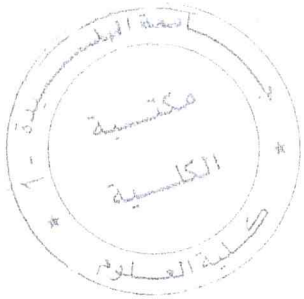
République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Saad DAHLAB Blida 01



Faculté des Sciences

Département d'Informatique

Mémoire Présenté par :

BENKADDOUR Aïcha

En vue d'obtenir le diplôme de master

Domaine : Mathématique et informatique

Filière : Informatique

Spécialité : Informatique

Option : Ingénierie de logiciel

Thème :

Partitionnement d'ontologies volumineuses en utilisant le datamining

Soutenu le : 28/09/2018, devant le jury composé de :

M^me^	FAREH	Promotrice
M^me^	BOUSTIA	Présidente
M^me^	MADANI	Examinatrice

2017/2018

Table des matières

Introduction Générale

Chapitre I : Les Ontologies

Chapitre II : Partitionnement d'Ontologies

Chapitre III : Approche Proposée

Chapitre IV : Implémentation et Evaluation

Conclusion Générale

Liste des figures

Liste des tableaux

Remerciement

Tout d'abord, je tiens à rendre grâce à dieu tout puissant pour m'avoir donné le courage et la détermination nécessaire pour finaliser ce travail et le mener à terme.

En second lieu, Je tiens à remercier ma promotrice Mme Fareh qui a endossé son rôle de de la meilleure façon qui soit. Je retiendrai son aide précieuse, ces conseils avisés, ces idées riches mais aussi sa sympathie et ses encouragements. Je prie Allah de lui rendre grâce pour avoir fait de mon travail avec elle, un réel honneur et un grand plaisir.

Je tiens à remercier chaleureusement et affectueusement mon cher mari Hocine et mon cher fils Sohaib pour leurs aides et encouragements.

Un grand remerciement à Melle Ali Khoudja Meriem et Mme Mezzi Melyara qui m'ont beaucoup aidé et soutenu pour la finalisation de ce travail.

Un grand remerciement aussi à mes chères collègues de la PGRS Fethia,Yasmine et Meriem, et mes responsables pour leurs aides et surtout leurs patience, je remercie aussi Mr Djamel Zebbiche. Puisse dieu les aider tous à atteindre tous leurs buts.

A tous ceux et à toutes celles dont les noms n'apparaissent pas sur cette page, qu'ils demeurent convaincus, que je ne les ai pas oublié et qu'ils soient assurés de ma profonde gratitude.

Merci.

Benkaddour Aïcha

Dédicace

C'est avec un immense plaisir que je dédie ce travail à

mon cher mari et mon cher fils.

Que dieu les garde et leur procure la santé et le bonheur.

A mon cher père.

Ainsi qu'à ma famille et ma belle-famille.

Résumé

Les ontologies sont confrontées de façon continue à un problème d'évolution, car les termes techniques relatifs à un domaine particulier changent et évoluent de façon perpétuelle, et comme l'usage du web sémantique est en pleine expansion le besoin des ontologies est devenu incontournable. Donc plusieurs opérations qui peuvent être appliquées sur les ontologies comme l'alignement, la fusion, le partage, l'intégration des données et la maintenance d'ontologies deviennent des tâches plus difficiles. Avec la complexité des ontologies et le passage à l'échelle, le partitionnement d'une ontologie en plusieurs partitions est une solution pour répondre à ces problèmes et difficultés.

Dans ce mémoire, nous proposons une méthode de partitionnement d'ontologie, par l'utilisation de l'algorithme de clustering k-means, qui sera combiné avec d'autres algorithmes. Nous allons utiliser différentes mesures de similarité en proposant une combinaison de ces dernières pour obtenir une mesure qui représente la distance utilisée dans ces algorithmes selon notre type de données. Notre méthode est testée sur des ontologies de différentes tailles.

Mots clés : Algorithme de clustering, ontologies OWL, mesures de similarité, sémantique, K-Means, k-medoide.

Abstract

Ontologies are continually confronted with a problem of evolution, because the technical terms relating to a particular domain change and evolve in a perpetual way, and as the use of the semantic web is in full expansion, the need for ontologies has become unavoidable. So several operations that can be applied to ontologies like alignment, merging, sharing, data integration, and ontology maintenance become more difficult tasks. With the complexity of ontologies and scalability, the partitioning of an ontology into several partitions is a solution to answer these problems and difficulties.

In this thesis, we propose an ontology partitioning method, using the k-means clustering algorithm, which will be combined with other algorithms. We will use different similarity measures by proposing a combination of these to obtain a measure that represents the distance used in these algorithms according to our data type. Our method is tested on ontologies of different sizes.

Key words : Clustering algorithm, OWL ontologies, similarity measures, semantics, K-Means, k-medoid

ملخص

تواجه الأنطولوجيا باستمرار مشكلة تطور ، لأن المصطلحات التقنية المتعلقة بمجال معين تتغير وتتطور بطريقة دائمة ، وبما أن استخدام الشبكة الدلالية أصبح في توسع تام ، فإن الحاجة إلى علم الأنطولوجيا أصبحت لا يمكن تجنبها. لذلك فإن العديد من العمليات التي يمكن تطبيقها على الأنطولوجيا مثل المواءمة ، والدمج ، والمشاركة ، وتكامل البيانات ، وصيانة الأنطولوجيا تصبح مهام أكثر صعوبة. مع تعقيد الأنطولوجيا وقابلية التوسع ، فإن تقسيم الأنطولوجيا إلى عدة أقسام هو حل للإجابة على هذه المشكلات والصعوبات.

في هذه الرسالة ، نقترح طريقة تقسيم الأنطولوجيا ، باستخدام خوارزمية التجميع k-means ، والتي سيتم دمجها مع خوارزميات أخرى. سنستخدم مقاييس تشابه مختلفة من خلال اقتراح مجموعة من هذه الأخيرة للحصول على مقياس يمثل المسافة المستخدمة في هذه الخوارزميات وفقا لنوع البيانات المستخدمة لدينا. سيتم اختبار الطريقة المقترحة على عدة أنطولوجيات ذات أحجام مختلفة.

الكلمات المفتاحية : خوارزمية التجميع ، الأنطولوجيا OWL ، مقاييس التشابه ، علم الدلالة ، K-Means، k-medoid

Introduction Générale

1. Contexte de travail

En représentation des connaissances, la définition de l'ontologie c-à-d la description formelle d'entités et de leurs propriétés, relations, contraintes et comportement dans un domaine particulier, offre une base solide au développement de nouvelles méthodes d'application de raisonnement sur les connaissances. Outre son apport en matière de réutilisabilité et de partage de connaissances, l'ontologie permet de définir un vocabulaire précis, sur lequel est basée la communication entre les différents acteurs d'un projet.

Actuellement, les ontologies constituent un enjeu stratégique dans la représentation et la modélisation des connaissances, elles permettent aussi la recherche d'informations, le traitement du langage naturel, l'intégration intelligente d'informations et la gestion des connaissances.

Les ontologies se développent à toute vitesse dans différents domaines d'application réels, elles deviennent de plus en plus volumineuses en terme de nombre de concepts (milliers de concepts) d'axiomes entre ces concepts et de règles.

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 l'alignement, la fusion et l'intégration.

2. Problématique

Dans les domaines d'applications réelles, les ontologies devenant de plus en plus volumineuses. Le partitionnement d'une ontologie en plusieurs partitions indépendantes les unes des autres sert à faciliter différentes opérations sur les ontologies comme la fusion, le mapping, l'intégration et l'alignement...etc.

Quand les ontologies sont de très grandes tailles, par exemple en agronomie et en médecine, des ontologies comportent plusieurs dizaines de milliers de concepts, l'efficacité des méthodes classiques pour toute opération réalisée sur les ontologies diminue considérablement en terme de la précision et de la qualité des résultats obtenus.

Le partitionnement, donc, est la meilleure solution pour résoudre ce problème. Pour cette raison, nous allons consacrer notre travail à répondre aux points suivants :

- Comment effectuer le partitionnement ?

- Comment utiliser et adapter les algorithmes de clustering pour le partitionnement d'ontologies ?
- Comment représenter la distance utilisée dans les algorithmes de clustering dans le domaine d'ontologie, en gardant la sémantique ?
- Quels sont les algorithmes utilisés pour la construction des partitions ?

3. Objectifs

L'objectif de ce projet consiste à concevoir et de réaliser un système de partitionnement d'ontologie de grande échelle en utilisant les techniques de clustering, en effet, de plus en plus, des domaines représentent leurs connaissances à travers des ontologies réelles de grande taille. Elles contiennent des milliers de concepts et de relations entre ces derniers. Par conséquent, 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, afin de n'avoir à traiter que des blocs de taille raisonnable.

Le partitionnement des données est une tâche importante en data mining, elle divise un ensemble de données en plusieurs sous-ensembles, ces derniers appelés groupes ou clusters. Ces clusters sont caractérisés idéalement par une forte similarité à l'intérieur et une forte dissimilarité entre les membres de différents groupes. Ces techniques sont particulièrement prometteuses lorsqu'il s'agit de grandes ontologies, comprenant des centaines et de milliers d'entités.

Dans les parties suivantes, nous allons expliquer en détail les étapes effectuées pour atteindre notre objectif qui est le partitionnement en utilisant le data mining, nous allons exprimer comment :

- Extraire les composants d'une ontologie ;
- Utiliser et adapter les algorithmes de clustering : le mal classé, le k-means et le k-medoïde ;
- Calculer la distance utilisée dans notre approche;

Et ce, pour obtenir à la fin, de bonnes partitions.

4. Organisation du mémoire

Afin d'atteindre nos objectifs, ce mémoire est organisé comme suit :

– Chapitre I : Les ontologies

Ce chapitre présente une vue d'ensemble sur les concepts de base des ontologies. Nous y aborderons l'origine, les définitions, les composants, langages de représentation et les différentes opérations qui peuvent être appliquées sur les ontologies.

– Chapitre II : Partitionnement d'ontologies

Ce chapitre se concentre autour de la présentation du concept de partitionnement des ontologies, les travaux qui ont été menés dans ce domaine et plus précisément, les principaux buts de ce dernier. Les différentes techniques et outils de partitionnement seront présentés tout en mettant l'accent sur les limites actuelles de ceux-ci.

– Chapitre III : Approche proposée

Ce chapitre comprend le modèle et l'approche proposée, en commençant par une étude comparative de quelques méthodes existantes pour le partitionnement ce qui va nous aider à élaborer notre système avec les différentes étapes détaillées et les différents algorithmes utilisés.

– Chapitre IV: Implémentation et évaluation

Ce chapitre détaille la mise en application de notre approche. Nous verrons donc le travail qui a été accompli et le système réalisé ainsi que les résultats obtenus.

En plus, nous détaillons l'ensemble des métriques de qualité sur lesquels se base l'évaluation de nos résultats de partitionnement obtenus.

- **Conclusion générale**

En fin, la conclusion de ce mémoire synthétisera nos principaux chapitres et donnera quelques perspectives à notre travail.

Chapitre I :

Les ontologies

1. Introduction

L'utilisation de connaissances en informatique a pour but de ne plus faire manipuler en aveugle des informations à la machine mais de permettre un dialogue, une coopération entre le système informatique et les utilisateurs. Pour cela, le système doit avoir accès non seulement aux termes utilisés par l'être humain mais également à la sémantique qui leur est associée, afin qu'une communication efficace soit possible.

Les ontologies visent à représenter cette connaissance en étant à la fois interprétables par l'homme et par la machine. Ces dernières sont un sujet de recherche dans diverses communautés notamment l'ingénierie des connaissances, la recherche d'information et le traitement du langage naturel, les systèmes d'information coopératifs, l'intégration intelligente d'information et la gestion des connaissances. Elles fournissent une connaissance partagée et commune sur un domaine qui peut être échangée entre des personnes et des systèmes hétérogènes. Elles ont été définies en intelligence artificielle pour faciliter le partage des connaissances et leur réutilisation.

Dans ce chapitre, nous attachons à décrire les différentes définitions du concept d'ontologie, nous verrons aussi les différents éléments dont elle est composée et les relations possible entre ces éléments.

2. Utilité d'ontologies

Les ontologies ont été largement appliquées dans plusieurs domaines pour fournir des structures communes, partager les connaissances et assurer l'interopérabilité entre des systèmes hétérogènes. Les ontologies sont nécessaires pour une réelle intégration sémantique des sources d'information et pour permettre aussi une description précise d'un univers de discours, exprimée dans un langage utilisable.

3. Définition d'ontologie

Il existe plusieurs définitions de l'ontologie, tel que la définition qui a été proposé par Neeches et ses collègues [Neeches et al., 1993] en 1993 à savoir « une ontologie définit les termes et les relations de base de vocabulaire d'un domaine, ainsi que les règles qui indiquent comment combiner les termes et les relations de façon à pouvoir étendre le vocabulaire».

En 1993, aussi, GRUBER [Gruber, 1993] a proposé une autre définition qui est : «une ontologie est une spécification explicite d'une conceptualisation». En 1995, GUARINO [Guarino, 1997] a modifié légèrement la définition de GRUBER, et l'a définit par : « une ontologie est une spécification partielle et formelle d'une conceptualisation». En 1997, ces deux dernières définitions ont été regroupées dans celle de BORST [Borst, 1997] comme « une ontologie est définie comme étant une spécification formelle et explicite d'une conceptualisation partagée», tel que :

- ✓ Spécification explicite : signifie que les concepts, les propriétés, les relations, les fonctions, les restrictions et les axiomes de l'ontologie sont définis de façon déclarative ;
- ✓ Formelle : une ontologie doit être traduite en langage interprétable par une machine ;
- ✓ Conceptualisation : réfère à un modèle abstrait d'un phénomène du monde réel en identifiant les concepts appropriés à ce domaine ;
- ✓ Partagée : réfère au fait qu'une ontologie capture la connaissance consensuelle, c.à.d non réservée à quelques individus, mais partagée par un groupe ou une communauté.

4. Composants d'ontologie

Les ontologies définissent le sens des termes et les relations entre eux, elles leur fournissent un vocabulaire commun. Malgré que les ontologies peuvent être très différentes surtout au niveau du traitement de leurs composantes de base telles que les choses, les relations…, elles caractérisent un même univers.

Les composants d'une ontologie sont :

4.1. Concepts

Un concept est une abstraction réunissant un certain nombre d'entités du « monde réel » (Aristote) qui sont ses instances.

Un concept est désigné par un label (terme, nom). Ce dernier est défini par ses différentes relations avec les autres concepts de l'ontologie, ses attributs et les contraintes qui lui sont associées, et il peut représenter plusieurs objets.

Un concept peut donc décrire une tâche, une fonction, une action, une stratégie, un processus de raisonnement, etc.

Il peut se définir comme une entité composée de trois éléments distincts :

Terme : Exprimant le concept en langue

Synonyme : plusieurs termes représentent le même concept.

Ambiguïté : plusieurs concepts représentés par le même terme.

Intention : Appelée également « notion », c'est l'ensemble de propriétés d'un concept.

Extension : Appelée également « réalisation », c'est les objets dénotés par le concept.

4.2. Relations et fonctions

Traduisent les associations (pertinentes) existant entre les concepts du domaine, une relation est définie comme une notion de lien entre des entités.

Une relation est déterminée par :

Terme : C'est la représentation de la relation en langue.

Extension : L'ensemble des réalisations effectives d'une relation entre concepts.

Intention : C'est l'ensemble de propriété et des attributs de la relation.

Signature : C'est l'ensemble des concepts pouvant être liés par la relation.

Les fonctions sont des cas particuliers de relations dans lesquelles un élément de la relation est défini à partir des autres éléments.

Les axiomes sont des expressions qui sont toujours vraies, ils permettent de contraindre les valeurs de classes ou d'instances.

4.3. Instances

Les instances représentent les éléments extensionnels spécifiques des concepts et des relations au domaine du problème modélisé.

5. Langages de représentation des ontologies

Les langages de représentation des ontologies sont des langages formels. Ils permettent l'expression d'ensemble de concepts et leurs relations conceptuelles, leurs objectifs est de :

– décrire les ressources et services présents sur le Web,

– exprimer des modèles abstraits (ontologies) de ces descriptions,

– échanger / partager ces descriptions sur le Web

Il existe plusieurs langages informatiques spécialisés dans la création et la manipulation des ontologies. Parmi ces langages nous citons :

5.1. KIF

KIF est un langage basé sur les prédicats du premier ordre avec des extensions pour représenter des définitions et des méta-connaissances, la logique du premier ordre étant un langage de bas niveau pour l'expression d'ontologies.

Une extension du langage KIF, ONTOLINGUA, est utilisée dans le serveur d'édition d'ontologies, ONTOLINGUA du même nom.

5.2. RDF, RDF(S)

Le langage Resource Description Framework (RDF) est le standard émergeant proposé par le W3C pour la représentation et l'échange de métadonnées sur le web, il est basé sur un modèle de triplet (ressource, propriété, valeur) et possède une syntaxe XML. Le langage RDF Schéma (RDFS) est le standard accompagnant RDF qui permet de représenter les connaissances ontologiques relatives aux annotations.

5.3. DAML +OIL

DAML+OIL est un langage construit sur des normes précédentes du W3C telles que RDF et RDF Schéma, et étend ces langages avec des primitives de modélisation plus riches. DAML+OIL a été conçu à partir du langage d'ontologie DAML-ONT (DARPA Agent Modelling Language-Ontology) en vue de combiner plusieurs composants du langage OIL.

5.4. OWL (Ontology Web Language)

OWL est l'extension du langage RDF(S). Il est développé par le groupe de travail sur le web sémantique du W3C qui est basé sur le langage DAML+OIL [McGuiness et al., 2002]. Il est aujourd'hui, le standard recommandé par la W3C pour le développement des ontologies et certainement le langage le plus utilisé. Il est fractionné en trois sous-langages qui sont présentés, ci-dessous, selon leur expressivité et complexité croissante :

1. OWL-Lite

C'est le moins expressif des sous-langages d'OWL. En fait, il offre la possibilité d'avoir une hiérarchie de classification et des contraintes simples (contrainte de cardinalité limité de 0 ou 1). L'avantage de ce langage est d'avoir une complexité formelle faible par rapport aux deux autres sous-langages d'OWL.

2. OWL-DL (Description Logic)

Il est basé sur la logique de description. C'est le sous-langage qui offre une expressivité maximale en garantissant la complétude et la décidabilité informatique, ce qui signifie que toutes les conclusions sont calculables dans un temps raisonnable.

3. OWL-Full

Il offre l'expressivité la plus complète, cependant, il ne possède pas les propriétés de complétude et de la décidabilité des calculs liées à l'ontologie. De plus, à ce jour, il n'existe pas d'implémentation complète d'OWL-Full.

6. Opérations sur les ontologies

Dans cette section, nous présentons quelques opérations qu'on peut appliquer sur les ontologies :

6.1. Intégration des ontologies

L'ontologie vise à constituer une représentation du monde réel qui puisse être acceptée par tous les membres d'une communauté mais les possibilités de représentation sont variables et par conséquent les ontologies diffèrent malgré les efforts de normalisation. Les différentes conceptualisations engendrent l'hétérogénéité des ontologies, une problématique de plus en plus explorée en raison de la multiplication de leur nombre et l'augmentation de leur accessibilité.

L'intégration d'ontologies représente une solution pour cette limite, elle permet la possibilité d'interopérabilité entre les ontologies.

L'intégration des ontologies signifie la construction d'une nouvelle ontologie en utilisant d'autres ontologies disponibles. Ces différentes ontologies font partie de la nouvelle ontologie.

6.2. Alignement d'ontologies

L'alignement d'ontologies est le processus de mise en correspondance sémantique des entités qui le composent. Le processus est exécuté selon une stratégie ou une combinaison de techniques de calcul de mesures de similarité utilisant un ensemble de paramètres (ex. paramètres de pondération, seuils, etc.) et un ensemble de ressources externes (ex. thésaurus, lexique. . .). Au final, nous obtenons un ensemble de liens sémantiques reliant les entités qui composent les ontologies [Euzenat et al., 2007].

L'alignement permet d'amener deux ou plusieurs ontologies hétérogènes à un "accord mutuel" en les rendant ainsi consistantes et mutuellement cohérentes. L'alignement d'ontologies nécessite la transformation des ontologies impliquées en procédant à l'élimination des entités non pertinentes et au rajout des entités manquantes.

La figure suivante montre le principe d'alignement d'ontologies :

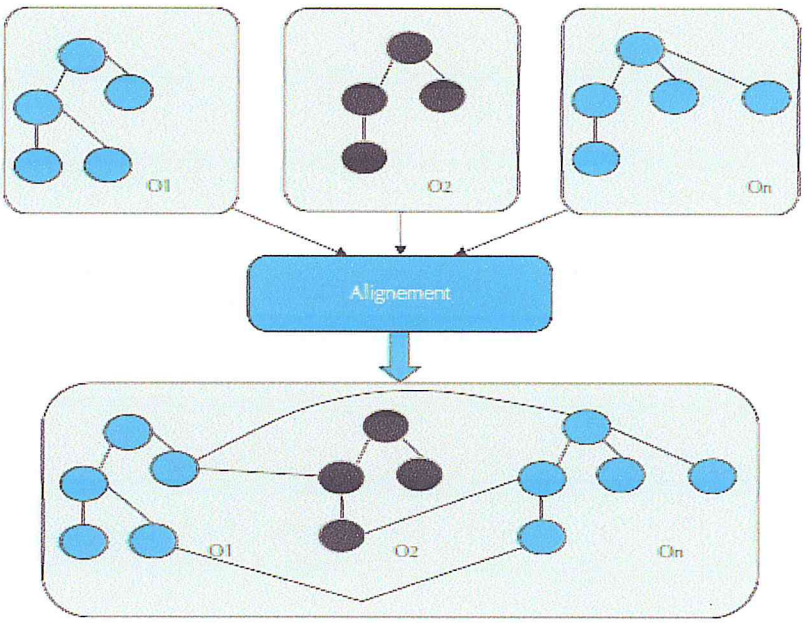


Figure 1 : Principe d'alignement d'ontologies [Mellal, 2007].

6.3. Fusion d'ontologies

La fusion d'ontologies est le processus de création d'une seule ontologie rassemblant les connaissances de deux ou plusieurs ontologies existantes et différentes qui décrivent le même sujet ou appartiennent au même domaine d'application. C'est donc la création d'une nouvelle ontologie, appelée l'ontologie fusionnée qui repose sur la capture des

connaissances des ontologies d'origine et les concepts supplémentaires nécessaires pour réaliser cette fusion. Le défi est alors d'assurer que toutes les correspondances et les différences entre les ontologies soient correctement prises en compte dans l'ontologie résultante [Elbyed, 09].

Ce type d'approche est généralement utilisé, dans le cadre d'intégration de données, pour obtenir une ontologie globale qui sert d'interface pour un certain nombre d'ontologies locales.

La figure suivante montre le principe de fusion d'ontologies :

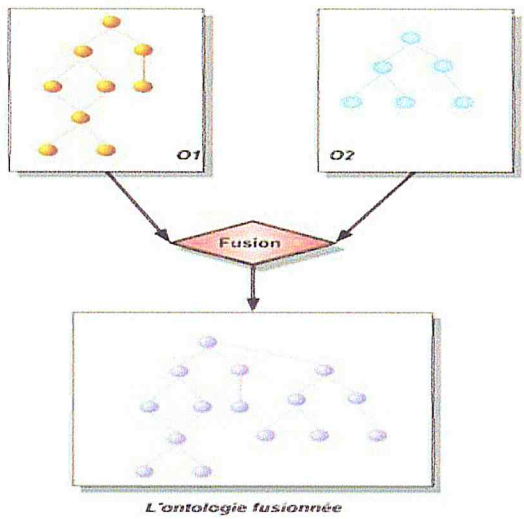


Figure 2 : Principe de la fusion d'Ontologies [Fürst, 2002].

6.4. Partitionnement d'ontologies

Le partitionnement d'ontologies divise une ontologie en un ensemble de sous-ensembles, chaque sous-ensemble étant appelé une partition. 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 [Zhang Cheng et al., 2007]

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.

6.5. Maintenance d'ontologies

Les ontologies sont confrontées de façon continue à un problème d'évolution, car les termes techniques relatifs à un domaine particulier changent et évoluent de façon perpétuelle. Par conséquent, les ontologies de domaine doivent être maintenues pour faire face aux incomplétudes et aux erreurs.

La maintenance de l'ontologie est une étape liée à la construction de l'ontologie, plus précisément, c'est la dernière phase des activités techniques de la construction. Elle consiste à évaluer l'ontologie produite en vérifiant qu'elle contient toutes les connaissances du domaine à modéliser. Elle consiste aussi à vérifier que l'ontologie est correctement formalisée. Dans le cas contraire, une mise à jour est réalisée pour corriger ces erreurs.

La maintenance de l'ontologie est une tâche très importante, car elle permet de mieux structurer le contenu sémantique de l'ontologie, et assurer une meilleure représentation des connaissances représentées par l'ontologie à l'aide des connaissances à jour, et garantir, par conséquent, l'utilité et la fiabilité de l'ontologie. Cette tâche est d'autant plus compliquée quand les ontologies à analyser sont de grande taille.

6.6. Mapping d'ontologies

Le mapping entre deux ontologies « Ontologie1 ; Ontologie2 » signifie que pour chaque entité dans une ontologie « Ontologie1 », il faut trouver l'entité correspondante dans l'ontologie « Ontologie2 » avec un sens équivalent ou le sens le plus proche. Une caractéristique importante de cette approche est qu'elle ne modifie pas les ontologies impliquées et qu'elle produit en sortie un ensemble de correspondances.

Le mapping d'ontologies a pour objectif la représentation des correspondances entre les ontologies. Cela permet, par exemple, d'interroger des bases de connaissances hétérogènes en utilisant une interface commune ou en transformant des données entre différentes représentations [Mellal, 2007].

La figure suivante montre le principe de mapping entre deux ontologies.

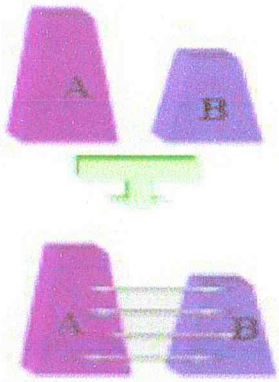


Figure 3 : Principe de mapping d'ontologies [IZZA, 2006].

Le matching d'ontologies est le processus de définition d'un ensemble de fonctions permettant de spécifier des « correspondances » entre termes [Xu et al., 2003] [He et al., 2003]. Le matching est le processus pour trouver le mapping.

7. Techniques de mapping

Nous présentons les différents éléments du processus de mapping entre ontologies :

7.1. Dimensions de l'alignement

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

a. ***L'input*** : est constitué essentiellement des structures destinées à être alignées, et qui peuvent être 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).

b. ***Le processus d'alignement*** : peut être considéré comme une fonction f, qui à partir d'une paire d'ontologies O et O', un alignement en entrée A (optionnel), un ensemble de paramètres p (ex : paramètres de pondération, seuils ...) et un ensemble de ressources externes r (ex : thesaurus, lexique...), détermine un alignement A' entre ces deux ontologies.

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

D'autre part, les deux orientations possibles d'un alignement entre deux ontologies (**O→O'** et **O'→O**), la multiplicité peut prendre les valeurs suivantes :

1 :1 - 1 : ? , ? : 1 , 1 :+ , +:1 , 1 :* , *:1 , ?:? , ?:+, +:? , ?:* , * :?, + :*, * :+, + :+, * :*.

Voici quelques exemples sur les configurations de multiplicité entre deux ontologies, constituée chacune de trois entités :

A' = f (O, O', A, p, r)

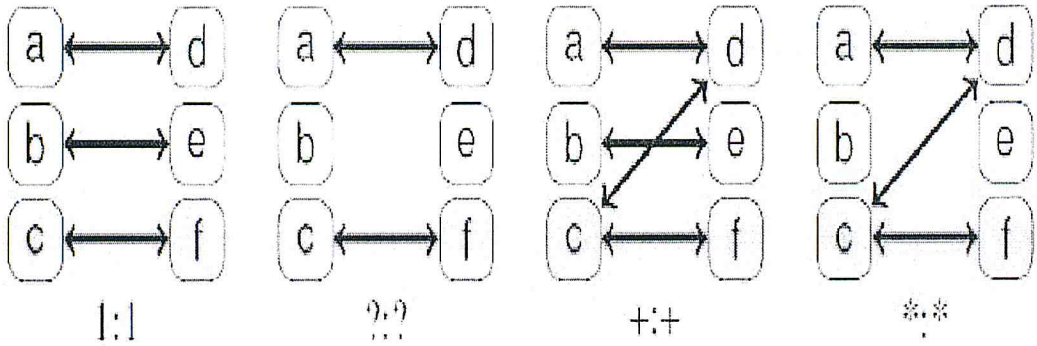


Figure 5 : Exemples de configurations de multiplicité entre 2 ontologies [Euzenat & Shvaiko, 2007]

b. Le format de l'output : Il existe plusieurs possibilités de représenter l'output, dont : OWL, XML, etc.

D'autres métadonnées peuvent être ajoutées au résultat final comme : l'algorithme d'alignement, sa date de création…

7.2. Similarité

Il s'agit de la similarité sémantique, qui est également appelée la proximité sémantique. Elle est déterminée grâce à l'association à des documents, des termes ou des entités, d'une métrique basée sur la similitude de leurs significations ou de leurs contenus sémantiques. La similarité est la quantité qui reflète la force du rapport entre deux objets ou deux caractéristiques.

Définition 1 La similarité $S : O \times O \rightarrow R$ est une fonction d'une paire d'entités à un nombre réel exprimant la similarité entre ces deux entités telle que:

- $\forall a, b \in O, S(a, b) \geq 0$ (positivité)
- $\forall a, b, c \in O, S(a, a) \geq S(b, c)$ et $S(a, a) = S(a, b) \Leftrightarrow a = b$ (auto similarité ou maximalité)

- $\forall a, b \in O, S(a, b) = S(b, a)$ (symétrie)
- $\forall a, b, c \in O, S(a, b) = S(b, c) \Rightarrow S(a, b) = S(a, c)$ (transitivité)
- $\forall a, b \in O, S(a, b) \leq \infty$ (finitude)

La dissimilarité est parfois utilisée au lieu de la similarité. Elle est définie de manière analogue à la similarité, sauf qu'elle n'est pas transitive :

Définition 2 La dissimilarité $DS : O \times O \rightarrow R$ est une fonction d'une paire d'entités à un nombre réel exprimant la dissimilarité entre ces deux entités telle que:

- $\forall a, b \in O, DS(a, b) \geq 0$ (positivité)
- $\forall a, b, c \in O, DS(a, a) \leq DS(b, c)$ et $DS(a, a) = 0$ (minimalité)
- $\forall a, b \in O, DS(a, b) = DS(b, a)$ (symétrie)
- $\forall a, b \in O, DS(a, b) \leq \infty$ (finitude)

La distance est une mesure utilisée aussi souvent que les mesures de similarité.

Elle mesure la dissimilarité de deux entités, elle est inverse de la similarité : si la valeur de la fonction de similarité de deux entités est élevée, la distance entre elle est petite et vice-versa. Elle est donc définie dans [Euzenat et al., 2004] comme suit :

Définition 3 La distance $D : O \times O \rightarrow R$ est une fonction de la dissimilarité satisfaisant la définitivité et l'inégalité triangulaire :

- $\forall a, b \in O, D(a, b) = 0 <=> a = b$ (définitivité)
- $\forall a, b, c \in O, D(a, b) + D(b, c) \geq D(a, c)$ (inégalité triangulaire)

Les valeurs de similarité sont souvent normalisées pour pouvoir être combinées dans des formules plus complexes. Si la valeur de similarité et la valeur de dissimilarité entre deux entités sont normalisées, notées S et DS, alors on a $S + DS = 1$.

Définition 4 Normalisation : Une mesure est dite normalisée si les valeurs calculées par cette mesure ne peuvent varier que dans un intervalle de 0 à 1. Ces valeurs calculées sont appelées valeurs normalisées. Les fonctions du calcul sont appelées fonctions normalisées et notées f.

Les mesures de la similarité, de la dissimilarité, de la distance peuvent être classées selon la nature des entités que l'on veut comparer : des termes, des chaînes de caractères, des structures, des instances (des individus des classes).

7.3. Mesures de similarité

La Figure 6 résume différentes mesures de similarité, catégorisées selon les techniques utilisées. Ce résumé est une synthèse des travaux présentés dans [Rahm & Bernstein, 2001], [Euzenat et al., 2004] et [Shvaiko & Euzenat, 2005].

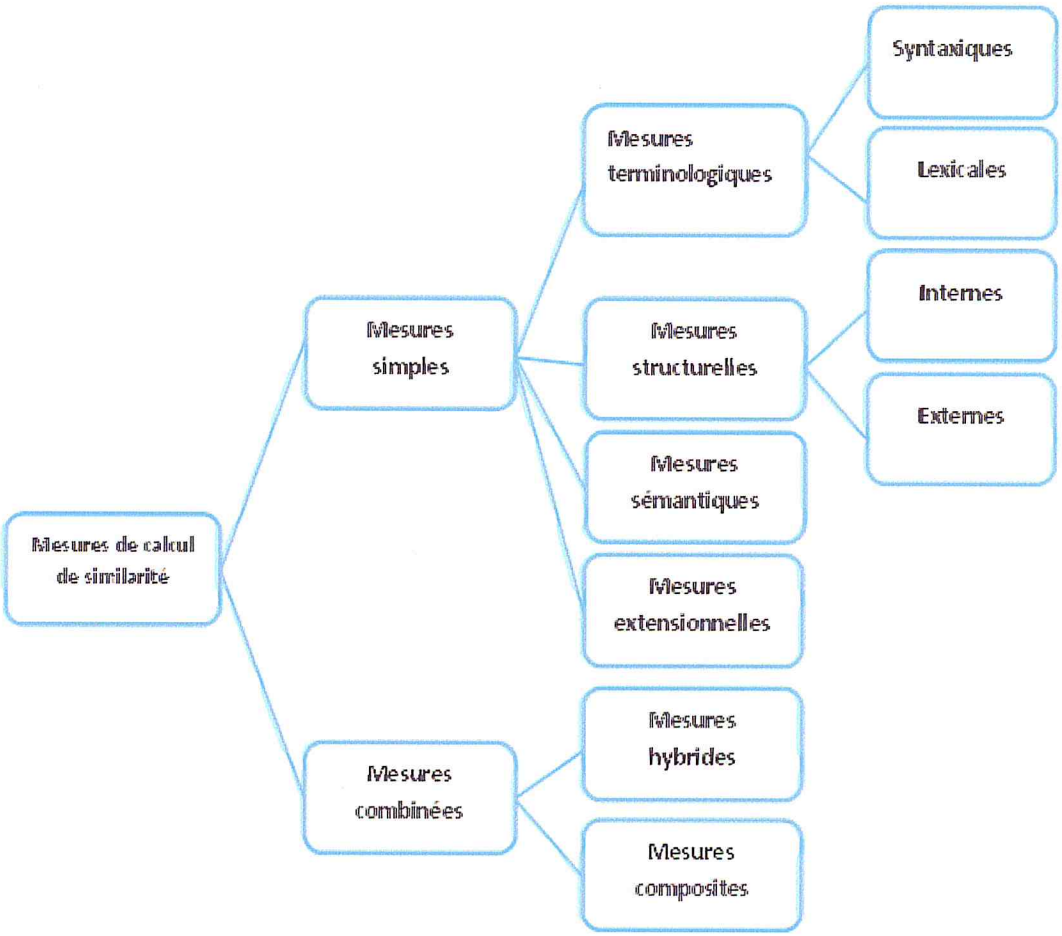


Figure 6 : Mesures de calcul de similarités [Euzenat, 2008].

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

7.3.1. Mesure simple

Elle regroupe les mesures suivantes :

a. Mesures terminologiques

Le calcul de la similarité terminologique est effectué entre les descripteurs d'entités comme les noms, les commentaires, les étiquettes et le sujet des entités.

Nous trouvons dans cette méthode deux approches : l'approche syntaxique et l'approche lexicale, appelée aussi linguistique [Monge et al., 1996] :

- Approche syntaxique : approche basées 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 estimation 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 d'hyponymes, mais aussi à des thésaurus et des ressources sémantiques ainsi qu'a des dictionnaires spécifiques aux domaines étudiés.

b. Mesures structurelles

Nous trouvons deux types de méthodes structurelles :

- Les méthodes structurelles internes : elles calculent la similarité entre deux concepts en exploitant les informations relatives à leur structure interne (restrictions et cardinalités sur les attributs, valeurs des instances, etc.) [Madhavan et al., 2001].

- Les méthodes structurelles externes ou conceptuelles : elles se servent de la structure hiérarchique de l'ontologie et se basent sur des techniques de comptage d'arcs pour déterminer la similarité sémantique entre deux entités [Wu & Palmer, 1994].

c. 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.

d. Mesures sémantiques

La plupart des mesures évaluent la relation entre les concepts en termes de similarité.

Selon [Resnik, 1999], la similarité sémantique est une évaluation de la relation sémantique entre deux concepts avec l'objectif d'obtenir une estimation du degré de proximité entre la signification de ces deux concepts. Un autre type de mesures qui a été proposée consiste à mesurer la « connexité sémantique », dont le but est d'évaluer la force de la relation sémantique entre deux concepts. Deux concepts peuvent ainsi être jugés fortement « connectés » sans être pour autant « similaire», comme par exemple les deux concepts de label « Table » et « Pot ».

7.3.2. 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].

8. Conclusion

Dans ce premier chapitre, nous avons donné une vision générale sur les ontologies par différentes définitions. Dans le chapitre suivant nous allons parler des méthodes de partitionnement d'ontologies qui ont été utilisées.

Chapitre II :

Partitionnement d'ontologies

1. Introduction

Les ontologies doivent leur succès actuel à leur aptitude à être partageables et réutilisables. Quoiqu'avec le partage, on se retrouve avec des ontologies de plus en plus larges.

Le chapitre présent est un état de l'art sur le partitionnement des ontologies, les concepts du partitionnement et les travaux que nous jugeons les plus importants réalisés dans ce domaine ainsi que les objectifs de partitionnement des ontologies. Nous partons d'une étude générale sur le partitionnement et ses objectifs. Ensuite, nous exposons les fameux travaux dans ce domaine et nous détaillons le domaine du partitionnement des ontologies à base de clustering dont notre travail fait partie.

2. Partitionnement d'ontologies

Le partitionnement est un processus au cours duquel une ontologie O est fractionnée en un ensemble de partitions P tel que l'union de l'interprétation de toutes les partitions ainsi crées soit équivalente à l'interprétation de l'ontologie initiale O.

3. Partitionnement à base de clustering

Dans cette section nous faisons tout d'abord un survol sur les concepts de base du clustering.

3.1. Datamining

En règle générale, le terme datamining désigne l'analyse de données depuis différentes perspectives et le fait de transformer ces données en informations utiles, en établissant des relations entre les données ou en repérant des patterns. Ces informations peuvent ensuite être utilisées par les entreprises pour augmenter un chiffre d'affaires ou pour réduire des coûts. Elles peuvent également servir à mieux comprendre une clientèle afin d'établir de meilleures stratégies marketing.

Les logiciels datamining font partie des outils analytiques utilisés pour l'analyse de données. Ils permettent aux utilisateurs d'analyser des données sous différents angles, de les catégoriser et de résumer les relations identifiées. Techniquement, le datamining est le procédé permettant de trouver des corrélations ou des patterns entre de nombreuses bases de données relationnelles. Le datamining repose sur des algorithmes complexes et sophistiqués permettant de segmenter les données et d'évaluer les probabilités futures.

3.2. Clustering

Le partitionnement de données (ou data clustering en anglais) est une des méthodes d'analyse des données. Elle vise à diviser un ensemble de données en différents « paquets » homogènes, en ce sens que les données de chaque sous-ensemble partagent des caractéristiques communes, qui correspondent le plus souvent à des critères de proximité (similarité informatique) que l'on définit en introduisant des mesures et classes de distance entre objets.

Donc, le clustering est un traitement sur un ensemble d'objets qui n'ont pas été étiquetés par un superviseur. Ce type de méthode vise à répondre au problème de diminution de la dimension de l'espace d'entrée, ou pour le groupement des objets en plusieurs catégories (clusters) non définies à l'avance.

Un « cluster » est donc une collection d'objets qui sont « similaires » entre eu et qui sont « dissemblables » par rapport aux objets appartenant à d'autre cluster.

Le clustering consiste à créer une partition ou une décomposition, tel que, l'objectif est de :

- minimiser l'inertie intra-classe pour obtenir des clusters les plus homogènes possibles ;
- maximiser l'inertie inter-classe afin d'obtenir des clusters bien différenciés.

3.3. Types de clustering

Il existe deux grands types de clustering [MacQueen, 1967] :

3.3.1. Clustering hiérarchique

Dans ce type de clustering, on décompose l'ensemble d'individus en une arborescence de clusters.

Ce type de clustering [Karypis et al., 1999] consiste à effectuer une suite de regroupement en clusters de moins en moins fines en agrégeant à chaque étape les objets (simple élément) ou les groupes d'objets (un cluster) les plus proche. Ce qui nous donne une arborescence de clusters. Cette approche utilise la mesure de similarité pour refléter l'homogénéité ou l'hétérogénéité des classes.

Parmi les algorithmes les plus connus de ce type :

a. Algorithme de classification ascendante hiérarchique (CHA) : où le mot ascendante est utilisé pour désigner qu'elle part d'une situation dont tous les individus représentent des clusters à part entière, puis on cherche les rassembler en classes de plus en plus grandes. Ainsi le qualificatif « hiérarchique » désigne le fait qu'elle produit une hiérarchie.

3.3.2. Clustering non-hiérarchique

Dans ce type de clustering, on décompose l'ensemble d'individus en K clusters.

Parmi les algorithmes de clustering non-hiérarchique, nous citons les plus connus :

a. K-means (Clustering exclusif)

L'algorithme de clustering K-Means [Hartigan & Wong, 1979] est une des techniques de clustering les plus fondamentales et les plus populaires. La technique de classification K-Means utilise le centroïde (la moyenne de tous les points dans le cluster) pour représenter le cluster. Il divise l'ensemble de données comprenant n éléments de données en k clusters de telle sorte que chacun des n éléments de données appartient à un cluster avec le centroïde le plus proche possible.

Algorithme K-means :

Entrée

Ensemble de N données, noté par x

Nombre de groupes souhaité, noté par k

Sortie

Une partition de K groupes $\{C1, C2,\ldots Ck\}$

Début

1- Initialisation aléatoire des centres C_k

Répéter

2- Affectation : générer une nouvelle partition en assignant chaque objet au groupe dont le centre est le plus proche :

$$x_i \in C_k \ si \ \forall j \ |x_i - \mu_k| = \min|x_i - \mu_j|$$

Avec μ_k le centre de la classe K ;

3- Représentation : Calculer les centres associés à la nouvelle partition ;

$$\mu_k = \frac{1}{N} \sum_{x_i \in C_k} x_i$$

Jusqu'à convergence de l'algorithme vers une partition stable.

Fin.

b. Overlapping clustering (Fuzzy clustering)

Fuzzy c-means (FCM) [Lance et al., 1967] est une méthode de clustering qui permet à un objet de données d'appartenir à deux ou plusieurs clusters. Cette méthode dérivée de l'algorithme c-means, identique à l'algorithme k-means décrit précédemment, elle a été développée par Dunn en 1973 et améliorée par Bezdek en 1981, est fréquemment utilisée dans la reconnaissance des formes. Il est basé sur la minimisation de la fonction objective suivante :

$$Jm = \sum_{i=1}^{N} \sum_{j=1}^{C} \cup_{ij}^{m} \left\| x_i - c_j \right\|^2 \qquad 1<=m<\alpha$$

Où **m** est un nombre réel (>1), $\mathbf{U_{ij}}$ est le degré d'appartenance de $\mathbf{x_i}$ dans le **j**ème cluster, $\mathbf{x_i}$ est le **i**ème élément des données mesurées, $\mathbf{c_j}$ est le centre d'un cluster et **k*k** est toute norme exprimant la similarité entre les données mesurées et le centre. Ce partitionnement logique floue (fuzzy) est réalisé grâce à une optimisation itérative de la fonction objective indiquée ci-dessus, avec la mise à jour de l'appartenance $\mathbf{u_{ij}}$ et les centres des clusters $\mathbf{c_j}$.

c. Algorithme d'Expectation Maximisation

En français « l'algorithme d'espérance-maximisation », souvent abrégé par EM, un des premiers articles sur EM a été écrit en 1988 [Gesu, 1988] mais la référence pratique qui a formalisé EM et a fourni une preuve de convergence est le document de Dempster, Laird et Rubin en 1977 [Dempster et al., 1977]. Son objectif est de trouver le maximum de vraisemblance des paramètres de modèle probabiliste. Il est utilisé dans plusieurs domaines, à titre d'exemple : dans la vision artificielle, le traitement d'image, plus spécifique en ce qui concerne la segmentation (image médicale, satellitaire…etc) ou le clustering pour regrouper les données homogènes dans un groupe.

Cet algorithme comprend deux étapes essentielles :

E-steps (E) : une étape d'évaluation de l'espérance, c'est dans cette étape qu'on calcule l'espérance de la vraisemblance en tenant compte des dernières variables observées.

M-steps (M) : une étape de maximisation de vraisemblance qu'on a trouvé à l'étape (E), en tenant compte d'estimer le maximum de vraisemblance des paramètres.

Et c'est ainsi qu'on itère l'algorithme en utilisant les paramètres trouvés à l'étape (M) pour évaluer à nouveau l'espérance.

4. Méthodes de partitionnement d'ontologies

Les méthodes existantes pour le partitionnement d'ontologies qui ont été proposées sont les suivantes :

- Méthode Garcia et al [Garcia et al., 2012]

Les auteurs ont proposé une méthode de partitionnement, dont l'objectif est d'étudier les techniques de partitionnement des graphes et les appliquer sur le partitionnement de grandes ontologies. Ce travail est réalisé en deux étapes :

Etape 1 : Comment convertir une ontologie donnée représentée en OWL ou RDF en graphe.

Etape 2 : Quel algorithme de partitionnement conviendrait.

Tout en concentrant sur la manière de préserver certaines propriétés/relations d'ontologies dans les modules générés.

Cinq algorithmes de partitionnement de graphes ont été utilisés (Spin Glass, FastGreedy, Walktrap, Leading Eigenvector et Edge Betweenness), mais seulement trois d'entre eux (Spin Glass, Walktrap et FastGreedy) ont été utilisés pour la vérification des variations des poids.

Une étude de cas a été menée à l'aide d'une ontologie de jeux dans le domaine de la pizza. Elle a montré des résultats intéressants, mais l'algorithme FastGreedy a obtenu les meilleurs résultats préliminaires, montrant une "sensibilité" par rapport au classement des propriétés du domaine.

- **Méthode Grau et al [Grau et al., 2007]**

Les auteurs ont proposé en 2007, une définition d'un module qui capture la signification complète d'un ensemble donné de termes, c-à-d d'inclure tous les axiomes pertinents à la signification de ces termes, et étudie le problème de l'extraction de modules en utilisant des approches logiques.

Cette méthode propose deux "approximations", mais qui sont trop strictes et peut conduire à des modules plus volumineux :

1- Approximation sémantique : peut être calculée en utilisant des DL reasoners existants;
2- Approximation syntaxique : peut être calculée en temps polynomial.

Les résultats de tests ont démontré que les modules générés par cette approche sont strictes et raisonnablement petits pour de nombreuses ontologies du monde réel, par rapport deux autres méthodes d'extraction de modules.

- **Méthode de Thi et al [Thi et al., 2008]**

En 2008, deux algorithmes ont été proposés par Thi Le Pham & al pour raisonner dans une ontologie décomposée basée sur des approches parallèles et distribuées. Ils ont utilisé la technique 'Ontologies Overlapping Decomposition' afin de diviser une ontologie en plusieurs sous-ontologies, en appliquant la logique de description pour la décomposition. C'est une nouvelle technique pour optimiser le raisonnement DL afin de minimiser autant que possible le temps d'exécution et l'espace de stockage nécessaires. Cette technique a pour but d'accélérer TBox et ABox, en particulier pour les grandes TBox. Elle est applicable à la plupart des ontologies de la vie réelle, en particulier pour les ontologies ayant des structures complexes avec un grand nombre d'axiomes GCI.

L'incorporation de cette méthode aux techniques d'optimisation précédentes dans les systèmes DL actuels peut résoudre efficacement les inférences intraitables. La décomposition d'une ontologie donnée en plusieurs sous-ontologies est implémentée de telle sorte que les services sémantiques et d'inférence de l'ontologie originale sont préservés.

Le principe est le suivant : Le raisonnement au sein d'une ontologie volumineuse (globale) peut être réduit à certaines procédures de raisonnement dans ses sous-ontologies (locales) basées sur des approches parallèles ou distribuées. Par conséquent, une requête

peut également être décomposée en sous-requêtes qui sont ensuite traitées séparément sur les ontologies locales, puis les résultats sont combinés.

- Méthode de Setti et al [Setti et al., 2015]

Dans le but de partitionner une ontologie volumineuse donnée en modules de haute qualité, Setti & al, ont proposé une approche de partitionnement qui introduit une amélioration de l'algorithme de classification k-means, en se basant sur une nouvelle mesure de similarité sémantique « Dennai ». C'est une amélioration de la mesure de similarité structurelle : Wu et Palmer.

L'approche consiste à extraire d'abord tous les constructeurs de l'ontologie (classes, axiomes, relations et propriétés), puis à convertir l'ontologie représentée en OWL sous forme d'un graphe. Après, différentes mesures de similarité sont calculés d'un nœud donné vers tous les nœuds du graphe. Il s'agit de déterminer le chemin le plus court entre un nœud donné et un autre nœud du graph. Cette étape génère différentes matrices de mesures de similarité. Ensuite, deux algorithmes de clustering sont utilisés. Le premier est introduit pour adapter l'algorithme traditionnel au clustering d'ontologies. Et il est modifié après pour intégrer les différentes mesures de similarité. Le deuxième algorithme a pour but d'éliminer les clusters inutiles générés auparavant. Enfin, un processus de validation du partitionnement est programmé comme une perspective.

Deux séries d'expérimentations ont montré que cette approche réduit le temps et l'espace requis pour traiter une ontologie, et produit des partitions de haute qualité. L'amélioration de l'algorithme k-means proposée et l'utilisation de la nouvelle mesure de similarité Dennai donnent de meilleures partitions et des clusters significatifs que l'algorithme k-means traditionnel.

- **Méthode de Grau et al [Grau et al., 2005]**

Grau, B.C & al, ont donné une définition formelle du module de sorte que chaque entité dans l'ontologie est sémantiquement encapsulée, afin de détecter les axiomes associés à chaque entité dans l'ontologie. Cette technique aborde le problème de la détermination du fragment d'une ontologie qui capture la signification essentielle d'une entité donnée dans l'ontologie. Ce travail présente également un algorithme d'extraction de fragments qui préserve un ensemble d'implications associé à cette entité. L'algorithme présenté génère à partir de l'ontologie d'entrée O un graphe orienté G(O), appelé *graphe*

de partitionnement, puis utilise le graphe pour trouver le module pour chaque entité dans O.

Les résultats des expérimentations avec des ontologies du monde réel montrent que pour un nombre significatif d'entités, les modules obtenus peuvent être notablement plus petits que l'ontologie originale, ce qui facilite la réutilisation, la capacité de traitement, la compréhensibilité et la maintenance.

- **Méthode de Ding et al [Ding et al., 2013]**

Dans ce travail, les auteurs utilisent l'algorithme K-means pour réaliser l'alignement entre plusieurs attributs.

En premier lieu, les auteurs convertirent les attributs en points, ensuite exécutent l'algorithme K-means pour partitionner les attributs à plusieurs clusters.

Les attributs dans le même cluster ont la même sémantique.

Dans cet algorithme, le nombre de K objets est choisi aléatoirement comme des centres de clusters ensuite la méthode TF/IDF est utilisée pour calculer le poids. Aussi le modèle de l'espace vectoriel est appliqué comme métrique de calcul de distance entre les points d'attributs.

5. Tableau de comparaison des différentes méthodes de partitionnement

Les méthodes de partitionnement décrites précédemment ont pour objectif l'amélioration des résultats de partitionnement en choisissant les meilleurs critères et spécificités des ontologies, par exemple, les mesures de similarité et les techniques de partitionnement utilisées, le choix de l'ontologie initiale utilisée (c-à-d est ce que c'est une ontologie OWL ou RDF), aussi le choix des ontologies de test par rapport à leur taille si elles sont petites, moyennes ou volumineuses et les mesures de test par les quelles les travaux ont été testés et validés.

Dans le tableau suivant, nous allons faire une comparaison entre toutes les méthodes de partitionnement étudiées par rapport aux critères cités ci-dessus.

Auteurs	Approche	Ontologie initiale	Mesure de similarité utilisée	Nombres de blocs	Technique de partitionnement	Ontologies de test	Mesures de test	Outils utilisés
Garcia et al., 2012	**Applying Graph Partitioning Techniques to Modularize Large Ontologies**	Ontologie volumineuse OWL/RDF Biomédicale	-	16	Construction de graphe à partir de l'ontologie initiale	Ontology de jeux	User preference	**PATO** pour la conversion de l'ontologie en graphe **iGraph Partitionnement du graphe**
Grau et al., 2007	**Justthe Right Amount: Extracting Modules from Ontologies.**	Ontologie OWL DL	axiomes syntaxiquement + sémantiquement locaux, DL reasoners	-	Modularisation en utilisant la logique de description pour extraire les modules de l'ontologie	***Ontologies simples :*** NCI ontology, SUMO UpperOntology, Gene Ontology, SNOMED Ontology. ***Ontologies complexes :*** GALEN Ontology, DOLCE UpperOntology, NASA's Semantic Web Ontology.	-	-
Thi et al., 2008	**Decomposition-based Reasoning for Large Knowledge Bases in Description Logics.**	Bases de connaissances volumineuses : Ontologies DL	Axiomes terminologiques, DL reasoners, Procédure DPL, heuristiques MOMS	-	Décomposition De l'ontologie en plusieurs sous-ontologies en introduisant une nouvelle techique appelée '***Overlap Ontology Decomposition***'	-	-	-

Setti et al., 2015	**Ontology Partitioning : Clustering Based Approach**	Ontologie volumineuse OWL	- Tbk, - Jaccard -Cosine - Dice - Euclidienne - Wu & Palmer - Mesure d'édition - Dennai	(5-40) (7-50) (10-30)	Clustering : k-means amélioré	***TAMBIS ontology*** (393 classes+relations+597 axiomes) ***Photographyontology*** (185 classes+268 relations + différent types d'objets + propriétés)	- Cohesion - density	-
Grau et al., 2005	**Modularizing OWL Ontologies.**	Ontologie OWL-DL	Dépendances sémantiques et graphiques	-	Utilise E-connections, pour construire des modules -Accorder pour chaque entité dans l'ontologie une encapsulation sémantique pour détecter ses axiomes associés	OWL-S Ontologies, NCI Thesaurus, GALEN Ontologies, NASA's SWEET-JPL Ontologies	-	-
Ding et al., 2013	**Multi-SchemaMatchingBased On ClusteringTechniques.**	Schémas synthétiques	Cosine		Appariement Clustering : k-means	05 librairies en ligne	Précision et rappel	**TF/IDF :** Facteur de pondération

- Le critère n'est pas cité dans l'article de base

Tableau 1 : Tableau de comparaison des différentes Méthodes de partitionnement.

6. Discussion

Après l'étude des différentes méthodes existantes pour le partitionnement d'ontologies, nous constatons les remarques suivantes :

1- Dans la méthode de [Garcia et al., 2012], cinq algorithmes de partitionnement de graphe ont été exécutés pour la représentation graphique d'une ontologie sur le domaine Pizza, seulement trois d'entre eux ont permis la génération de graphes pondérés. Parmi ces trois algorithmes, FastGreedy a obtenu les meilleurs résultats préliminaires, montrant une "sensibilité" par rapport au classement des propriétés du domaine. Mais, d'autres tests devraient être exécutés avec des ontologies plus grandes et différentes.

2- Dans la méthode de [Grau et al., 2007], nous remarquons un problème d'évaluation, tel que les auteurs n'ont pas pris en considération la mesure de test, et n'ont pas cité le nombre de blocs par rapport à l'ontologie testée.

3- D'après [Thi et al., 2008], les techniques de décomposition concrètes n'ont pas été présentées dans l'article.

4- Dans la méthode de [Setti et al., 2015], les auteurs ont basé sur la mesure structurelle wu & palmer améliorée, ils ont négligé l'utilisation des mesures syntaxiques et sémantiques qui permettent une forte similarité entre les concepts.

Les résultats expérimentaux sur des ontologies simples ont montré que cette approche réduit le temps et l'espace requis pour traiter une ontologie et produit des partitions de haute qualité, mais les tests ne sont pas effectués sur des ontologies volumineuses.

5- Dans les travaux de [Grau et al., 2005], les mesures de test sur les ontologies ne sont pas mentionées.

6- Dans la méthode [Ding et al., 2013], les auteurs ont utilisé l'algorithme de clustering k-means traditionnel, dont l'initialisation des centres est aléatoire, donc les résultats obtenus (les partitions) dépendent entièrement de cette initialisation qui n'est pas toujours optimale.

7. Conclusion

Dans ce chapitre, nous avons parlé des différentes techniques de partitionnement existantes avec une étude comparative, en terminant par une discussion. Nous avons défini aussi le clustering et les différents types de clustering.

Dans le chapitre suivant nous présenterons la conception et l'architecture globale de notre système ainsi que les différents algorithmes utilisés.

Chapitre III :

Approche proposée

1. Introduction

Après avoir donné un aperçu général sur le domaine de partitionnement d'ontologies, nous avons choisis d'utiliser l'algorithme de clustering k-means pour réaliser notre travail. Ce dernier est applicable à tout type de données (mêmes textuelles), en choisissant une bonne notion de distance, il est connu aussi par sa simplicité conceptuelle et sa rapidité pour la convergence.

Pour adapter le k-means pour le partitionnement d'ontologie, nous allons combiner ce dernier avec l'algorithme le mal classé, et l'algorithme k-medoïde pour améliorer la qualité des résultats du partitionnement, et ce, en choisissant la distance qui s'adapte avec notre type de données.

2. Comparaison entre les différentes méthodes et notre travail

D'après l'étude comparative réalisée dans le chapitre précédent, nous avons constaté que parmi les méthodes de partitionnement existantes, il y a des méthodes qui ne s'intéressent pas vraiment aux mesures de test de validation et ne précisent pas le nombre de blocs par rapport à la taille de l'ontologie partitionnée. Il y en a d'autres qui n'utilisent pas les mesures de similarité pour calculer la table des correspondances, et dans le cas contraire, elles se basent sur la similarité entre les concepts en ignorant le calcul de cette dernière sur les instances.

Pour pallier à ces contraintes non négligeables qui influent d'une manière directe au résultat du partitionnement de notre ontologie, nous avons essayé de mettre au point un système de partitionnement qui se concentre sur :

- Le calcul d'une similarité sémantique entre les concepts et les instances de l'ontologie, il s'agit de combiner plusieurs mesures de calcul de similarité existante (terminologique, structurelle et intentionnelle).
- La prise en considération des paramètres d'entrée, où le système permet aux utilisateurs de déterminer le nombre de partitions souhaité.
- L'adaptation de l'algorithme k-means pour le partitionnement d'ontologies.

3. Partitionnement d'ontologies à base de clustering

Nous utilisons dans notre système des ontologies écrites dans le format OWL, qui est un langage ontologique, capable de décrire formellement la signification de la

terminologie employée dans les documents Web, constitue le premier niveau nécessaire du Web sémantique après RDF.

Nous présentons le schéma global de notre approche, ensuite nous détaillons les différentes étapes et les différents algorithmes utilisés.

Le schéma global de l'approche proposée est illustré dans la Figure 7 suivante :

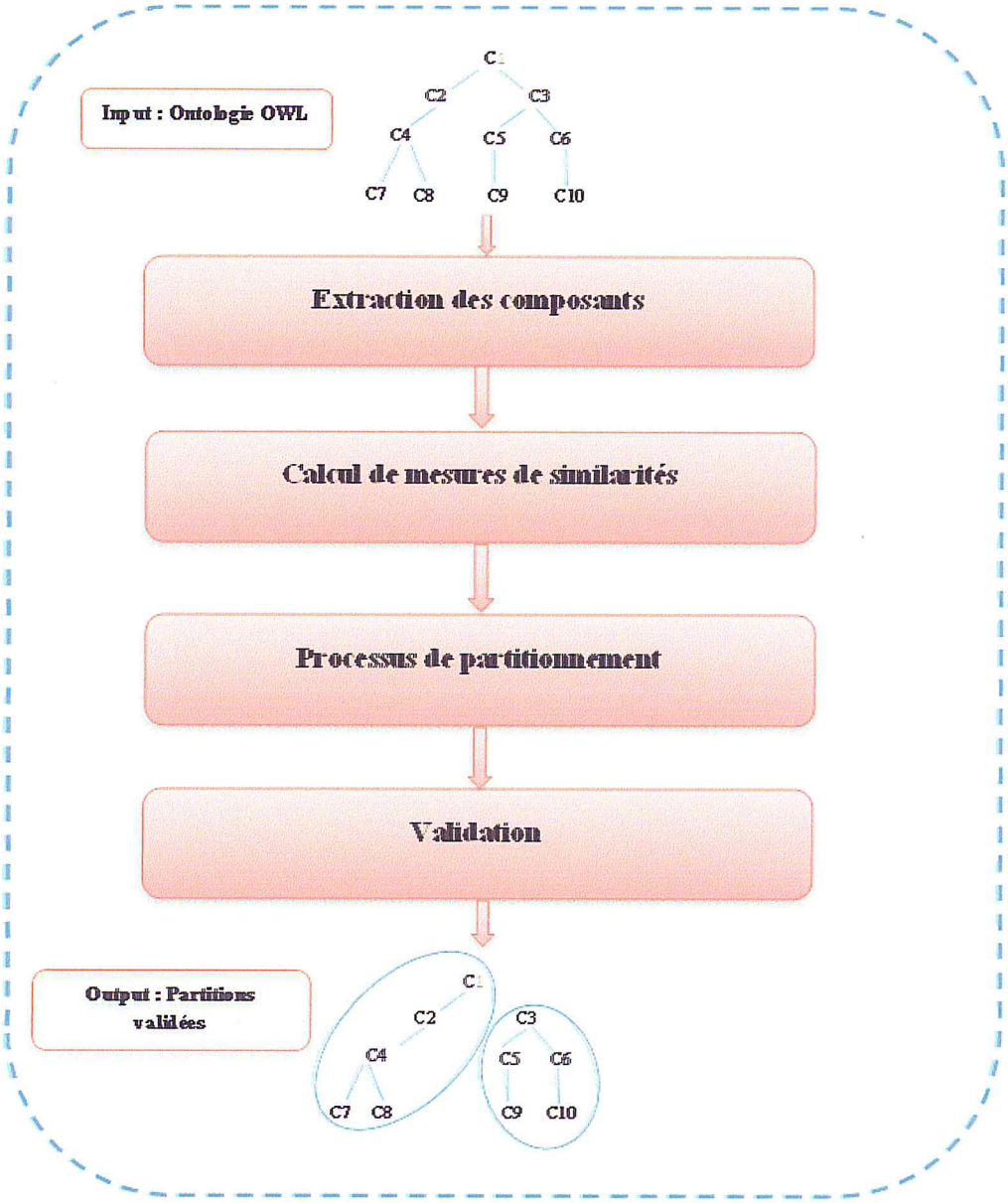


Figure 7 : Schéma global du Processus de partitionnement d'ontologie

4. Description générale de notre approche

Notre approche est composée des phases suivantes :

1) Phase d'extraction des composants de l'ontologie OWL,

2) Phase de calcul de mesures de similarités,

3) Processus de partitionnement,

4) Et enfin la phase de validation.

Phase 1 : Extraction des composants de l'ontologie OWL

Dans la première phase nous avons en entrée une ontologie OWL, ensuite on doit parcourir toute la hiérarchie de l'ontologie, pour extraire tous les composants qui se trouvent dans cette dernière (concepts, instances, attributs et propriétés).

Phase 2 : Calcul des mesures de similarité

En second lieu, vient le calcul de similarité qui s'effectue entre les concepts et les instances de notre ontologie afin de mesurer leur degré de correspondance et de l'utiliser dans le processus de partitionnement.

Phase 3 : Processus de partitionnement

Après le calcul de toutes les mesures de similarités, nous procédons au processus de partitionnement, il s'agit d'implémenter un algorithme de clustering amélioré, qui se compose de trois algorithmes combiné: k-means, k-medoide et le mal classé.

A la fin, nous obtiendrons un ensemble de partitions.

Phase 4 : Validation

Dans cette phase nous procédons à la validation de l'ensemble des partitions générées lors de l'application du processus de partitionnement.

5. Description détaillée de l'approche de partitionnement

Dans cette section, nous allons expliquer en détail le processus de partitionnement.

5.1. Extraction des composants de l'ontologie

Cette partie consiste à extraire tous les composants de notre ontologie OWL.

Où le système permet de parcourir toute l'ontologie et d'extraire ses composants (classe racine, classe, instances).

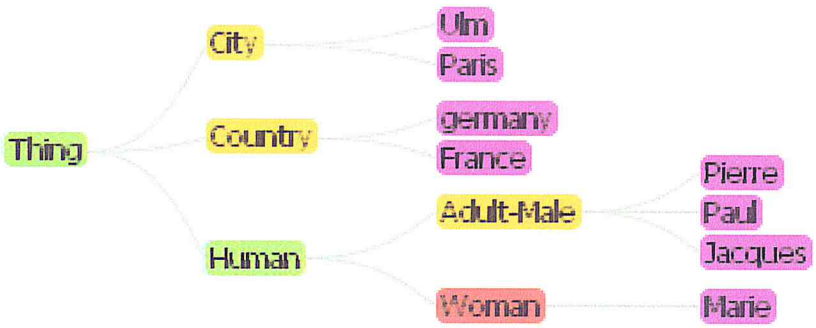


Figure 8: Exemple de l'ontologie Family

Voici un exemple des résultats de l'extraction des composants de cette ontologie :

Classe racine	Classes	Instances
Thing	Country	- Germany - France
	Human	-
	Adult-Male	- Pierre - Paul - Jacques
	Woman	- Marie
	City	- Ulm - Paris

Tableau 2 : Exemple des composants de l'ontologie Family

5.2. Mesures de similarité utilisées dans l'algorithme de partitionnement

Lors de cette étape, nous allons calculer les différentes mesures de similarité qui vont nous aider à faire le partitionnement de notre ontologie.

Pour ce faire, nous avons choisis une approche combinée de calcul des mesures de similarité. Cette approche se compose de deux types de combinaison :

- **Combinaison séquentielle :** elle est représentée dans notre cas par l'exécution des mesures lexicale et syntaxique avant d'exécuter la mesure terminologique, et d'exécuter les trois mesures terminologique, structurelle et extensionnelle avant d'exécuter la mesure sémantique.

- **Combinaison parallèle :** dans ce type, l'exécution parallèle s'effectue entre les trois mesures : structurelle, terminologique et extensionnelle avant d'exécuter la mesure sémantique.

Le schéma suivant résume la combinaison des mesures de similarité :

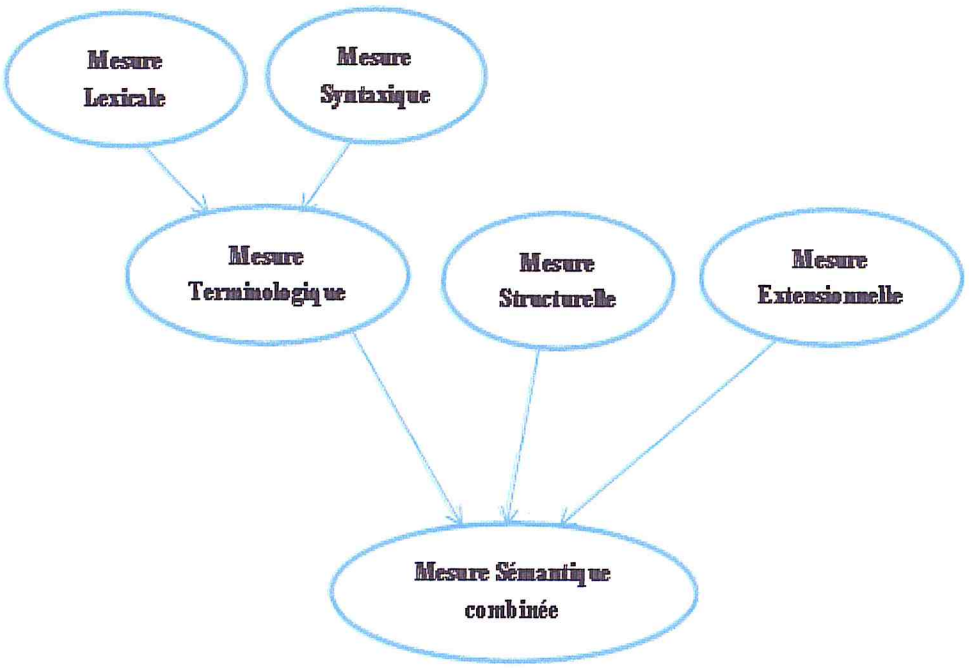


Figure 9 : Schéma de combinaison des mesures de similarité.

Nous détaillons maintenant les différentes mesures de similarité utilisées :

5.2.1. Mesure de similarité terminologique

Afin de mesurer la similarité terminologique entre les concepts, nous allons utiliser les deux mesures suivantes :

a. Mesure syntaxique

Dans notre système nous allons utiliser la distance de JARO, cette dernière mesure la similarité entre deux chaînes de caractères. Plus la distance de Jaro entre deux chaînes est élevée, plus elles sont similaires.

Cette mesure est particulièrement adaptée au traitement de chaînes courtes comme des noms ou des mots de passe. Le résultat est normalisé de façon à avoir une mesure entre 0 et 1, le zéro représente l'absence de similarité.

La distance de Jaro entre les deux chaînes $s1$ et $s2$ est définie par :

$$\textbf{Simsyn} = \frac{1}{3}\left(\frac{m}{|s1|} + \frac{m}{|s2|} + \frac{m-t}{m}\right)$$

Où :

- $\boldsymbol{m}$ est le nombre de caractères correspondants.

- $\boldsymbol{t}$ est le nombre de transpositions.

Deux caractères identiques de s1 et de s2 sont considérés comme correspondants si leur éloignement (i.e. la différence entre leurs positions dans leurs chaînes respectives) ne dépasse pas :

$$\left(\frac{\max(|s1|, |s2|)}{2}\right) - 1$$

Le nombre de transpositions est obtenu en comparant le $i^{ème}$ caractère correspondant de s1 avec le $i^{ème}$ caractère correspondant de s2. Le nombre de fois où ces caractères sont différents, divisé par deux, donne le nombre de transpositions.

$DS_{Jaro} : S\,xS \in [0, 1]$ telle que :

Exemple de la mesure Jaro :

$$\text{Simsyn}(\text{Marie}, \text{Maires}) = \frac{1}{3}\left(\frac{5}{|5|} + \frac{5}{|6|} + \frac{5-2}{5}\right) = \boxed{0.81}$$

b. Mesure lexicale

Les méthodes basées sur un langage se fondent sur des techniques de traitement du langage naturel afin de trouver des associations entre les entités ou les classes. Ces méthodes exigent l'utilisation de ressources externes. Plusieurs types de ressources peuvent être employés, notre choix s'est porté sur WordNet.

WordNet est une ressource lexicale de langue anglaise, disponible sur internet, qui regroupe des termes (noms, verbes, adjectifs et adverbes) en ensembles de synonymes appelés *synsets*. Un synset regroupe tous les termes dénotant un concept donné.

WordNet contient des liens entre les synsets qui représentent plusieurs relations : is-a (est-un(e)), part-of (fait-partie-de), synonymie, antonymie, hyponymie, homonymie etc.

Pour le calcul de la similarité linguistique, la fonction Syn(c) calcule l'ensemble des Synsets de WordNet du concept c retenus après l'enrichissement de l'ontologie; soit **S = Syn (c1) ∩ Syn (c2)** l'ensemble des sens communs entre c1 et c2 à comparer, la cardinalité de S est :

$$\lambda(S) = |Syn(c1) \cap Syn(c2)| ;$$

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

$$Simlex\ (c\ 1,\ c\ 2\ ) = \frac{\lambda\ (S)}{min(Syn(\ c\ 1\ ),Syn(\ c\ 2\ ))}$$

Nous avons ensuite, combiné ces deux mesures, le résultat est la formule suivante :

$$Sim_{term}(c1,c2) = \frac{Simlex(c1,c2) + Simsyn(c1,c2)}{2}$$

5.2.2. Mesure de similarité Structurelle

Cette dernière sera calculée par la mesure de Wu et Palmer :

$$Sim_{wp}(c1,\ c2) = \frac{2 * \text{Profondeur (C)}}{\text{Profondeur (C1)} + \text{Profondeur (C2)}}$$

Exemple

Soit l'ontologie de la Figure 10, on dénote par C1, C2 et C3 les concepts «Person», « PostDoc» et «AdministrativeStaff». En appliquant la mesure de Wu et Palmer, la valeur de similarité est calculée comme suit :

Simwp(C1, C2) = 2*1/(1+4) = 0.4

Simwp(C2, C3) = 2*2/(4+3) = 4/7 = 0.57

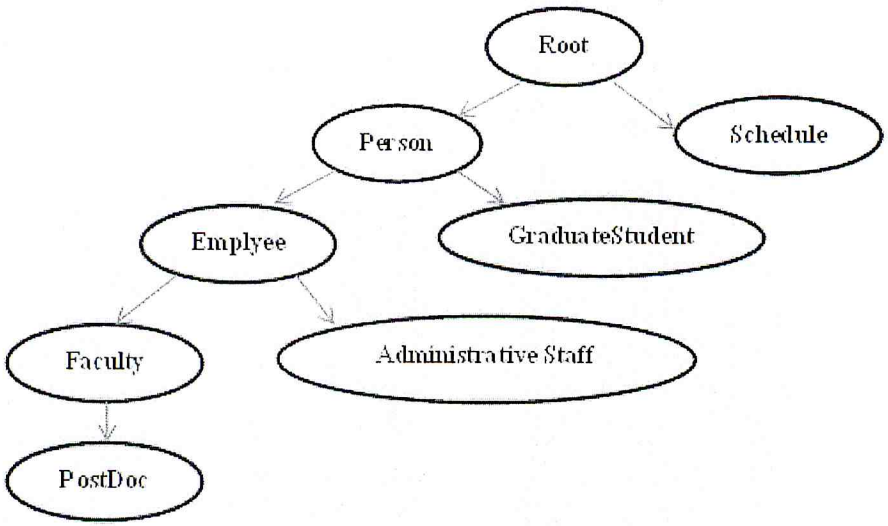


Figure 10 : Extrait de l'ontologie UnivBench.

5.2.3. Mesure de similarité extensionnelle

Pour calculer la mesure de similarité extensionnelle, nous allons utiliser la mesure terminologique syntaxique de Jaro, que nous avons appliqué précédemment sur les concepts, adaptée sur les termes des instances.

Soit C1, C2 deux concepts.

Pour calculer la mesure de similarité entre les instances appartenant aux C1 et C2, on suit les étapes suivantes :

Etape 1 :

- Définir un seuil **S1**
- Calculer la mesure de similarité entre toutes les instances appartenant à C1 et C2.
- **Si** la similarité $\mathbf{Sim_{Jaro}}$ entre Instances Inst1 et Inst2 (appartenant à C1, C2 successivement) est supérieure ou égale au seuil (Sim_{Jaro} >= S1)

  Alors Inst1 et Inst2 sont similaires, Stocker $\mathbf{Sim_{Jaro}}$

  Sinon Sim =0

Etape 2 :

- Définir un autre seuil **S2**, tel que **S2** est le nombre d'instances similaires.
- Calculer la formule suivante : $\mathbf{F} = \frac{\sum instances\ similaires\ (C1,C2)}{\sum nombre\ d'instances(C1,C2)}$
- **Si F >= S2**

  Alors $\mathbf{SimIns} = \frac{\sum \mathrm{SimJaro}(C1,C2)}{\sum nombre\ d'instances(C1,C2)}$ où $\mathbf{Sim_{Jaro}}$**>= S1**

  C.à.d (***SimIns*** = La Moyenne des mesures qui dépassent S1)

  Sinon ***SimIns*** **= 0**

Calcul de similarité sémantique (globale)

La similarité sémantique sera calculée par la combinaison hybride des similarités terminologique, structurelle et extensionnelle, selon la formule suivante :

$$SimSem\ (c1,\ c2) = \frac{SimTer\ (c1,c2) + SimStr(c1,c2) + SimIns(c1,c2)}{3}$$

A l'issue de ces étapes on obtiendra une matrice des mesures de similarité « matrice de correspondance » qui va nous aider à réaliser le partitionnement.

5.3. Processus de partitionnement

Il est à noter que les phases précédentes sont une préparation à la phase présente dans le sens où la matrice de similarité générée est utilisée dans les algorithmes que nous avons développés pour partitionner nos ontologies OWL.

Pour ce faire, nous présentons les algorithmes que nous allons utiliser pour réaliser le partitionnement d'ontologies :

5.3.1. Algorithme k-means

Le principe de l'approche se base sur l'algorithme k-means, ou algorithme de clustering par centre mobile, il permet d'effectuer un clustering d'un ensemble de données en K clusters. Chaque cluster est décrit par son centre.

L'algorithme est décrit comme suit :

1- Un ensemble de K centres est choisi aléatoirement dans l'ensemble des données,
2- Les K clusters sont formés en regroupant dans chaque centre l'ensemble des données plus proches du centre courant que de tout autre centre.
3- Le centre de chaque cluster est calculé et devient le nouveau centre.
4- L'algorithme boucle alors sur l'étape précédente : Les données sont réaffectées en fonction de ces nouveaux centres et la condition d'arrêt est que les centres deviennent immobiles.

Cet algorithme est l'un des plus simples algorithmes de classification automatique des données, dont l'idée principale est de choisir aléatoirement un ensemble de centres fixé à priori et de chercher itérativement la partition optimale.

Chaque individu est affecté au centre le plus proche, après l'affectation de toutes les données la moyenne de chaque groupe est calculée, elle constitue les nouveaux représentants des groupes, lorsqu'ils ont abouti à un état stationnaire (aucune donnée ne change de groupe) l'algorithme est arrêté.

Nous avons choisis cet algorithme, car il est parmi les algorithmes de classification, il doit sa popularité à sa simplicité, sa capacité de traiter de larges ensembles de données et sa convergence rapide, il est appliqué pour tous les types de données (dans notre cas ce sont les concepts) en choisissant une bonne notion de distance.

5.3.2. Limite de l'algorithme k-means

1ère limite

La principale limite de cet algorithme est la dépendance des résultats des valeurs de départ (centres initiaux). A chaque initialisation correspond une solution différente (optimum local) qui peut dans certain cas être très loin de la solution optimale (optimum global). Une solution naïve à ce problème consiste à lancer l'algorithme plusieurs fois avec

différentes initialisations et retenir le meilleur regroupement trouvé, donc, l'usage de cette solution reste limité.

Pour cette raison, nous avons trouvé une solution au problème d'initialisation du k-means qui est l'initialisation par le mal classé.

5.3.2.1. Algorithme le mal classé

Cet algorithme est appliqué lors de l'initialisation de l'algorithme k-means. L'objectif de cette initialisation est d'atteindre une bonne solution dont le nombre d'itérations sera réduit. La stratégie de cette dernière se base sur l'individu le plus mal classé.

Le k-means est initialisé aléatoirement par un certain nombre de centres initiaux, nous proposons de commencer l'initialisation du k-means avec deux centres, tel que ces deux derniers assurent la séparabilité des données au cours de classification, il est évident de choisir les deux individus les plus éloignés [Guellil & Zaoui, 2009].

Ce principe est illustré par l'algorithme suivant :

Algorithme : **Initialisation par le mal classé**

Début

1- Création d'une matrice de distance
2- Choisir les deux éléments les plus éloignés (ils représentent les deux premiers centres)

Tant Que le nombre de classe souhaité n'est pas atteint

Faire

3- Affecter les individus aux noyaux disponibles
4- Sélectionner un élément mal classé (celui qui possède la plus petite distance de son centre le plus proche)
5- Ajouter cet individu à l'ensemble des noyaux
6- Augmenter le nombre des noyaux

Fin Tant Que

Fin

La figure suivante montre le choix des deux éléments les plus éloignés :

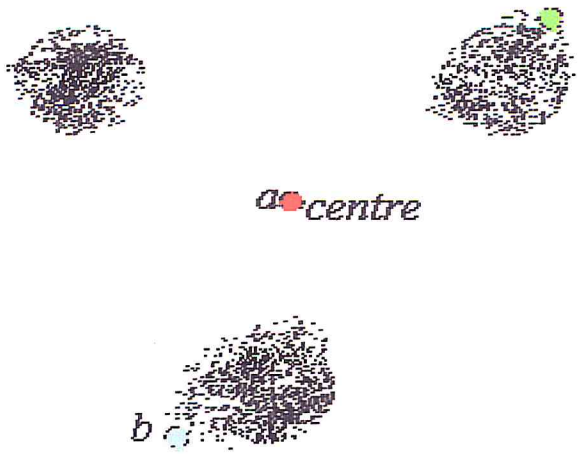


Figure 11 : (a) Le centre des données en rouge, (b) le bleu et le vert représentent les deux objets les plus éloigné.

2ème limite

Notre approche est une adaptation de l'algorithme K-means dans le domaine des ontologies où les données sont les concepts et les instances de l'ontologie. Cependant, le problème de calcul du centre d'un cluster se pose, car dans les méthodes usuelles de K-means, le centre d'un cluster se calcule à partir des cordonnées des données. Or la notion de cordonnées n'existe pas dans la structure d'une ontologie. C'est pourquoi nous avons proposé l'utilisation de l'algorithme k-méloïde pour le calcul de nouveaux centres des clusters.

5.3.2.2. Algorithme k-medoïde

L'algorithme K-Medoïde [Kaufman & Rousseeuw, 1987] est l'un des algorithmes de clustering. Il utilise des médoïdes pour représenter les groupes. Le médoïde est une statistique qui représente ce membre de données d'un ensemble de données dont la dissimilarité moyenne par rapport à tous les autres membres de l'ensemble est minimale. Le medoïde représente l'élément de données le plus central de l'ensemble de données.

Le fonctionnement de l'algorithme de clustering K-Medoids est similaire à celui de l'algorithme K-Means. Il commence également par la sélection aléatoire de k éléments de données en tant que médoïdes initiaux pour représenter les k clusters. Tous les autres éléments restants sont inclus dans les clusters dont le médoïde est plus proche.

Par la suite, un nouveau médoïde est déterminé, qui peut mieux représenter le groupe. Tous les éléments restants sont encore attribués aux clusters ayant le médoïde le plus proche. A chaque itération, les médoïdes modifient leur position. La méthode

minimise la somme des différences entre chaque élément du cluster et son medoïde correspondant. Ce cycle est répété jusqu'à ce qu'aucun médoïde ne change son emplacement. Cela marque la fin du processus et nous avons les clusters finaux résultants avec leurs medoïdes définis. K groupes sont formés qui sont centrés autour des médoïdes et tous les individus sont placés dans le cluster approprié sur la base du médoïde le plus proche.

Algorithme : k-medoïdes

Input

K : nombre de cluster; **D** : ensemble de données contenant **N** objets

Output : Un ensemble de k clusters qui minimise la somme des dissimilarité de tous les objets avec leurs médoïdes les plus proches, en utilisant la formule suivante :

$$E = \sum_{i=1}^{k} \sum_{X \in C_i} d\,(X, M_i)^2$$

Tel que :

K : nombre de clusters

X : l'ensemble de données.

M_i : le medoïde du cluster C_i

Les étapes :

1: Choisissez arbitrairement k éléments de données comme médoïdes initiaux.

2: Attribuez chaque élément de données restant à un cluster dont le médoïde est le plus proche.

3. Sélectionnez aléatoirement un élément de données non médoïde et calculez le coût total du remplacement de l'ancien médoïde par l'élément non médoïde actuellement sélectionné.

4. Si le coût total de l'échange est inférieur à zéro, effectuez l'opération du remplacement pour générer le nouvel ensemble de k medoids.

5. Répétez les étapes 2, 3 et 4 jusqu'à ce que les médoïdes stabilisent leurs emplacements.

Fin

6. Structure de l'algorithme global K-Means-Combiné

Entrée

Ensemble de N données (concepts et instances de notre ontologie OWL);

Nombre de clusters (partitions) souhaité, noté par K ;

Nombre d'itération, noté par nb_iter_max = 100;

Début

1- Calculer toutes les mesures de similarité entre les concepts de notre ontologie pour établir la matrice de similarité.

Nos calculs des mesures de similarité seront effectués selon les formules suivantes :

- **Mesure terminologique** : qui est composée des deux mesures :
- **Mesure syntaxique « Jaro » :** $\mathbf{Simsyn} = \frac{1}{3}\left(\frac{m}{|s1|} + \frac{m}{|s2|} + \frac{m-t}{m}\right)$
- **Mesurelexicale « WordNet » :** $Simlex(c\ 1,\ c\ 2) = \frac{\lambda(S)}{min(Syn(\ c1\ ),Syn(\ c2\ ))}$

En combinant les deux mesures, nous obtenons la mesure terminologique suivante :

$$Sim_{term}(c1,c2) = \frac{Simlex(c1,c2) + Simsyn(c1,c2)}{2}$$

a. **Mesure structurelle :** elle est représentée par la mesure de Wu &Palmer, comme suit :

$$\mathbf{Sim_{wp}(c1,\ c2)} = \frac{2 * \text{Profondeur (C)}}{\text{Profondeur(C1)} + \text{Profondeur (C2)}}$$

b. **Mesure extensionnelle :**

Pour calculer la mesure de similarité extensionnelle, nous allons utiliser la mesure terminologique syntaxique de Jaro, que nous avons appliqué précédemment sur les concepts (page 50)

c. **Mesure sémantique combinée :** à la fin, nous combinons toutes ces mesures pour obtenir une mesure de similarité sémantique pertinente et globale, représentée par la formule suivante :

$$SimSem\ (c1,\ c2) = \frac{SimTer(c1,c2) + SimStr(c1,c2) + SimExt(c1,c2)}{3}$$

2- Commencer l'initialisation de l'algorithme K-Means-Combiné par l'algorithme le mal classé, en choisissant les deux premiers centres initiaux C1 et C2 les plus éloignés tel que :

$$SimSem(c1,\ c2) = min_{\substack{i,j \in [1..N] \\ i \neq j}} \left(SimSem(ci, cj)\right)$$

Tant Que Le nombre de clusters souhaité n'est pas atteint

Faire

3. Affectation : affecter chaque individu à son centre initial le plus proche, et ce, pour générer une nouvelle partition, tel que :

$$c_i \in C_k \ si \ \forall\ Simsem(c_i, \mu_k) = \mathbf{max}\ \mathbf{SimSem}\ (c_i, \mu_j)$$

Avec μ_k représente les centres initiaux C1 et C2 qui ont été sélectionnés lors de initialisation par l'algorithme le mal classé;

4. Représentation : une fois tous les individus affectés, recalculer les centres associés à la nouvelle partition : dans cette partie nous utilisons la formule de calcul de nouveau médoïde qui est un individu représentatif selon l'algorithme k-médoïde, car l'algorithme k-means n'est pas applicable en présence d'individus qui ne sont pas du type intervalle, où les centres de gravité sont calculés par la moyenne de tous les individus du cluster:
   - Choisir aléatoirement un non-médoïde M_R ;
   - Pour chaque médoïde M_j

     Calculer le coût du remplacement CR de M_j par M_R, selon la formule de l'erreur carrée suivante : (Kaufman & Rousseeuw, 1987)

     CR = E(M_R) – E(M_j)

     Où **E** = $\sum_{i=1}^{k} \sum_{c \in C_i} SimSem(c, M_i)^2$

     Si CR > 0 *Alors*

- Remplacer Mj par M_R ;
- Réaffecter tous les individus qui n'ont pas été sélectionnés aux k cluster ;

5. Sélectionner un individu le mal classé (celui qui possède la plus grande distance de son médoïde le plus proche, c-à-d la plus petite mesure de SimSem entre l'individu et son médoïde le plus proche) ;
6. Ajouter cet individu à l'ensemble des médoïdes ;
7. Augmenter le nombre des médoïdes ;

Fin Tant Que

Nb_iter = 0 ;

Répéter

8. Affectation : affecter les (N – k) individus aux k clusters, tel que, chaque individu est affecté au médoïde le plus proche;
9. Calculer les nouveaux médoïdes des clusters formés à l'étape N° 08 ;
10. Augmenter le nombre d'itération (nb_iter++)

Si l'algorithme converge (stabilisation des médoïdes)

Alors Sortir de la boucle répéter;

Jusqu'à Nb_iter =Nb_iter _max ;

11. Afficher les clusters K (C1, C2,…, Ck) ;

Fin.

7. Parallélisme de l'algorithme k-means-Combiné

Le parallélisme dans notre cas est définit par l'exécution simultanée des tâches qui sont indépendantes. L'objectif est l'exécution rapide de notre algorithme.

Après la proposition de notre algorithme, nous devons définir les tâches qui pourront être parallélisées.

Notre approche est essentiellement composée de trois procédures:

1. Lire l'ontologie et extraire les composants nécessaires (concepts et instances),

2. Calculer les mesures de similarité,

3. Exécuter l'algorithme k-means-Combiné

Pour effectuer le parallélisme sur nos calculs, nous avons utilisé la notion des threads, et afin de paralléliser notre algorithme, nous avons appliqué cette notion au niveau des affectations des individus aux nouveaux centres correspondants pour chaque itération.

8. Conclusion

Au cours de ce chapitre nous avons présenté les étapes suivies pour arriver à la réalisation de notre objectif qui est le partitionnement d'ontologies à base de clustering. Nous consacrerons le chapitre qui suit à l'implémentation de notre système.

Chapitre IV :

Implémentation et Evaluation

1. Introduction

Nous avons présenté dans le chapitre précédent notre approche sur le partitionnement des ontologies OWL en utilisant le datamining. Dans ce chapitre, nous allons présenter l'implémentation de cette approche. Nous allons tout d'abord présenter l'environnement de développement ainsi que les différents outils utilisés, puis nous donnons une description de notre système de partitionnement à travers quelques fenêtres de capture qui représentent les interfaces de ce dernier

2. Implémentation

Avant de commencer l'implémentation de la phase conceptuelle de notre système de partitionnement, nous allons tout d'abord spécifier les outils utilisés.

2.1. Environnement de développement

Avant de commencer l'implémentation de la phase conceptuelle de notre système de partitionnement, nous allons tout d'abord spécifier les outils utilisés.

1. Matériel utilisé

Toutes les expériences et les tests réalisés dans ce mémoire sont faits sur une machine portable Intel(R) Core (TM) i3-3110 M CPU@ 2.40 GHz avec 4.00 Go de mémoire RAM sous Windows 8.

2. Langages de programmation utilisés

Pour l'implémentation de notre système, nous avons utilisé le langage de programmation Python.

Définition : Python est 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.

L'interpréteur Python et la bibliothèque standard étendue sont librement disponibles sous forme de source ou de binaire pour toutes les principales plates-formes à partir du site Web Python [1], et peuvent être distribués librement. Le même site contient également

[1] https://www.python.org/

des distributions et des pointeurs vers de nombreux modules, programmes et outils Python gratuits, ainsi que de la documentation supplémentaire.

L'interpréteur Python est facilement étendu avec de nouvelles fonctions et de nouveaux types de données implémentés en C ou C ++ (ou d'autres langages pouvant être appelés à partir de C). Python convient également comme langage d'extension pour les applications personnalisables.

2.2. Outils utilisés

Pour réaliser notre implémentation, nous avons utilisé les outils nécessaires suivants :

1. NumPy

NumPy est une extension du langage de programmation Python, destinée à manipuler des matrices ou tableaux multidimensionnels ainsi que des fonctions mathématiques opérant sur ces tableaux.

Plus précisément, cette bibliothèque logicielle libre et open source fournit de multiples fonctions permettant notamment de créer directement un tableau depuis un fichier ou au contraire de sauvegarder un tableau dans un fichier, et manipuler des vecteurs, matrices et polynômes.

NumPy est la base de SciPy, regroupement de bibliothèques Python autour du calcul scientifique.

2. NLTK

Natural LanguageToolkit (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).

3. Owlready

Owlready est un module de programmation orientée ontologie en Python, incluant un quadrore RDF optimisé.

Owlready peut:

- Importez les ontologies OWL 2.0 au format NTriples, RDF / XML ou OWL / XML.

- Exportez les ontologies OWL 2.0 vers NTriples ou RDF / XML.
- Manipule de manière transparente les classes, les instances et les propriétés d'ontologie, comme s'il s'agissait d'objets Python normaux.
- Ajoutez les méthodes Python aux classes d'ontologie.
- Effectuez une classification automatique des classes et des instances, en utilisant le raisonneur HermiT.

4. Threading

Threading est un package qui permet d'exécuter plusieurs opérations simultanément dans le même espace de processus. Il construit des interfaces de thread de niveau supérieur au-dessus du module de thread de niveau inférieur en utilisant différents objets et fonctions.

5. 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.

2.3. Présentation de l'application

Afin d'interagir facilement avec l'utilisateur, lui donner la possibilité de choisir l'ontologie à partitionner et le nombre de clusters souhaité, ainsi de lui afficher les résultats des différentes étapes du partitionnement de l'ontologie choisie, nous avons implémenté une interface graphique présentée dans la figure suivante.

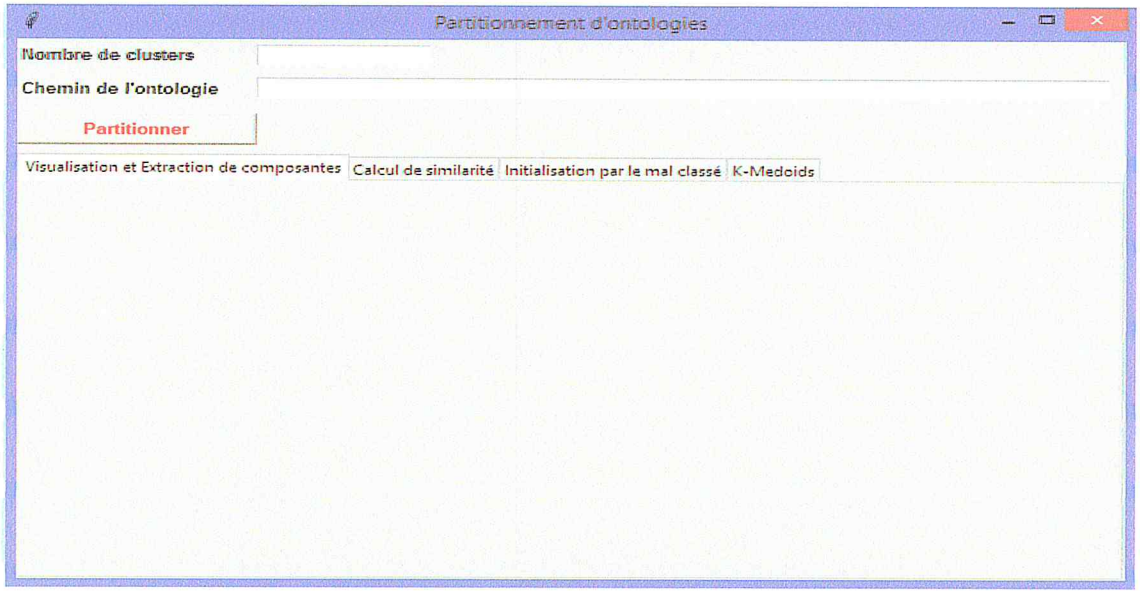


Figure 12 : Interface graphique de notre application

Dans le premier champ, l'utilisateur doit entrer le nombre de clusters souhaité pour partitionner l'ontologie, et il peut sélectionner l'ontologie à partitionner en utilisant le deuxième champ et en suivant le chemin. Les 4 onglets représentent les principales étapes de notre approche du partitionnement.

En choisissant le nombre de clusters et l'ontologie souhaités, et en appuyant sur le bouton '**Partitionner**', le processus de partitionnement de l'ontologie se lance et son résultat s'affiche.

Nous présentons un affichage d'un exemple d'une petite ontologie à 14 concepts avec un nombre de clusters égal à 4 afin de suivre les étapes de notre approche.

La figure suivante présente l'affichage après l'extraction de composants de l'ontologie, grâce au premier onglet qui nous permet de visualiser l'ontologie à partitionner ainsi ses composantes. Nous pouvons remarquer, par exemple, que le concept '***abstract***' n'a aucune instance.

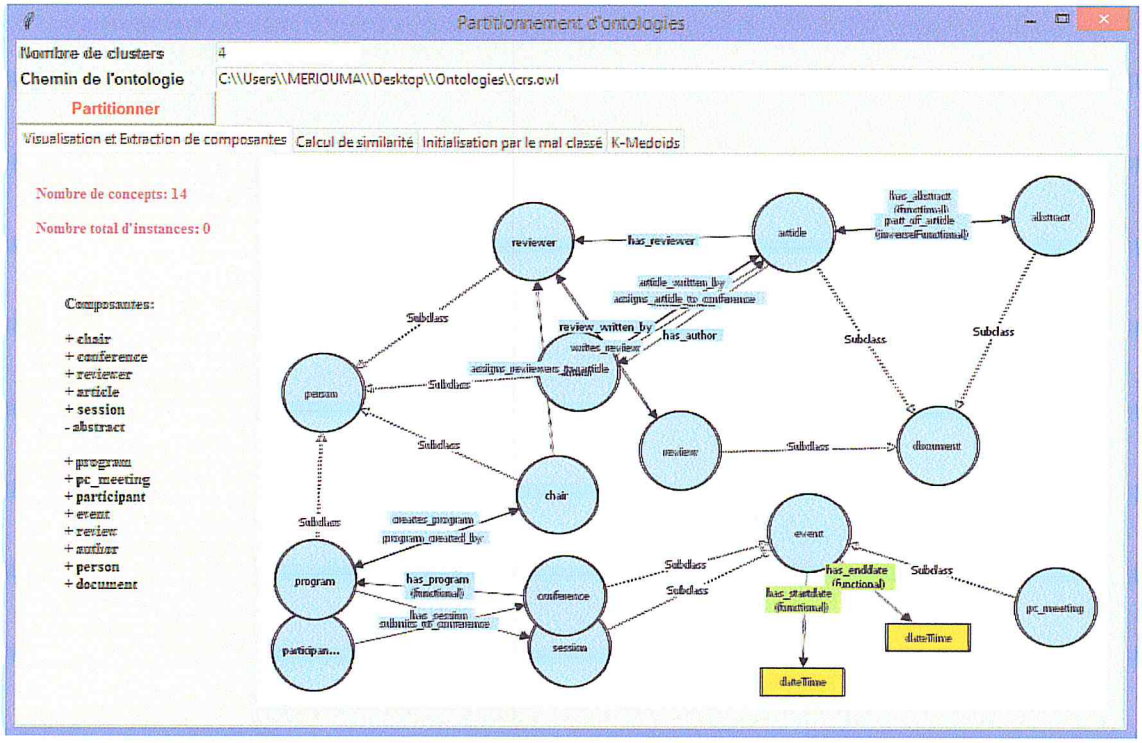
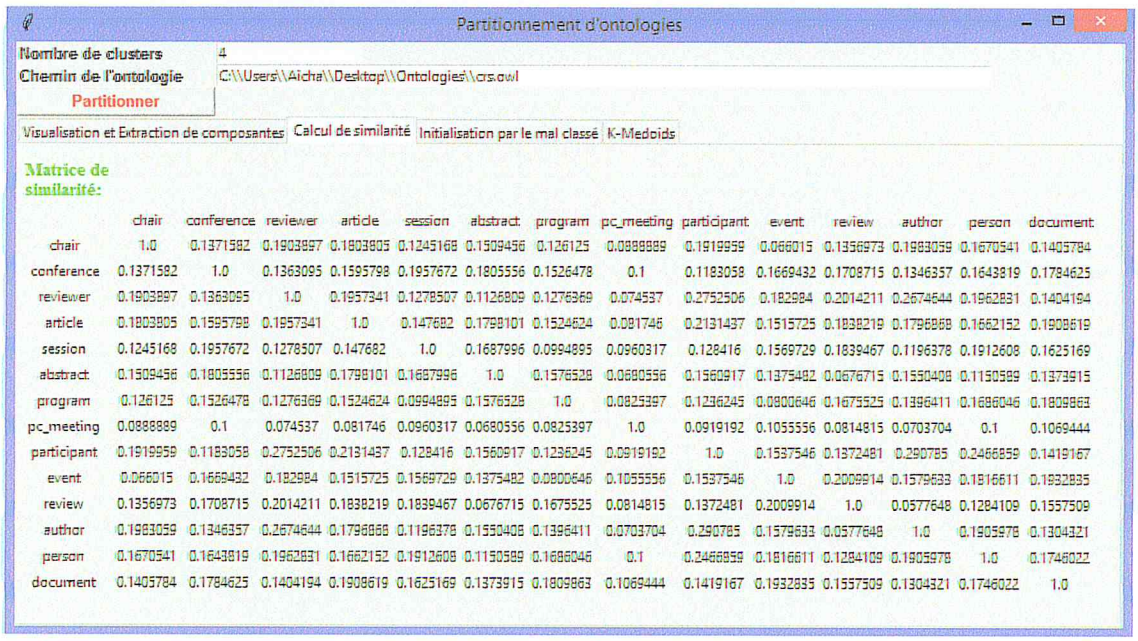


Figure 13 : Extraction des composantes de l'ontologie

Après l'extraction des composants de l'ontologie, une matrice de similarité entre les différents concepts de cette dernière est créée en utilisant les différentes mesures de similarité présentées dans le chapitre précédent. La figure ci-dessous présente la matrice générée pour notre exemple.



	chair	conference	reviewer	article	session	abstract	program	pc_meeting	participant	event	review	author	person	document
chair	1.0	0.1371582	0.1903897	0.1803805	0.1245168	0.1509456	0.126125	0.0888889	0.1919959	0.066015	0.1356973	0.1983059	0.1670541	0.1405784
conference	0.1371582	1.0	0.1363095	0.1595798	0.1957672	0.1805556	0.1526478	0.1	0.1183058	0.1669432	0.1708715	0.1346357	0.1643819	0.1784625
reviewer	0.1903897	0.1363095	1.0	0.1957341	0.1278507	0.1126809	0.1276369	0.074537	0.2752506	0.182984	0.2014211	0.2674644	0.1962831	0.1404194
article	0.1803805	0.1595798	0.1957341	1.0	0.147682	0.1798101	0.1524624	0.081746	0.2131437	0.1515725	0.1838219	0.1796868	0.1662152	0.1908619
session	0.1245168	0.1957672	0.1278507	0.147682	1.0	0.1687996	0.0994895	0.0960317	0.128416	0.1569729	0.1839467	0.1196378	0.1912608	0.1625169
abstract	0.1509456	0.1805556	0.1126809	0.1798101	0.1687996	1.0	0.1576528	0.0680556	0.1560917	0.1375482	0.0676715	0.1550408	0.1150589	0.1373915
program	0.126125	0.1526478	0.1276369	0.1524624	0.0994895	0.1576528	1.0	0.0825397	0.1236245	0.0800646	0.1675525	0.1396411	0.1686046	0.1809863
pc_meeting	0.0888889	0.1	0.074537	0.081746	0.0960317	0.0680556	0.0825397	1.0	0.0919192	0.1055556	0.0814815	0.0703704	0.1	0.1069444
participant	0.1919959	0.1183058	0.2752506	0.2131437	0.128416	0.1560917	0.1236245	0.0919192	1.0	0.1537546	0.1372481	0.290785	0.2466859	0.1419167
event	0.066015	0.1669432	0.182984	0.1515725	0.1569729	0.1375482	0.0800646	0.1055556	0.1537546	1.0	0.2009914	0.1579633	0.1816611	0.1932835
review	0.1356973	0.1708715	0.2014211	0.1838219	0.1839467	0.0676715	0.1675525	0.0814815	0.1372481	0.2009914	1.0	0.0577648	0.1284109	0.1557509
author	0.1983059	0.1346357	0.2674644	0.1796868	0.1196378	0.1550408	0.1396411	0.0703704	0.290785	0.1579633	0.0577648	1.0	0.1905978	0.1304321
person	0.1670541	0.1643819	0.1962831	0.1662152	0.1912608	0.1150589	0.1686046	0.1	0.2466859	0.1816611	0.1284109	0.1905978	1.0	0.1746022
document	0.1405784	0.1784625	0.1404194	0.1908619	0.1625169	0.1373915	0.1809863	0.1069444	0.1419167	0.1932835	0.1557509	0.1304321	0.1746022	1.0

Figure 14 : Calcul de similarité entre les concepts de l'ontologie

Ensuite, les centres initiaux des clusters sont initialisés par la méthode du mal classé. La figure suivante montre les résultats de cette étape :

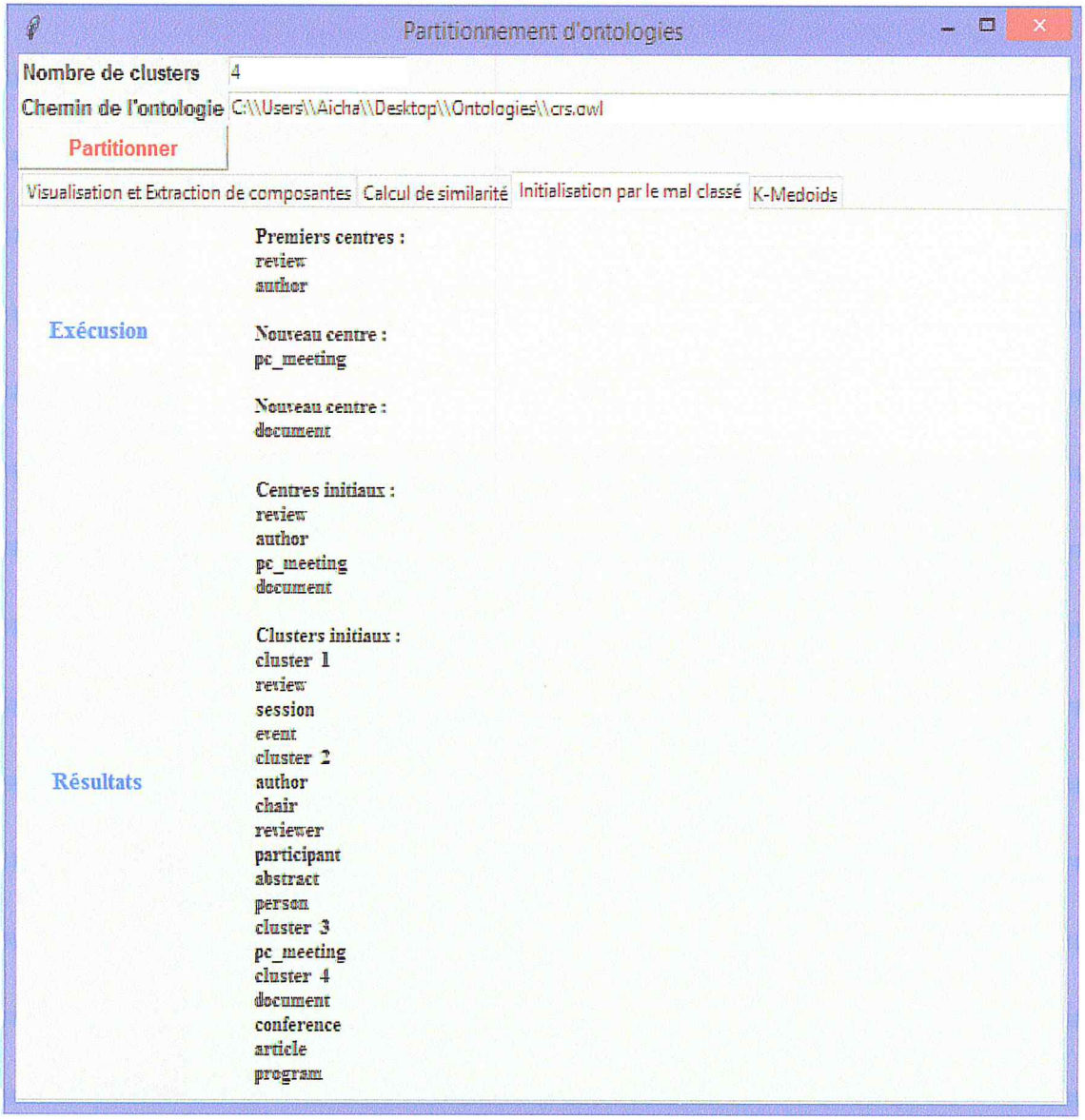


Figure 15 : Initialisation des centres initiaux par le mal classé

La phase suivante consiste à la redistribution des centres et la réaffectation des concepts aux centres les plus similaires jusqu'à la stabilisation de ces derniers. La figure ci-dessous montre le résultat de cette phase qui est aussi le résultat final de l'approche :

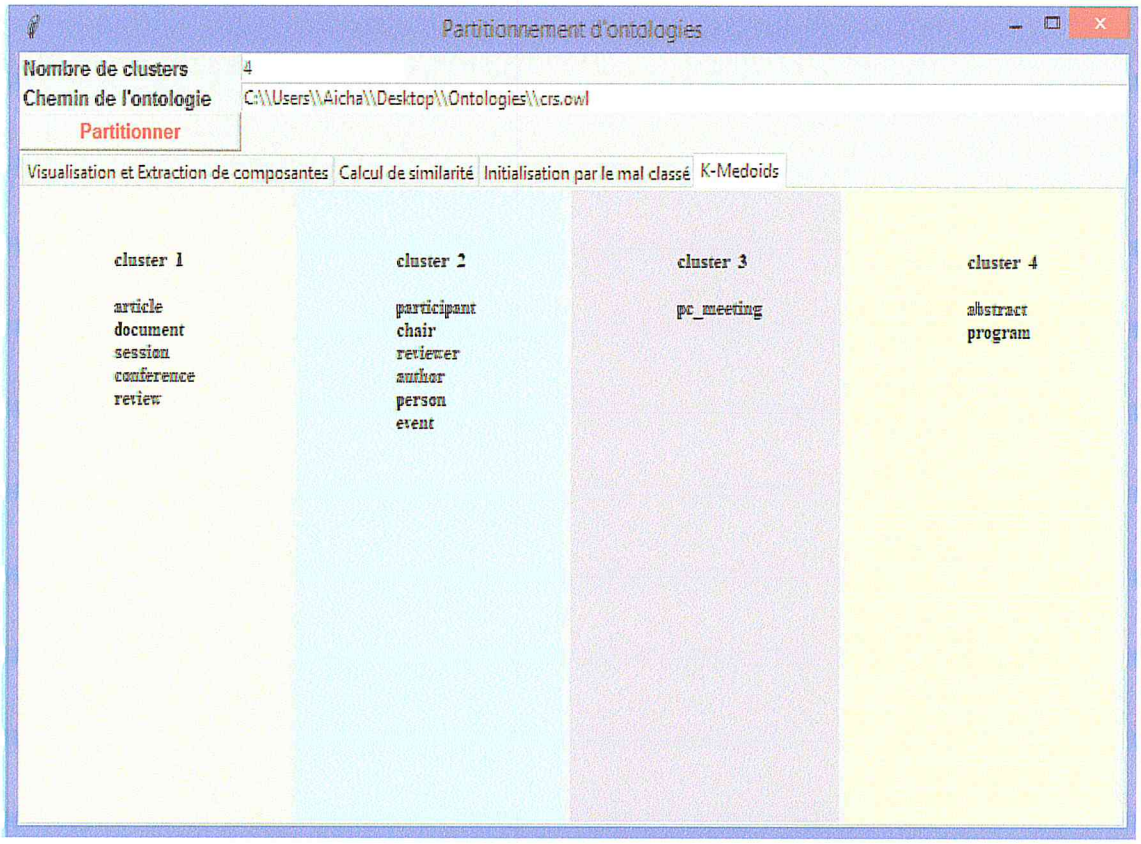


Figure 16 : Résultats du partitionnement de l'ontologie

3. Evaluation

Afin de tester la fiabilité et de montrer la faisabilité de notre approche, nous avons développé le système de partitionnement par l'implémentation de l'algorithme K-Means-Combiné dont les fonctionnalités principales ont été présentées dans les sections précédentes. Ce système a besoin d'être expérimenté sur des ontologies volumineuses afin de vérifier ses fonctions.

Nous avons effectué des expériences sur des ontologies de différentes tailles, petite, moyenne et grande, selon le tableau suivant :

Taille	Ontologie (OWL)	Nombre de classes	Nombre d'instances	Lien
Petite	confious	56	17	http://www.confious.com
	micro	31	4	www.microarch.org

	Linklings	37	5	http://www.linklings.com/
	cmt	29	0	http://msrcmt.research.microsoft.com/cmt
	MyReview	38	2	http://myreview.intellagence.eu/
Moyenne	edas	103	114	http://edas.info/
	iasted	140	4	http://iasted.com/conferences/2005/cancun/ms.htm
	pizza	100	0	https://ontohub.org/pizza/
Grande	mouse	2744	0	-
	human	3304	0	-
	agro	1685	613	http://www.obofoundry.org/ontology/agro.html

Tableau 3 : Ontologies de test

Nous rappelons que le calcul de similarité entre les différents concepts de l'ontologie se fait en combinant différentes mesures de similarité, y compris la mesure extensionnelle qui se base sur les instances des classes. Vu que parmi les ontologies trouvées pour le test, y a certaines qui ne contiennent pas un nombre suffisant d'instances distribuées en équilibre entre les concepts de l'ontologie, nous évaluons notre approche de partitionnement sans la mesure extensionnelle pour ces ontologies, et sans et avec la mesure extensionnelle pour les autres.

3.1. Mesures d'évaluation utilisées

Pour mesurer la qualité d'une classification, différentes mesures d'évaluation existent. Les plus connues et les plus utilisées sont :

- ***Cohésion :*** mesure la compacité entre les objets dans un même cluster. Plus les concepts à l'intérieur des clusters sont homogènes, plus les valeurs de similarité entre eux sont élevées.

- ***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 l'une 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 ont éloignés des objets des autres clusters. Pour évaluer la qualité de notre approche de partitionnement, nous avons choisis 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 :

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

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 maximales de cohésion, et des valeurs minimales de séparation.

3.2. Résultats expérimentaux et discussion

Dans cette section nous présentons les résultats obtenus. La première série de tests a été faite sur les petites ontologies : *confious*, *micro*, *Linklings*, *cmt* et *MyReview*, avec différentes valeurs de K. Les résultats obtenus sans utilisation de la mesure extensionnelle sont résumés dans le tableau suivant :

Ontologie	K=5	K=10	K=15	K=20
confious	0.0196128183	0.0282201249	0.0369751570	0.0417756937
micro	0.0302741122	0.0443773303	0.0549974878	0.0680156416
Linklings	0.0324595883	0.0457705904	0.0489720928	0.0675406823
cmt	0.0287775362	0.0434034858	0.0559312169	0.0768825564
MyReview	0.0189622610	0.0349699594	0.0448162413	0.0573678598

Tableau 4 : Valeurs de la mesure de séparation de la première série de tests sans ME

Les résultats de la première série de tests avec la mesure extensionnelle sont présentés dans le tableau suivant :

Ontologie	K=5	K=10	K=15	K=20
confious	0.0116243115	0.0116951103	0.0194201728	0.0226330885
micro	0.0154749837	0.0229781977	0.0284438676	0.0403641200
Linklings	0.0179942830	0.0309356089	0.0320707591	0.0358571059
MyReview	0.0114955606	0.0174823840	0.0257497535	0.0341549598

Tableau 5 : Valeurs de la mesure de séparation de la première série de tests avec ME

Les résultats présentés dans les deux tableaux précédents nous ont permis d'avoir les barres suivantes :

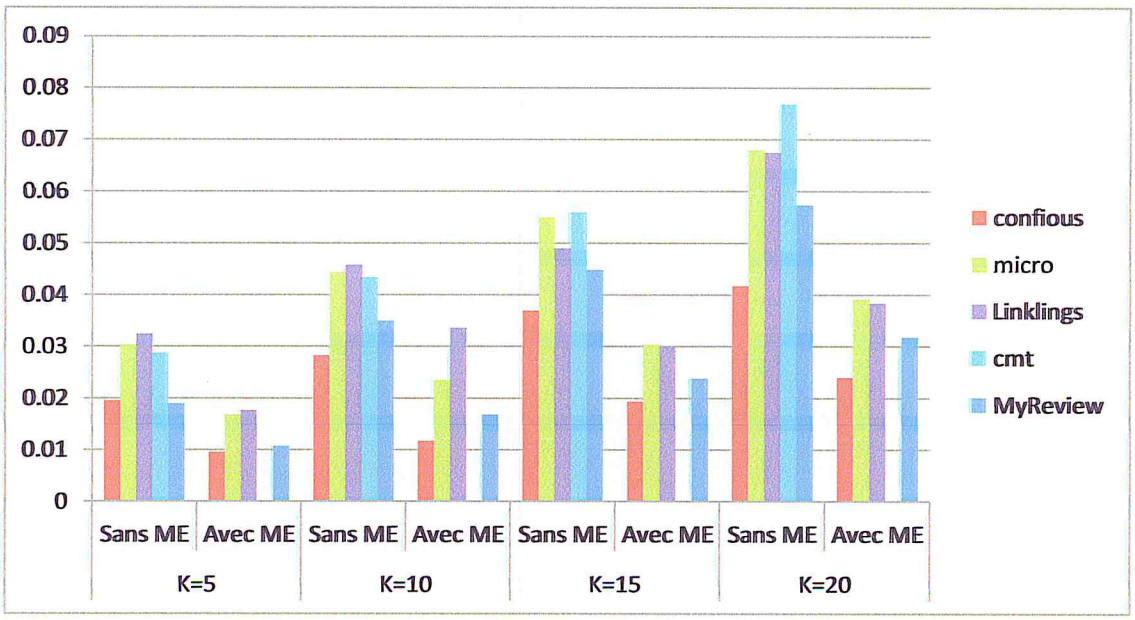


Figure 17 : Evaluation de la première expérience par la mesure de séparation

Il est remarquable, à partir de cette figure, que les résultats de cette expérience sont bons, et que les meilleurs résultats sont pour des petites valeurs de K. Il est aussi remarquable que les résultats avec mesure extensionnelle sont mieux que sans cette mesure.

La deuxième expérience a été effectuée sur les ontologies de taille moyenne : *edas*, *iasted* et *pizza*.

Les résultats obtenus sans mesure extensionnelle sont représentés dans le tableau 6 :

Ontologie	K=5	K=10	K=15	K=20
edas	0.0195020782	0.0196724575	0.0254203145	0.0262475468
iasted	0.0156834866	0.0206978403	0.0238703309	0.0296127909
pizza	0.0141404881	0.0223256776	0.0273742799	0.0253355131

Tableau 6 : Valeurs de la mesure de séparation de la deuxième série de tests sans ME

Les résultats obtenus avec mesure extensionnelle sont résumés dans le tableau suivant :

Ontologie	K=5	K=10	K=15	K=20
edas	0.0104775031	0.0124290266	0.0143204551	0.0153311770
iasted	0.0067077714	0.0102771334	0.0125170043	0.0153311770

Tableau 7 : Valeurs de la mesure de séparation de la deuxième série de tests avec ME

La figure suivante traduit les résultats présentés dans les deux tableaux ci-dessus.

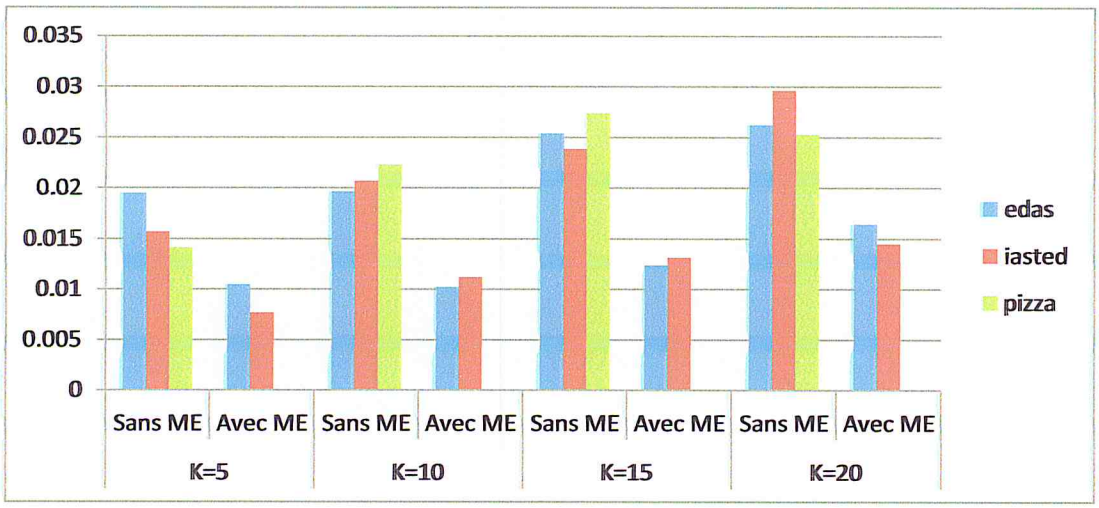


Figure 18 : Evaluation de la deuxième expérience par la mesure de séparation

A partir de cette figure, nous pouvons remarquer que notre approche de partitionnement, surtout avec la mesure extensionnelle, a donné de bons résultats pour les ontologies de cette série de tests, et que, plus le nombre de clusters est augmenté, plus la valeur de séparation est élevée.

La troisième série de tests a été effectué sur trois grandes ontologies : *mouse*, *human* et *agro*. Les résultats obtenus sans calcul de mesure extensionnelle sont représentés dans le tableau suivant :

Ontologie	K=5	K=10	K=15	K=20
mouse	0.0201399231	0.0349648516	0.0364482172	0.0403230298
human	0.0001404105	0.0346798516	0.0341240397	0.0380407007
agro	0.0172733647	0.0215883081	0.0226409271	0.0209044987

Tableau 8 : Valeurs de la mesure de séparation de la troisième série de tests sans ME

Les résultats de cette expérience avec mesure extensionnelle sont représentés par le tableau suivant :

Ontologie	K=5	K=10	K=15	K=20
agro	0.0100426718	0.0173369303	0.0980321047	0.0131748632

Tableau 9 : Valeurs de la mesure de séparation de la troisième série de tests avec ME

Les résultats des deux tableaux précédents peuvent être présentés dans la figure suivante :

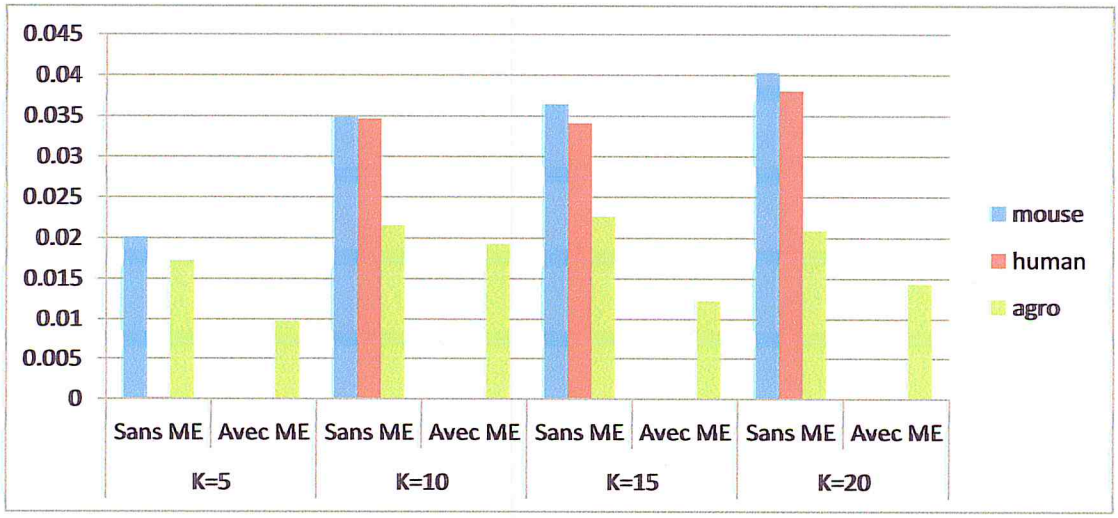


Figure 19 : Evaluation de la troisième expérience par la mesure de séparation

On peut remarquer, à partir de cette figure que, aussi pour les ontologies de grande taille, notre approche de partitionnement est efficace. Il est aussi remarquable que les meilleures valeurs de séparation sont obtenues pour des petits nombres de clusters, et que

pour l'ontologie '*agro*', la performance du partitionnement est plus élevée avec la mesure extensionnelle.

Les résultats des tests effectués et présentés précédemment peuvent être résumés dans les deux figures suivantes :

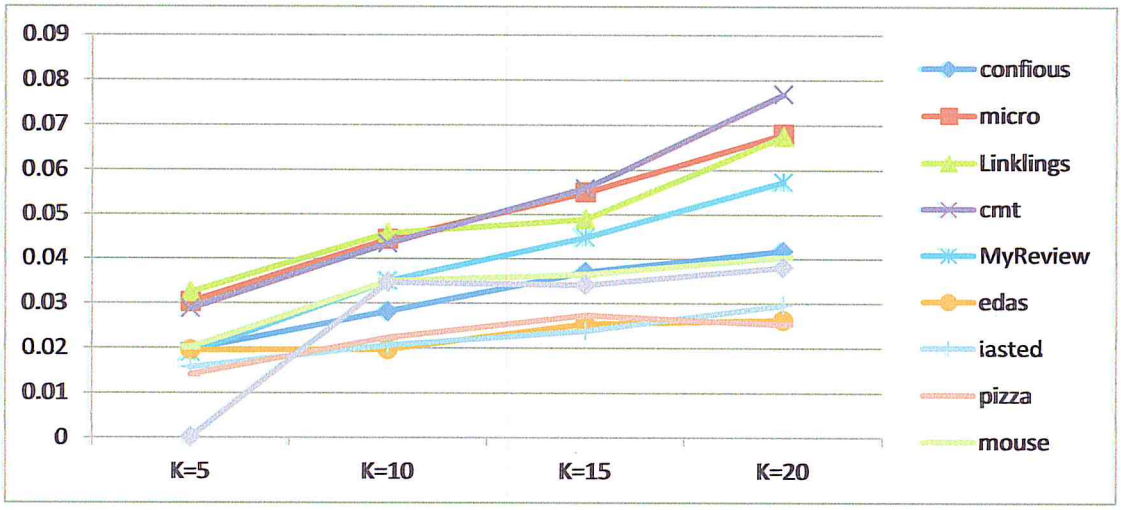


Figure 20 : Evaluation du nombre de clusters par la mesure de séparation

Il est facilement remarquable, à partir de cette figure, aussi à partir des figures 17, 18 et 19, que plus le nombre de clusters est diminué, plus la valeur de séparation est faible. Les meilleures valeurs de séparation sont donc pour K égal à 5.

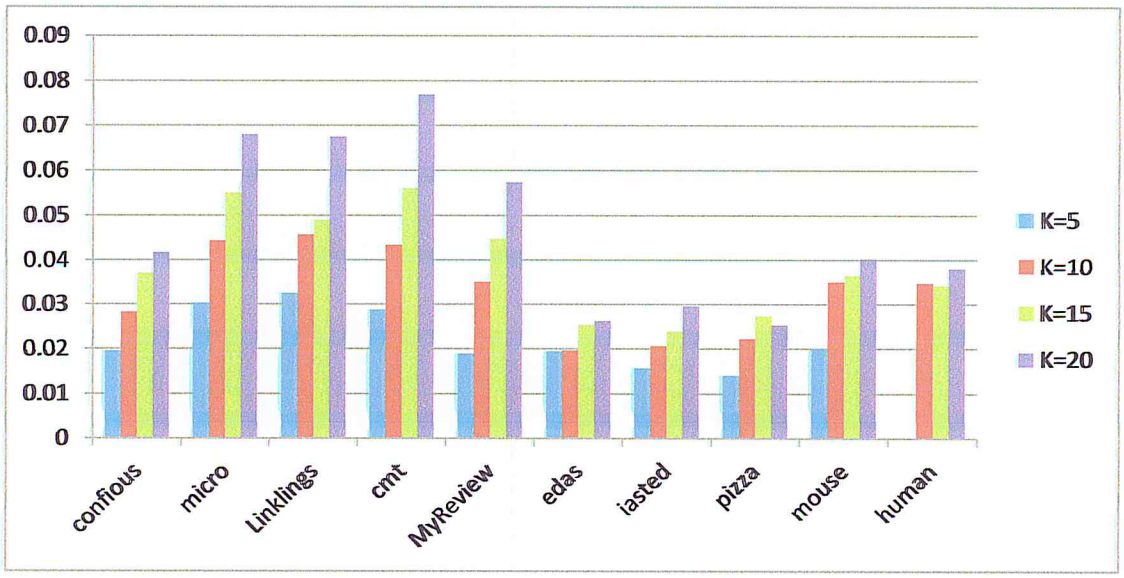


Figure 21 : Evaluation de notre approche de partitionnement par la mesure de séparation

Nous remarquons à partir de la figure précédente que les meilleurs résultats ont été obtenus pour les ontologies de taille moyenne, puis pour les ontologies volumineuses. Cependant, la meilleure valeur a été pour la grande ontologie '*human*' avec 5 clusters qui était presque nulle.

La quatrième expérience a été effectuée sur les ontologies contenant des instances, en modifiant le calcul du deuxième seuil S2.

Nous avons résumé les résultats obtenus dans le tableau suivant :

Taille	Ontologie	K=5	K=10	K=15	K=20
Petite	confious	0.0095764251	0.0160549677	0.0194201728	0.0234738764
	micro	0.0154749837	0.0235304831	0.0299622685	0.0392060964
	Linklings	0.0179528268	0.0335825540	0.0274270001	0.0358571059
	MyReview	0.0114955606	0.0170753311	0.0228069637	0.0341549598
Moyenne	edas	0.0104627376	0.0117383679	0.0150424707	0.0152432721
	iasted	0.0074247756	0.0101987321	0.0138671186	0.0142871652
Grande	agro	0.0115602839	0.0174463201	0.0103782941	0.0113726403

Tableau 10 : Valeurs de la mesure de séparation de la quatrième série de tests

Les données dans ce tableau nous ont permis de réaliser le graphe dans la figure suivante :

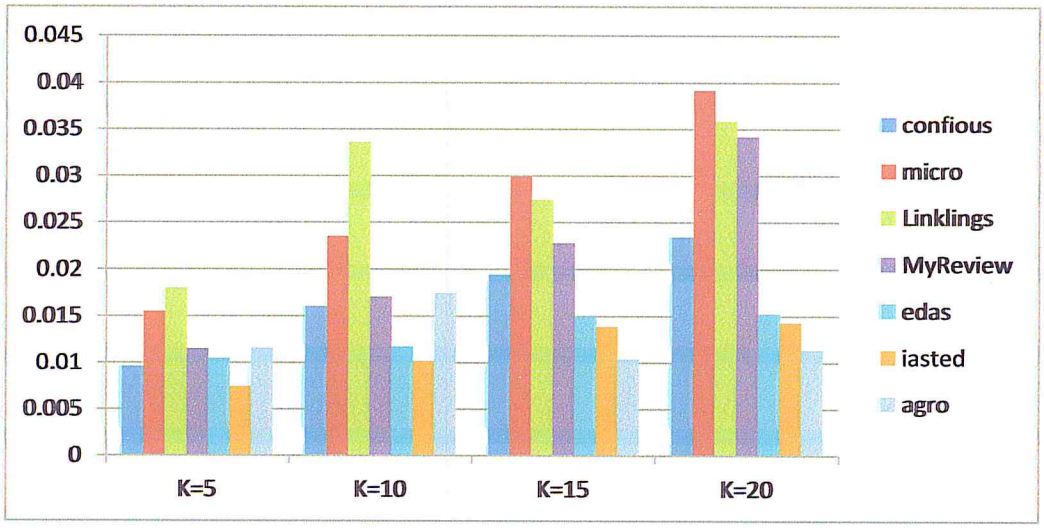


Figure 22 : Evaluation de la quatrième expérience par la mesure de séparation

A partir de la figure 22, on peut remarquer que les valeurs de séparation sont aussi bonnes pour cette expérience. Les meilleures valeurs sont pour les ontologies de grande et moyenne taille.

Nous avons effectué une cinquième série de tests en changeant le calcul de la mesure sémantique tel qu'on combine que les mesures de similarité ayant une valeur supérieur ou égale à un certain seuil, ici égal à 0.2. Nous présentons les résultats dans le tableau ci-dessous :

Taille	Ontologie	K=5	K=10	K=15	K=20
Petite	confious	0.0603058971	0.0823692810	0.0749330305	0.1004017828
	micro	0.0768482309	0.0970828625	0.1110958338	0.1388647917
	Linklings	0.0491281511	0.0578201528	0.0959609760	0.1028434982
	MyReview	0.0480845068	0.0688643983	0.0991060819	0.1161398068
Moyenne	edas	0.0394248471	0.0589955124	0.0665914173	0.0712903901
	iasted	0.0445566395	0.0635636903	0.0578511360	0.0691450570
Grande	agro	0.0398378395	0.0788203931	0.0877210293	0.1192395031

Tableau 11 : Valeurs de la mesure de séparation de la quatrième série de tests

Les résultats présentés dans le tableau précédent peuvent être traduits par la figure suivante :

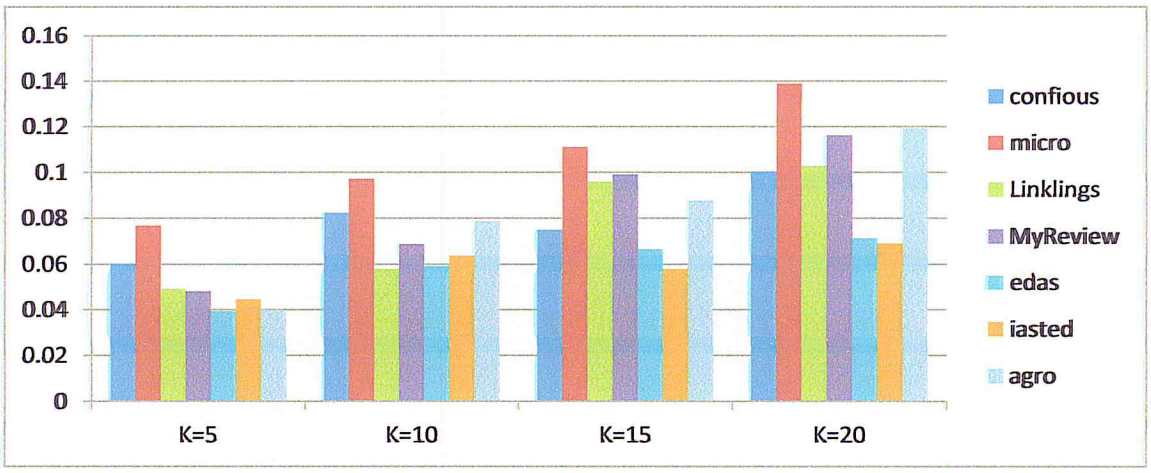


Figure 23 : Evaluation de la quatrième expérience par la mesure de séparation

Nous pouvons remarquer à partir de la figure ci-dessus que, dans cette expérience, les valeurs de séparation ont été légèrement augmentées par rapport les autres séries de tests, mais ils sont aussi de bonnes valeurs. Les meilleures valeurs sont pour K avec une petite valeur, et pour les ontologies de moyenne et grande taille.

A travers les tests que nous avons effectués, nous concluons que l'approche de partitionnement présenté dans ce travail est bonne, sans et avec la mesure extensionnelle. Cela est clairement remarquable à partir de la valeur de la mesure d'évaluation utilisée dans les cinq séries de tests effectuées. Ce qui signifie que notre approche donne de bon résultats quelque soit la taille de l'ontologie à partitionner.

4. Conclusion

Dans ce chapitre, nous avons présenté l'implémentation de notre approche de partitionnement d'ontologies et les différents outils utilisés. Nous avons aussi présenté ses résultats d'évaluation en utilisant la mesure de séparation sur des ontologies de différentes tailles. Les résultats expérimentaux montrent que notre approche du partitionnement est positive.

Conclusion et Perspectives

Conclusion et Perspectives

Les travaux menés dans ce mémoire nous ont permis d'approfondir nos connaissances dans le domaine de l'ingénierie des connaissances et plus particulièrement l'ingénierie ontologique.

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. Ensuite, nous avons parlé des composants des ontologies et les différents langages de présentations de ces dernières.

Nous avons aussi cité plusieurs opérations qui peuvent être appliquées sur les ontologies, parmi elles, le partitionnement d'ontologie qui est l'objectif de ce travail.

Une bonne partie du deuxième chapitre a été consacrée au partitionnement des ontologies, ses concepts de base et ses objectifs ainsi que les différentes techniques de partitionnement qui existent. Cette partie a été finie par une étude comparative en étudiant tous les points qui peuvent servir à mener notre travail. Par la suite nous avons évoqué le partitionnement à base de clustering, aussi les fondements théoriques et les concepts de bases ont été exposés.

Dans le chapitre 3, nous avons proposé une nouvelle approche de partitionnement des ontologies à base de clustering, qui se base sur la combinaison de plusieurs mesures de calcul de similarité. Cette approche est une combinaison de plusieurs algorithmes dans le but d'améliorer la qualité de partitionnement en utilisant l'algorithme de clustering K-means.

Sur la base de cette approche, nous avons implémenté notre système de partitionnement. Des expérimentations ont été menées sur plusieurs ontologies de tailles différentes, afin de qualifier le travail effectué et de démontrer l'efficacité de notre approche dans la production des partitions de bonne qualité, des tests sur les différentes partitions générées ont été effectués en utilisant la mesure d'évaluation 'séparation'.

Finalement nous avons discuté les résultats obtenus en soulignant les points positifs de cette expérimentation.

Toutefois, dans le souci d'améliorer notre système de partitionnement, et afin de juger mieux la qualité des partitions générées, nous souhaitons appliquer d'autres métriques d'évaluation telles que : la cohésion et l'indice de silhouette.

Enfin, ce travail a été une expérience très enrichissante qui nous a permis d'apporter des tentatives de solution à un problème d'actualité.

Références Bibliographiques

Liste des Références

[Borst, 1997] : Borst W. N. Construction of Engineering Ontologies. Center for Telematica and Information Technology, University of Tweenty, Enschede, NL. 1997.

[Corcho et al., 2003] : Corcho O, Fernández López M et al. Methodologies, tools and languages for building ontologies. Where is their meeting point? , 46, 41-64, 2003

[Dempster et al., 1977] : A.P. Dempster, N.M. Laird and D.B. Rubin : « Maximum vraisemblance fromincomplete data via the EM algorithm », J. Royal Statis. Society, B39, P 1-38, 1977.

[Ding et al., 2013] : Guohui Ding, Tianhe Sun et Yingnan Xu, «Multi Schema Matching Based On Clustering Techniques », 2013 10th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)

[Elbyed, 09] : M. Abdeltif ELBYED, ROMIE, une approche d'alignement d'ontologies à base d'instances, Thèse présentée pour l'obtention du grade de Docteur de l'INSTITUT NATIONAL DES TELECOMMUNICATIONS, (16 Octobre 2009)

[Euzenat & Shvaiko, 2007] : J. Euzenat, and P. Shvaiko, "Ontology matching », Springer, Heidelberg (DE), 2007

[Euzenat et al., 2004] : Euzenat J., Bach T.L., Barrasa J., Bouquet P., Bo J.D., Dieng-Kuntz R.;Ehrig M., Hauswirth M., Jarrar M., Lara R., Maynard D., Napoli A., Stamou G., Stuckenschmidt H., Shvaiko P., Tessaris S., Acker S.V. and Zaihrayeu, "I. State of the art on ontology alignment, deliverable 2.2.3", IST Knowledge web No E, 80 p, June 2004.

[Euzenat et al., 2007]: Euzenat J. and al, "Heterogeneity in the semantic web, deliverable 2.2", Knowledge Web, December 2007.

[Euzenat, 2008] : Jérôme Euzenat. Quelques pistes pour une distance entre ontologies. Actes 1er atelier EGC 2008 sur similarité sémantique, Jan 2008, Sophia-Antipolis, France. No commercial editor, pp.51-66, 2008

[Fürst, 2002] : Frédéric Fürst, Ingénierie des Connaissances, Rapport de recherche, Institut de recherche en informatique de Nates –France- (Octobre 2002).

[Garcia et al., 2012] : Ana Carolina Garcia, Letícia Tiveron, Claudia Justel, Maria Cláudia Cavalcanti, « Applying Graph Partitioning Techniques to Modularize Large Ontologies », 2012

[Gesu, 1988] : V. di Gesu « Mathematical Morphology and Image Analysis : A Fuzzy Approach ». Workshop on Knowledge-Based Systems and Models of Logical Reasoning, 1988.

[Grau et al., 2007] : B. Cuenca Grau, I. Horrocks, Y. Kazakov, and U. Sattler. Extracting modules from ontologies: Theory and practice. Technical report, University of Manchester, 2007. Available from: http://www.cs.man.ac.uk/_bcg/Publications.html.

[Grau et al., 2005] : G. Grau, B. Parsia, E.Sirin and A.Kalyanpur, "Modularizing owl ontologies". In Proceedings of the KCAP-2005 Workshop on Ontology Management, Ban, Canada.

[Gruber, 1993] : Gruber T. «A Translation Approach to Portable Ontology Specifications». Knowledge Acquisition, 5(2), 199-220, 1993.

[Guarino, 1997] : Guarino N. «Understanding, building and using ontologies». International J. Human-Computer Studies, 46, 293-310, 1997b

[Guellil & Zaoui, 2009] : Z. Guellil et L.Zaoui "Proposition d'une solution au problem d'initialisation cas du K-means", 2nd Conférence Internationale sur l'Informatique et ses Applications · CIIA 2009 Saida, Algeria, May 3-4

[Hartigan & Wong,1979] : Hartigan, J. A.; Wong, M. A. (1979), "Algorithm AS 136: A K-Means Clustering Algorithm". Journal of the Royal Statistical Society, Series C 28 (1): 100–108.

[He et al., 2003] : He, B., Chang, K., & Han, J. (2003, December). Automatic Complex Schema Matching across Web Query Interfaces: A Correlation Mining Approach. Technical Report UIUCDCS R-2003-2388, Department o Computer Science, UIUC.

[Hubert et al., 1985] : Hubert, L., Arabie, P.: Comparing partitions. J. Intell. Inf. Syst. 2(1), 193–218 (1985).

[IZZA, 2006] : S. IZZA, « Intégration des Systèmes d'information industriels Une approche flexible basée sur les services sémantiques », thèse Pour obtenir le grade de Docteur de l'Ecole Nationale Supérieure des Mines de Saint-Etienne, 2006, p3-32-33

[Karypis et al., 1999] : Karypis G., Eui-Hong, H., et Kumar, V. Chameleon : Hierarchical Clustering Using Dynamic Modeling. Computer, n° 32(8) : 68-75, 1999.

[Kaufman & Rousseeuw,1987] : Kaufman, L. and Rousseeuw, P.J. (1987), Clustering by means of Medoids, in Statistical Data Analysis Based on the L1 - Norm and Related Methods, edited by Y. Dodge, North- Holland, 405– 416.

[Lance et al., 1967] : Lance, G.N. & Williams, W.T.: A general theory of classificatory sorting strategies : I. Hierarchicals systems. Computer Journal, n° 9, pp 373-380, 1967.

[Leacock et al., 1998] : Leacock C, ChodorowM. Combining local context andWordNet similarity for word sense identification. In :WordNet: an electronic lexical database. Cambridge (USA) : MIT Press, 1998

[MacQueen, 1967] : J. B. MacQueen, 1967: « Somme methods for classification and Analysis of Multivariate Observations, Proceeding of 5-th Berkeley Symposium on Mathematical Statistics and Probability", Berkeley, University of California, Press, n°1, pp281-297, 1967.

[Madhavan et al., 2001] : Madhavan J, Bernstein P, Rahm E. Generic schema matching with cupid. In : Proceedings of the 27th International Conference on Very Large Data Bases. Rome (Italy) : Morgan Kaufmann Publishers Inc, 2001, p. 49-58.

[McGuiness et al., 2002] : McGuiness et al. Daml+oil: An ontology langage for the semantic web. IEEE Intelligent Systems, 17, p72-80, 2002.

[Mellal, 2007] : Mellal N. Réalisation de l'interopérabilité sémantique des systèmes, basée sur les ontologies et les flux d'information. Thèse dc Doctorat. Annecy Lc Vieux : Polytech Savoie, 2007.

[Monge et al., 1996] : Monge A, Elkan C. The field-matching problem: algorithm and applications. In : Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining, 1996, p. 267-70.

[Neeches et al., 1993] : Neeches, Finin T, Fikes R.E, Gruber T.R, Senator T et Swartou W.R. « Enabling technology for knowledge sharing » AI Magazine. Vol.12, no 3, 1993.

[Rahm & Bernstein, 2001] : A Survey of Approaches to Automatic Schema Matching.

VLDB Journal, 10(4):334–350, (2001).

[Resnik, 1999] : Resnik, P. "Semantic Similarity in a Taxonomy : An Information-Based Measure and its Application to Problems of Ambiguity in Natural Language." Journal of Artificial Intelligence 11 (1999) : 95-130.

[Sellami, 2008] : S. Sellami, A. Benharkat, Y. Amghar R Rifaieh, "Study of Challenges and Techniques in Large Scale Matching". In Proceedings of the 10th International Conference on Enterprise Information Systems, Barcelona, Spain.pp.355-361. 2008.

[Setti et al., 2015] : Soraya Setti Ahmed, Mimoun Malki, Sidi Mohamed Benslimane,"Ontology Partitioning: Clustering Based Approach", International Journal of Information Technology and Computer Science(IJITCS), vol.7, no.6, pp.1-11, 2015. DOI: 10.5815/ijitcs.2015.06.01

[Shvaiko & Euzenat, 2005] : A Survey of Schema-Based Matching Approaches.Journal on Data Semantics, IV:146–171, (2005).

[Thi et al., 2008] : Thi Le Pham, Nhan Le-Thanh, Peter Sander. « Decomposition based Reasoning for Large Knowledge Bases in Description Logics ». Integrated Computer-Aided Engineering, IOS Press, 2008, 15 (1), pp.53-70.

[Wache et al., 2001] : Wache H, Voegele, T., Visser, U., Stuckenschmidt, H., Schuster, G.,Neumann, H. et Huebner, S. Ontology-Based Integration of Information - A Survey of Existing Approaches. Proc. of the IJCAI-01 Workshop on Ontologies and Information Sharing , Seattle, August 4-5, 2001.

[Wu & Palmer, 1994] : Wu Z, Palmer M. Verb semantics and lexical selection. In : Proceedings of the 32nd Annual Meeting of the Associations for Computational Linguistics, 1994, p. 133-8.

[Xu, 2003] : Xu L, & Embley, D. (2003, October 20). Using Domain Ontologies to Discover Direct and Indirect Matches for Schema Elements.In : Second International Semantic Web Conference (ISWC- 03).

[Zhang Cheng et al., 2007] : X. Zhang Cheng G, Y. Qu., "Ontology summarization based on rdf sentence graph". In Proceedings of the 16th International Conference on World Wide Web, New York, NY, USA.ACM press. pp. 707-716. 2007.

[Ziani et al., 2010] : Ziani M, Boulanger D, Talens G. Système d'aide à l'alignement d'ontologies métier. 28 congres INFORSID, Marseille 2010 : 345-60.