept par rapport aux autres concepts du cluster. Le concept qui représente le mediode est celui qui a une similarité élevée avec les autres éléments du cluster.

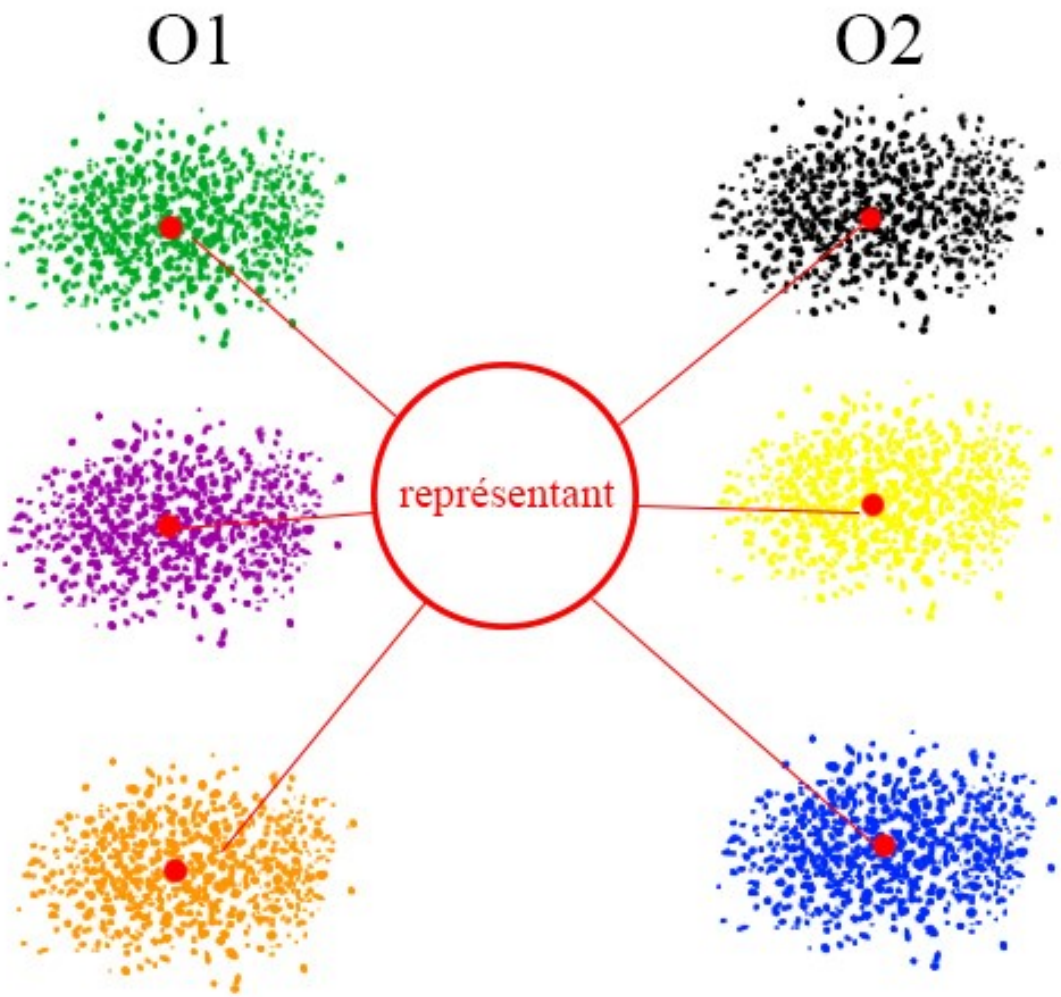

FIGURE 3.4 – Les représentants des clusters

3.3.3.2 Alignement entre représentants

Après la détermination d'un représentant pour chaque cluster, l'étape suivante consiste à trou- ver des correspondances entre les clusters des deux ontologies en utilisant les représentants. L'obtention d'un ensemble de correspondances entre clusters des deux ontologies, se fait par par calculer la similarité entre chaque représentant de la première ontologie et tous les représentants de la deuxième ontologie.

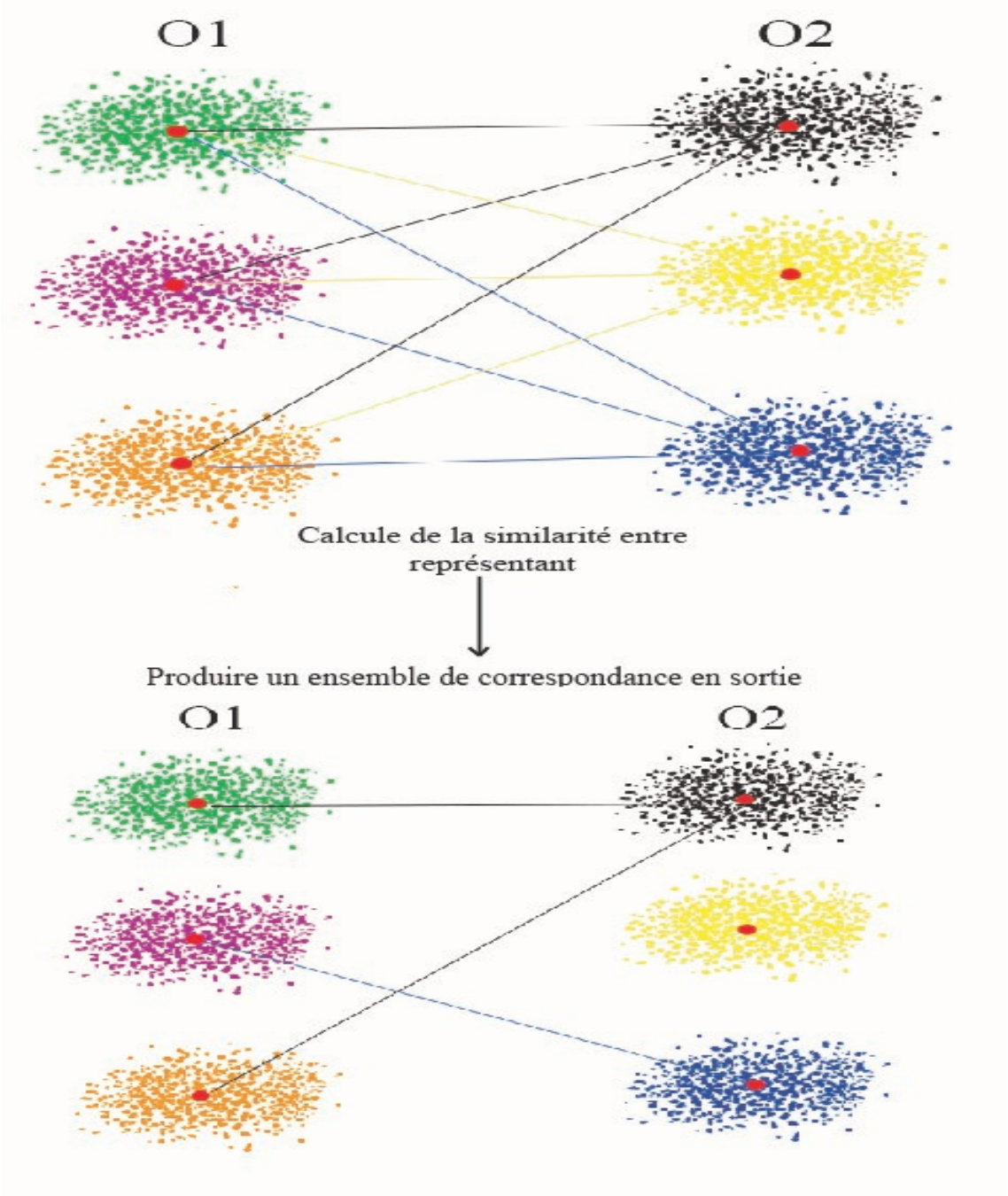

FIGURE 3.5 – Alignement de représentant

3.3.3.3 Alignement entre concept

Après avoir calculé les mesures de similarité entre représentants et produire un ensemble de correspondance entre les clusters des deux ontologies on va maintenant passer à l'étape suivante qui a pour but de faire l'alignement entre les concepts . Pour cette étape on va faire un calcul de si- milarité sémantique entre les concepts des partitions similaires des deux ontologies

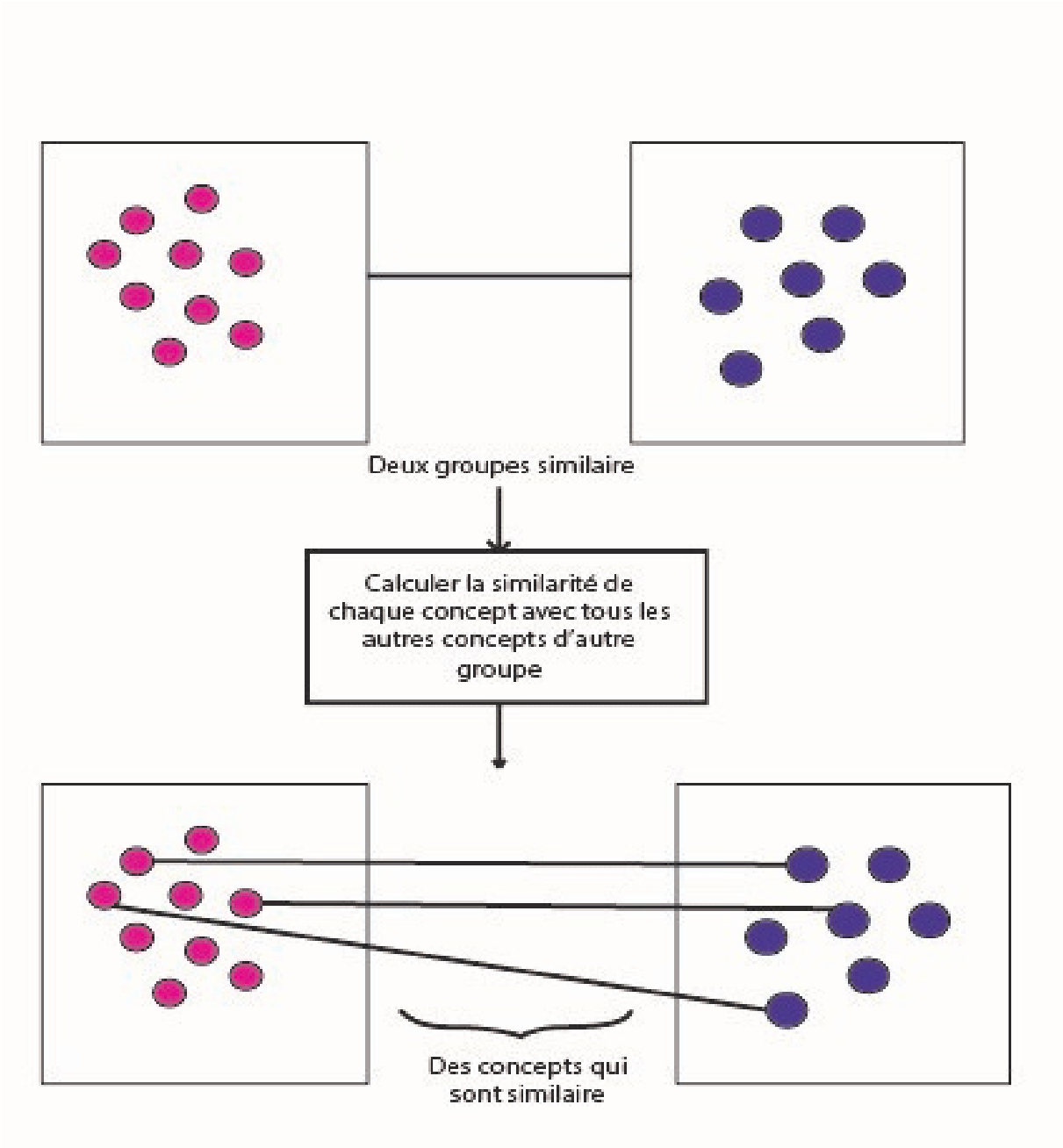

FIGURE 3.6 – Alignement entre concepts

3.3.3.4 Les mesures de similarité

Les mesures de similarité utilisées dans les deux étapes précédentes sont présentées par la suite :

3.3.3.5 Les mesures terminologiques

Pour pouvoir calculer la mesure terminologique on va utiliser les deux mesures suivantes :

1. **La similarité syntaxique** Ils existe plusieurs choix pour calculer la similarité syntaxique, notre choix s'est porté sur la distance de Levenshtein. La distance de Levenshtein également connue sous le nom de distance d'édition est une distance, au sens mathématique du terme, donnant une mesure de la différence entre deux chaînes de caractères. Elle est égale au nombre minimal de caractères qu'il faut supprimer, insérer ou remplacer pour passer d'une chaîne à l'autre. On appelle distance de Levenshtein entre deux mots M et P le coût minimal pour transformer M en P en effectuant les seules opérations élémentaires suivantes :
   — substitution d'un caractère de M par un caractère différent de P ;
   — insertion (ou ajout) dans M d'un caractère de P .
   — suppression (ou effacement) d'un caractère de M.
   Soit **Insert(M,P)** la somme des insertions nécessaire pour avoir les mêmes caractères entre le mot M et le mot P .
   Soit **Supp(M,P)** la somme des suppressions nécessaire pour avoir les mêmes caractères entre le mot M et le mot P.
   Soit **Permut(M,P)** la somme des permutations nécessaire pour avoir le même ordre des caractéres dans M et P.
   La distance de Levenshtein est définie comme suit :

$$Lev(M,P) = Insert(A,B) + Supp(A,B) + Permut(A,B).$$

   Et pour avoir la similarité Syntaxique :

$$SimSyn(M,P) = 1 - \left(\frac{Lev}{\max(|M|,|P|)}\right)$$

   Si par exemple on prend 'Levenshtein' comme chaine de caractère source (P),
   Si M='Levenshtein' alors Lev(M,P)=0 ,

$$SimSyn = 1 - \frac{0}{11} = 1$$

Si M ='Levenshten' alors Lev(M,P)=1 ,

$$SimSyn = 1 - \frac{1}{11} = 0.90$$

Si M='Levshten' alors Lev(M,P)=2 ,

$$SimSyn = 1 - \frac{2}{11} = 0.81$$

2. **La similarité lexicale** La similarité lexicale est la mesure du degré de ressemblance entre des séries de mots en utilisant des ressources externes, plusieurs choix est disponible pour etre employé, notre choix s'est porté sur Wordnet. Wordnet est une base de données lexical large d'anglais, Cette ressource est structurée autour de la notion de synsets, c'est-à-dire en quelque sorte un ensemble de synonymes qui forment un concept. Chaque synset représente un sens de mot. Les synsets sont reliés entre eux par des relations, soit lexicales (antonymie par exemple) ou taxonomiques (hyperonymie, métonymie, etc.)

   Pour le calcul de la similarité, la fonction $Syn(c)$ calcul l'ensemble des synsets de wordnet du concept c, soit $S = Syn(c1) \cap Syn(c2)$ l'ensemble des sens communs entre c1 et c2 à comparer, la cardinalité de S est : $Card(S) = |Syn(c1)Syn(c2)|$;

   Soit $\min(|Syn(c1)|, |Syn(c2)|)$ le minimum entre la cardinalité des deux ensembles $Syn(c1)$ et $Syn(c2)$ alors la mesure de similarité lexicale entre les deux concepts c1 et c2 est définit comme suit :

$$SimLex(c1, c2) = \frac{Card(S)}{\min(|Syn(c1)|, |Syn(c2)|)}$$

   Ensuite pour avoir la similarité terminologique finale on combine les deux mesures calculées c.à.d.

$$SimTer(c1, c2) = \frac{SimSyn + SimLex}{2}$$

3. **La similarité structurelle** Cette dernière sera calculer par la mesure wu et palmer (Wup) : La mesure de similarité de Wup est basée sur le principe suivant : Etant donnée une ontologie O formée par un ensemble de nœuds et un nœud racine R (Figure 3.7). Soit X et Y deux éléments de l'ontologie dont nous allons calculer la similarité. Le principe de calcul de similarité est basé sur les distances (N1 et N2) qui séparent les nœuds X et Y du nœud racine et la distance

qui sépare le concept subsumant (CS) de X et de Y du nœud R.

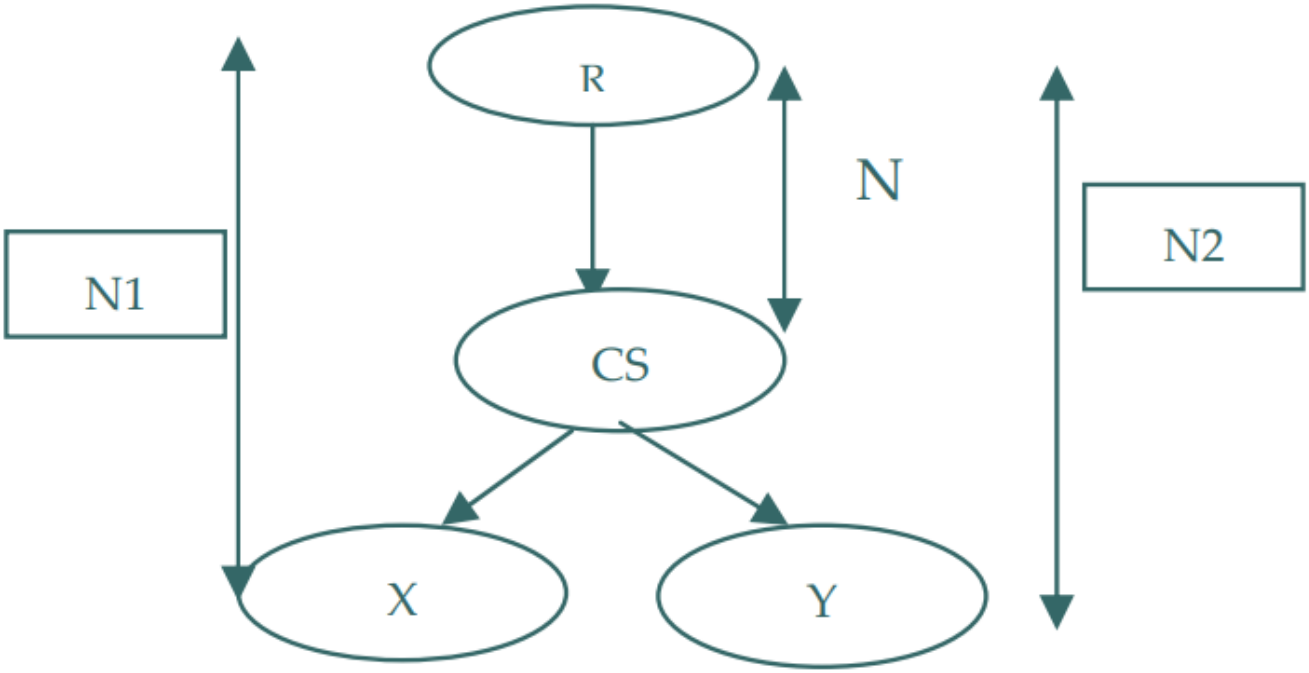

FIGURE 3.7 – Les paramètres de la mesure de Wu et palmer

La mesure de Wu et Palmer est définie par la formule suivante :

$$SimStruct(X,Y) = \frac{(2*N)}{(N1+N2)}$$

4. **La similarité sémantique** La similarité sémantique est une évaluation du lien sémantique afin d'estimer à quel degré deux concepts sont proches dans leurs sens, elle est calculée par la combinaison des similarités terminologique et structurelle, donc elle est définis selon la formule :

$$SimSem(c1,c2) = \frac{SimTer(c1,c2) + SimStru(c1,c2)}{2}$$

— Afin d'améliorer cette dernière nous avons ajouté un poids pour augmenter sa valeur, Soit :

$$\mathbf{P1} = e^{SimSyn}$$

$$\mathbf{P2} = e^{SimLex}$$

$$\mathbf{P3} = e^{SimStruct}$$

La formule de la similarité sémantique final est :

$$SimSem(c1,c2) = \frac{\mathbf{P1} * SimSyn(c1,c2) + \mathbf{P2} * SimLex(c1,c2) + \mathbf{P3} * SimStruct(c1,c2)}{P1 + P2 + P3}$$

3.4 Conclusion

Dans ce chapitre, nous avons présenté les étapes détaillées de notre algorithme d'alignement, en détaillant les deux phases : de partitionnement proposé, et d'alignement, ainsi que les mesures de similarité utilisées.

Dans le chapitre suivant, nous allons implémenter et mettre en œuvre ce que nous avons déjà proposé dans ce chapitre, autrement dit l'implémentation et la validation des résultats de notre système.

Chapitre 4

Implémentation et Evaluation

4.1 Introduction

Nous avons présenté dans le chapitre précédent l'approche proposée dans notre travail, dans ce chapitre on va expliquer l'implémentation de cette approche. Le chapitre s'organise comme suit : tout d'abord on va présenter l'environnement de développement ainsi que les différents outils utilisés dans ce travail, ensuite nous allons décrire de façon visuelle l'implémentation de notre système à travers des captures d'écran des différentes interfaces du système. Nous passons par la suite à la validation et les tests de notre système, nous montrons l'évaluation du mapping avec quelques résultats et discussions.

4.2 Les outils utilisés

Avant de commencer l'implémentation de notre système d'alignement, nous allons spécifier les différents outils utilisés dans notre travail :

4.2.1 Python

Python est un langage de programmation interprété, multi-paradigme et multiplateformes. Il favorise la programmation impérative structurée, fonctionnelle et orientée objet. Il est doté d'un typage dynamique fort, d'une gestion automatique de la mémoire par ramasse-miettes et d'un système de gestion d'exceptions ; Il est aussi un langage de programmation puissant et facile à apprendre. Il dispose de structures de données de haut niveau efficaces et d'une approche simple mais efficace de la programmation orientée objet. La syntaxe élégante et le typage dynamique de Python, associés à sa nature interprétée, en font un langage idéal pour la création de scripts et le développement rapide d'applications dans de nombreux domaines sur la plupart des plates-formes

4.2.1.1 Paquets utilisés dans Python

Dans l'implémentation de notre système, on a utilisé les paquets suivants dans python :

1. **NLTK :** Natural Language Toolkit (NLTK) est une bibliothèque logicielle en Python permettant un traitement automatique des langues, développée par Steven Bird et Edward Loper du département d'informatique de l'Université de Pennsylvanie. En plus de la bibliothèque, NLTK fournit des démonstrations graphiques, des données-échantillon, des tutoriels, ainsi que la documentation de l'interface de programmation (API).
2. **Owlready2 :** Owlready est un module de programmation orientée ontologie en Python, incluant un quadrore RDF optimisé. Owlready peut :
   — Importer les ontologies OWL 2.0 au format NTriples, RDF/XML ou OWL/XML.
   — Exporter les ontologies OWL 2.0 vers NTriples ou RDF/XML.
   — Manipuler de manière transparente les classes, les instances et les propriétés d'ontologie, comme s'il s'agissait d'objets Python normaux.
   — Ajouter les méthodes Python aux classes d'ontologie.
3. **Wordnet :** WordNet est une base de données lexicale pour la langue anglaise, créée par Princeton et faisant partie du corpus NLTK. WordNet peut être utilisé avec le module NLTK pour trouver la signification des mots, des synonymes, des antonymes, etc.

4.2.2 Le logiciel Protégé pour la visualisation des ontologies

Protégé est un système auteur pour la création d'ontologies. Il a été créé à l'université Stanford et est très populaire dans le domaine du Web sémantique et au niveau de la recherche en informatique. Protégé est développé en Java. Il est gratuit et son code source est publié sous une licence libre (la Mozilla Public Licence).Protégé peut lire et sauvegarder des ontologies dans la plupart des formats d'ontologies : RDF, RDFS, OWL, etc.

4.2.3 HermiT-Reasoner

HermiT-reasoner est un raisonneur pour les ontologies écrites à l'aide du langage OWL (Web OntologyLanguage). Avec un fichier OWL, HermiT peut déterminer si l'ontologie est cohérente et consistante.

4.3 Présentation de l'application

Notre application contient une interface globale, permettant de faire le partitionnement et aussi le mapping des deux ontologies.

L'interface de notre système contient deux boutons, un bouton pour le partitionnement, et le deuxième bouton permet de faire l'alignement entre les deux ontologies, (indiqué par 4 et 5 respectivement dans la figure au-dessous) , en cliquant sur un des boutons le processus se lance et son résultat s'affiche.

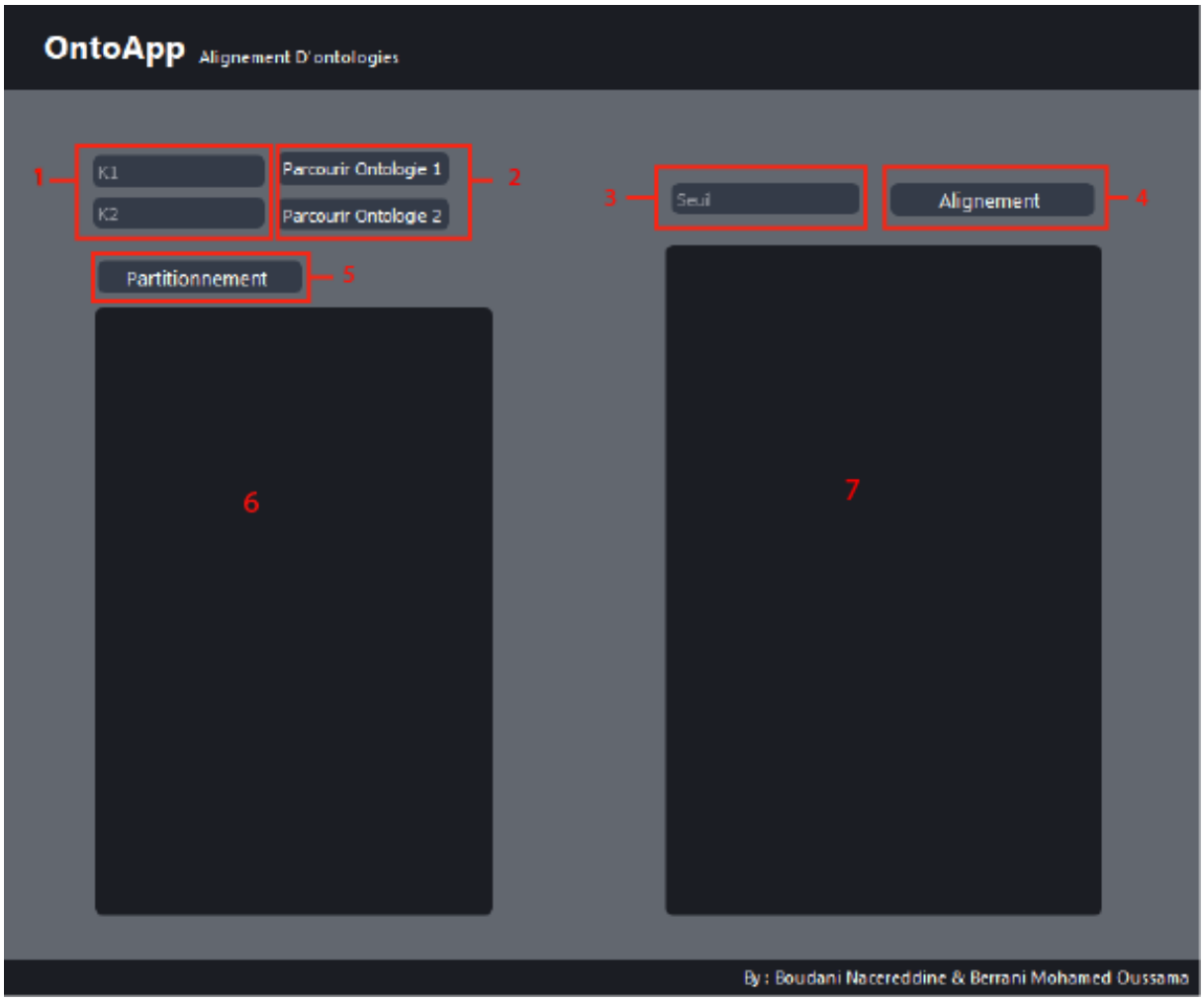

FIGURE 4.1 – L'interface de l'application

Afin qu'un utilisateur puisse aligner deux ontologies, il doit cliquer sur le bouton Parcourir pour choisir l'ontologie 1 puis la sélectionner, de même pour l'ontologie 2 (indiqué par 2 dans la figure 4.1)

Après avoir choisi les deux ontologies à aligner, l'utilisateur doit spécifier le nombre de partitions voulu pour chaque ontologie (O1 et O2) en introduisant la valeur de K dans les deux champs respec-

tivement (indiqué par 1 dans la figure 4.1) , ensuite pour effectuer l'alignement l'utilisateur doit aussi indiquer le seuil (indiqué par 3 dans la figure 4.1)

L'interface du notre système contient deux parties d'affichage, une pour afficher les résultats de partitionnement et l'autre pour les résultats de Mapping (indiqué par 6 et 7 respectivement dans la figure 4.1).

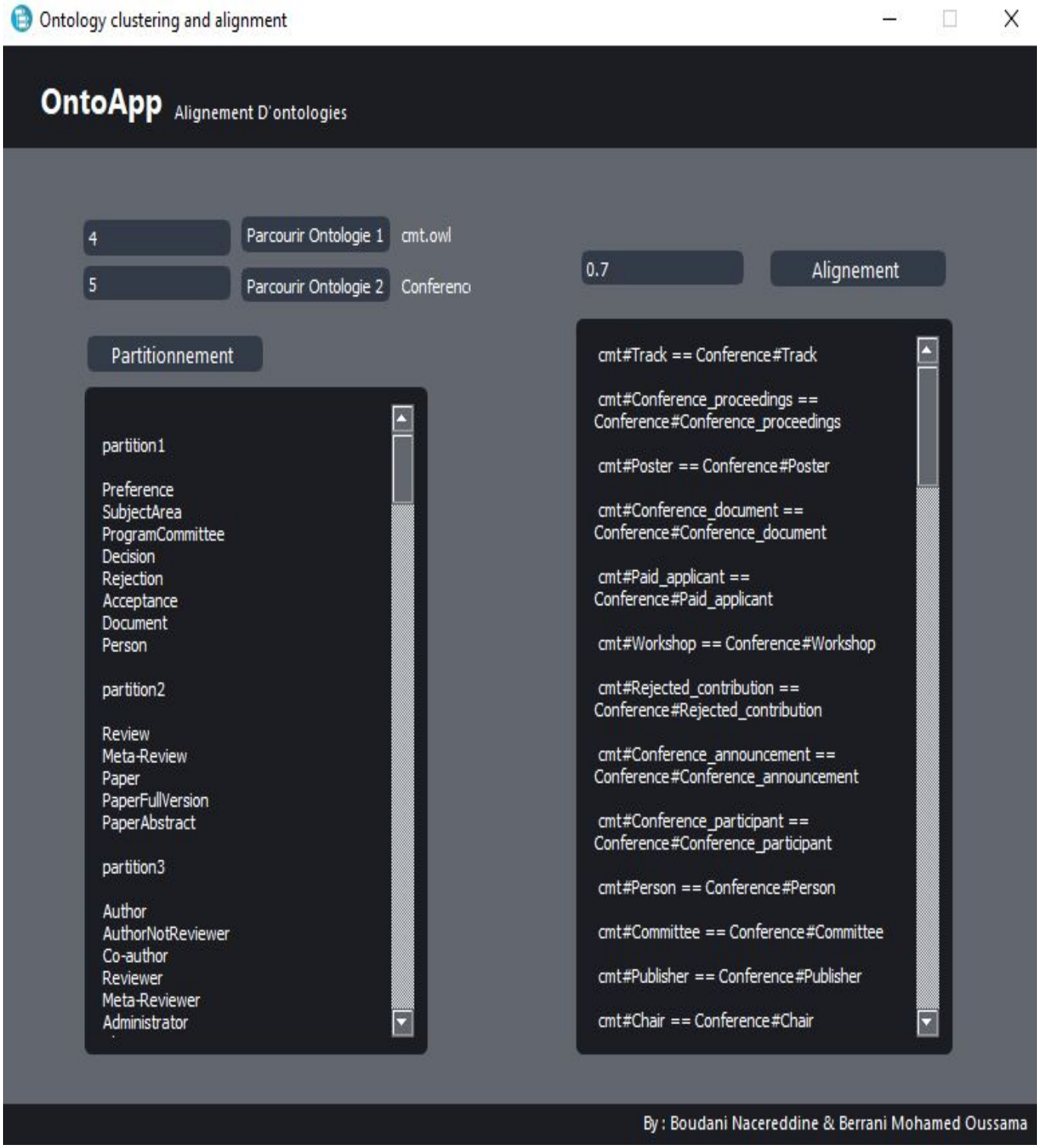

FIGURE 4.2 – Le résultat généré sur l'interface

L'application génère deux fichiers de format xml, un pour le partitionnement (Figure 4.3) et l'autre pour l'alignement (Figure 4.4) qui sont enregistrés dans le même dossier de l'application

FIGURE 4.3 – Une partie d'un fichier XML du partitionnement généré

```xml
<?xml version="1.0" ?>
<Alignement>
    <Ontology1>cmt</Ontology1>
    <Ontology2>Conference</Ontology2>
    <Cell>
        <Entity1>Committee_member</Entity1>
        <Entity2>Committee_member</Entity2>
        <Measure>1</Measure>
    </Cell>
    <Cell>
        <Entity1>Submitted_contribution</Entity1>
        <Entity2>Submitted_contribution</Entity2>
        <Measure>1</Measure>
    </Cell>
    <Cell>
        <Entity1>Conference_www</Entity1>
        <Entity2>Conference_www</Entity2>
        <Measure>1</Measure>
    </Cell>
    <Cell>
        <Entity1>Conference</Entity1>
        <Entity2>Conference</Entity2>
        <Measure>1</Measure>
    </Cell>
    <Cell>
        <Entity1>Presentation</Entity1>
        <Entity2>Presentation</Entity2>
        <Measure>1</Measure>
    </Cell>
    <Cell>
        <Entity1>Committee</Entity1>
        <Entity2>Committee</Entity2>
        <Measure>1</Measure>
    </Cell>
```

FIGURE 4.4 – Une partie d'un fichier XML d'alignement généré

4.4 Tests et Evaluation

Afin de tester la fiabilité et la validité de notre approche, nous avons effectué des tests sur des ontologies de différentes tailles des benchmark de L'OAEI [Benchmark OAEI 2020], petite, moyenne et grande, selon le tableau suivant :

Ontology (OWL)	CMT	ConfOf	Sigkdd	Conference	Ekaw	Iasted	Mouse	Human
Nombres de classes	29	39	49	60	74	140	2744	3304

TABLE 4.1 – Les ontologies de tests

Toutes les ontologies présentées dans le tableau ci-dessus ont été prouvées cohérentes et consistantes en utilisant Hermit-Reasoner

4.4.1 Evaluation du partitionnement

Dans cette section on procède à l'évaluation de notre approche pour le partitionnement des ontologies

4.4.1.1 Mesure d'évaluation utilisées

— **Séparation** mesure à quel point un cluster est distinct ou bien séparé des autres clusters. Plus les clusters de l'ontologie sont hétérogènes entre eux, plus les valeurs de similarité entre leurs concepts sont faibles. Ces deux mesures sont inversement proportionnelles. En d'autres mots, plus lune d'elles est grande, l'autre est petite et vis-versa, parce que, plus la proximité entre les objets dans un cluster est élevée, plus ils sont éloignés des objets des autres clusters. Pour évaluer la qualité de notre approche de partitionnement, nous avons choisi d'utiliser la mesure de séparation [Hubert et al., 1985] qui prends ses valeurs dans l'intervalle [0,1], et se calcule par la formule suivante

$$Séparation = \frac{2}{n(n-1))} \sum_{(i,j)\in[1..k])} \sum_{x\in C_i} \sum_{x\in C_j} Sim(x,y).Sim(i,j)$$

Où :

— **K** : est le nombre de clusters ;

— **n** : le nombre de concepts de l'ontologie ;

— **i** et **j** : sont les centres des clusters de **x** et **y**.

Les valeurs entre les concepts de l'ontologie représentent des valeurs de similarité, par conséquent, une bonne classification dans ce cas doit présenter des valeurs minimales de séparation.

4.4.1.2 Résultats expérimentaux et discussion

Dans cette section nous présentons les résultats obtenus. Le test a été fait sur des ontologies de différente taille (petite, moyenne , grande) avec différentes valeurs de K. Les résultats des tests sont présentés dans le tableau suivant :

		K=2	K=6	K=9	K=15
Petite	Conference	0.020	0.019	0.019	0.018
Moyenne	Iasted	0.004	0.010	0.010	0.012
Grande	Mouse	0.003	0.012	0.008	0.009
	Human	0.002	0.006	0.011	0.009

TABLE 4.2 – Valeurs de la mesure de séparations

Les résultats présentés dans le tableau précédent nous a permis d'avoir les barres suivantes :

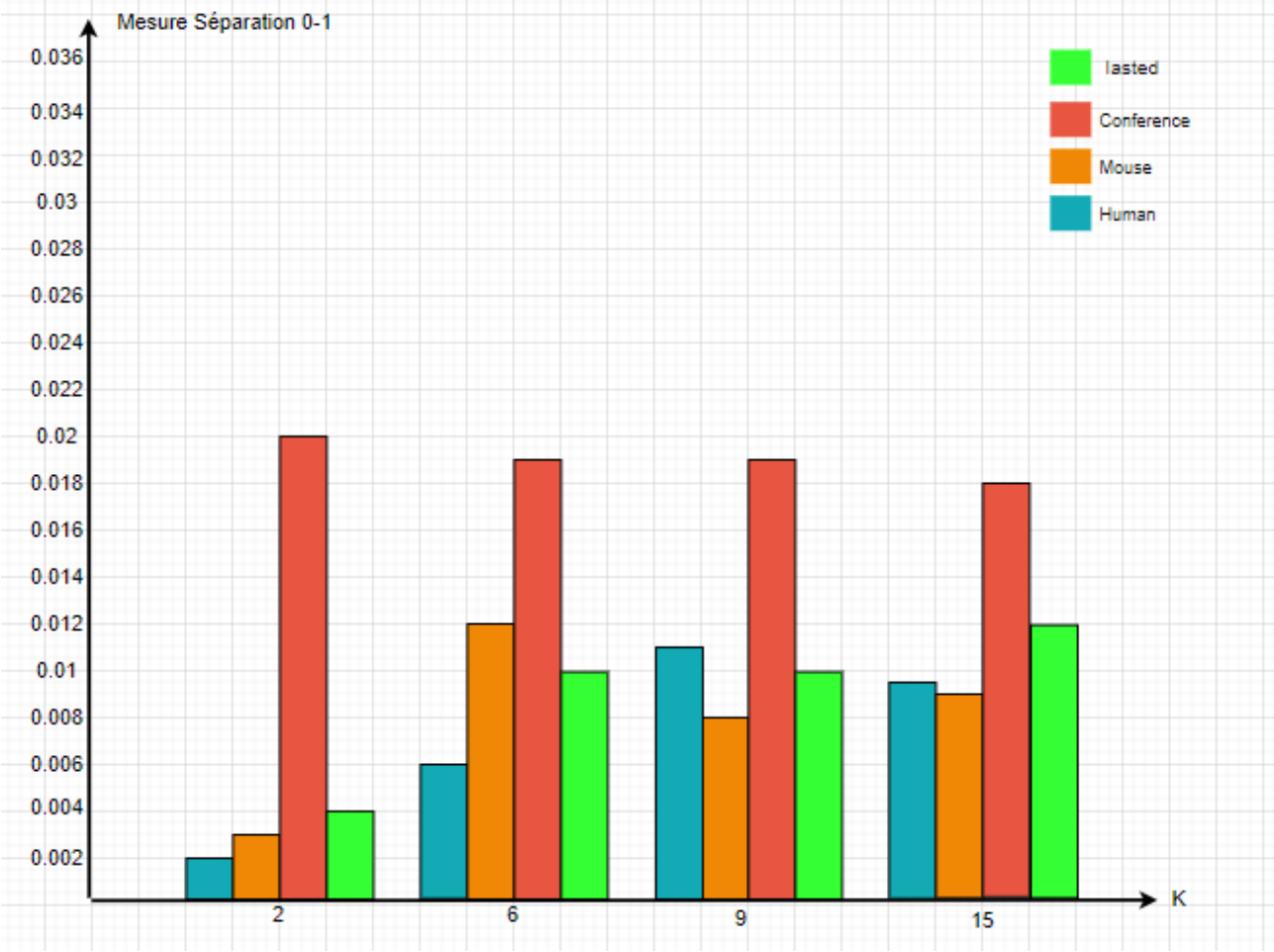

FIGURE 4.5 – Evaluation par la mesure de séparation

— On peut remarquer, à partir de cette figure que notre approche a donné de bons résultats pour les ontologies de ce test, et que même Quand le nombre de k augmente, la séparation n'augmente pas beaucoup

— On peut aussi remarquer que les meilleures valeurs de séparation sont obtenues avec les ontologies larges, la valeur de séparation la plus haute appartient à la plus petite ontologie dans ce test

— A travers le test que nous avons effectué, nous concluons que l'approche de partitionnement présenté dans ce travail est bonne. La valeur de séparation est basse pour toutes les ontologies testées pour des différents valeurs de k, Ce qui signifie que notre système a effectué le partitionnement avec une distance intra cluster élevée, et une distance inter cluster basse

4.4.2 Evaluation du Mapping

Dans cette section on procède à l'évaluation de notre approche pour le mapping des ontologies

4.4.2.1 Mesure d'évaluation utilisées

Pour mesurer la qualité d'un mapping, on va utiliser la mesure dévaluation suivante :

— **Précision :** La précision est une mesure qualitative d'un système de mapping qui se calcule par rapport à un alignement de référence (ensemble des correspondances considérées correctes généralement par un humain) Soit H l'alignement de référence. La précision d'un alignement A produit par un système S est une fonction qui renvoie une valeur appartenant à l'intervalle [0,1].

La précision est le rapport entre les correspondances correctes (True positives) sur la totalité des correspondances retournées par le système S (True positives et False positives). Sa formule est donnée Comme suit :

$$\boldsymbol{Precision} = \frac{|A \cap H|}{|A|}$$

4.4.2.2 Résultats expérimentaux et discussion

— Dans cette section nous présentons les résultats obtenus. Les séries de tests ont été faites sur les ontologies : Iasted, Ekaw, Sigkdd, Cmt, ConfOf, Conference , Human et Mouse

— Dans le cadre des expérimentations menées, les tests de comparaison ont été réalisés avec l'aide d'un expert du domaine.

— Nous avons comparé nos résultats avec le résultat des experts du domaine pour calculer la mesure de précision.

— Le tableau ci-dessous présente les valeurs de précision avec 4 comme nombre de partition et seuil=0.8 :

	Iasted/Ekaw	Ekaw/Sigkdd	Cmt/ConfOf	Conference/ConfOf	Human/Mouse
Précision	1	1	0.8	0.87	0.82

TABLE 4.3 – Valeurs de précision sur les ontologies de test

— Le tableau montre que le Mapping effectué avec notre système donne des bonnes valeurs de précision par rapport à l'alignement de référence.

— Notre approches donne plus de (True Positives) plus que de (False Positives)

4.4.3 Evaluation du temps d'exécution

Dans cette section on procède à l'évaluation de notre approche par rapport au temps d'exécution

4.4.3.1 Résultats expérimentaux et discussion

Cette série de tests est faite avec un pc portable i5 , 8gb de ram

	cmt/confOf	Conference/confOf	Ekaw/sigkdd	Iasted/ekaw	Human/Mouse
Temps d'exécution de notre système	2.6 s	5.01 s	5.88 s	12.83 s	3035.8 s
Temps d'exécution d'un alignement sans partitionnement	5.22 s	9.15 s	9.29 s	16.26 s	9230.2 s

TABLE 4.4 – Valeurs de temps d'exécution

— On remarque à partir du tableau 4.4 que l'alignement entre cmt et confOf, et l'alignement entre Conference et confOf et l'alignement entre Ekaw et sigkdd le temps d'exécution est 2 fois plus rapide que le temps d'exécution sans partitionnement

— On remarque aussi que l'alignement de deux ontologies larges (Human et Mouse) se fait 3 fois plus rapidement dans notre système avec une précision de 0.82, donc notre système améliore énormément le temps d'exécution du processus de Mapping en gardant la précision.

4.4.4 Bilan

A travers les tests que nous avons effectués, nous concluons que notre approche proposée est bonne, Cela est remarquable à partir des valeurs des mesures dévaluation utilisées dans les séries de tests effectuées. Le partitionnement donne des clusters bien séparés à travers la mesure de séparations, et la valeur de la précision de notre système pour l'alignement sur différentes ontologies est bonne, et ce, en réduisant le temps d'exécution considérablement.

4.5 Conclusion

Dans ce chapitre, nous avons présenté l'implémentation de notre système d'alignement d'ontologies proposé et les différents outils utilisés. Nous avons aussi présenté les résultats d'évaluation en utilisant la mesure de séparation sur le partitionnement , la précision sur le Mapping et le temps d'exécution .

Conclusion générale

Quand les ontologies sont de très grandes tailles, l'efficacité des méthodes classiques d'alignement des ontologies diminue considérablement en raison de redondance de calculs des mesures de similarité, L'objectif de ce projet était de réaliser un système d'alignement d'ontologies qui se base sur le partitionnement sans redondance de calculs en utilisant la structure de l'ontologie

Dans les deux premiers chapitres, nous nous sommes intéressés à ce qu'était une ontologie. Pour cela, nous sommes partis des origines philosophiques du terme pour ensuite définir son sens en ingénierie des connaissances, nous avons aussi parlé de composants des ontologies et les différents langages de présentations de ces dernières. Ensuite, nous avons entamé l'alignement et le Mapping et leurs dimensions.

Une bonne partie du deuxième chapitre a été consacrée aux systèmes d'alignement des ontologies qui existent. Cette partie a été finie par une étude comparative en étudiant tous les points qui peuvent servir à mener notre travail.

Dans le troisième chapitre , nous avons proposé une nouvelle approche de partitionnement des ontologies qui se base sur la structure d'ontologie. Cette approche est dans le but d'améliorer la qualité de l'alignement et réduire le temps d'exécution en éliminant les redondances de calcul de similarité.

Sur la base de cette approche, nous avons implémenté notre système d'alignement. afin de qualifier le travail effectué et de montrer l'efficacité de notre approche avec des ontologies de différentes tailles. Des tests sur le partitionnement et l'alignement ont été effectués en utilisant la mesure d'évaluation 'séparation' et ' la précision' et aussi le temps d'exécution. Finalement, nous avons discuté les résultats obtenus en soulignant les points positifs de cette expérimentation.

Comme perspectives de notre travail, nous voudrons améliorer le système de manière pour qu'il soit applicable sur des ontologies qui contiennent des concepts composés, c.-à-d. lorsqu' un même concept est représenté par deux termes différents et qui sont composés.

Enfin, ce travail a été une expérience très enrichissante qui nous a permis d'apprendre des tech-

niques que ce soit du côté de programmation ou bien théorique et d'apporter des tentatives de solution à un problème d'actualité.

Bibliographie

[Abolhassani, 2006] Abolhassani H., B.B. Hariri, S. H. Haeri, On Ontology Alignment Experiments, Webology, **3(3)**, September, 2006.

[Agarwal, 1995] Agarwal S., Keller A. M., Wiederhold G., Saraswat K., Flexible relation : an approach for Integrating data From Multiple, Possibly Inconsistent Databases, IEE Int. Conf. On Data Engineering, Taipei (Taiwan), 1995.

[Alasoud, 2010] Alasoud Khalifa Ahmed, A Multi-Matching Technique for Combining Similarity Measures in Ontology Integration, Thèse de doctorat, Concordia, Université de Montréal, Québec, Canada. 2010.

[Alaya et al., 2012] Alaya, N., Yahia, S. B., Lamolle, M., de Montreuil, L. I. (2012). Modlson : une nouvelle approche de modularisation d'ontologies à grande échelle. In EGC (pp. 279-284).

[Algergawy et al., 2014] Algergawy, A., Babalou, S., Kargar, M. J., Davarpanah, S. H. (2015, September). Seecont : A new seeding-based clustering approach for ontology matching. In East European Conference on Advances in Databases and Information Systems (pp. 245-258). Springer, Cham.

[Arpirez et al., 1998] Arpírez, J. C., Gómez-Pérez, A., Lozano-Tello, A., Pinto, H. S. A. N. (2000). Reference ontology and (ONTO) 2 agent : the ontology yellow pages. Knowledge and Information Systems, 2(4), 387-412.

[Bach, 2006] Bach, T. L. (2006). A Translation Approach to Portable Ontology specications Knowledge System Laboratory Stanford. Article.1993.

[Bachimont, 2000] Bachimont, B. (2000). Engagement sémantique et engagement ontologique : conception et réalisation d'ontologies en ingénierie des connaissances. Ingénierie des connaissances : évolutions récentes et nouveaux défis, 305-323.

[Benchmark OAEI 2020] . Ontologie de tests des benchmark de l'OAEI Retrieved from http ://oaei.ontologymatching.org/2020/

[Bernaras et al., 1996] Bernaras, A., Laresgoiti, I., Bartolome, N., Corera, J. (1996, February). An ontology for fault diagnosis in electrical networks. In Proceedings of International Conference on Intelligent System Application to Power Systems (pp. 199-203). IEEE.

[Borst, 1997] Borst . construction of engineering ontologies. PHD thesis. centre for comunica and information.1997.

[Blázquez et al., 1998] M. Blázquez et al . Building Ontologies at the Knowledge Level using the Ontology Design Environment. Laboratorio de Inteligencia Articial Facultad de Informática, Universidad Politécnica de Madrid.

[Borgo et al., 1996] Borgo, S., Guarino, N., Masolo, C. (1996, August). Stratified ontologies : the case of physical objects. In Proceedings of ECAI-96 Workshop on Ontological Engineering (pp. 5-15). ECAI.

[Charlet et al., 2003] Jean Charlet et al. Web sémantique.livre . 2003

[David et al., 2006] David, J. (2006). AROMA : une méthode pour la découverte d'alignements orientés entre ontologies à partir de règles d'association (Doctoral dissertation).

[De Bruijn et al., 2006] De Bruijn, J., Ehrig, M., Feier, C., Martín-Recuerda, F., Schare, F., and Weiten, M. (2006). Ontology mediation, merging and aligning. Semantic Web Technologies, pages 95–113.

[Djedidi et al., 2010] Djedidi, R., Aufaure, M. A. (2010, February). ONTO-EVO A L an ontology evolution approach guided by pattern modeling and quality evaluation. In International Symposium on Foundations of Information and Knowledge Systems (pp. 286-305). Springer, Berlin, Heidelberg.

[Doan et al., 2002] Doan, A., Madhavan, J., Domingos, P., Halevy, A. (2002, May). Learning to map between ontologies on the semantic web. In Proceedings of the 11th international conference on World Wide Web (pp. 662-673).

[Elbyed, 2009] Elbyed, A. (2009). ROMIE, une approche d'alignement d'ontologies à base d'instances (Doctoral dissertation, Evry, Institut national des télécommunications).

[Euzenat et al., 2004] Euzenat, J., Loup, D., Touzani, M., Valtchev, P. (2004, November). Ontology alignment with OLA.

[Euzenat et al., 2007] Jérôme Euzenat · Pavel Shvaiko, Ontology Matching, Springer, livre, 2007.

[Euzenat, 2007] Euzenat, J., Shvaiko, P. (2007). Ontology matching (Vol. 18). Heidelberg : Springer.

[Euzenat, 2008] Jérôme Euzenat. Quelques pistes pour une distance entre ontologies .Article. 8èmes Journées Francophones pour Extraction et Gestion des Connaissances .2008.

[Flouris et al., 2006] Flouris, G., Plexousakis, D., and Antoniou, G. (2006). A Classication of Ontology Change. In Semantic Web Applications and Perspectives (SWAP), volume 201. CEUR-WS.org.

[Furst, 2004] Fürst, F. (2004). Contribution à l'ingénierie des ontologies : une méthode et un outil d'opérationalisation (Doctoral dissertation, Nantes).

[Gruber, 1993] Gruber, T. R. (1993). A translation approach to portable ontology specifications. Knowledge acquisition, 5(2), 199-220.

[Hubert et al., 1985] Hubert, G., Mothe, J., Ralalason, B., Ramanonjisoa, B. (1985, May). Modèle d'indexation dynamique à base d'ontologies. In CORIA (pp. 169-184).

[Lera et al., 2008] Isaac Lera et al.Quick Ontology Mapping Algorithm for distributed environments.Article.2008

[Tang et al., 2006] Jie Tang, Juanzi Li. Using Bayesian Decision for Ontology Mapping. Department of Computer Science and Technology.Article.2006

[Kachroudi et al., 2013] Kachroudi, M., Zghal, S., Yahia, S. B. (2013, January). Paramétrage intelligent de l'alignement d'ontologies par l'intégrale de Choquet. In EGC (pp. 377-382).

[kalfoglou et al., 2003] Kalfoglou, Y., Schorlemmer, M. (2003). Ontology mapping : the state of the art. The knowledge engineering review, 18(1), 1-31.

[Kasri, 2010] Soumaya Kasri, « Etude des techniques d'alignement des ontologies : proposition d'un algorithme de passage à l'échelle ». Thèse Magister en Informatique université 20 Août 1955 Skikda.2010

[Ki-Heon et al., 2010] Kim, Ki-Heon Yang, Jae Choi, Jae-Hun Yang, Kyungah Ha, Young-Guk. (2007). A Semantic Inheritance/Inverse-Inheritance Mechanism for Systematic Bio-Ontology Construction. Conference proceedings : ... Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Conference. 2007. 398-401. 10.1109/IEMBS.2007.4352308.

[Klein, 2001] Klein, M. (2001). Combining and relating ontologies : an analysis of problems and solutions. In IJCAI-01 Workshop on Ontologies and Information Sharing, pages 53–62. CEUR-WS.org.

[Leacock et al., 1998] Leacock, C., Chodorow, M. (1998). Combining local context and WordNet similarity for word sense identification. WordNet : An electronic lexical database, 49(2), 265-283.

[Massmann et al., 2011] Aumueller, D., Do, H. H., Massmann, S., Rahm, E. (2005, June). Schema and ontology matching with COMA++. In Proceedings of the 2005 ACM SIGMOD international conference on Management of data (pp. 906-908).

[Mellal, 2007] Mellal, N. (2007). Réalisation de l'interopérabilité sémantique des systèmes, basée sur les ontologies et les flux d'information (Doctoral dissertation, Chambéry).

[Mizoguchi, 1997] Mizoguchi, R., Ikeda, M., Sinitsa, K. (1997, August). Roles of shared ontology in AI-ED Research. In Proc. of AI-ED (Vol. 97, pp. 537-544).

[Monge et al., 1996] Monge, A. E., Elkan, C. (1996, August). The Field Matching Problem : Algorithms and Applications. In Kdd (Vol. 2, pp. 267-270).

[Neches et al., 1991] R.Neches et al. Enabling technology for knowledge sharing, Article, 1991.

[Rahm et al, 2001] Rahm, E., Bernstein, P. A. (2001). A survey of approaches to automatic schema matching. the VLDB Journal, 10(4), 334-350.

[Stephen et al., 2002] Stephen L et al . Mapping Ontologies into Cyc. Article. 2002

[Shvaiko et al., 2005] Shvaiko, P., Euzenat, J. (2005). A survey of schema-based matching approaches. In Journal on data semantics IV (pp. 146-171). Springer, Berlin, Heidelberg.

[Tang, 2006] Tang, J., Li, J., Liang, B., Huang, X., Li, Y., Wang, K. (2006). Using Bayesian decision for ontology mapping. Journal of web semantics, 4(4), 243-262.

[Thomas et al.,1993] Thomas R et al. A Translation Approach to Portable Ontology specications Knowledge System Laboratory Stanford. Article.1993.

[Touzani, 2005] Moharned TOUZANI« Alignement des ontologies OWL-Lite », Faculté des arts et des sciences ,Avril,2005

[Umer et al., 2012] Umer, Q. and Mundy, D. (2012). Semantically Intelligent Semi-Automated Ontology Integration. In Proc. of the World Congress on Eng.

[Uschold et al., 1995] M.Uschold et al. Towards a methodolgy for building ontologies. In Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing. Montreal, Canada . 1995.

[Vialle, 2017] Vialle, S., Stephane, C. (2017). Big Data : Informatique pour les données et calculs massifs.

[Ziani et al., 2010] Ziani, M., Boulanger, D., Talens, G. (2010, May). Système d'aide à l'alignement d'ontologies métier. In INFORSID (pp. 345-360).

[Ziemba et al., 2015] Ziemba, P., Jankowski, J., Watrobski, J., Wolski, W., and Becker, J. (2015). Integration of Domain Ontologies in the Repository of Website Evaluation Methods. In Federated Conference on Computer Science and Information Systems (FedCSIS), pages 1585–1595. IEEE