Espaces vectoriels sémantiques : enrichissement et interprétation de requêtes dans un système d'information distribué et hétérogène

Anthony Ventresque

École Centrale de Nantes Université de Nantes École des Mines de Nantes

ÉCOLE DOCTORALE STIM
« SCIENCES ET TECHNOLOGIES DE L'INFORMATION ET DES MATÉRIAUX »

Année 2009

Espaces vectoriels sémantiques : enrichissement et interprétation de requêtes dans un système d'information distribué et hétérogène

---

THÈSE

pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ DE NANTES

Discipline : INFORMATIQUE

présentée et soutenue publiquement par

Anthony VENTRESQUE

le 26 Septembre 2008

au LINA

devant le jury ci-dessous

Président :	Pr. PrénomPrésident NomPrésident	Institution
Rapporteurs :	Bernd Amann, Professeur	Université Pierre et Marie Curie de Paris
	Mohand Boughanem, Professeur	Université Paul Sabatier de Toulouse
Examinateurs:	Sylvie Cazalens, Maître de Conférence	Université de Nantes
	Philippe Lamarre, Maître de Conférence	Université de Nantes
	Gabriella Pasi, Professeur	Universitá di Milano Bicocca
	Patrick Valduriez, Directeur de Recherche	INRIA

Directeur de thèse : Pr. Patrick VALDURIEZ
Encadrant de thèse : Philippe LAMARRE
Laboratoire : LINA (Laboratoire d'Informatique de Nantes Atlantique)

N° ED 503-030

ESPACES VECTORIELS SÉMANTIQUES : ENRICHISSEMENT ET INTERPRÉTATION DE REQUÊTES DANS UN SYSTÈME D'INFORMATION DISTRIBUÉ ET HÉTÉROGÈNE

Semantic Vector Spaces: Query Enrichment and Interpretation in a Distributed and Heterogeneous Information System

Anthony VENTRESQUE

favet neptunus eunti

Université de Nantes

Anthony VENTRESQUE

Espaces vectoriels sémantiques : enrichissement et interprétation de requêtes dans un système d'information distribué et hétérogène

xvii+136 p.

Ce document a été préparé avec LaTeX2$_\varepsilon$ et la classe `these-IRIN` version 0.92 de l'association de jeunes chercheurs en informatique LOGIN, Université de Nantes. La classe `these-IRIN` est disponible à l'adresse :

`http://login.lina.sciences.univ-nantes.fr/`

Impression : these.tex – 16/4/2009 – 14:17

Révision pour la classe : $Id: these-IRIN.cls,v 1.3 2000/11/19 18:30:42 fred Exp

Lack of interoperability threatens the future of the Internet. The Internet will extend far beyond people using computers and handheld gadgets. By 2007, billions of devices and bits of software will sense and act on the physical world–no humans required. The result: Everything from tiny sensors to mammoth business apps will exchange data. But products won't interoperate if they have incompatible data formats. This will hurt manufacturers and customers, and jeopardize the future of the Internet

— David Truog *et al.* , How The X Internet Will Communicate, 2001.

Remerciements

Je tiens à remercier en tout premier lieu Patrick Valduriez, mon directeur de thèse. Il a toujours cru en l'intérêt de mon sujet de thèse, même (et surtout) quand je ne le voyais pas moi-même. J'ai eu une chance immense de pouvoir côtoyer un chercheur de cette envergure, capable de comprendre l'importance d'une idée et de sentir émerger des problématiques avant tout le monde. Chacune de nos réunions me laissait beaucoup de bons souvenirs et à chaque fois une grosse motivation pour continuer mon travail.

Je veux aussi remercier Bernd Amann et Mohand Boughanem : ils ont accepté de me relire pendant les vacances d'été, alors même qu'il faisait beau (pour une fois) et que tout devait les appeler loin de mon manuscrit. J'ai eu beaucoup d'honneur à les avoir comme rapporteurs de mon travail. Quant à Gabriella Pasi, ce fût un grand plaisir de l'avoir dans mon jury, parce que je trouve son travail très intéressant et sa façon d'en parler remarquablement claire.

Mais ceux que je dois remercier par-dessus tout, ce sont Philippe et Sylvie. J'ai commencé à travailler avec eux dès le master 1 (ex-maîtrise), puis en master 2 (ex-DEA), et en thèse. Si je compte bien, cela fait donc six années de recherche communes. Il est plus qu'évident que j'en ai été marqué. Ma façon de vivre la recherche, c'est eux. Ma ténacité, c'est eux. Ma réussite, c'est aussi la leur. Je les remercie mille fois pour tout ça.

Je ne pouvais pas réussir ma thèse sans une ambiance de travail conviviale. Et sans mentir, j'ai été gâté à Nantes. Je ne remercierais jamais assez Lorraine, Jim, Sandra, Eduardo, Rabab, Manal, Jorge, Jérémie, Charlotte, Erwan, Vidal, Reza, Fabrice, Brice, Estelle, Olivier, Matthieu, Guillaume(s), Fred(s), Gaël, Annie, Marc, Antoine, Cédric, Stéphane, etc. (mille fois pardon pour tous ceux que j'oublierais ici, mais que je n'oublierai jamais) pour tous ces bons moments qui m'ont permis de décompresser et de relativiser tout ça !

Pour réussir un doctorat, il faut être travailleur, mais surtout être curieux. Et dans mon cas, il faut aussi être persévérant et convaincu par le côté « gratuit » de la recherche. Toutes ces qualités, je les dois à un grand nombre de modèles qui m'ont marqué au cours de ma vie : mes enseignants (je voudrais en particulier citer MM. Bossard, Bailhache, Costard et Gabard qui ont toujours mis la curiosité et le travail en avant et ont aiguisés mon amour de l'étude, parfois malgré moi...), l'ACSJ (qui m'a permis de mûrir et de de m'ouvrir au monde), et surtout mes parents, mes grands-parents et ma soeur. Ces derniers ne s'en doutent sans doute pas, mais c'est à leur côté que je suis devenu qui je suis, c'est à leur contact que j'ai acquis les valeurs essentielles qui me définissent aujourd'hui. Pour terminer, car il le faut, j'ai une pensée pour toutes les personnes qui ont été intéressées par ma recherche et/ou mes efforts, et ont semblé fiers de moi. C'est aussi pour tous ceux-là que j'ai voulu faire cette thèse.

Et si c'était à refaire... heureusement que ce n'est pas à refaire !

Sommaire

Introduction

Nous assistons depuis l'émergence du Web comme moyen de communication de masse (milieu des années 1990) à un accroissement considérable des informations qui sont accessibles. Il ne s'agit plus uniquement de sites Web personnels ou de commerce. Les blogs, encyclopédies, journaux, réseaux sociaux, systèmes d'échanges de fichiers ou de contenus vidéos, etc. se multiplient. Les documents mis en ligne sont hétérogènes : textes en langage naturel, vidéos, sons, données enrichies, etc. Cette profusion d'information devient compliquée à gérer. Dans les premiers temps du Web, la connaissance de quelques sites généralistes qui référençaient des sites thématiques suffisait. Avec le développement de l'accès au Web et des mises à disposition de données, les listes de sites n'ont plus suffi : les moteurs de recherche ont vu le jour. Ils sont aujourd'hui fortement liés au Web tel que nous le connaissons : il nous suffit de connaître ces portails pour accéder à la profusion d'information qu'ils indexent. Une étude récente montre ainsi que chaque internaute européen a lancé en moyenne plus de 110 requêtes sur des moteurs de recherche (principalement Google) au mois de mars 2008, soit 24,550 milliards de recherches en Europe [WWW 32]. Néanmoins, lancer une recherche sur « Barack Obama » en Juillet 2008 sur Google retourne 55 800 000 résultats. En comparaison, une recherche sur « Napoléon Bonaparte » ne renvoie *que* 3 230 000 réponses. Comment parcourir ne serait-ce qu'une portion infime de ces résultats ? Comment savoir ce qui est réellement pertinent ? Comment les moteurs de recherche peuvent faire la différence entre tous ces documents, dont certains ne sont que peu pertinents et dont beaucoup sont semblables ?

Le développement d'appareils communicants est lui-aussi extrêmement rapide. Depuis le téléphone à impulsions, les progrès semblent tentaculaires : ADSL, Wifi, Bluetooth, Wimax, MIMO, Wep, RFID, Edge, TNT, etc. permettent à toute une série d'appareils de communiquer entre eux, des téléphones portables aux télévisions en passant par les objets ménagers et les véhicules. Des prototypes de frigidaires sont ainsi capables de se connecter via internet à des supermarchés en ligne pour « se » réapprovisionner, à partir des informations portées par les puces RFID des aliments qu'ils contiennent. D'autres permettent à des véhicules de se connecter en Wap à des services d'info-traffic pour connaître le temps de trajet estimé, les routes les plus intéressantes, etc. Ces quelques exemples permettent de sentir l'importance des échanges d'information qui vont apparaître et se développer très bientôt. Or, les systèmes traditionnels ne sont peut-être pas préparés pour cette révolution. La capacité de traitement des serveurs ne se développe pas avec le même ordre de grandeur et les débits disponibles n'augmentent pas aussi vite. Ainsi, nous avons pu voir des tremblements de terre couper des pays entiers d'accès au Web. De même, les serveurs du ministère des finances sont traditionnellement surchargés lors des déclarations d'impôts, etc.

Le modèle client/serveur a été pour ces raisons remis en question, parce qu'il ne permet pas d'augmenter indéfiniment le nombre de machines connectées : ce qui est appelé le problème du « passage à l'échelle ». La dissymétrie des rôles du client et du serveur place la vulnérabilité du côté de ce dernier, car il reçoit toutes les demandes des clients. D'autres architectures complètement décentralisées émergent depuis quelques années, telles que les systèmes pair-à-pair (P2P). Le paradigme est souvent pour eux celui de pairs égaux et autonomes dont l'organisation essaie de permettre une distribution du stockage et du traitement optimale, c'est-à-dire permettant le passage à l'échelle sans nécessiter de grands serveurs. Les systèmes P2P ont été popularisés par les logiciels de partage de données, par exemple Gnutella [JAB01] ou KaZaA [WWW 16] et sont simples (recherche textuelle). Le problème qui est apparu avec la massification de l'utilisation de ces systèmes est qu'ils sont peu économes en ressources (une requête inonde le réseau) et qu'ils offrent des fonctionnalités limités (recherche par mots-clés). Des solutions dérivées ont donc vu

le jour, qui limitaient l'autonomie ou l'égalité des pairs, pour augmenter leurs performances : super-pairs et P2P structurés [NWQ+02a, SMK+01].

Le problème de l'efficacité de la recherche d'information (RI) sur le Web et les problèmes d'expressivité et d'efficacité des systèmes P2P poussent les deux communautés à se tourner vers une approche identique : la *sémantique*. Il s'agit d'utiliser des données de haut niveau sur le sens contenu dans des documents ou des requêtes, ou qui découle de l'intérêt d'utilisateurs ou de communautés. En informatique, les approches basées sur la sémantique nécessitent souvent de définir des ontologies, c'est-à-dire des formalisations des connaissances. Dans une ontologie, les concepts sont ainsi organisés en réseaux hiérarchiques qui permettent de modéliser des énoncés comme « les chats sont des félins » ; d'autre part des relations s'appliquent à certains concepts suivant certaines règles, dans le but de relier les concepts entre eux : « les chats chassent les souris »; des règles logiques permettent aussi de raisonner sur les relations et les concepts : « si les félins ont de grandes moustaches et que Tom est un chat, alors Tom a de grandes moustaches ». La sémantique permet de représenter documents et requêtes à un niveau d'abstraction supérieur. Les ambiguïtés du langage naturel (synonymie, polysémie) peuvent alors être contournées « par le haut »: la sémantique permet d'adresser directement le sens des énoncés et non leur forme de surface [GVCC98]. Elle permet aussi d'adresser l'hétérogénéité des formats, types, langues des documents, puisque cette représentation commune s'affranchit de cette diversité. Un texte en langage naturel, une image, etc. sont comparables s'ils sont indexés sur une même ontologie. La sémantique permet aussi d'augmenter l'expressivité des systèmes P2P [HIMT03] et de les organiser selon des proximités d'intérêt [CG02].

Parmi les différents modèles de représentation des documents, requêtes, pairs, etc. nous nous sommes intéressés au *modèle vectoriel sémantique* [Woo97, KC92]. Il utilise un espace vectoriel dont les dimensions sont les concepts d'une ontologie. Documents et requêtes sont alors des vecteurs dans cet espace et prennent des valeurs sur différentes dimensions de celui-ci. La pertinence entre documents et requêtes peut être calculée de différentes manières ; l'une d'elles consiste à considérer l'angle entre leurs vecteurs en utilisant le cosinus. Il obtient de très bons résultats en RI. Utiliser les vecteurs sémantiques et cette façon de calculer la pertinence permet de caractériser de manière homogène différents niveaux de représentation : requêtes, documents, collection de documents, pairs, communautés de pairs, etc. Il a néanmoins une limite forte qui est que les dimensions sont indépendantes entre elles. Les dimensions « chat » et « félin » sont aussi liées dans ce modèle que « chat » et « fer à repasser ». Les solutions qui adressent l'indépendance des dimensions sont souvent lourdes à mettre en œuvre et dépendantes d'un calcul centralisé, basé généralement sur des connaissances supplémentaires, comme la répartition des concepts dans une collection de documents de référence.

Plusieurs propositions supposent la présence d'une seule ontologie de référence. Or des travaux montrent qu'il n'est pas possible d'en concevoir une seule [Sta02]. D'une part, les ontologies ne sont que des représentations des connaissances, pas les connaissances elles-même. Elles sont donc subjectives, et plusieurs personnes peuvent faire des choix différents et incompatibles pour décrire tel ou tel concept, telle ou telle relation, tel ou tel domaine de connaissances. D'autre part, les développements sont distribués et autonomes, et il est possible que des développeurs d'ontologies ne veuillent pas que leur modélisation soit ramenée à une solution standard. Par exemple, une société décrivant ses produits à l'aide d'une ontologie Ω_1 peut ne pas vouloir changer ses descriptions pour être incorporée dans l'ontologie Ω_2 d'une autre société, parce que Ω_2 s'applique mal à ses produits ou parce qu'elle ne souhaite pas le faire. C'est pourquoi il existe une *hétérogénéité sémantique* entre ontologies. La solution classique est d'utiliser des correspondances entre parties des ontologies (concepts, relations). Ces correspondances permettent de réécrire des représentations décrites dans une ontologie vers des représentations d'une autre. Par exemple, si Ω_1 a le concept *chat* mais pas le concept *félin*, au contraire de Ω_2, et qu'il existe

une correspondance entre ces deux concepts dans les deux ontologies, une requête « les chats ont-ils des moustaches ? » se réécrirait en « les félins ont-ils des moustaches ? » Mais les correspondances ne sont pas totales sur les ontologies. Il existe donc des parties qui sont non partagées après la mise en place des correspondances entre ontologies. Or nous voulons que les intentions des initiateurs de requêtes soient respectées dans notre système d'information distribuée. Il reste donc un problème d'hétérogénéité sémantique après la mise en place de correspondances entre ontologies.

Dans cette thèse, nous cherchons à améliorer l'interopérabilité sémantique et l'échange d'information entre un pair initiateur de requête p_1 et un fournisseur d'information p_2, qui utilisent différentes ontologies mais partagent certains concepts. L'idée est de permettre à chacun de décrire ses informations et ses requêtes en utilisant tous ses concepts, y compris les non partagés. Cette solution est originale car ces concepts non partagés ne sont jamais pris en compte dans les systèmes sémantiques. Nous prouvons dans notre travail que les parties non partagées des ontologies peuvent être utilisées pour répondre de manière pertinente à des requêtes. En effet, même si p_2 ne connaît pas le concept *chat* de la requête de p_1, mais qu'il sait que pour p_1 un *chat* est similaire à un *félin*, dont il dispose, alors il peut retourner un de ses documents sur les *tigres* (qui sont des *félins*), même si p_1 ne le connaît pas.

La première partie de cette thèse s'attache à présenter le contexte très général de notre travail. Nous nous concentrons sur l'utilisation de vecteurs sémantiques pour la recherche d'information et sur le problème de l'indépendance des dimensions. Les solutions classiques visant à modifier l'espace nous paraissent lourdes à mettre en place et difficilement applicables au Web. L'expansion de requête est une bonne idée, mais génère souvent du bruit et un déséquilibre des dimensions dans le vecteur de la requête étendue. Nous proposons donc EXSID, une solution utilisant une *expansion structurante* des dimensions principales de la requête ainsi qu'une *image* des documents au travers de la requête étendue. L'image d'un document permet de replier les dimensions d'une requête dans un seul vecteur, représentatif des dimensions de la requêtes dans le document. Nous avons effectué une série d'évaluations. Un choix judicieux des paramètres permet d'obtenir des résultats qui ne présentent pas de dégradation, et même plutôt une légère amélioration, par rapport aux solutions de référence : l'une mesurant la pertinence par le cosinus entre un document et une requête et l'autre par le cosinus entre un document et une expansion usuelle dans un seul vecteur.

La deuxième partie présente le cadre de l'hétérogénéité sémantique dans les systèmes d'information distribués. Il n'est en effet pas possible d'imaginer une seule ontologie de référence, et il faut passer par des correspondances, partielles, entre ontologies. Les parties non partagées, sans mise en correspondance, des différentes ontologies sont pour les solutions classiques des sortes « d'angles morts » dont elles ne tiennent pas compte. Or nous pensons qu'il est possible d'avancer un peu plus vers l'interopérabilité sémantique des systèmes d'information en utilisant les correspondances entre ontologies et les parties non partagées. EXSID s'enrichit alors d'un module d'interprétation et devient EXSI^{2}D. Cette solution permet de comprendre l'expansion structurante d'un requête dans l'ontologie du fournisseur d'information, en essayant de déduire des parties communes l'expansion que l'initiateur de la requête aurait effectuée s'il avait eu l'ontologie du fournisseur. Nous montrons que EXSI^{2}D résiste beaucoup mieux à l'hétérogénéité sémantique que les mesures de référence : cosinus et expansion automatique. En faisant varier le degré d'hétérogénéité, nous voyons que jusqu'à 70% de mappings manquant entre les ontologies de l'initiateur de la requête et du fournisseur d'information, nous avons toujours plus de 80% d'efficacité dans les recherches, alors que les autres méthodes s'écroulent (aux alentours de 40%).

La troisième partie introduit l'utilisation de EXSI^{2}D dans un environnement fortement distribué et hétérogène sémantiquement. Elle commence par présenter les systèmes P2P et l'intégration sémantique dans ces systèmes. Parmi les architectures P2P, nous nous intéressons en priorité aux non structurées, parce qu'elles laissent les pairs autonomes et qu'elles sont fortement tolérantes aux pannes. EXSI^{2}D

n'est cependant pas limité à une architecture particulière. Nous décrivons ensuite les solutions de routage de requêtes, et en particulier les requêtes top-k, dans une architecture non structurée. L'intégration sémantique dans les systèmes P2P peut utiliser une coopération des différents pairs, qui créent une abstraction ou un réseau sémantique, ou bien mettre en place des correspondances entre les ontologies des pairs. Nous montrons ensuite comment EXSI^{2}D peut s'intégrer à un réseau P2P non structuré hétérogène sémantiquement. Nous proposons des premiers algorithmes dans ce cadre.

PARTIE I

Expansion structurante et image d'un document dans un cadre homogène sémantiquement

CHAPITRE 1

Quelques éléments sur la recherche d'information

1.1 La recherche d'information, toujours aussi cruciale

Dans ce chapitre nous présentons un certain nombre de définitions relatives à la recherche d'information (RI) qui serviront dans le reste du document. Il ne s'agit pas là de présenter toutes les facettes d'un domaine en pleine effervescence. Nous rappelons tout d'abord quelques chiffres sur l'impact et les besoins, dont la plupart sont générés directement ou indirectement par l'avènement du Web et des systèmes distribués en général. Nous présentons ensuite les définitions propres à l'évaluation d'un système de RI d'une part, et les modèles de représentation et d'indexation qui sont propres à chaque système.

1.1.1 Images de la recherche d'information d'hier et d'aujourd'hui

La recherche d'information est un domaine qui a suivi le développement de l'informatique, et l'a même parfois précédé voire stimulé. Les grandes quantités de données de la « révolution administrative » du XIX^{e} siècle[1] sont devenues humainement inutilisables au fil des ans. Dès les années 1880, le dépouillement du recensement américain prenait une dizaine d'années, malgré les 1500 employés du bureau du recensement. C'est dans cet environnement qu'Herman Hollerith proposa une machine à cartes perforées pour traiter l'information du recensement, trier les cartes et faire automatiquement certaines statistiques [Aus82] (cf. figure 1.1). L'automatisation permit de diviser le temps de traitement par trois, ce qui impressionna très fortement les contemporains (cf. par exemple la page Web sur Hollerith chez IBM [www]). Hollerith a fondé une petite société spécialisée dans l'automatisation de différents processus : International Business Machines, ou IBM, à partir des machines qu'il avait mises en place et du brevet qu'il avait déposé. Cette société a connu à terme un grand développement.

22

Le problème devint de plus en plus critique en terme de temps de calcul et de pertinence, par exemple pour l'analyse des communications durant la seconde guerre mondiale [Hod88]. Actuellement, le développement des informations à traiter est aussi impressionnant. Le projet *How much info?* de Berkeley [www] a pour but de comptabiliser les informations contenues sur les différents médias et les quantités échangées à travers le monde (cf. tableau 1.1 pour quelques-unes de ces données). Il y est estimé que la quantité d'information stockée sur les disques durs en 2003 est comprise dans une fourchette de 403 à 1986 teraoctets et que la progression entre 1999 et 2003 est de 114% en ce qui concerne les nouvelles

35

[1] Expression venue d'historiens anglo-saxons principalement. Voir par exemple l'ouvrage de Lowe [Low87]. En France, le terme ne semble pas encore imposé : Gardey [Gar08] parle lui de « révolution de papier ».

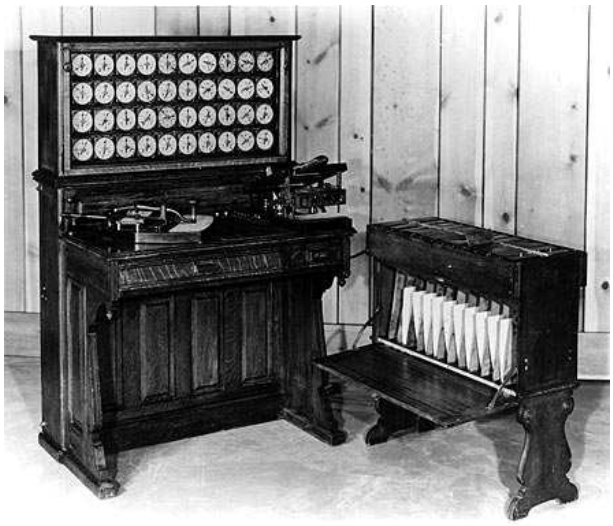


Figure 1.1 – L'appareillage inventé par Hollerith. A droite, le classeur où sont entreposées les fiches une fois traitées. A gauche, le « catalogueur », avec à gauche sur le pupitre une perforeuse de cartes et à droite un lecteur de cartes.

informations. Le développement de nouveaux systèmes (téléphones mobiles, appareils photos numériques, etc.) accélère le mouvement : la messagerie instantanée génère cinq milliards de messages par jour (750 gigaoctets), ou 274 teraoctets par an, les appels téléphoniques dans le monde mettent en jeu 17,3 exaoctets de nouvelles informations si on les stockait sous un format électronique, etc.

Medium	To (1 000 Go) en 2002
Web de surface	167
Web profond	91 850
courriels	440 606
messagerie instantanée	274

Table 1.1 – Quelques exemples de quantités d'information dans le monde en 2002

La recherche d'information dans ces conditions reste un sujet critique où toutes les avancées sont cruciales. La compagnie Google en est une preuve éclatante. Il s'agit d'une société encore jeune (fondée en 1998), sur un projet apparemment naïf et peu ambitieux : proposer un moteur de recherche simple à utiliser et rapide. Sur cette base, ce site est bientôt devenu incontournable pour les recherches sur le Web, puis une marque, un univers. Mais le cœur de métier de Google, et ce qui lui rapporte le plus d'argent, reste son moteur de recherche d'information. Dans son rapport annuel et trimestriel [WW34], la compagnie indique avoir un bénéfice pour le premier trimestre 2008 de 1,307 milliards de dollars, en hausse de 25% par rapport à 2007, et un chiffre d'affaire de plus de 16,5 milliards de dollars.

D'un autre côté, si, dans les années Quatre-vingt-dix, les gens préféraient interroger d'autres personnes plutôt que des systèmes informatisés [WW33, MRS08], désormais la situation est inversée. Ainsi Fallows [WW27] indique que « 92% des internautes trouvent que l'Internet est un bon endroit pour obtenir

l'information de tous les jours »[2]. C'est ainsi que d'un outil académique, utilisé par des bibliothécaires et des scientifiques, la recherche d'information est devenue un champ ouvert au grand public.

Le développement des recherches sur le Web illustre ce phénomène. La société ComScore [www32] publie ainsi des rapports sur le monde numérique, et en particulier sur l'utilisation des moteurs de recherche. Pour le continent européen, au mois de mars 2008, elle indique que 24,550 milliards de recherches ont été effectuées sur les moteurs de recherche. Il s'agit, toujours selon la même étude, de 221,2 millions d'internautes, et donc une moyenne de 111 recherches par internaute européen sur ce mois de mars 2008. Notons aussi que 79,2% de ces recherches ont été effectuées sur Google (cf. tableau 1.2).

Site	Recherches (en millions)	pourcentage des recherches
total des internautes	24,550	100
Google	19,434	79,2
Ebay	752	3,1
Yandex	528	2,2
Yahoo!	486	2,0
Microsoft	469	1,9

Table 1.2 – Classement des recherches européennes (domicile et travail, internautes de quinze ans et plus) en mars 2008 par la société ComScore [www32].

La recherche d'information est donc un champ ancien, avec des problématiques anciennes et nouvelles. Les technologies développées depuis longtemps voient depuis les années Quatre-vingt-dix les quantités d'information à traiter exploser, et le type de données se multiplier. L'apparition de nouveaux paradigmes (systèmes distribués, systèmes P2P, réseaux sociaux, etc.) génère de nouvelles problématiques pour lesquelles les modèles anciens sont en train d'évoluer. De nombreuses solutions sont proposées à la fois en traitement du langage naturel, système d'information, Web sémantique, bases de données, etc. Elles sont souvent combinées pour s'adapter aux défis actuels.

1.1.2 Une définition de la recherche d'information

La RI est assez difficile à définir. La principale difficulté vient du fait qu'elle se trouve au point de rencontre de plusieurs disciplines, et qu'on ne sait pas forcément clairement ce qu'on recherche et pourquoi on le recherche. Cette « information » est en effet assez imprécise : bases de données ? Textes ? Flux ? De même, la réponse doit-elle être structurée ? En texte brut ? Raffinée ? Etc. Les réponses à cette question sont assez diverses.

Comme base de notre réflexion sur la recherche d'information, revenons au célèbre livre de Van Rijsbergen [Van79]. Malgré son ancienneté, ce dernier reste en grande partie d'actualité, pour les questions qu'il pose et les bases qu'il propose à la recherche d'information. La citation suivante en est extraite :

> Information retrieval is a wide, often loosely-defined term [...]. Unfortunately the word information can be very misleading. In the context of information retrieval (IR), information, in the technical meaning given in Shannon's theory of communication [SW49], is not readily measured. In fact, in many cases one can adequately describe the kind of retrieval by simply substituting "document" for "information". Nevertheless, "information retrieval" has become accepted as a description of the kind of work published by Cleverdon, Salton, Sparck Jones,

[2] « 92% of Internet users say the Internet is a good place to go for getting everyday information. »

> Lancaster and others. A perfectly straightforward definition along these lines is given by Lancaster [Lan68]: "Information retrieval is the term conventionally, though somewhat inaccurately, applied to the type of activity discussed in this volume. An information retrieval system does not inform (i.e. change the knowledge of) the user on the subject of his inquiry. It merely informs on the existence (or non-existence) and whereabouts of documents relating to his request." This specifically excludes Question-Answering systems [and also] data retrieval systems such as used by, say, the stock exchange for on-line quotations. »

La citation de Van Rijsbergen [Van79] montre son hésitation à définir précisément un domaine dont il est pourtant un des fers de lance. Il finit par citer des travaux, plutôt que de définir de manière formelle la RI, comme si cette dernière était un champ d'investigation dont les limites sont mouvantes plutôt qu'une discipline figée. En étudiant par exemple l'appel à communication de SIGIR'08 [www], qui est la grande conférence mondiale sur la RI, nous nous rendons bien compte de la largeur du champ traité par la RI. Les différentes techniques, les différentes étapes du processus de RI, les différents modèles utilisés, les différents contenus recherchés, etc. donnent l'image d'un domaine riche. La principale définition que Van Rijsbergen propose est finalement une définition négative, face à la recherche de données. C'est ce que Blair [Bla84, Bla06] reprend d'ailleurs, en soulignant la présence de deux paradigmes différents. Selon nous, comme nous l'avons déjà souligné, cette distinction n'est plus vraiment d'actualité : les domaines de la recherche d'information, gestion de données, ingénierie des connaissances, etc. identifient des problèmes proches ; ces différents point de vue collaborent et s'enrichissent mutuellement. Ces avancées sont illustrées par Boughanem et Savoy [BS08] qui proposent un état des lieux et des perspectives de la RI.

Nous adoptons finalement la définition suivante de Giunchiglia *et al.* [GKZ08] : « le but d'un système de recherche d'information est de faire correspondre une requête en langage naturel q, qui spécifie des besoins en information d'un utilisateur, à un ensemble de documents dans une collection de documents $\mathcal{D}$, qui répondent à ces besoins, et (de manière facultative) de classer ces documents selon leur degré de pertinence[3] ».

Les trois paramètres qui doivent être instanciés par tout système de recherche d'information sont :

- à quoi correspond un *élément atomique* dans les documents et les requêtes ?
- quelle définition de *similarité* entre documents et requêtes est utilisée ?
- quel *modèle* est adopté pour les requêtes et les documents ?

Ainsi, un système de recherche d'information est un triplet :

$$S =< \text{modèle, élément, similarité} >$$

et chacun des systèmes doit faire des choix pour chacun de ces paramètres.

La figure 1.2 illustre le processus de recherche d'information. Documents et requêtes sont transformées en représentations adéquates, c'est-à-dire utilisables en machine. Cette phase de représentation s'appuie sur un modèle de représentation, qui peut être de plusieurs formes : algébrique, probabiliste, ensembliste, etc. Les cartes perforées utilisées par le bureau du recensement américain à la fin du XIX^{e} siècle puis par IBM étaient un modèle particulier et très performant pour les machines chargées de traiter l'information à l'époque. Chaque carte représentait un document, ou une source de données, et était passée dans la machine qui la traitait. La phase de représentation des documents et requêtes est une étape très importante, qui implique certaines contraintes et conditionne une partie des étapes suivantes et des

[3] « The goal of an information retrieval system is to map a natural language queries Q, which specify user information needs, to a set of documents in the document collection D, which meet these needs, and (optionally) to order these documents according to the degree of relevance ».

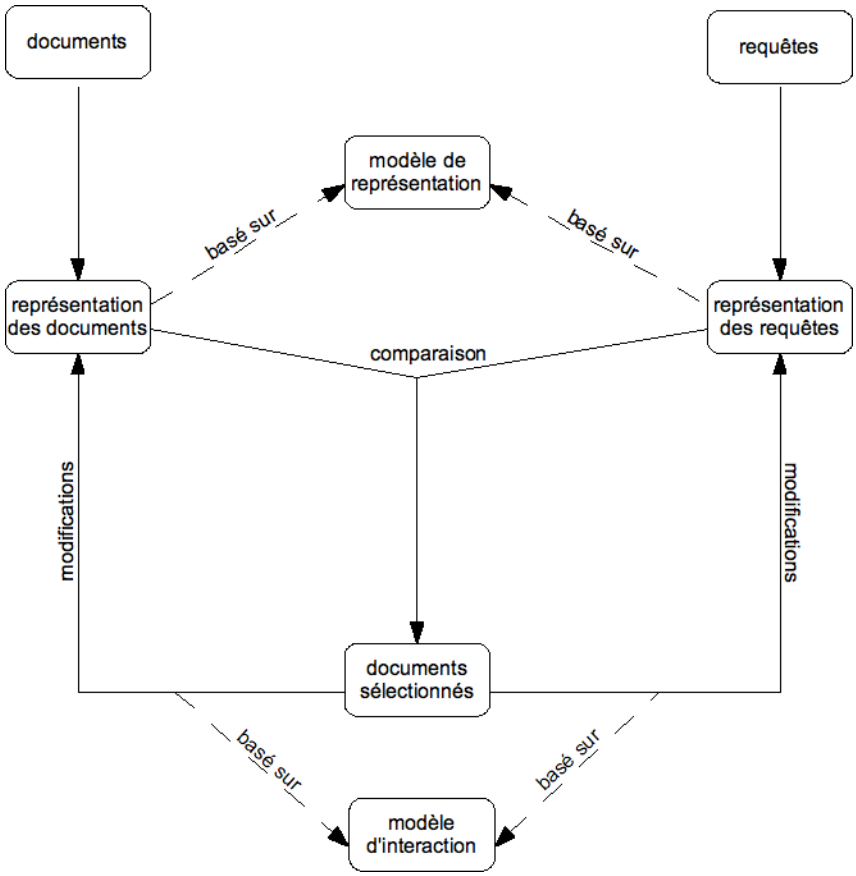


Figure 1.2 – Processus de recherche d'information.

résultats. Les cartes perforées sont par exemple difficilement interrogeables, en tout cas dans leur version initiale ; il n'est ainsi pas possible de demander une carte précise.

Les représentations doivent être compatibles pour effectuer la comparaison entre les documents et une requête. La comparaison utilise des mesures qui permettent d'estimer la pertinence des documents par rapport aux requêtes. Les documents trouvés peuvent être ordonnés, ou non.

Le modèle d'interaction permet de représenter la façon dont les fournisseurs d'information et les utilisateurs peuvent interagir avec le système de recherche d'information, une fois donnés les résultats. Il a ainsi pu être possible pour les agents du bureau du recensement américain de modifier les questions posées lors du recensement pour changer les cartes perforées et obtenir de meilleurs réponses à leurs interrogations, comme il est possible pour un utilisateur de reformuler sa requête. Dans tous les cas, il faut un modèle d'interaction pour savoir ce qui est permis dans cette réécriture.

1.2 Mesurer la qualité des réponses

Pour évaluer un système de RI, il est nécessaire de pouvoir mesurer la pertinence des réponses fournies par celui-ci par rapport aux besoins exprimés par l'utilisateur dans sa requête. Nous avons donc besoin d'une définition de mesures de pertinence d'une part ; et d'autre part de requêtes et de documents

de référence que nous pouvons assimiler à un ensemble de scénarios : un *corpus de test*. Pour un même corpus et une même mesure de pertinence, il est possible de comparer plusieurs systèmes entre eux.

1.2.1 Efficience ou efficacité ?

Jones [Jon81] décrit synthétiquement les deux défis de la recherche d'information :

> « L'efficacité est à quel point le système fait ce qu'il est supposé faire ; ses bénéfices sont les gains par rapport à ce que le système fait ; son efficience est à quel point il fait ce qu'il fait de manière économe[4] ».

Les deux termes mis en lumière dans cette citation sont *efficacité* et *efficience*, deux faces bien connues de la notion de performance :

- l'efficience est l'articulation entre moyens et résultats : « est-ce que les résultats sont suffisants compte tenu des moyens mis en œuvre ? ».
- l'efficacité est l'articulation entre résultats et objectifs : « est-on arrivé à ce que l'on avait l'intention de faire, à quel point l'objectif fixé est-il atteint ? »

En ce qui concerne la recherche d'information, ces deux notions renvoient à la pertinence des résultats (efficacité) et à la vitesse d'obtention de ces résultats (efficience). Dans ce chapitre, nous ne nous intéressons qu'à l'efficacité. En faisant des choix parmi les nombreuses alternatives possibles, concernant les modèles de représentation, type d'indexation, etc. de nombreux systèmes peuvent être définis. Bien évidemment, ils ne se valent pas tous, et il va nous falloir les juger puis les comparer.

1.2.2 Que mesurer ?

Il y a de très nombreuses façons de mesurer la qualité d'un système de recherche d'information. Choisir une mesure plutôt qu'une autre peut être critique, ou même critiqué. Il est en effet possible de diminuer la portée d'un système en lui appliquant une mesure pour lequel il n'a pas été conçu. Inversement, avec une « bonne » mesure, il est aussi possible de faire passer tout système pour un « bon » système. Dans tous les cas, nous commençons par introduire le formalisme nécessaire à toutes les mesures que nousprésentons par la suite.

Soit $\mathcal{D}$ un ensemble de documents, et q une requête posée sur cet ensemble. Pour cette requête, un sous-ensemble des documents peut être dit *pertinent* pour l'utilisateur (u), et un autre *non pertinent*. Nous notons ces deux ensembles $\mathcal{P}_q^u \subseteq \mathcal{D}$ et $\overline{\mathcal{P}}_q^u = \mathcal{D} \setminus \mathcal{P}_q^u$ respectivement.

D'un autre côté, tout système S_i *sélectionne* des documents appartenant à $\mathcal{D}$ comme réponses adéquates pour la requête q. Nous notons les deux ensembles ainsi définis $\mathcal{P}_q^{S_i} \subseteq \mathcal{D}$ et $\overline{\mathcal{P}}_q^{S_i} = \mathcal{D} \setminus \mathcal{P}_q^{S_i}$. La figure 1.3 présente les ensembles $\mathcal{P}_q^u$ et $\mathcal{P}_q^{S_i}$ définis précédemment.

1.2.2.1 Précision

En utilisant les termes et les formalismes précédents, la précision, pour un système S_i, est définie par le rapport du nombre de documents pertinents pour l'utilisateur sélectionnés par le système sur le nombre de documents sélectionnés par le système :

$$P_q = \frac{||(\mathcal{P}_q^{S_i} \cap \mathcal{P}_q^u)||}{||\mathcal{P}_q^{S_i}||} \in [0, 1]$$

[4] « Effectiveness is how well the system does what it is supposed to do; its benefits are the gains deriving from what the system does ; its efficiency is how cheaply it does what it does ».

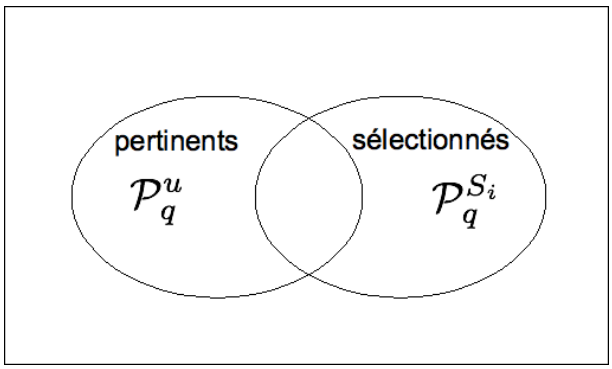


Figure 1.3 – Parmi l'ensemble des documents, ceux qui sont sélectionnés par le système S_i ($\mathcal{P}_q^{S_i}$) , et ceux qui sont pertinents selon l'utilisateur ($\mathcal{P}_q^u$).

Ce qui peut s'exprimer ainsi :

$$P_q = \frac{\text{nombre de documents pertinents sélectionnés}}{\text{nombre de documents sélectionnés}}$$

ou se représenter graphiquement comme sur la figure 1.4 :

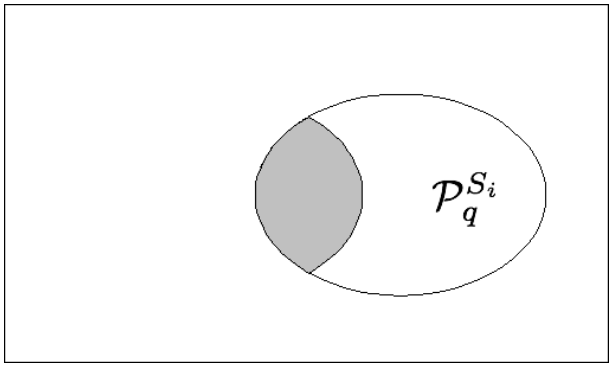


Figure 1.4 – La précision mesure le rapport du nombre de documents pertinents pour l'utilisateur sélectionnés par S_i sur le nombre de documents sélectionnés par S_i.

1.2.2.2 Rappel

Le rappel, pour un système S_i, est défini par le rapport du nombre de documents pertinents pour l'utilisateur sélectionnés par le système, divisé par le nombre de documents pertinents pour l'utilisateur :

$$R_q = \frac{||(\mathcal{P}_q^{S_i} \cap \mathcal{P}_q^u)||}{||\mathcal{P}_q^u||} \in [0,1]$$

Ce qui peut s'exprimer ainsi :

$$R_q = \frac{\text{nombre de documents pertinents sélectionnés}}{\text{nombre de documents pertinents pour l'utilisateur}}$$

ou se représenter graphiquement comme sur la figure 1.5 :

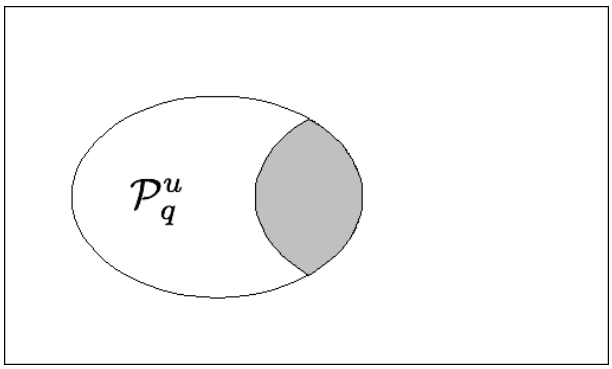


Figure 1.5 – Le rappel mesure le rapport du nombre de documents pertinents pour l'utilisateur sélectionnés par S_i sur le nombre de documents pertinents pour l'utilisateur.

1.2.2.3 Autres mesures

Il existe d'autres mesures simples que nous ne présentons pas ici car nous ne les utilisons pas pour caractériser les systèmes :

- l'erreur, qui correspond au rapport du nombre de documents pertinents non sélectionnés et sélectionnés non pertinents sur la totalité des documents ;
- le taux de chute, qui correspond au rapport du nombre des documents sélectionnés non pertinents sur le nombre de documents non pertinents ;
- le silence qui vaut $1 - R_q$;
- le bruit, qui vaut $1 - P_q$;
- etc.

1.2.2.4 La F-mesure, une mesure de synthèse

La F-mesure permet de donner en une seule mesure un rapport entre la précision et le rappel, cf. Van Rijsbergen [Van79]. La définition générale est la suivante, pour tout réel β non nul :

$$F_\beta = (1 + \beta^2) \times \frac{P \times R}{P \times \beta^2 + R}$$

Ainsi nous avons :

$$F_1 = (1 + 1) \times \frac{P \times R}{P \times 1 + R} = 2 \times \frac{P \times R}{P + R}$$

Il s'agit alors de la moyenne harmonique entre la précision et le rappel. La F_1-mesure permet d'indiquer si un système penche plutôt du côté de la précision (ne sélectionne que les bons documents) ou du côté du rappel (sélectionne de nombreux documents pertinents). Selon que β est inférieur ou supérieur à 1, deux cas de présentent. Dans le premier cas, par exemple une $F_{0,5}$-mesure, nous avons une priorité donnée au rappel dans l'évaluation du système. Dans le second, F_2-mesure, c'est la valeur de précision qui est mise en avant.

La F-mesure la plus connue est la F_1-mesure. Elle est d'ailleurs souvent citée comme F-mesure.

1.2.2.5 Valeurs à n et caractérisation graphique

Il est aussi possible de classer les résultats. Cela dépend du modèle de représentation et de la mesure de pertinence utilisée. Si les documents ne sont pas seulement sélectionnés, mais aussi classés, nous n'avons pas alors deux ensembles, mais une liste ordonnée. C'est-à-dire que chaque document a dans ce cas une valeur de pertinence, par exemple située dans l'intervalle $[0, 1]$, avec 0 pour les documents les moins intéressants selon le système S_i, et 1 pour les documents les plus intéressants selon le système S_i.

Dans ce cas précision et rappel peuvent être mesurés selon un certain entier. Il s'agit de recréer les intuitions décrites précédemment. En effet, en recherchant la précision ou le rappel sur les n meilleurs documents retrouvés, on fait comme si le système ne sélectionnait que ces n meilleurs, même s'il a classé tous les documents. Nous parlons alors de précision à n, ou d'un rappel à n, notés $P@n$ et $R@n$, pour $n \in \mathbb{N}$.

Précision et rappel sont les deux mesures les plus connues et les plus utilisées en recherche d'information. Il est à noter qu'elles ont un comportement inverse : quand la valeur de l'une augmente, l'autre diminue et inversement. L'idéal est évidemment quand les deux valeurs sont égales à 1 (valeur maximale). Ce qui est le cas quand le système est totalement cohérent avec les résultats d'un utilisateur humain, et que $\mathcal{P}_q^{S_i} = \mathcal{P}_q^u$. En pratique, aucun système ne satisfait cette condition, et la plupart des systèmes ont un comportement proche de celui de la figure 1.6 : chaque point correspond à un couple ($P@1$, $R@1$), ($P@2$, $R@2$), …, ($P@n$, $R@n$), $n \in \mathbb{N}$.

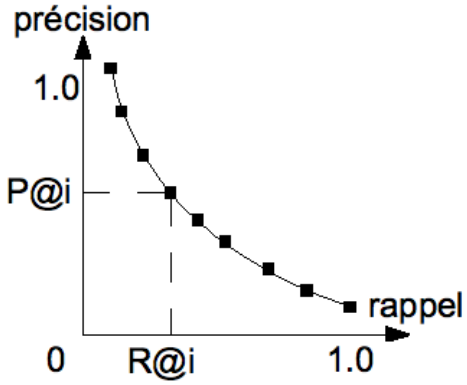


Figure 1.6 – Courbe précision-rappel, caractéristique d'un système de recherche d'information.

Ce type de courbe est caractéristique d'un système et permet de l'identifier. En effet, un système défini pour sélectionner un maximum de résultats, donc d'avoir un très bon rappel, a des valeurs de précision faible. En effet, pour avoir un bon rappel, il est nécessaire d'avoir beaucoup de documents pertinents dans les documents sélectionnés, et peu importe qu'il y ait beaucoup de faux positifs, c'est-à-dire de documents non pertinents sélectionnés. De la même façon, un système qui veut ne récupérer que les meilleurs résultats peut générer un rappel faible. En effet, un système précis cherche à discriminer fortement les documents et les documents les moins discriminés sont refusés. Ce qui fait que des documents pertinents peuvent être éliminés, non sélectionnés. Ces deux types de systèmes sont représentés dans la figure 1.7.

Nous disposons maintenant d'une mesure pour évaluer l'efficacité des systèmes de recherche d'information.

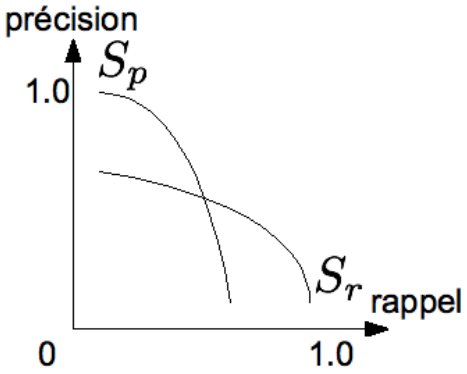


Figure 1.7 – Deux systèmes caractéristiques : S_p a une bonne précision mais un mauvais rappel, S_r a un bon rappel, mais une mauvaise précision.

1.2.3 Corpus de test

À présent, nous devons définir le contexte d'évaluation, c'est-à-dire les éléments qui vont servir à tester le processus de sélection des documents pour chaque système. En recherche d'information, il s'agit d'un *corpus de test*, constitué une collection de documents, aussi appelée *corpus de documents*, d'une collection de requêtes, ou *corpus de requêtes*, et des jugements de pertinence des documents par rapport à ces requêtes.

Un corpus de documents est un ensemble de documents sur lesquels poser des requêtes. Les requêtes du corpus de requête simulent l'activité de l'utilisateur. Les jugements de pertinence indiquent pour chaque document du corpus s'il est pertinent, et parfois même à quel degré il l'est, pour chaque requête. Trois éléments sont donc en jeu : collection de documents, requêtes et jugements de pertinence. Ils vont être développés dans les sous-sections suivantes.

1.2.3.1 Corpus de documents

Il existe de très nombreux ensembles de documents en accès libre, par exemple sur le Web. Les spécialistes de recherche d'information peuvent y choisir ce qui leur convient le mieux pour leurs différentes expériences, suivant ce qu'ils veulent obtenir : documents plus ou moins vulgarisés, plus ou moins spécialisés dans un domaine, dans une langue ou une autre, etc.

Dans tous les cas, il s'agit de choisir les textes de manière adéquate. En effet, si le Trésor de la Langue Française Informatisé [WWW] (TLFI) définit ainsi un corpus de documents : « recueil réunissant ou se proposant de réunir, en vue de leur étude scientifique, la totalité des documents disponibles d'un genre donné, par exemple épigraphiques, littéraires, etc. », il ne faut pas perdre de vue que c'est « l'opération de choix raisonné parmi les composants disponibles qui crée un corpus » [Hab00], surtout pour la recherche d'information. C'est-à-dire qu'un corpus de documents est un ensemble de documents choisis selon certaines caractéristiques. Le travail concernant la sélection des documents des corpus est d'ailleurs très important et fait encore l'objet de nombreuses recherches [Goe08].

1.2.3.2 Corpus de requêtes

De la même façon qu'il existe déjà des documents utilisables pour l'évaluation mais qu'il est nécessaire de faire un choix important et parfois délicat parmi tous ces documents, il existe de très nombreuses requêtes qui ne sont pas toujours intéressantes pour les tâches d'évaluation. Ainsi les requêtes que posent les utilisateurs sur les moteurs de recherche ou sur des systèmes de recherche en général pourraient être une bonne base pour la construction de l'ensemble de requêtes. Mais ces requêtes sont partielles, parfois mal posées, etc. Pour exploiter au mieux les caractéristiques d'une collection de documents, il est important de créer des requêtes adéquates, qui sont souvent artificielles. Elles doivent, par leur longueur, les thèmes abordés, leur forme, etc. investiguer au mieux les propriétés du corpus (nombre de documents touchés, regroupements thématiques, etc.).

1.2.3.3 Jugements de pertinence

C'est un autre problème critique. L'évaluation qui doit être menée concerne l'efficacité du système par rapport au jugement humain : l'idée régulatrice de tout système étant de répondre correctement aux demandes des utilisateurs, et de sélectionner exactement les documents qui l'intéressent. Il est donc évident qu'il faut une intervention de l'utilisateur pour indiquer les documents pertinents dans un corpus pour une requête donnée. C'est justement ce qui fait débat. Ainsi Voorhees [Voo05] souligne :

> One objection to test collections that dates back to the Cranfield tests is the use of relevance judgments as the basis for evaluation. Relevance is known to be very idiosyncratic, and critics question how an evaluation methodology can be based on such an unstable foundation. An experiment using the TREC-4 and TREC-6 retrieval results investigated the effect of changing relevance assessors on system comparisons. The experiment demonstrated that the absolute scores for evaluation measures did change when different relevance assessors were used, but the relative scores between runs did not change. That is, if system A evaluated as better than system B using one set of judgments, then system A almost always evaluated as better than system B using a second set of judgments (the exception was in the case where the two runs evaluated as so similar to one another that they should be deemed equivalent). The stable comparisons result held for different evaluation measures and for different kinds of assessors and was independent of whether a judgment was based on a single assessor's opinion or was the consensus opinion of a majority of assessors.

En fait, les attaques contre les corpus de test, par exemple Cranfield ou TREC, se concentrent sur la question des jugements de pertinence. Ainsi, la présentation inaugurale de la conférence ECIR'08 (*European Conference on Information Retrieval*) par Nick Belkin, concernant les défis de la recherche d'information, pointait les limites des systèmes d'évaluation, et en particulier la tâche de collecte des jugements de pertinence. De nombreux bloggueurs [28, 29] du domaine ont alors relayé la position d'Ellen Voorhees qui défend le modèle d'évaluation de Cranfield.

Les jugements sont faits par des experts des domaines concernés par les corpus de test. Ils travaillent à plusieurs et de manière coordonnée. C'est donc évidemment un processus onéreux. Mais comme la pertinence est une notion relative et subjective, et pour « diminuer au maximum » cette subjectivité, il est important que ce soient des experts qui désignent les documents pertinents pour chaque requête.

1.2.3.4 Corpus existants

La création de corpus fait l'objet de nombreux travaux depuis les années Soixante-dix. Les corpus diffèrent par le nombre de documents et le nombre de requêtes. Ainsi il est possible d'établir des tableaux

de comparaison tels que le tableau 1.3. Mais les corpus diffèrent aussi sur leur domaine de spécialité, la façon de juger la pertinence, etc.

Nom du corpus	nb. de documents	nb. de requêtes	taille (en Mo)
ADI	82	35	0,04
Time	425	83	1,5
Medline	1033	30	1,1
Cranfield	1400	225	1,6
CISI	1460	112	2,2
CACM	3204	64	2,2
LISA	5872	35	3,4
NPL	11429	93	3,1

Table 1.3 – Quelques corpus de test avec leurs caractéristiques.

Il existe d'autres corpus, tels que Reuters [LYL04, www.23] et TREC [Voo07, www.24], qui sont beaucoup plus importants (plusieurs gigaoctets). Mais, outre qu'ils sont souvent payants (c'est le cas pour le corpus de documents de la plupart des pistes de TREC), ils nécessitent un temps de traitement que nous n'avons pas souhaité prendre, car l'objectif final n'est pas pour nous d'améliorer une technique de RI. D'autre part, nous souhaitions considérer un nombre de requêtes important et de ce point de vue, ils ne sont pas forcément plus intéressants que les corpus du tableau 1.3. En effet notre solution repose avant tout sur des modifications des requêtes (et des documents, mais au travers des requêtes) et disposer d'un nombre important de celles-ci permet de tester d'autant plus de cas différents. Nous avons finalement choisi d'utiliser le corpus Cranfield pour nos premiers tests d'évaluation car il propose un nombre conséquent de requêtes. Nous travaillons actuellement sur un corpus plus gros issu de TREC (le corpus W3C de la piste « entreprise »).

1.3 Représentation des documents et des requêtes

Dans cette section, nous allons évoquer les deux problèmes centraux de la recherche d'information, l'*indexation* et la *représentation* de l'information. Dans plusieurs cas, le modèle de représentation dirige le type d'indexation. Dans d'autres, l'indexation est libre. Nous préférons donc présenter d'abord l'indexation en recherche d'information, puis certains modèles de représentation.

1.3.1 Indexation

L'indexation est un processus ancien, qui prend ses racines avec la création de l'imprimerie (au moins) et qui sert dès l'origine pour les livres et pour les bibliothèques. Ainsi beaucoup d'ouvrages possèdent des index qui relient certains mots-clés aux numéros de pages où ils apparaissent. Et les ouvrages des bibliothèques sont généralement indexés par un certain nombre de termes-clés les décrivant. Dans un contexte plus automatisé, nous adoptons une définition adaptée de [BS08] :

Definition 1 (Indexation)**.**
L'indexation consiste à produire un descripteur du document ou de la requête qui est une liste de termes significatifs auxquels sont associés des poids pour différencier leurs degrés de représentativité.

1.3.1.1 Réflexions sur l'indexation

L'indexation joue un rôle primordial en recherche d'information. Il existe plusieurs types d'indexation, qui varient sur plusieurs critères. Dans tous les cas, il s'agit de transformer un document en donnée informatisée.

Bien sûr, la première question est de savoir si ce processus est nécessaire, car la plupart des textes sont déjà dans un format informatisé. Pourquoi ne pas utiliser le document informatisé directement et utiliser les termes qui le composent comme « représentation » de ce document. Cette idée se heurte à quatre écueils. Le premier est l'ensemble des documents non représentables directement car non informatisés. Beaucoup de textes sont ainsi encore utilisés sous forme papier. Il existe bien entendu des méthodes plus ou moins efficaces pour faire passer ces documents au format informatisé. Les techniques d'OCR (*reconnaissance optique de caractères*, pour *optical character recognition*), permettent ainsi de passer de documents papier à des documents informatisés, de manière plus ou moins précise.

Le deuxième écueil est que tous les documents ne sont pas forcément des textes : une photographie, un fichier musical, une séquence vidéo sont aussi des documents qui peuvent être indexés. Ces documents, qu'ils soient sous un support numérique ou non, ne sont évidemment pas directement représentables textuellement.

Le troisième écueil est que le texte complet peut ne pas correspondre parfaitement au « sens » d'un document. En effet, la langue naturelle est suffisamment ambigüe et complexe pour que le sens d'un document ne soit pas correctement exprimé par les termes qui le composent. La polysémie (un terme a plusieurs sens) et la synonymie (un sens se retrouve dans plusieurs termes) permettent la richesse des langues naturelles, mais génèrent beaucoup d'ambigüités.

Enfin, le quatrième écueil est que les requêtes ne sont pas facilement exprimées sur les termes même des documents. Ce problème, appelé *problème du vocabulaire* [FLGD83, FLGD87, Tur94, Tur95], est celui qui fait que l'utilisateur d'un moteur de recherche par exemple doit bien réfléchir aux termes qu'il utilise : pour tout concept, quel terme l'indexeur d'un site Web qui nous intéresse va utiliser ? Il est ainsi prouvé que pour décrire le même objet, il n'y a que 20% de chances que deux personnes utilisent le même terme.

1.3.1.2 Indexation manuelle ou automatique ?

L'indexation la plus ancienne est bien évidemment l'indexation manuelle. Un ou plusieurs indexeurs humains peuvent caractériser les ouvrages dans les bibliothèques par exemple. L'indexation automatique, elle est faite par un système qui automatise le processus. Il existe quelques systèmes plus ou moins hybrides, qui utilisent des indexeurs humains après une première phase de pré-indexation automatique. Ces systèmes permettent de diminuer certains des problèmes de l'indexation automatique ou manuelle, mais n'ont pas de caractéristique propre selon nous. C'est pourquoi dans la suite de cette sous-section, nous étudions les différences entre l'indexation manuelle et l'indexation automatique.

Le principal défaut de l'indexation manuelle est le coût. En effet, ce sont des spécialistes d'un domaine qui effectuent ces indexations, parce qu'il faut qu'ils connaissent bien le domaine et les termes-clés utilisés. C'est la même contrainte, la nécessité d'avoir une bonne indexation, qui amène parfois plusieurs indexeurs à travailler sur le même document. Dans ces conditions, la solution automatique paraît plus intéressante, un système d'indexation automatique étant sans aucun doute moins onéreux.

En ce qui concerne la qualité des systèmes, les travaux et études sont pour l'instant encore assez partagés. Savoy [Sav05] montre ainsi que les résultats d'indexations manuelles ou automatiques sont assez similaires, et que les facteurs explicatifs et les stratégies liées à l'indexation manuelle sont mal

compris. Par exemple, comme nous l'avions déjà indiqué, l'indexation manuelle est soumise à une grande variabilité entre les indexeurs (problème du vocabulaire).

Un autre problème de l'indexation manuelle est l'évolution des connaissances du domaine. Ainsi, quelle que soit la technique utilisée pour une indexation, toute modification sur les connaissances du domaine implique une ré-indexation, qui peut être plus ou moins complexe. Isaac *et al.* [IMvdM+08] montrent que c'est un processus qui n'est pas évident. Il est cependant plus simple de l'automatiser.

Le dernier problème est celui de la taille du corpus à indexer. Les acquisitions d'une bibliothèque sont importantes, mais ce n'est rien comparé au Web. S'il est envisageable, bien que très coûteux, pour une institution, une entreprise ou une bibliothèque de gérer une indexation manuelle, cela devient très difficile pour de grands corpus comme le Web.

1.3.1.3 Vocabulaire libre, contrôlé, ou sémantique ?

L'approche la plus naïve d'indexation informatisée est de prendre tous les termes d'un document, de les mettre dans un fichier, et de les utiliser comme représentation de cette information. A part le problème de taille de ces indexations, il est vite apparu qu'un certain nombre de mots sont inutiles. Ces mots vides de sens (*stop words*) sont des mots tellement communs qu'il ne sert à rien de les utiliser : « le », « la », « du », « ça », etc. Ces mots vides sont spécifiques à une langue et/ou une collection de documents. Il existe des listes de ces mots vides (*stop lists*) pour différentes langues ; voir par exemple le projet SnowBall [25] qui contient des listes de mots vides pour plusieurs langues.

Souvent les termes sont ramenés à une racine commune [SM83]. Ainsi des termes comme « précieuse », « précieuses » ou « préciosité », sont transformés en une racine simple : « précieux ». Des algorithmes tels ceux de Porter [Por80] pour l'anglais ou de Namer [Nam00] pour le français permettent d'effectuer ces traitements, grâce à un ensemble de règles spécifiques à chaque langue. Ce processus est appelé *racinisation* ou *lemmatisation*.

L'utilisation de *vocabulaires contrôlés*, c'est-à-dire que seul un sous-ensemble des termes du vocabulaire des documents sont utilisés pour les indexer et pour soumettre les requêtes permet de limiter l'ambigüité du langage naturel. En effet, il faut souvent laisser une certaine expressivité à ces langages, sous peine de les rendre inutilisables. L'équilibre entre limitation du problème du vocabulaire et richesse de la représentation possible est difficile à trouver. Ce genre de technique est utilisé dans des domaines spécialisés : bibliothèques, entreprises, etc.

Une autre solution est d'utiliser des ontologies pour indexer ou annoter les documents et les requêtes. Les ontologies servent à représenter des connaissances dans les systèmes informatiques [Für04]. Il s'agit de rendre explicites les conceptualisations partagées et tacitement admises dans différents domaines [Gru95]. L'ampleur de ce qui est formalisé n'est pas toujours identique : il peut s'agir de connaissances lexicographiques, conceptuelles ou même sémantiques, c'est-à-dire que nous pouvons être à un niveau non formel ou au niveau d'ontologies légères ou lourdes. Fürst [Für04] présente différents niveaux d'engagement sémantique et de formalisme (cf. figure 1.8). Par engagement sémantique il est entendu le degré d'intégration des connaissances du domaine dans l'ontologie. Nous considérons une ontologie comme une hiérarchie de concepts, une hiérarchie de relations, des signatures de ces relations sur les concepts et des axiomes sur les relations. Ces ressources sont devenues de plus en plus étudiées en RI, de par l'importance que la RI a dans le cadre du Web sémantique, pour de nombreuses applications [AG08]. Nous parlons d'indexation conceptuelle lorsque sont utilisées des hiérarchies de concepts, et d'indexation sémantique lorsque les ressources utilisées sont plus complètes [MM00a]. La première utilise comme termes de l'indexation les concepts de l'ontologie ; c'est donc exactement le même processus que pour le cas où les termes sont des mots du langage naturel. La seconde utilise des descriptions du

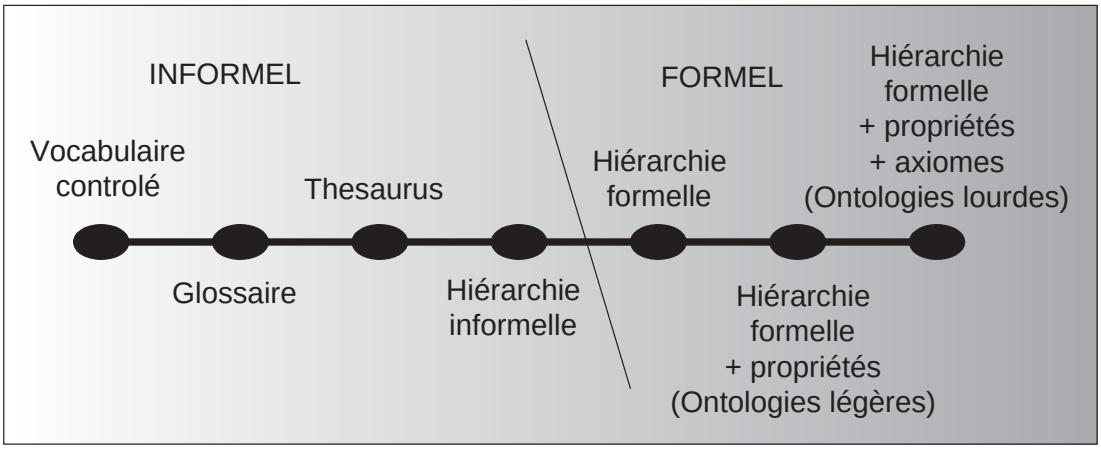


Figure 1.8 – Les différents niveaux de formalisme et d'engagement sémantique en représentation des connaissances.

contenu des documents sous forme de métadonnées. Cette distinction n'est pas essentielle selon nous.

L'utilisation de ressources sémantiques en RI est prometteur, mais discuté [Voo94, San00, SOT03]. Gonzalo *et al.* [GVCC98] prouvent ainsi qu'une relativement pauvre désambiguïsation de mots (jusqu'à 60% d'erreurs dans l'indexation) conduit à de meilleurs résultats dans la recherche d'information que l'utilisation de mots-clés, et une amélioration globale de 29% dans tous les cas. Baziz *et al.* [BBA05] proposent aussi une représentation originale qui permet d'améliorer les résultats de RI. Nous présentons plus loin cette approche.

1.3.1.4 Pondération des termes

Il existe plusieurs approches pour pondérer les termes significatifs d'un document ou d'une requête. Nombre d'entre elles se basent sur les facteurs tf et idf qui permettent de considérer les pondérations locales et globales d'un terme. On distingue la fréquence d'apparition d'un terme dans un document (*term frequency*, tf) et la fréquence d'apparition de ce même terme dans toute la collection considérée (*inverse document frequency*, idf). La mesure $tf * idf$ permet d'approximer la représentativité d'un terme dans un document, surtout dans les corpus de documents de tailles homogènes [MRS08, SM83].

1.3.2 Modèles de recherche d'information

Un modèle de recherche d'information doit se positionner sur plusieurs points :

- représentation des documents : sous quelle forme un document est-il représenté ? quel modèle d'indexation est possible ? quel modèle mathématique est utilisé ?
- représentation des requêtes : ces dernières n'ont pas forcément la même forme que les documents ; un modèle de recherche d'information peut proposer une représentation particulière pour les requêtes ;
- fonction de comparaison de documents par rapport à une requête : quelle fonction donne une note de pertinence d'un document par rapport à une requête ?

Nous présentons rapidement certains modèles et nous détaillons le modèle vectoriel qui est utilisé dans la suite du document.

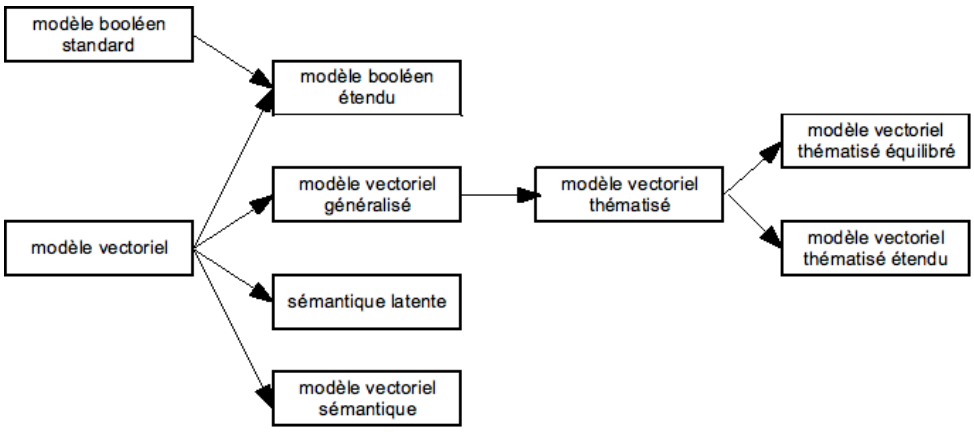


Figure 1.9 – Le modèle vectoriel de recherche d'information et quelques variantes.

1.3.2.1 Modèles booléen et booléen étendu

Le *modèle booléen* représente les documents comme des ensembles des termes et les requêtes comme des expressions booléennes (composées d'opérateurs booléens) sur ces termes. Il est aussi possible d'utiliser des concepts plutôt que des termes dans ce modèle. C'est le modèle le plus utilisé dans les moteurs de recherche sur le Web. Les opérateurs booléens les plus courants sont la négation, la disjonction et la conjonction. Parfois sont aussi utilisés des opérateurs un peu plus riches, comme des opérateurs d'ordre : *before*, *after*, et un autre de proximité. La fonction de pertinence renvoie, pour un document et une requête donnés, la valeur vrai ou faux selon que le document est jugé pertinent ou non. L'évaluation est faite selon les fonctions d'interprétation usuelles des opérateurs booléens.

Parmi les avantages de ce modèle, nous pouvons citer sa grande facilité de mise en œuvre, l'efficacité du calcul, ainsi que l'expressivité et la clarté des requêtes, basées sur une logique booléenne. C'est pourquoi les moteurs de recherche l'ont choisi de manière évidente : il permet une recherche rapide, et il a une syntaxe apparemment simple.

En apparence seulement, car la syntaxe et la sémantique des requêtes est finalement assez compliquée et pas toujours très intuitive. La plupart des requêtes sont ainsi interprétées comme booléennes par les interfaces de recherche des moteurs de recherche : il s'agit la plupart du temps d'un ensemble de mots-clés traduit en une requête booléenne (conjonction des termes). D'autre part, le modèle est par défaut sur le mode « tout ou rien ». La valeur 1 est donnée aux documents qui satisfont la formule booléenne de la requête, et 0 aux autres. Ce qui ne permet évidemment pas un classement des documents, mais renvoie juste la liste des documents adéquats. Il n'y a pas non plus dans le modèle standard de pondération des termes des documents ou des requêtes. Bien évidemment, certains des problèmes décrits ici sont adressés, d'une façon ou d'une autre, par les systèmes.

1.3.2.2 Modèle vectoriel et variantes

Le *modèle vectoriel*, ou *Vector Space Model*, est sans doute un des modèles les plus connus en RI. Il le doit à sa capacité inhérente à classer les documents, sa robustesse et ses bons résultats [BP98]. Pour un historique, voir aussi Dubin [Dub04], et pour une organisation des différentes variantes du modèle, la figure 1.9.

Le modèle vectoriel consiste à représenter documents et requêtes comme des vecteurs dans un espace à n dimensions, ces dimensions étant les termes du vocabulaire d'indexation [Seb02, SWY75, BDJ99, BBM02].

Definition 2 (Vecteur de document)**.**
Soit $\mathcal{T}$ l'ensemble des termes du vocabulaire d'indexation et d, un document. Sa représentation $\overrightarrow{d}$ dans le modèle vectoriel est une application définie sur l'ensemble des termes de $\mathcal{T}$:

$$\forall\, t_i \in \mathcal{T}, \overrightarrow{d} :\ t_i \rightarrow [0,1]$$

On peut donner une définition similaire pour un vecteur de requête :

$$\forall\, t_i \in \mathcal{T}, \overrightarrow{q} :\ t_i \rightarrow [0,1]$$

La fonction de pertinence utilisée pour comparer document et requête est le cosinus, qui estime l'angle entre les deux vecteurs :

$$\cos(\overrightarrow{d}, \overrightarrow{q}) = \frac{\overrightarrow{d} \cdot \overrightarrow{q}}{|\overrightarrow{d}| \times |\overrightarrow{q}|}$$

Le cosinus est très souvent utilisé comme mesure de pertinence dans le modèle vectoriel, car il donne de bons résultats. Cependant, deux problèmes importants de cette approche sont le grand nombre de dimensions et l'indépendance des dimensions :

- le grand nombre de dimensions provient du fait que chaque terme du vocabulaire d'indexation devient une dimension de l'espace ; cela demande d'étudier en détail la représentation des vecteurs sous peine de travailler avec des matrices creuses et très grandes ;
- les dimensions de l'espace sont orthogonales, ainsi, que les termes soient proches ou non « sémantiquement », ne change rien pour le modèle. Par exemple, « chat » et « félin » sont aussi indépendants que « chat » et « automobile »: $\cos(\text{chat, félin}) = \cos(\text{chat,automobile}) = 0$.

Un certain nombre de variantes de ce modèle ont été proposées (cf. figure 1.9).

Le modèle vectoriel généralisé, *General Vector Space Model* [WZW85], utilise des combinaisons (co-occurrences) de termes plutôt que des termes individuels, avec pour objectif de limiter le problème de l'indépendance des dimensions. A notre connaissance, son efficacité n'a pas été prouvée. Et il fait augmenter le nombre de dimensions de l'espace, ainsi que la complexité du calcul des co-occurrences.

Le modèle de sémantique latente, *Latent Semantic Analysis* [DDL$^+$90], indexe par des regroupements de termes. En effet, il y a un certain nombre de termes du vocabulaire d'indexation qui sont fortement co-occurrents. Les rassembler en entités plus importantes permet de limiter les dimensions et l'indépendance des dimensions (les dimensions proches dans la collection de documents sont fusionnées). Néanmoins, le processus mathématique utilisé par cette méthode (traitement sur la matrice $termes \times documents$) est long à effectuer, supporte assez mal les évolutions dans la collection de documents, et ne permet que de lier des dimensions qui apparaissent comme liées dans la collection de documents.

Les modèles vectoriels thématisés [BK03, PK07] permettent d'ôter la contrainte d'orthogonalité des dimensions. Il s'agit d'une extension du modèle généralisé. C'est bien évidemment un modèle très intéressant, mais particulièrement complexe à mettre en place.

Kraft et Buell [KB83] et Salton *et al.* [SM83] ont étendu le modèle booléen en le mixant avec le modèle vectoriel. Cela permet de calculer une pertinence entre la requête et chaque document (amélioration par rapport au modèle booléen), tout en gardant les expressions booléennes sur les requêtes (amélioration par rapport au modèle vectoriel). C'est le modèle utilisé par les moteurs de recherche.

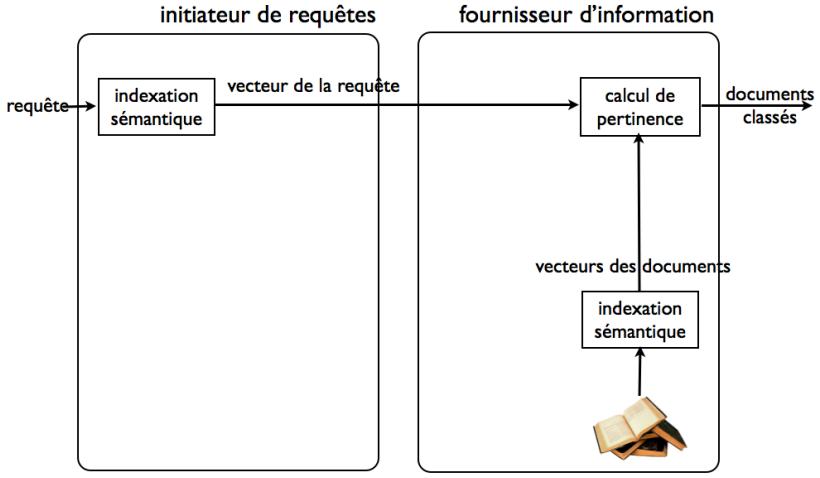


Figure 1.10 – Cadre d'un système de recherche d'information utilisant le modèle vectoriel sémantique.

Un autre type d'évolution du modèle vectoriel concerne l'utilisation de concepts d'une ontologie plutôt que des mots-clés. Cette idée est assez ancienne [Woo97, KC92] pour le modèle vectoriel et tend à se développer. De manière générale, les apports de l'utilisation de la sémantique ne sont pas immédiats, et demandent souvent d'étudier le cadre d'utilisation et les buts recherchés [AG08]. L'avantage principal est d'enrichir le vocabulaire d'indexation par l'utilisation de formalisations des connaissances et de rendre interopérables les différentes représentations. Cependant il faut noter que le problème de l'indépendance des dimensions n'est pas adressé par l'utilisation de concepts plutôt que de mots-clés.

Formellement, dans l'approche dite du modèle vectoriel sémantique, les dimensions de l'espace sont les concepts d'une ontologie. Le cadre général d'un tel modèle est celui de la figure 1.10 : requêtes et documents sont indexés sémantiquement et les documents sont classés par rapport à chaque requête en utilisant le cosinus comme mesure de pertinence.

Definition 3 (Vecteur sémantique)**.**
Un vecteur sémantique $\overrightarrow{v_\Omega}$ *est une application définie sur l'ensemble des concepts* $\mathcal{C}_\Omega$ *d'une ontologie* Ω
$\forall c \in \mathcal{C}_\Omega, \overrightarrow{v_\Omega} : c \rightarrow [0,1]$

Considérer les valeurs dans $[0,1]$ arbitraire et ne change en rien le raisonnement. La référence à l'ontologie sera omise s'il n'y a pas d'ambigüité. Dans ce modèle vectoriel, les concepts de l'ontologie sont souvent appelés les dimensions des vecteurs. Par exemple, considérons l'ontologie de la figure 1.11. Elle est composée de douze concepts, avec des liens de subsomption (*is-a*) entre eux. Par exemple le concept *public school* est une sorte (*is-a*) du concept *school*.

La figure 1.12 présente un document d_i caractérisé par un vecteur sémantique dans l'espace défini par l'ontologie de la figure 1.11. Nous voyons que $\overrightarrow{d_i}[financial\ institution] = 0.5$, $\overrightarrow{d_i}[central\ bank] = 0.8$, $\overrightarrow{d_i}[bank] = 1$, $\overrightarrow{d_i}[university] = 0.8$ et $\overrightarrow{d_i}[school] = 0.2$. Ce qui signifie que pour le document est lié à ces cinq dimensions, plus fortement au concept *bank* qu'aux autres, même si les concepts *central*

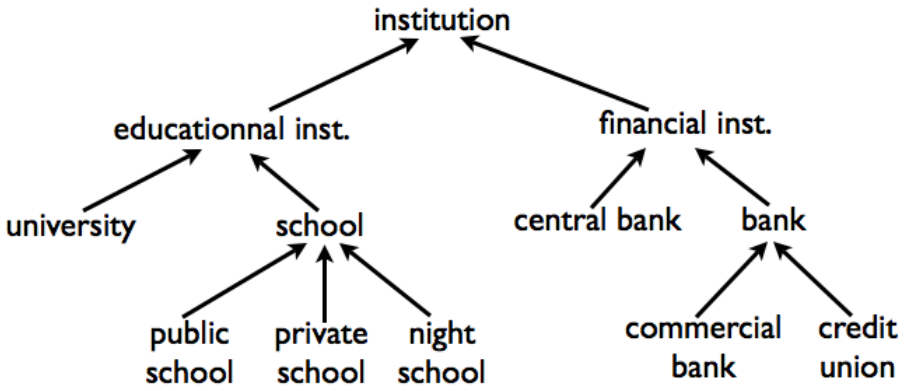


Figure 1.11 – Une ontologie restreinte, composée de douze concepts avec les liens de subsomption (*is-a*).

bank et *university* sont importants. En ce qui concerne les deux autres concepts, *financial institution* et *school*, leur pondération indique que ce ne sont pas des dimensions centrales pour le document.

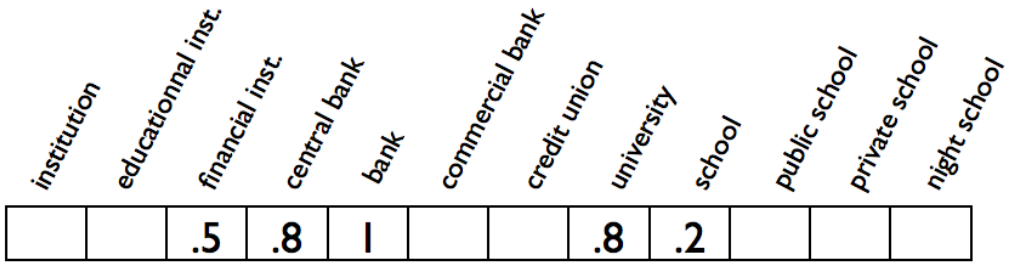


Figure 1.12 – Le vecteur sémantique d'un document caractérisé sur l'ontologie de la figure 1.11.

1.3.2.3 D'autres modèles utilisés en RI

Le modèle probabiliste a été développé dans les années Soixante-dix, mais a connu des développements récents, car les approches basées sur ce modèle ont obtenu de très bons résultats dans TREC, par exemple le système OKAPI [RWHBG95]. Documents et requêtes sont représentés par des vecteurs, comme dans le modèle vectoriel. La fonction de pertinence utilise les probabilités. Soit $P(R|\overrightarrow{d_i})$ la probabilité que le document d_i soit pertinent pour la requête $\overrightarrow{q}$, et $P(\overline{R}|\overrightarrow{d_i})$ la probabilité que ce document

ne soit pas pertinent pour cette requête. Alors la pertinence entre le document d et la requête q est :

$$sim(d,q) = \frac{P(R|\overrightarrow{d_i})}{P(\overline{R}|\overrightarrow{d_i})}$$

Ce modèle est assez lourd à mettre en œuvre, car il nécessite d'avoir une estimation des probabilités initiales. Néanmoins le cadre est très solide mathématiquement, et il permet un grand nombre de techniques formellement éprouvées et qui pratiquement, sont prometteuses. Taylor *et al.* [TZC+06] montrent ainsi que l'apprentissage automatique est très efficace pour la recherche d'information, à partir du moment où il est fourni au système suffisamment de paramètres, et de données recueillies sur les recherches.

Baziz *et al.* [BBA05] proposent une indexation utilisant le *noyau sémantique d'un document*. Il s'agit d'un ensemble de concepts pondérés suivant leur représentativité dans le documents et liés entre eux par des mesures de similarité. Cette structure dépend de la mesure de similarité considérée, mais n'est calculée qu'une fois par mesure de similarité. Baziz *et al.* a remarqué que la pondération des concepts est un point très crucial, autant que le choix de ces concepts, pour les performances du système. L'idée de noyau sémantique permet de rendre graphiquement de façon claire à l'utilisateur les concepts importants dans un document et leurs liens. La solution de Baziz *et al.* est d'une certaine manière proche du modèle des *graphes conceptuels*, tout en étant plus simple à mettre en place.

Le modèle des graphes conceptuels [Sow76, WWW, WWW] consiste à annoter sémantiquement les documents (et les requêtes) avec des concepts et des relations entre ces concepts, de manière graphique et avec une logique précise. Ce sont des descriptions de très haut niveau, permettant de décrire les énoncés du langage naturel de manière univoque et de raisonner sur des connaissances humaines. Ils sont de plus facile à lire pour un humain. Néanmoins les annotations automatiques sont difficiles à obtenir.

1.3.2.4 Choix du modèle vectoriel sémantique

La solution qui est développée dans cette thèse s'appuie sur le modèle vectoriel sémantique. Le modèle vectoriel est toujours très utilisé en RI et, utilisé avec le cosinus comme mesure de pertinence, obtient souvent de très bons résultats. L'utilisation de concepts d'une ontologie à la place de termes est cohérente avec le contexte dans lequel se situe notre travail, où les différents pairs du système utilisent des ontologies, en particulier pour indexer leurs documents, et où l'interopérabilité est assurée par des descriptions sémantiques. De plus, le fait d'utiliser des vecteurs est très adapté au calcul du type d'expansion que nous proposons.

CHAPITRE 2

Structurer l'expansion de requêtes

Dans ce chapitre, nous présentons notre approche pour comparer documents et requêtes. Elle est basée sur une expansion de requête spécifique, appelée *Expansion Structurante*, qui permet la transformation de chaque vecteur de document en une *Image de Document*. Cette méthode sera dénotée EXSID dans la suite du travail. L'image peut être considérée comme une « vue » du document au travers de la requête.

La figure 2.1 récapitule les différentes étapes grâce à deux modules : module d'expansion structurante qui génère la requête étendue ; et module de calcul de l'image, qui « replie » les différentes dimensions sémantiquement enrichies dans le document pour générer l'image du document.

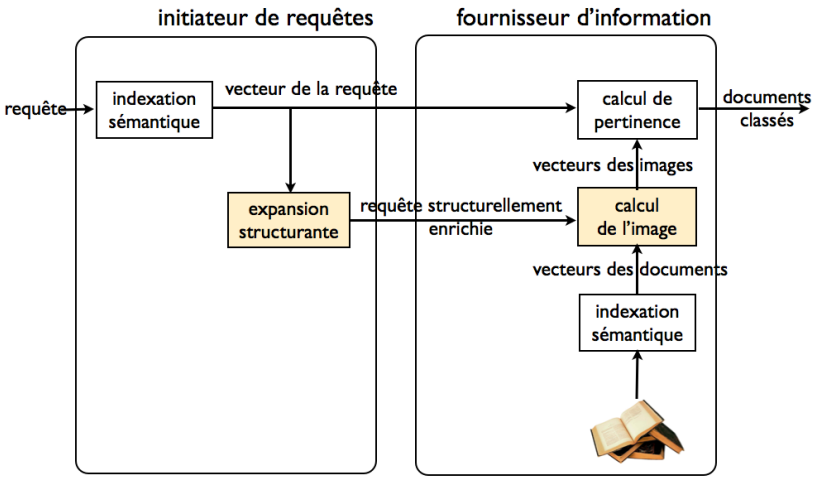


Figure 2.1 – Notre solution consiste en deux modules : un module d'expansion structurante des requêtes et un module de calcul de l'image d'un document au travers de cette expansion. Nous appelons cette solution EXSID

La dernière section de ce chapitre présente les deux paramètres importants de l'expansion structurante que sont la fonction de propagation et la fonction de similarité sémantique.

2.1 Expansion « classique » de requêtes : apports et limites

De manière générale, l'expansion de requêtes est une méthode très couramment utilisée lorsque l'on obtient peu ou pas de réponses. L'expansion est souvent réalisée par le fournisseur qui étend la requête selon ses connaissances pour proposer des réponses supplémentaires de manière coopérative. Plusieurs domaines utilisent ce type d'approche. Par exemple, l'expansion de requête a été introduite dans les années quatre-vingt dans les systèmes de réponses coopératives [GGM92] pour les bases de données. Parmi les techniques envisagées, certaines étendent les requêtes SQL en utilisant des taxonomies. Dans le cadre d'une représentation vectorielle [QF93, Voo94], l'idée est de pondérer de nouvelles dimensions liées aux dimensions exprimées dans la requête initiale. Nous illustrons cette idée de manière très générale en considérant les deux affirmations suivantes :

- A : « Les chats sont carnivores »;
- B : « Les félins qui ronronnent mangent principalement de la viande ».

Le tableau 2.1 présente la matrice correspondant à une indexation des deux affirmations dans un modèle vectoriel.

	chat	carnivore	félin ronronnant	manger	viande
A	1	1	0	0	0
B	0	0	1	1	1

Table 2.1 – La matrice des concepts pour les deux affirmations A et B.

Nous étendons les affirmations A et B. Ainsi, en sachant qu'un chat est un félin qui ronronne, et qu'un carnivore mange de la viande, il est possible de repondérer les affirmations précédentes, comme dans le tableau 2.2

	chat	carnivore	félin ronronnant	manger	viande
A	1	1	0.5	0.5	0.5
B	0.5	0.5	1	1	1

Table 2.2 – La matrice des concepts pour les deux affirmations A et B après expansion.

L'expansion de requêtes permet a priori de retrouver plus de documents pertinents pour une requête car elle résout dans une certaine mesure le problème de l'indépendance des dimensions. Si nous considérons les deux affirmations précédentes représentées par les vecteurs de la table 2.1, bien que les deux affirmations parlent de la même chose, leur cosinus est nul, ce qui signifie qu'il n'y a aucun lien de pertinence entre les deux. Cela vient de l'hypothèse que les dimensions sont indépendantes. La proximité entre dimensions ne peut pas être rendue directement dans un tel espace. L'expansion, en pondérant des concepts supplémentaires, permet de considérer plus de dimensions. Ainsi le cosinus des vecteurs de la table 2.2 est cette fois non nul. Ceci permet d'affirmer une certaine pertinence entre les deux affirmations.

La justesse de l'expansion repose bien évidemment sur la formalisation des liens entre les concepts (similarité) et le mode de pondération des concepts non initialement impliqués (propagation). Dans l'exemple précédent, nous avons supposé que les dimensions similaires mais non initiales, avaient une pondération moindre. Similarité et propagation sont étudiées plus en détail dans les sections suivantes.

Si elle permet de s'abstraire du problème de l'indépendance des dimensions dans une certaine mesure, l'expansion a un certain nombre de conséquences. Tout d'abord, le fait d'introduire de nouvelles

dimensions dans les vecteurs a tendance à générer du « bruit » et donc à dégrader les résultats. C'est très souvent le cas si la propagation des pondérations affecte des concepts peu similaires aux concepts initiaux de la requête. Ainsi Voorhees [Voo94] et Mihalcea et Moldovan [MM00b] montrent que les résultats sont moins bons que la solution classique si l'expansion n'est pas parfaitement dirigée. Il y a de plus une conséquence contre-intuitive à l'expansion : les documents très pertinents, c'est-à-dire qui ne comportent que des concepts initiaux de la requête, voient leur pertinence baisser. Par exemple, pour une requête sur le concept *chat* étendue avec le concept *chat de gouttière*, un document représenté par le concept *chat* seul voit sa mesure de pertinence baisser par rapport à celle qu'il aurait eue sans étendre la requête. Enfin, si tous les concepts pondérés par l'expansion sont mémorisés dans le même vecteur, il devient impossible de détecter l'origine de cette pondération. Par exemple, dans l'affirmation A du tableau 2.2, il n'est pas possible de savoir si le concept *viande* est pondéré parce qu'il est lié à *chat* ou parce qu'il est lié à *carnivore*. Nie et Jin [NJ02] critiquent aussi la propagation dans un seul vecteur des différents concepts d'une requête, en soulignant qu'il peut y avoir possibilité de déséquilibre de l'importance des concepts initiaux d'une requête. Par exemple, supposons que le vecteur initial de la requête est $\overrightarrow{q}[c_1] = 0.5$ et $\overrightarrow{q}[c_2] = 0.5$, que la propagation à partir de c_2 pondère aussi c_3, c_4 and c_5, respectivement avec les poids 0.3, 0.3 et 0.2 ; mais que la propagation à partir de c_1 ne pondère aucun autre concept. Dans un tel cas, il semble que la propagation affecte au concept initial c_2 une importance beaucoup plus grande que celle qu'il avait initialement.

2.2 Expansion structurante

Dans le processus d'expansion décrit précédemment, nous remarquons que chaque concept pondéré de la requête est le point de départ d'une propagation. Pour éviter les problèmes de bruit, déséquilibre de l'importance des concepts initiaux, etc. nous proposons de mémoriser le résultat de chacune de ces propagations dans des vecteurs séparés. L'ensemble de ces vecteurs constitue *l'expansion structurante de la requête*. La requête initiale n'étant pas modifiée, elle peut être utilisée dans le processus de classement par pertinence des documents. Le rôle de l'expansion structurante est d'adapter la description d'un document à la requête, ce que nous appelons *image d'un document*. Soit une requête portant sur *chat* et un document décrit par le concept *chat de gouttière* seul. Si l'expansion structurante relie ces deux concepts et permet donc d'obtenir une image du document où le concept *chat* est pondéré de manière non nulle, alors le document peut être pertinent.

L'expansion structurante étant constituée des expansions obtenues à partir de chaque concept pondéré de la requête, nous allons étudier le processus d'expansion partant d'un concept unique. Ce concept étant donné, le problème consiste à propager l'intérêt aux autres concepts de l'ontologie en leur attribuant des valeurs en fonction de leur similarité avec celui-là. Nous présentons les définitions relatives à une fonction de similarité, et une fonction de propagation. La propagation de l'intérêt est obtenue par la composition de ces deux fonctions. L'expansion structurante peut alors être introduite en s'appuyant sur ces deux notions.

2.2.1 Similarité

La similarité entre concepts a fait l'objet de très nombreuses études, que ce soit dans le domaine de la RI [RS95a], celui de l'ingénierie des connaissances (IC) [AE03] ou du traitement automatique du langage naturel (TALN) [Res95, PBP03]. Cette section se contente de présenter les points strictement nécessaires à la définition et à la compréhension de ce qu'est l'expansion structurante. Un état de l'art

des similarités sémantiques est proposé en section 2.4.2, page 35.

La fonction de similarité est généralement une fonction à deux paramètres : les deux concepts considérés. Notre contexte nous conduisant à avoir une approche centrée sur un concept, nous proposons une définition adaptée :

Definition 4 (Mesure de similarité sémantique centrée sur un concept)**.**
Soient Ω une ontologie, $\mathcal{C}_\Omega$ l'ensemble des concepts de cette ontologie et c un concept de $\mathcal{C}_\Omega$.
Une fonction sim_c est une fonction de similarité centrée sur c *si et seulement si :*

- $sim_c : \left| \begin{array}{ccc} \mathcal{C}_\Omega & \mapsto & [0,1] \\ c' & \rightarrow & sim_c(c') \end{array} \right.$
- $sim_c(c) = 1$;
- $\forall c' \in \mathcal{C}_\Omega, sim_c(c') \leq 1$.

$sim_c(c_1) \geq sim_c(c_2)$ *signifie que c_1 est plus similaire à c que c_2.*

Il est trivial de noter que cette fonction induit un pré-ordre total sur les concepts de Ω.

2.2.2 Propagation

La propagation à partir d'un concept est alors définie en utilisant l'ordre défini par (dis)similarité avec le concept pivot.

Definition 5 (Fonction de propagation)**.**
Une fonction $\mathcal{P}f$ est une fonction de propagation *si et seulement si :*

- $\mathcal{P}f : \left| \begin{array}{ccc} [0,1] & \mapsto & [0,1] \\ x & \rightarrow & \mathcal{P}f(x) \end{array} \right.$
- $\mathcal{P}f(1) = 1$;
- $\forall \alpha, \beta \in [0,1]\ \alpha \leq \beta \Rightarrow \mathcal{P}f(\alpha) \ \leq \ \mathcal{P}f(\beta)$.

L'idée générale est d'utiliser cette fonction en la composant avec la similarité. La valeur quantifiant le lien entre un concept c' et un concept central de la requête c, pondéré par v, s'obtient par la formule : $v \times \mathcal{P}f(sim_c(c'))$. La figure 2.2 présente un exemple de ce processus où le concept central, *bank*, est supposé pondéré par 1 dans la requête. Les concepts de l'ontologie sont positionnés selon la valeur de la fonction sim_{bank}. Ainsi le concept $commercial\ bank$ obtient la pondération $1 \times \mathcal{P}f(sim_{bank}(commercial\ bank)) = 0.6$.

2.2.3 Définition de l'expansion structurante

L'objectif de l'expansion structurante est de définir et mémoriser les propagations obtenues à partir de chacun des concepts de la requête. Le faire pour tous les concepts serait bien trop coûteux, et certainement peu intéressant. En effet, une propagation à partir d'un concept ayant une pondération nulle associe la valeur 0 à tous les concepts. Seuls les concepts pondérés doivent donc être considérés. Plus précisément encore, nous ne gardons dans une requête que ceux représentant au mieux la recherche, c'est-à-dire dépassant un certain seuil. La requête est alors uniquement constituée de concepts pondérés non nuls, dits concepts centraux.

Definition 6 (Concepts centraux d'une requête)**.**
Soit $\overrightarrow{q}$ une requête. $\mathcal{C}_{\overrightarrow{q}}$, l'ensemble des concepts centraux de la requête, *dénote l'ensemble des concepts pondérés de manière non nulle par $\overrightarrow{q}$.*

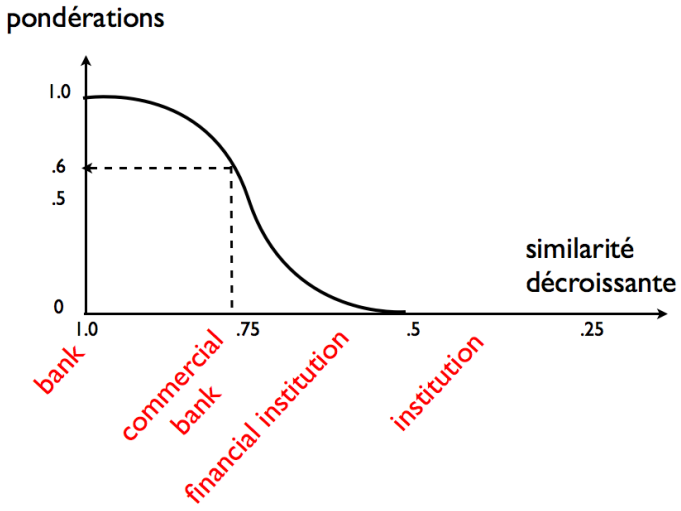


Figure 2.2 – Exemple d'une fonction de propagation.

Etant donné un concept c identifié comme central, le vecteur que nous lui associons dans l'expansion structurante permet de caractériser les liens entre c et les autres concepts. Dans l'espace vectoriel, cela correspond à établir les liens qu'il y a entre la dimension associée à ce concept et les autres dimensions de l'espace. Nous appelons ce vecteur *dimension sémantiquement enrichie* (DSE).

Definition 7 (Dimension sémantiquement enrichie et concept central d'une DSE)**.**
Soit c un concept appartenant à $\mathcal{C}_\Omega$ et v sa valuation.
Un vecteur sémantique $\overrightarrow{dse_c}$ est une dimension sémantiquement enrichie (DSE) *de c, si et seulement si $\overrightarrow{dse_c}[c] = v$ et $\forall c' \in \mathcal{C}_\Omega, \overrightarrow{dse_c}[c'] \leq \overrightarrow{dse_c}[c]$. c est appelé le* concept central *de cette DSE.*

Cette définition est très peu contraignante. Elle ne fait que garantir qu'aucun concept dans la DSE n'a une importance, c'est-à-dire une pondération, supérieure à celle du concept central.

L'expansion structurante quant à elle est plus précise et utilise les notions de similarité et de propagation introduites précédemment.

Definition 8 (Expansion structurante d'une requête)**.**
Soient Ω une ontologie, $\mathcal{C}_\Omega$ l'ensemble des concepts de cette ontologie.
Soit $\overrightarrow{q}$ le vecteur sémantique d'une requête et $\mathcal{C}_{\overrightarrow{q}}$ l'ensemble de ses concepts centraux. Une expansion structurante *de $\overrightarrow{q}$, notée $\mathcal{E}_{\overrightarrow{q}}$, est un ensemble défini par :*
$\mathcal{E}_{\overrightarrow{q}} = \{\overrightarrow{dse_c} : c \in \mathcal{C}_{\overrightarrow{q}}, \forall c' \in \mathcal{C}_\Omega, \exists \mathcal{P}f$ *une fonction de propagation* $: \overrightarrow{dse_c}[c'] = \overrightarrow{q}[c] \times \mathcal{P}f(sim_c(c'))\}$

Si la mesure de similarité sémantique est généralement la même quel que soit le concept central, la fonction de propagation peut au contraire varier d'un concept à l'autre. Par exemple un utilisateur peut choisir de faire une propagation large sur un concept, et une propagation très réduite sur un autre concept d'une même requête.

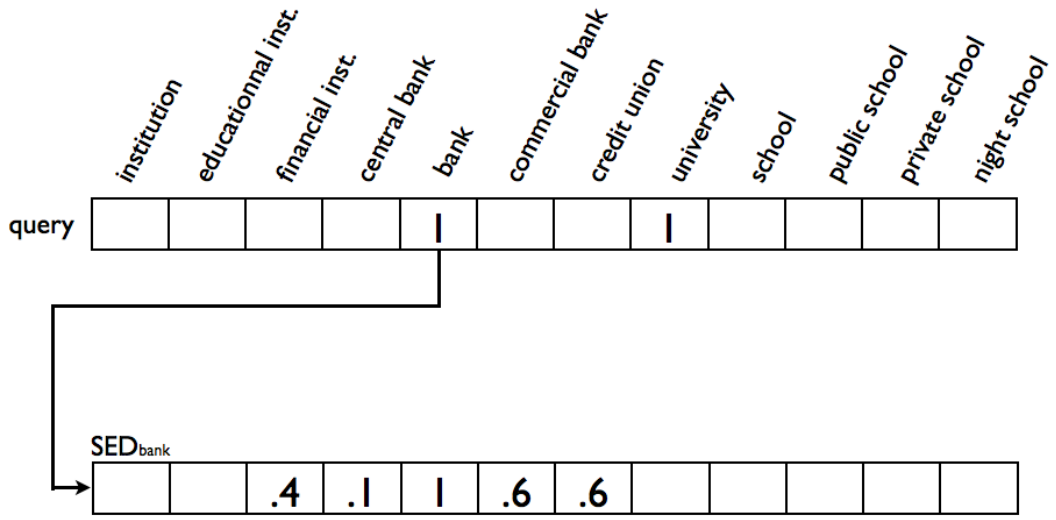


Figure 2.3 – A partir d'un concept central de la requête, le module d'enrichissement créé une dimension sémantiquement enrichie.

Théorème 1 (Une expansion structurante est composée de DSEs)**.**
Soit $\overrightarrow{q}$ *le vecteur sémantique d'une requête. Soit* $\mathcal{E}_{\overrightarrow{q}}$ *l'expansion structurante de* $\overrightarrow{q}$*.*
$\forall \overrightarrow{dse_c} \in \mathcal{E}_{\overrightarrow{q}}$, $\overrightarrow{dse_c}$ *est une DSE.*

Démonstration. Soit $\overrightarrow{dse_c} \in \mathcal{E}_{\overrightarrow{q}}$, par définition la valeur associée à c est : $\overrightarrow{dse_c}[c] = \overrightarrow{q}[c] \times \mathcal{P}f(sim_c(c))$; par définition de la fonction de similarité, $sim_c(c) = 1$, donc l'expression devient $\overrightarrow{q}[c] \times \mathcal{P}f(1)$, et par définition de la fonction de propagation, $\mathcal{P}f(1) = 1$. La valeur associée à c est donc $\overrightarrow{q}[c]$. Pour les autres concepts c', leur similarité étant inférieure ou égale à 1 et la fonction de propagation étant croissante, les valeurs qui leurs sont associées sont inférieures ou égales à $\overrightarrow{q}[c]$. □

2.3 Image d'un document au travers d'une requête

Dans la section précédente, nous avons vu comment obtenir l'expansion structurante d'une requête. L'objet de cette section est de montrer comment l'utiliser pour calculer l'image d'un document. Intuitivement, cette image décrit le document en amenant des pondérations sur les dimensions de la requête en fonction de l'expansion structurante.

La figure 2.1, page 23, montre le positionnement du module de construction de l'image dans le cadre d'un système de recherche d'information sémantique. L'image du document se fait du côté du fournisseur d'information. Ce dernier utilise les vecteurs sémantiques de ses documents et l'expansion structurante de la requête que lui a envoyé l'initiateur de la requête. Le résultat de la construction de l'image sert ensuite à donner une valeur de pertinence du document par rapport à la requête initiale.

Supposons maintenant que nous avons un document et une requête étendue, comme dans la figure 2.5. La requête étendue est celle présentée précédemment, avec deux dimensions sémantiquement enrichies,

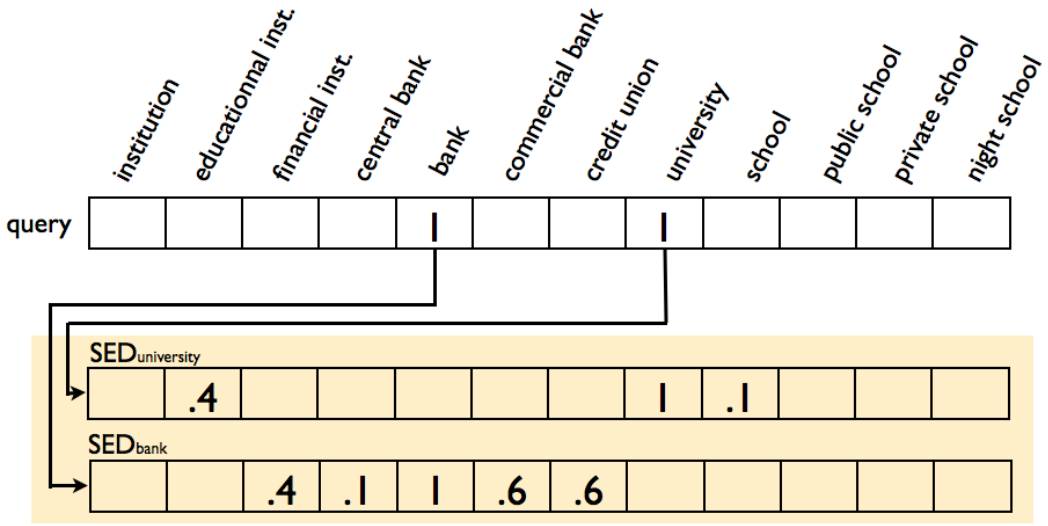


Figure 2.4 – L'ensemble des concepts centraux de la requête créent un ensemble de dimensions sémantiquement enrichies, et donc une requête enrichie.

celle sur le concept *university*, $\overrightarrow{dse}_{university}$, et celle sur *bank*, $\overrightarrow{dse}_{bank}$. Le document est lui pondéré sur les concepts *institution*, *educationnal institution*, *central bank*, *commercial bank*, *university*, *school* et *private school*.

Les concepts peuvent être classés en trois catégories. Un concept peut être :

- concept central d'une DSE.
  Par exemple *bank* et *university*.
  La DSE est là pour décrire les liens entre ce concept et les autres concepts. La valeur associée à l'image du document pour cette dimension résulte donc d'un calcul impliquant tous les concepts pondérés non nuls de la DSE.
- pondéré (de manière non nulle) dans au moins une DSE sans être concept central.
  Par exemple *educationnal inst.*, *financial inst.*, etc.
  Le fait qu'un tel concept apparaisse dans une DSE entraîne qu'il est impliqué dans la pondération du calcul central de la DSE dans laquelle il apparaît (cf. point précédent). Conserver sa valeur dans la dimension qui lui est associée dans l'espace vectoriel aurait pour conséquence de prendre en compte cette valeur deux fois : une première fois positivement dans la dimension initiale de la DSE concernée, et une seconde négativement dans sa dimension propre, car la requête n'a pas de pondération sur cette dimension. Cette dernière manière de prendre en compte le concept suppose qu'il n'est pas intéressant pour la requête, ce qui contredit le fait qu'il apparaît dans une DSE. Nous neutralisons donc cette valeur dans la dimension propre en donnant une valeur nulle à l'image pour ce concept. Cette dimension est en quelque sorte *repliée* sur la dimension du concept central de la DSE.
- pondéré dans aucune DSE de la requête.
  Par exemple *institution* et *private school*, *public school*, etc.
  Le fait qu'un tel concept n'apparaît dans aucune DSE signifie qu'il n'est d'aucun intérêt pour la

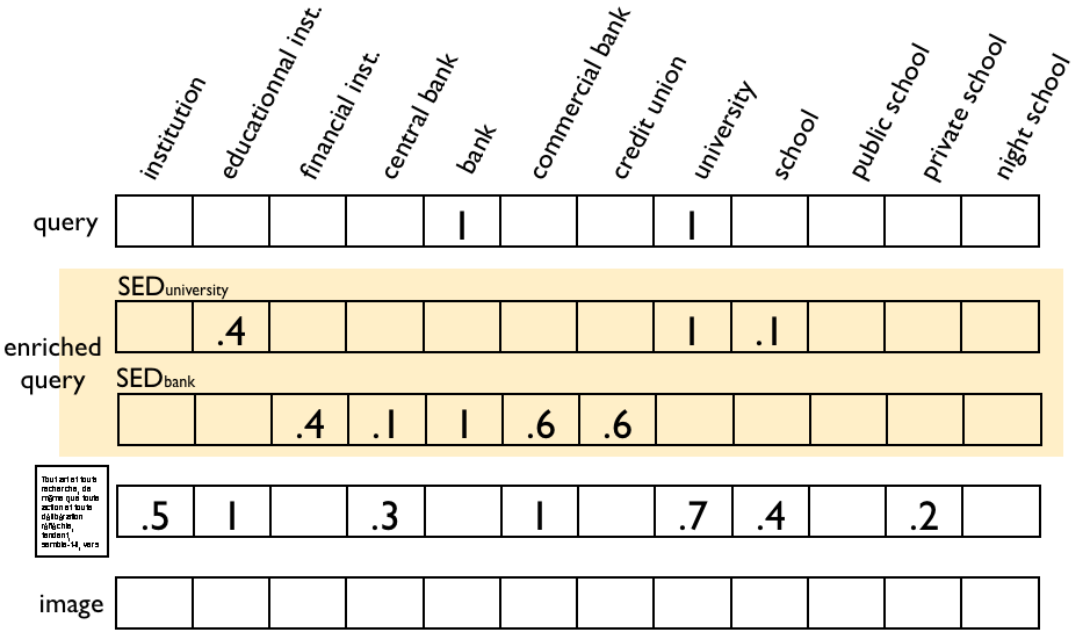


Figure 2.5 – Une requête enrichie, un document, et son image, pour l'instant vide.

requête. Il est important de conserver la valeur qui lui est associée. Elle influera sur l'évaluation de la pertinence du document, de manière négative. La valeur associée à ce concept dans l'image du document est donc la valeur de la représentation du document pour ce concept.

Definition 9 (Image d'un document)**.**
Soient Ω une ontologie, $\mathcal{C}_\Omega$ l'ensemble des concepts de cette ontologie.
Soit $\overrightarrow{q}$ le vecteur sémantique d'une requête. Soit $\mathcal{E}_{\overrightarrow{q}}$ l'expansion structurante de $\overrightarrow{q}$.
Soit $\overrightarrow{d}$ le vecteur sémantique décrivant un document.
*Le vecteur sémantique $\overrightarrow{i}_d$ est l'*image de $\overrightarrow{d}$ *si et seulement si pour tout concept $c \in \mathcal{C}_\Omega$:*

$$\begin{cases} \overrightarrow{i}_d[c] = \max_{c' \in \mathcal{C}_\Omega}(\overrightarrow{d}[c'] \times \overrightarrow{dse}_c[c']) & \textit{si } c \in \mathcal{C}_{\overrightarrow{q}} \\ \overrightarrow{i}_d[c] = 0 & \textit{si } \exists c' \in \mathcal{C}_{\overrightarrow{q}}, \overrightarrow{dse}_{c'}[c] \neq 0 \\ \overrightarrow{i}_d[c] = \overrightarrow{d}[c] & \textit{sinon} \end{cases}$$

La pertinence d'un document par rapport à une requête est alors obtenue en comparant l'image du document $\overrightarrow{i}_d$ avec le vecteur de la requête $\overrightarrow{q}$. Il est intéressant de noter que cette approche ne nécessite aucune adaptation du module de calcul de pertinence, pas plus que module de calcul de représentation des documents (indexation sémantique). Nous pouvons donc espérer intégrer sans trop de difficulté notre approche à des solutions existantes.

L'algorithme 1 propose une opérationnalisation de cette définition. Analysons son fonctionnement en l'appliquant à l'exemple de la figure 2.6. Il créé l'image du document en traitant les dimensions sémantiquement enrichies les unes après les autres. Commençons avec la $\overrightarrow{dse}_{bank}$. Le processus replie

les concepts de la DSE et du document sur le concept central de la DSE, ici le concept *bank*. Les informations contenues dans les cinq concepts de la DSE centrée sur *bank* et celles de leurs deux concepts correspondants dans le document se replient sur le concept central de la DSE, *bank*. En pratique, nous multiplions les pondérations de chaque concept de la DSE et du document, et nous prenons la valeur maximum. Ainsi pour *central bank*, le résultat est 0.03, et pour *commercial bank*, le résultat est 0.6. C'est donc ce résultat qui pondère dans l'image le concept central de la DSE, ici *bank*. Les pondérations du document, parce qu'elles sont impliquées dans le processus précédent, sont alors supprimées, et elles n'apparaissent pas dans l'image. C'est pourquoi les pondérations du document sur les concepts *central bank* et *commercial bank* sont mises à 0 dans l'image. Les dimensions correspondantes sont repliées sur le concept central *bank* qui est celui pondéré dans la requête.

Algorithme 1 : Image d'un document par rapport à une requête enrichie.

entrée : le vecteur sémantique d'un document $\overrightarrow{d}$ sur une ontologie Ω; une requête enrichie $\mathcal{E}_{\overrightarrow{q}}$
sortie : le vecteur sémantique $\overrightarrow{i}_d$, image de $\overrightarrow{d}$.
begin
  forall $c \in \mathcal{C}_{\overrightarrow{q}}$ **do**
    $\overrightarrow{i}_d[c] \leftarrow 0$;
    forall $c' : \overrightarrow{dse}_c[c'] \neq 0$ **do**
      $\overrightarrow{i}_d[c] \leftarrow \max(\overrightarrow{d}[c'] \times \overrightarrow{dse}_c[c'], \overrightarrow{i}_d[c])$;
  forall $c \notin \mathcal{C}_{\overrightarrow{q}}$ **do**
    if $\exists c' \in \mathcal{C}_{\overrightarrow{q}} : \overrightarrow{dse}_{c'}[c] \neq 0$ **then** $\overrightarrow{i}_d[c] \leftarrow 0$
    else $\overrightarrow{i}_d[c] \leftarrow \overrightarrow{d}[c]$;
  return $\overrightarrow{i}_d$;
end

De la même façon, pour la DSE centrée sur *university* : cf. figure 2.7. Cette fois-ci, tous les concepts pondérés dans la dimension sémantiquement enrichie le sont aussi dans le document. Là-aussi, le processus effectue un repliement des concepts sur le document. La valeur maximum est ici celle du concept *university* : 0.7, et c'est elle qui est donnée à *university* dans l'image du document. Les concepts *educationnal institution* et *school*, qui apparaissent dans la dimension sémantiquement enrichie et le document, ne sont pas dans l'image du document.

La dernière étape consiste à prendre en compte les concepts du document qui ne sont impliqués dans aucune des DSEs de la requête. En effet, ces concepts, comme *institution* ou *private school* dans la figure 2.8, sont des dimensions du document qui n'appartiennent à aucune des DSEs de la requête ; du bruit pour la requête. C'est pourquoi l'image doit conserver ces dimensions qui pénalisent le document face à la requête. Les deux concepts précédents sont placés dans l'image, et pénaliseront l'image lorsque le cosinus donnera une valeur à l'image pour la requête.

Le module de calcul de pertinence des documents par rapport à une requête (cf. figure 2.1, page 23) utilise le cosinus entre les images et la requête initiale : $\cos(\overrightarrow{i}_d, \overrightarrow{q})$. Cela permet de prendre en compte les documents qui ont des concepts liés à ceux de la requête. Utiliser le cosinus, et donc la norme des vecteurs, permet de donner une valeur de pertinence plus faible aux images qui ont une norme importante, c'est-à-dire qui ont beaucoup de concepts en dehors des DSEs.

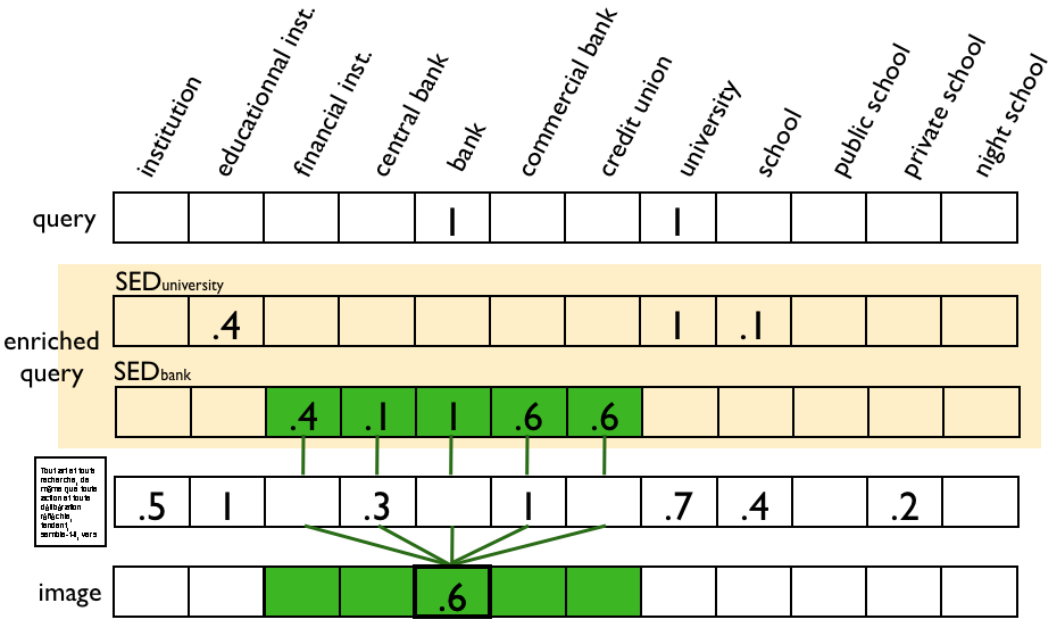


Figure 2.6 – Une requête enrichie, un document, et son image après traitement d'une des deux DSEs de la requête enrichie.

2.4 Fonctions de propagation et de similarité sémantique : propositions

Nous avons présenté en section 2.2.1 la fonction de similarité sémantique et la fonction propagation des poids lors de l'enrichissement sémantique de dimensions. Ces deux paramètres sont importants pour notre approche, car ils définissent la manière dont sont calculées les DSEs. Nous allons proposer dans cette section une classe de fonctions de propagation, une étude des mesures de similarité ainsi que notre proposition. Ce travail a fait l'objet d'une publication dans [VCLV08b].

2.4.1 Fonction de propagation

La fonction de propagation intervient dans le calcul d'une DSE en attribuant une valeur d'intérêt en fonction de la similarité par rapport à un concept central.

Il existe un grand nombre de solutions envisageables. Les fonctions d'appartenance du type de celles de la logique floue [Zad65] sont particulièrement adaptées selon nous. Contrairement à la logique classique, où les propriétés ont deux états possibles (vrai ou faux, 1 ou 0), la logique floue donne plus de liberté dans les valeurs. Un sous-ensemble flou F est défini sur un ensemble de valeurs, le référentiel U. F est caractérisé par une fonction d'appartenance :

$$\mu : x \in U \rightarrow \mu(x) \in [0, 1]$$

Par exemple, la figure 2.9 (a) présente les propriétés *froid, tiède* et *chaud* (pour de l'eau par exemple)

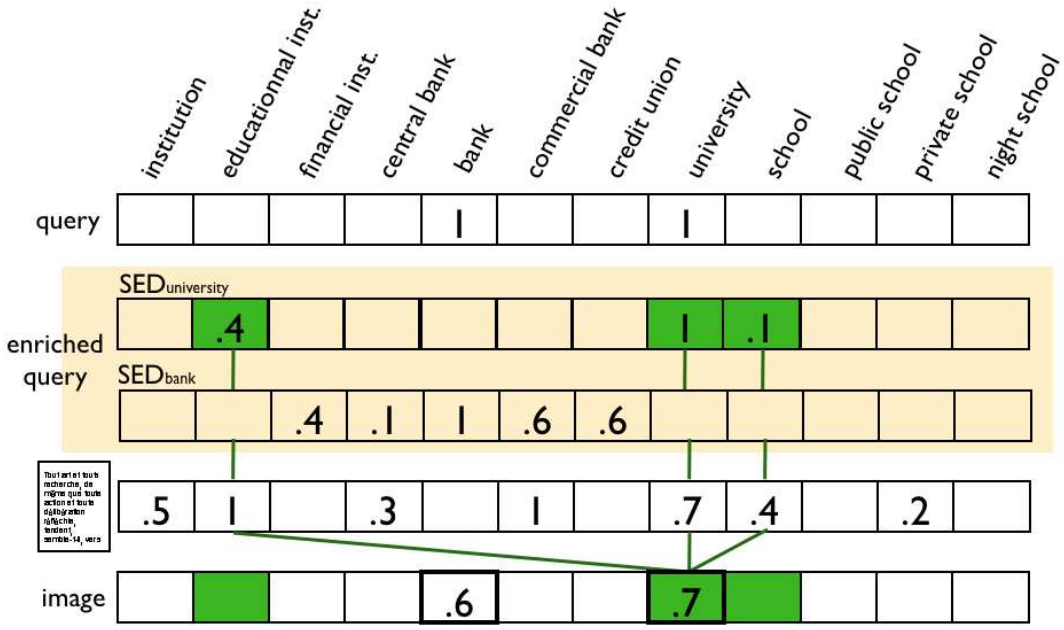


Figure 2.7 – Une requête enrichie, un document, et son image après traitement de la deuxième DSE de la requête enrichie.

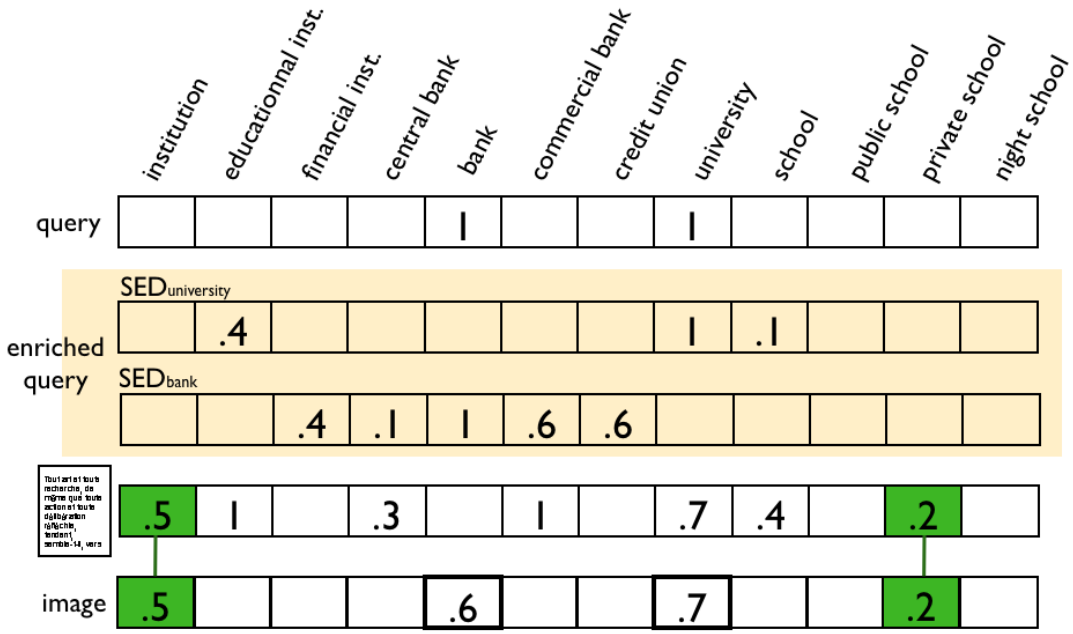
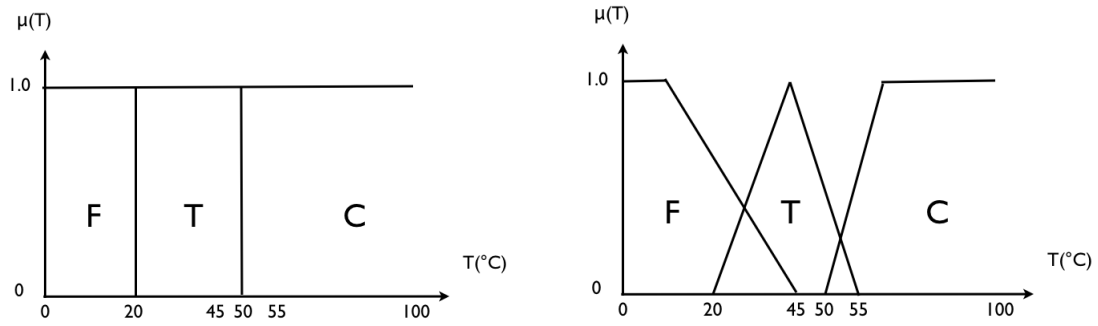


Figure 2.8 – Une requête enrichie, un document, et son image à la fin du traitement.

en logique classique. Il est évident que ce n'est pas une représentation très « réaliste ». Ces même propriétés en logique floue sont celles de la figure 2.9 (b). Nous pouvons voir que les ensembles correspondant à *froid*, *tiède* et *chaud* n'ont pas que deux valeurs, mais que leur fonction d'appartenance accepte des valeurs plus complexes. Aux environs de $30^{\circ}C$ par exemple, l'eau est un peu froide et un peu tiède.



(a) (b)

Figure 2.9 – Représentation de trois états de l'eau en logique classique (a) et en logique floue (b).

Notons que les fonctions d'appartenance ne sont pas forcément linéaires par morceau, mais peuvent avoir des transitions hyperboliques, exponentielles, gaussiennes, etc.

Nous voulons que les pondérations des concepts soient dépendantes de leur similarité avec un concept central. Nous proposons une fonction linéaire par morceau avec une seule transition, et donc les deux paramètres suivants :

- l_1 : valeur de similarité jusqu'à laquelle les concepts ont la même pondération que le concept central ;
- l_2 : valeur de similarité jusqu'à laquelle les concepts ont une pondération décroissante.

De telle façon que $\forall x \in [0..1]$:

$$\mathcal{P}f(x) = f_{l_1,l_2}(x) = \begin{vmatrix} 1 & si\ x \geq l_1 \\ \frac{1}{l_1 - l_2}x + \frac{l_2}{l_1 - l_2} & si\ l_1 > x > l_2 \\ 0 & si\ l_2 \geq x \end{vmatrix}$$

La figure 2.10 est un exemple d'une telle fonction, $f_{0.7,0.4}$. Le concept central est $bank$, dont la pondération est 1, et le classement est effectué grâce à sim_{bank}. De $l_1 = 0.7$ à $sim_{c_2}(c_2) = 1$, tous les concepts ont la pondération de $bank$, par exemple le concept $commercial\ bank$ qui a la valeur de similarité $sim_{bank}(commercial\ bank) = 0.85$. De l_1 à $l_2 = 0.4$, les concepts sont pondérés de manière décroissante en fonction de leurs valeurs décroissantes de similarité, par exemple le concept $financial\ institution$. La fonction de propagation donne la pondération 0 à toutes les valeurs de similarité plus faibles que l_2: $institution$.

Rappelons que le rôle de cette fonction n'est pas de fixer l'intérêt d'un concept par rapport à un concept de la requête, mais simplement d'exprimer comment cet intérêt se propage. En effet, comme nous l'avons vu (cf. définition 8), lors du calcul d'une DSE, le résultat de cette fonction est multiplié par la pondération du concept central : $\overrightarrow{dse}_c[c'] = \overrightarrow{q}[c] \times \mathcal{P}f(sim_c(c'))$.

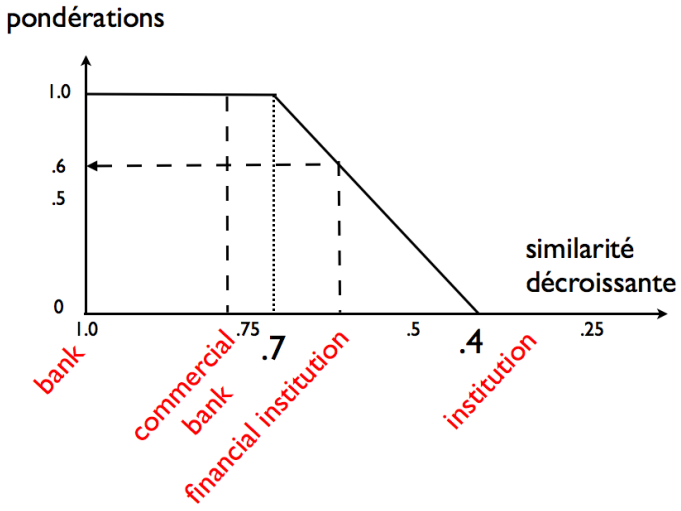


Figure 2.10 – Exemple d'une fonction de propagation $f_{0.7,0.4}$ avec le concept central *bank*.

2.4.2 Similarités sémantiques

La similarité sémantique intervient dans le calcul d'une DSE en permettant de préciser pour chaque concept de l'ontologie sa proximité par rapport à un concept central. C'est en fonction de cette valeur de similarité que la fonction de propagation donne sa valeur. Dans cet objectif, il est important de considérer non seulement l'ordre induit sur les concepts de l'ontologie, mais aussi et surtout les valeurs de similarité associées à chaque concept.

Dans cette section, nous allons d'abord présenter un petit état de l'art des mesures de similarité sémantique. Certaines propriétés de ces mesures vont nous permettre de faire un choix que nous allons améliorer pour qu'il s'adapte à notre cadre.

2.4.2.1 Etat de l'art des similarités sémantiques

La similarité sémantique a été un champ d'intérêt pour la recherche en intelligence artificielle, psychologie, sciences cognitives. Le premier modèle visant à caractériser et simuler cette capacité de l'être humain date de Quillian à la fin des années 1960 [Qui68]. Il existe deux techniques principales : basée sur les noeuds (concepts) ou basée sur les arcs (relations) de l'ontologie.

Approche basée sur les arcs La mesure la plus intuitive pour évaluer la similarité sémantique entre deux concepts est de compter les arcs (relations *is-a*) qui les séparent [RMBB89]. Cependant, il est nécessaire de savoir si ces liens de subsomption entre concepts sont bien les mêmes partout dans l'ontologie. En effet, il n'est pas sûr que le sens derrière les liens *is-a* dans la hiérachie des concepts soit le même partout [JC97, RS95a]. Par exemple, un concept semble d'autant plus proche de ses descendants directs qu'il est bas dans la hiérarchie : *entité* et *objet* semblent moins proches que *chat* et *chat persan*. De

même, des descendants sont parfois plus « représentatifs » que d'autres, et c'est sans doute un jugement culturel : un *chat européen* est sans doute plus proche d'un *chat* pour nous qu'un *chat persan* [KlRJ07], etc. Nous ne nous intéressons cependant pas aux techniques permettant de moduler, mettre en contexte, les proximités entre concepts dans cette thèse. Nous supposons donc qu'elles sont toutes identiques.

Rada *et al.* [RMBB89] utilise une métrique, indiquant le nombre d'arcs minimum à parcourir pour aller d'un concept à l'autre :

$$sim_{dist}(c_1, c_2) = \frac{1}{1 + dist(c_1, c_2)} \tag{2.1}$$

Si les concepts sont identiques, la similarité est maximale et vaut 1. Plus ils sont éloignés, plus la valeur est faible. Notons qu'il est d'ailleurs possible, si aucun chemin n'est trouvé entre c_1 et c_2, de donner une valeur nulle à la similarité.

Wu et Palmer [WP94] utilisent le concept de plus petit généralisant commun, c'est-à-dire le généralisant commun à $c1$ et $c2$ le plus éloigné de la racine :

$$sim_{W\&P}(c_1, c_2) = \frac{2 \times prof(c)}{prof(c_1) + prof(c_2)} \tag{2.2}$$

avec $prof(c_i)$ qui est une mesure de la profondeur du concept c_i, c'est-à-dire le nombre d'arêtes à la racine et c qui est le plus petit ancêtre commun à c_1 et c_2. Ce qui nous donne encore une fois, et si les concepts c_1 et c_2 sont identiques, une valeur maximale de 1, puisque le plus petit généralisant commun à c_i et c_i, c'est lui-même. De même, plus les concepts sont éloignés de leur généralisant commun, plus la mesure de similarité est faible.

Resnik [Res95] propose de plus une mesure utilisant la profondeur maximale de l'ontologie, MAX :

$$sim_{edge}(c_1, c_2) = (2 \times MAX) - dist(c_1, c_2) \tag{2.3}$$

La mesure de Leacock et Chodorow [LC98] utilise la longueur du chemin entre les deux concepts, pondérée par la profondeur de la taxonomie.

$$sim_{lea-chod}(c_1, c_2) = -log\left(\frac{dist(c_1, c_2)}{2 \times MAX}\right) \tag{2.4}$$

Bidault [Bid02] propose une solution élégante qui n'utilise que la hiérarchie des concepts, et qui est très effective et efficace, une fois un pré-traitement (numérotation des concepts) effectué.

Il numérote tous les concepts d'une hiérarchie en partant du principe que tout concept hérite des descripteurs (*i.e.* numéros) de ses parents en y ajoutant ses propres caractéristiques. Prenons une portion de hiérarchie très simple composée de cinq concepts A, B, C, D et E. Nous avons C qui hérite de A et B, D qui hérite de A, et E qui hérite de C (figure 2.11 (a)).

Supposons que les numéros de A et B sont :

- $A = 00a2$;
- $B = 0e1$.

Le concept C, héritant de ses deux parents A et B, aura donc deux descripteurs différents, l'un commençant par $00a2$, le descripteur de A et l'autre par $0e1$,le descripteur de B. De même D qui hérite de A aura un seul descripteur commençant par $00a2$. Nous pouvons donc donner les descripteurs des deux concepts C et D :

- $C = 00a21, 0e11$;
- $D = 00a22$.

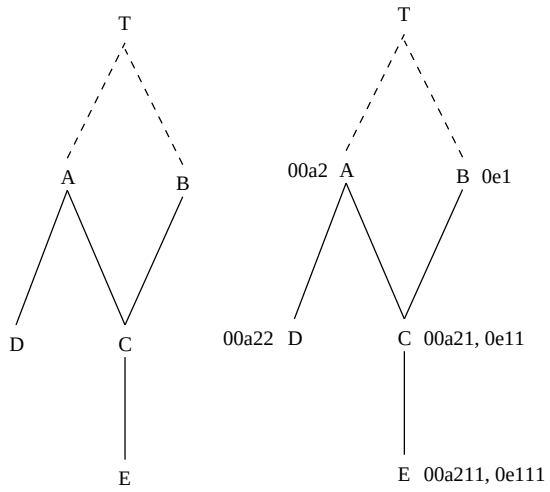


Figure 2.11 – Hiérarchie partielle permettant d'expliquer la numérotation de Bidault (a) et numérotation de Bidault (b).

De même, nous donnerons à E les descripteurs $00a211$ et $0e111$ (figure 2.11 (b)).

A partir de ces descripteurs Bidault construit les fonctions suivantes entre deux descripteurs n_i et m_j avec $i \in [1..k]$ et $j \in [1..l]$:

- com_{ij} qui permet d'obtenir la partie commune entre n_i et m_j;
- fin_{n_i} et fin_{m_j} qui retournent respectivement la partie finale[1] de n_i et de m_j.

Nous avons donc $n_i = com_{ij} + fin_{n_i}$ et $m_j = com_{ij} + fin_{m_j}$. Ces fonctions qui permettent de comparer deux descripteurs servent pour le traitement de chaînes de caractères : partie commune à deux chaînes de caractères et partie finale. Elles correspondent à un ensemble de propriétés communes et à un ensemble de différences. Le principe en est que les concepteurs de thésaurus spécialisent quand de nouvelles propriétés sont révélées, et ils différencient si des propriétés homologues mais différentes sont présentes.

Pour deux descripteurs, nous avons la note de proximité de m_j centrée sur n_i :

$$R_{m_j \to n_i} = ((|com_{ij}| + P_h - |n_i|)/P_h) - M \times |fin_{m_j}| \tag{2.5}$$

Cette mesure permet de définir la proximité du descripteur m_j par rapport à n_i. M correspond à un *malus* appliqué pour pénaliser plus ou moins les fils.

Puisque chaque concept contient plusieurs descripteurs, Bidault a généralisé sa mesure pour que nous puissions mesurer la similarité d'un concept C' centré sur un descripteur, puis d'un concept C' centré sur C :

$$R_{max_{C' \to n_i}} = MAX\{R_{m_j^p \to n_i}, p \in [1..q]\}$$

et

$$N_{C' \to C} = MOY\{R_{max_{C' \to n_i^p}}, p \in [1..q]\} \tag{2.6}$$

[1] C'est-à-dire ce qui n'est pas commun à n_i et m_j.

Approche basée sur le contenu informationnel Ce deuxième type d'approche utilise une caractéristique propre aux nœuds : le contenu informationnel (*information content*). Il existe deux versions pour le calcul du contenu informationnel.

Version avec corpus d'apprentissage Elle a été introduite par Resnik [Res95] suite aux travaux très antérieurs de Shannon [SW49]. Notons $P(c)$ la probabilité de trouver un concept c dans un corpus d'apprentissage. La probabilité associée à la racine de la hiérarchie, c'est-à-dire au concept universel $\top$ est de 1. Le contenu informationnel associé à un concept est $ic = -logP(c)$: plus le concept est générique, plus son contenu informationnel est faible, c'est-à-dire qu'il nous apporte peu d'information, alors que plus un concept est spécialisé, plus son contenu informationnel est important. En effet, le concept *private school* est plus informatif que celui de *institution*. Dans ce cadre, $\top$ a un contenu informationnel nul[2].

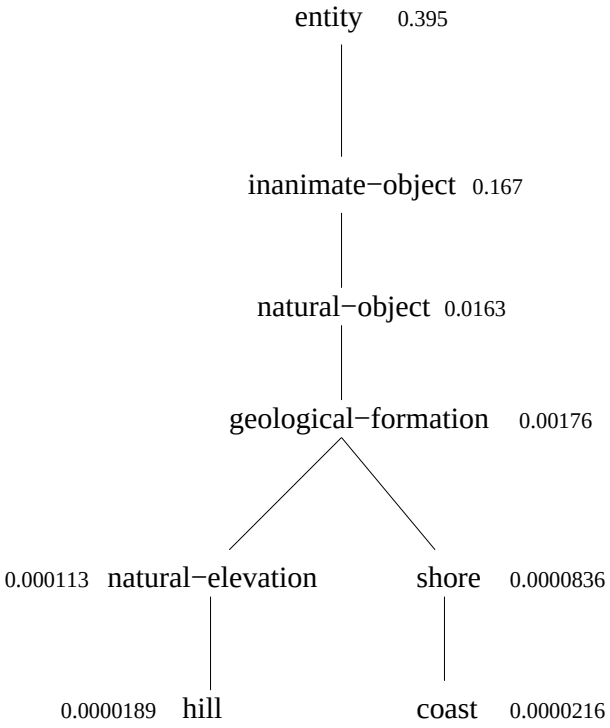


Figure 2.12 – Extrait de WordNet présenté par [Lin98] avec les probabilités correspondants aux différents concepts.

Version ne tenant compte que de l'ontologie Seco *et al.* [SVH04] pensent que l'utilisation d'un corpus d'apprentissage est un défaut de l'approche précédente. Pour eux une ontologie seule suffit à trouver le contenu informationnel des nœuds. Leur thèse est qu'il est possible de retirer de la structure de cette ontologie un sens au nombre d'hyponymes qu'a un concept : plus un concept a de descendants, plus il est spécialisé par d'autres concepts, moins il est lui-même caractéristique. Pareillement, les feuilles de la taxonomie ont une valeur informationnelle maximale, car elles sont

[2] $-logP(\top) = -log(1) = 0$. Notons aussi que si un concept n'apparaît pas dans le corpus (corpus mal constitué, trop petit, *etc.*) nous nous trouvons dans le cas $-log(0) = +\infty$. La solution la plus simple est de ne pas donner de résultat pour des concepts n'apparaissant pas dans le corpus.

les plus spécialisées. Seco *et al.* utilisent la fonction suivante pour leur calcul du contenu informationnel :

$$ic_{wn}(c) = \frac{log(\frac{hypo(c)+1}{max_{wn}})}{log(\frac{1}{max_{wn}})} = 1 - \frac{log(hypo(c)+1)}{log(max_{wn})} \tag{2.7}$$

avec $hypo(c)$ qui indique le nombre d'hyponymes dont dispose le concept c, et max_{wn} une constante qui indique le nombre de concepts de l'ontologie. Nous pouvons ajouter sur cette formule que le dénominateur, qui représente le concept avec la plus forte valeur informationnelle, permet d'assurer que les valeurs sont incluses dans l'intervalle $[0, 1]$. De plus, cette formule permet d'assurer que le contenu informationnel ainsi défini croit de façon monotone depuis la racine, qui a une valeur de 0, jusqu'aux feuilles qui ont elles $ic_{wn} = 1$ (*cf.* figure 2.13).

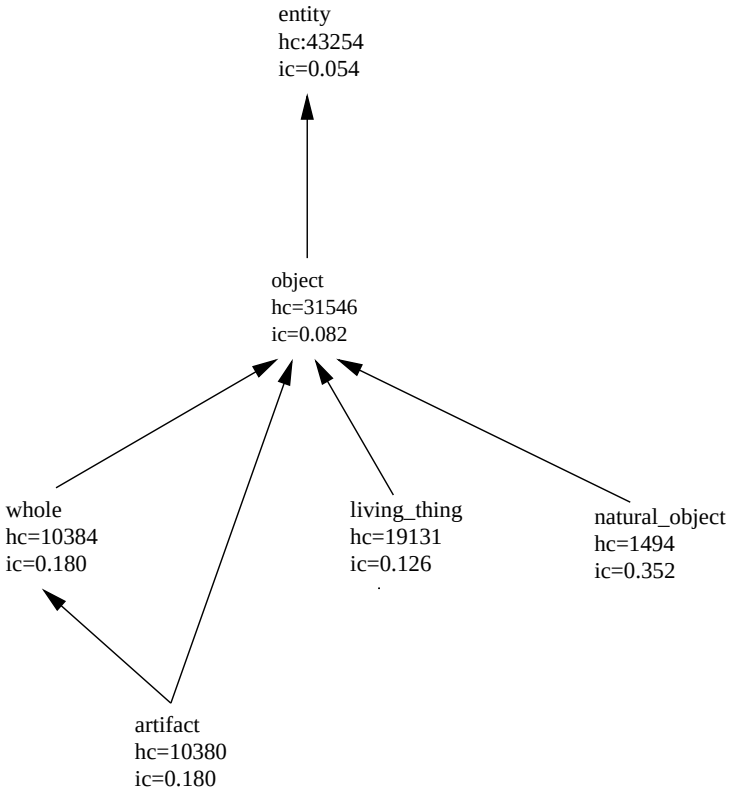


Figure 2.13 – Exemple montrant le nombre d'hyponymes (hc) et le contenu informationnel (ic) pour la mesure de Seco *et al.*.

Soit l'ensemble des concepts subsumant $c1$ et $c2$: $S(c1, c2)$. Nous avons donc, selon Resnik [Res95] et Richardson *et al.* [RSM94] :

$$sim_{resnik}(c1, c2) = \max_{c \in S(c1,c2)} [ic(c)] \tag{2.8}$$

Avec ic qui est une des deux mesures du contenu informationnel présentées précédemment. Le problème avec cette mesure est qu'elle ne différencie pas entre les descendants divers d'un concepts : le fait de descendre dans la hiérarchie n'est pas pénalisant.

Formulations hybrides A partir des deux modèles présentés auparavant, différentes approches peuvent voir le jour en combinant l'information contenue dans les noeuds et la structure de la hiérarchie. Lin [Lin98] propose de réutiliser le contenu informationnel et le plus petit ancêtre commun :

$$sim_{lin}(c_1, c_2) = \frac{2 \times logP(c)}{logP(c_1) + logP(c_2)} \tag{2.9}$$

Avec c qui est le plus petit généralisant commun à c_1 et c_2. De même pour Jiang et Conrath [JC97] :

$$sim_{jiang-conrath}(c_1, c_2) = -logP(c_1) - logP(c_2) - 2 \times (-logP(c)) \tag{2.10}$$

2.4.2.2 Comparaison des solutions et choix

Le tableau 2.3 permet d'indiquer les propriétés de différentes mesures de similarité que nous avons présentées précédemment.

Propriété \ Mesure	2.8	2.1	2.2	2.3	2.9	2.6
augmentation avec les similarités	oui	non	oui	non	oui	oui
diminution avec les différences	non	oui	oui	oui	oui	oui
symétrie	oui	oui	oui	oui	oui	**non**
inégalité triangulaire	non	oui	non	oui	non	**non**

Table 2.3 – Comparaison entre différentes mesures de similarité.

Tversky [Tve77] a montré en psychologie que la similarité sémantique ne peut pas être une distance, car elle ne satisfait pas les propriétés de symétrie et d'inégalité triangulaire. Par exemple, les deux énoncés : « un homme ressemble à un arbre » et « un arbre ressemble à un homme » ont deux sens complètement différents. De même il est possible de dire « les rugbymen fidjiens se sont battus comme des lions », mais pas (ou moins) « les lions se sont battus comme des rugbymen fidjien ». Parce qu'il y a un sens dans les jugements de similarité : la similarité sémantique n'est pas symétrique. De même, de ce que la Martinique et les Bahamas sont similaires parce que ce sont des îles Caraïbes, et de ce que les Bahamas et le Canada sont similaires parce que ce sont d'anciennes colonies britanniques, il n'est pas possible d'inférer que la Martinique et le Canada sont plus similaires que la somme des deux similarité précédentes.

Puisque nous n'avons pas besoin que la mesure de similarité que nous allons utiliser soit une distance (nous n'avons pas besoin de faire des opérations dans un espace euclidien), nous choisissons la mesure qui est la plus en adéquation avec les propriétés indiquées précédemment : celle de Bidault. Ce travail a été publié dans [Ven06b].

2.4.2.3 Améliorations de la solution choisie

Cette solution a été choisie car c'est celle qui correspond le plus à nos besoins. Cependant, elle présente encore quelques problèmes que nous allons adresser.

Nos constatations sur la mesure de Bidault sont de deux ordres :

- l'approche de Bidault ne s'intéresse qu'à l'ordre induit sur les concepts. Les valeurs de similarité n'ont aucune signification intrinsèque. Or pour notre approche ces valeurs ont une importance primordiale. Nous souhaitons donc qu'elles aient une signification propre. Par exemple, la valeur de similarité d'un concept avec son père doit être normalisée.
- nous avons remarqué que la solution proposée par Bidault ne satisfait pas une condition qui nous semble importante : la spécialisation donne un concept plus similaire que la généralisation. En conséquence, les descendants d'un concept c doivent toujours avoir une proximité plus forte que les ascendants de c ; de même les pères de c et leurs descendants doivent avoir une proximité plus forte que le premier grand-père de c, etc. (voir figure 2.14)

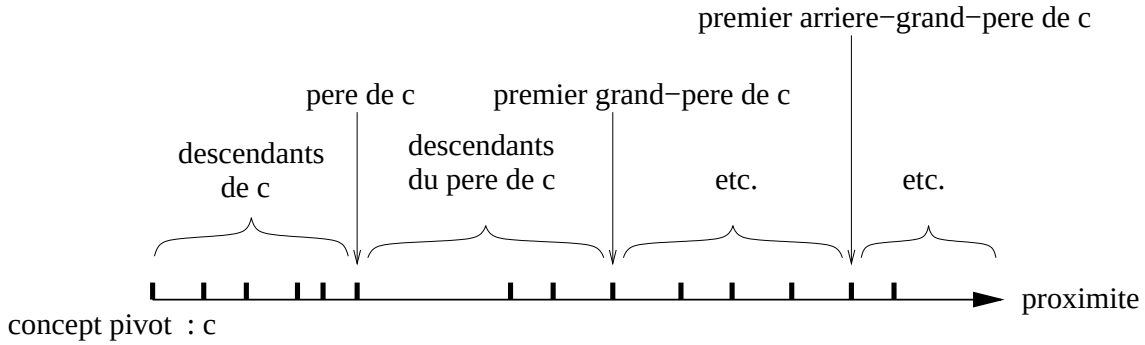


Figure 2.14 – Comment doivent être placés les concepts de l'ontologie par rapport au concept pivot.

Normalisation des mesures de proximité L'application visée par Bidault n'implique qu'une seule mesure de proximité centrée sur un seul concept. Son objectif étant de trouver les concepts les plus proches du concept pivot. Notre application nécessite plusieurs mesures centrées sur différents concepts, et ces différentes mesures doivent pouvoir interagir entre elles. Il nous faut donc une certaine cohérence des valeurs d'une mesure à une autre.

Nous proposons une normalisation à l'aide des puissances de 2. Nous utilisons les profondeurs de la hiérarchie et des identifiants pour ce faire, ce qui remplace la fonction 2.5, page 37, par celui-ci :

$$R_{m_j \to n_i} = 1 - \frac{(2^{Ph - P_{com_{ij}}+1} - 2^{Ph-P_{n_i}+1})}{2^{Ph+1}} - M \times (|m_j| - |com_{ij}|) \tag{2.11}$$

Spécialisation plus similaire que la généralisation D'un point de vue technique, c'est en modifiant le malus du calcul de Bidault que nous allons obtenir les propriétés suivantes :

- tout descendant du concept pivot est plus proche de ce dernier qu'un de ses ascendants;
- tout descendant d'un ascendant au niveau n du concept pivot est plus proche du concept pivot qu'un ascendant situé au niveau $n+1$ au minimum : par exemple tout descendant du père du concept pivot est plus proche de ce dernier que ses autres ascendants.

Soit la fonction suivante définie par Bidault :

$$R_{m_j \to n_i} = \frac{|com_{ij}| + P_h - |n_i|}{P_h} - M \times |fin_{m_j}|$$

Nous avons n_i le descripteur du concept pivot. Supposons que m_j décrivent un fils de n_i. Nous avons alors $|com_{ij}| = |n_i|$, puisque la partie commune aux deux identifiants, leur parent commun, est $|n_i|$.

Donc :

$$R_{m_j \to n_i} = \frac{|n_i| + P_h - |n_i|}{P_h} - M \times |fin_{m_j}|$$
$$R_{m_j \to n_i} = \frac{P_h}{P_h} - M \times |fin_{m_j}|$$
$$R_{m_j \to n_i} = 1 - M \times |fin_{m_j}|$$

avec m_j un fils de n_i.

Soit maintenant m_j un ancêtre de n_i. Nous avons alors $|fin_{m_j}| = 0$, car la partie commune $|com_{ij}|$ aux deux est m_j, qui n'a donc pas de partie finale.

$$R_{m_j \to n_i} = \frac{|com_{ij}| + P_h - |n_i|}{P_h}$$

De plus, com_{ij} est contenue dans n_i, puisque com_{ij} est l'ancêtre de n_i : m_j. Ce qui nous fait :

$$com_{ij} - n_i = niveau_{m_j}$$

avec $niveau_{m_j}$ le niveau d'ascendance (pour le père : niveau 1, pour le grand père niveau 2 : *etc.*) de m_j par rapport à n_i.

$$R_{m_j \to n_i} = \frac{P_h - niveau_{m_j}}{P_h}$$
$$R_{m_j \to n_i} = 1 - \frac{niveau}{P_h}$$

Ce qui nous donne l'idée de généraliser la relation que nous avions pour un fils du concept pivot. En effet, pour un descendant d'un ancêtre de niveau quelconque nous avons les même réflexions que pour un ancêtre et que pour un fils du concept pivot :

$$R_{m_j \to n_i} = 1 - \frac{niveau}{P_h} - M \times |fin_{m_j}|$$

Ce qui nous donne bien $R_{m_j \to n_i} = 1 - M \times |fin_{m_j}|$ pour un fils du concept pivot ($niveau = 0$).

Nous voulons que la relation suivante soit toujours vérifiée entre le fils d'un ascendant de niveau $niveau$ et un ascendant de niveau $niveau + 1$

$$\begin{aligned} 1 - \frac{niveau}{P_h} - M \times |fin_{m_j}| &> 1 - \frac{niveau + 1}{P_h} \\ -M \times |fin_{m_j}| &> -\frac{1}{P_h} \\ M \times |fin_{m_j}| &< \frac{1}{P_h} \\ M &< \frac{1}{P_h \times |fin_{m_j}|} \end{aligned}$$

Nous avons bien évidemment la relation suivante pour tous identifiants m_j et n_i :

$$Ph \geq |fin_{m_j}| \geq 0$$

($fin_{m_j} = Ph$ si $|n_i| = 0$ et $|m_j| = Ph$, et $fin_{m_j} = 0$ si $|n_i| = |m_j|$).

Nous prenons pour nous :

$$M = \frac{1}{P_h^2}$$

Ce qui est une valeur supérieure au 0.002 de Bidault et va nous permettre de « ventiler » les concepts.

CHAPITRE 3

Evaluations et discussions

L'objectif de ce chapitre est double : (1) évaluer l'indexation sémantique choisie et (2) évaluer expérimentalement l'approche EXSID. Ce chapitre s'organise donc en deux sections : évaluation de la similarité sémantique proposée, puis évaluation de l'approche EXSID.

3.1 Evaluation expérimentale des mesures de similarité sémantique

3.1.1 Contexte d'évaluation

Rubenstein et Goodenough [RG65] ont les premiers mis en place des tests sur la similarité sémantique de termes. Ils ont pour ce faire donné une valeur de similarité à 65 paires de termes avec l'aide d'experts humains. D'autres tests ont suivi, parmi lesquels celui de Miller et Charles [MC91]. Il s'agit d'un sous-ensemble des paires de mots de Rubenstein et Goodenough, dont un groupe de 38 étudiants a noté la similarité.

Nous utilisons le thésaurus WordNet [Fel98, www.26]. Il est composé d'une hiérarchie de « concepts »; Ces derniers étant en fait des ensembles de synonymes, des *synsets*. Par exemple, il existe sept synsets qui font intervenir le terme *cat* dans WordNet

- *cat, true cat – (feline mammal usually having thick soft fur and being unable to roar; domestic cats; wildcats)*;
- *guy, cat, hombre, bozo – (an informal term for a youth or man; "a nice guy"; "the guy's only doing it for some doll")*;
- *cat – (a spiteful woman gossip; "what a cat she is!")*;
- *cat-o'-nine-tails, cat – (a whip with nine knotted cords; "British sailors feared the cat")*;
- *Caterpillar, cat – ((trademark) a large vehicle that is driven by caterpillar tracks; frequently used for moving earth in construction and farm work)*;
- *big cat, cat – (any of several large cats typically able to roar and living in the wild)*;
- *computerized tomography, computed tomography, CT, computerized axial tomography, computed axial tomography, CAT – (a method of examining body organs by scanning them with X rays and using a computer to construct a series of cross-sectional scans along a single axis)*.

C'est une ontologie reconnue par la communauté TALN, et en particulier, elle est très utilisée pour les mesures de similarité sémantique. Pour simuler les mesures de pertinence que nous allons tester, nous avons utilisé deux librairies :

- le package WordNet::Similarity [www.8], écrit en Perl, qui implémente certaines des mesures de similarité que nous avons étudiées ;

- la librairie Java d'accès à WordNet, JWNL [WW11], pour le test de notre propre mesure.

3.1.2 Méthodes de référence

Les différentes mesures de similarité que nous avons testées sont :
- B : la mesure de Leacock et Chodorow (cf. équation 2.4 page 36);
- C : la mesure de Jiang et Conrath (cf. équation 2.10 page 40) ;
- D : la mesure de Lin (cf. équation 2.9 page 40) ;
- E : la mesure de Wu et Palmer (cf. équation 2.2 page 36) ;
- F : la mesure de Resnik utilisant le contenu informatif de Seco (cf. équation 2.8 page 39) ;
- G : la mesure de Bidault (cf. équation 2.6 page 37) ;
- H : notre mesure.

3.1.3 Résultats d'évaluation

La table 3.1 donne les résultats pour la plupart des paires de concepts de l'expérimentation de Miller et Charles. La colonne A correspond aux évaluations manuelles. Nous pouvons y voir que notre solution a des résultats qui sont parmi les meilleurs.

3.1.4 Discussion

D'autres évaluations, par exemple WordSimilarity-353 de Finkelstein *et al.* [FGM+01, WW10] qui contient au total 353 paires de mots dont la similarité a été jugée par 13 à 16 sujets humains. Nous ne l'avons pas utilisée car le jeu de test de Miller et Charles est plus connu (cela tend à changer) et que nous voulions seulement montrer que notre solution a des résultats proches de ceux des mesures de similarité classiques.

Notre mesure est symétrique. Nous avons pris les termes dans le même sens que celui du test de Miller et Charles. Il nous aurait été possible d'inverser les concepts, pour regarder la différence que nous aurions obtenue. Mais, outre que la mesure de référence, celle tirée des jugements humains, est aussi orientée, et que nous n'aurions pas su que faire de nos résultats, il aurait été difficile pour nous d'exploiter ces résultats. En effet, nous ne sommes ni du TALN, ni de l'IC.

3.2 Evaluation expérimentale d'ExSID

Dans cette section, nous mettons en place une évaluation d'ExSID, avec pour mesures de référence le cosinus et l'expansion (classique). Nous voyons qu'il est nécessaire de mettre en place une indexation des documents et des requêtes d'un corpus de test, de choisir une ontologie, une mesure de similarité comme nous le décrivons dans la section précédente et une fonction de propagation. Ces différents paramètres ont un impact sur les résultats de notre solution, mais pas sur la définition de notre solution.

3.2.1 Méthodes de référence

Les méthodes de référence sont le cosinus et l'expansion. Le cosinus est non seulement la mesure généralement utilisée avec le modèle vectoriel en RI, mais aussi le choix que nous avons fait pour notre mesure de pertinence. En effet, ExSID est générique, elle peut être utilisée avec différentes mesures de

paire de mots		A	B	C	D	E	F	G	H
car	automobile	3.92	3.47	0.00	1.00	0.89	0.68	0.937	0.937
gem	jewel	3.84	3.47	0.00	1.00	0.86	1.00	0.937	0.937
journey	voyage	3.84	2.77	4.95	0.69	0.92	0.66	0.937	0.937
boy	lad	3.76	2.77	3.41	0.82	0.80	0.76	0.937	0.937
coast	shore	3.70	2.77	0.62	0.97	0.91	0.78	0.875	0.875
asylum	madhouse	3.61	2.77	0.41	0.98	0.82	0.94	0.875	0.875
magician	wizard	3.50	3.47	0.00	1.00	0.80	0.80	0.875	0.875
midday	noon	3.42	3.47	0.00	1.00	0.88	1.00	0.875	0.875
furnace	stove	3.11	1.39	18.13	0.220	0.46	0.18	0.750	0.750
food	fruit	3.08	1.39	11.65	0.13	0.22	0.05	0.719	0.719
bird	cock	3.05	2.77	3.76	0.80	0.94	0.40	0.719	0.719
bird	crane	2.97	2.08	*	*	0.84	0.40	0.937	0.937
tool	implement	2.95	2.77	1.23	0.92	0.91	0.42	0.937	0.937
brother	monk	2.82	2.77	14.90	0.25	0.92	0.18	0.469	0.469
lad	brother	1.66	1.86	12.47	0.29	0.60	0.18	0.781	0.781
journey	car	1.16	0.83	11.93	0.00	0.00	0.00	0.00	0.00
monk	oracle	1.10	1.39	17.42	0.23	0.46	0.18	0.656	0.656
food	rooster	0.89	0.83	15.19	0.10	0.13	0.05	0.687	0.687
coast	hill	0.87	1.86	5.37	0.71	0.67	0.50	0.687	0.687
monk	slave	0.55	1.86	15.52	0.25	0.44	0.08	0.469	0.469
coast	forest	0.42	1.52	17.60	0.13	0.60	0.18	0.844	0.844
lad	wizard	0.42	1.86	13.60	0.27	0.40	0.08	0.625	0.625
chord	smile	0.13	1.07	14.86	0.27	0.60	0.18	0.750	0.750
glass	magician	0.11	1.39	18.07	0.13	0.36	0.18	0.00	0.00
noon	string	0.08	0.98	18.32	0.00	0.00	0.00	0.00	0.00
rooster	voyage	0.08	0.47	21.61	0.00	0.00	0.00	0.00	0.00
corrélation		1.00	0.82	0.81	0.80	0.74	0.77	0.82	0.82

Table 3.1 – Comparaison de différentes mesures de similarité sur un exemple tiré de [MC91].

pertinence. Puisque nous avons choisi d'utiliser le cosinus, nous devons étudier l'impact de l'ajout de l'expansion structurante et de l'image au cosinus. D'autre part, nous voulons savoir si EXSID permet un gain par rapport à l'expansion, qui est la solution courante aux problèmes que nous avons traités au chapitre précédent.

3.2.2 Contexte d'évaluation

3.2.2.1 Ontologie

Nous utilisons WordNet pour nos expérimentations. Comme nous l'avons vu, il s'agit d'une ontologie « légère » au sens de Gomez-Pérez *et al.* [GPFC04], c'est-à-dire une ontologie composée d'une taxonomie de concepts et quelques relations (méronymie, antonymie, etc.), mais qui ne sont pas très importantes. Malgré sa « légèreté » c'est l'ontologie la plus complète pour l'anglais. En recherche d'information il y a eu un débat pour savoir si WordNet est acceptable pour les expérimentations (voir par

exemple les discussions de Voorhees [Voo94], Sanderson [San00] et Stokoe *et al.* [SOT03]). Cependant, d'autres travaux influents ont montré qu'il est possible d'utiliser WordNet, et parfois d'autres ressources, et d'avoir de bons résultats. Gonzalo *et al.* [GVCC98] prouvent ainsi qu'une relativement pauvre désambiguïsation de mots (jusqu'à 60% d'erreurs dans l'indexation) conduit à de meilleurs résultats dans la recherche d'information que l'utilisation de mots-clés, et une amélioration globale de 29% dans tous les cas.

Pour nos expérimentations, WordNet est de toute façon suffisant, parce que nous ne recherchons pas une indexation parfaite, ni le meilleur système de recherche d'information, juste une comparaison entre des mesures de pertinence. La version de WordNet que nous utilisons est la 1.7.1. Dans cette version, WordNet a 81 426 synsets dans la hiérarchie nominale (il y a aussi des hiérarchies pour les verbes, les adjectifs et les adverbes), avec 117 097 termes apparaissant dans ces synsets, ce qui fait une polysémie moyenne de 1,23 par terme.

3.2.2.2 Corpus utilisé : Cranfield

Nous utilisons le corpus Cranfield pour nos expérimentations (cf. section 1.2.3.4). Il s'agit d'un corpus spécialisé, les documents étant composés de résumés d'articles de recherche du domaine de l'aéronautique. Vous pouvez voir un document du corpus dans la figure 3.1 et une requête dans la figure 3.2.

3.2.2.3 Indexation avec RIIO

Nous utilisons dans nos expérimentations un programme d'indexation développé dans notre laboratoire par une équipe de Traitement Automatique du Langage Naturel : RIIO de Desmontils et Jacquin [DJ02]. Ce programme utilise l'étiqueteur de Brill [Bri95], un lemmatiseur et élimine les mots vides. Puis il recherche dans WordNet les synsets les plus proches des termes choisis précédemment. La pondération des termes se fait seulement par une fréquence d'apparition des termes dans un document et par une méthode d'analyse de synsets représentatifs, utilisant la fonction de Wu et Palmer [WP94]. Nous avons adapté leur pré-traitement au corpus Cranfield (formatage des documents et des requêtes) et l'indexation sémantique au domaine considéré (aéronautique) pour éviter des indexations trop bruitées, en particulier au niveau des requêtes. Pour ce faire, nous avons utilisé une base de connaissance recensant les sens principaux du corpus. Ainsi les termes les plus souvent utilisés obtiennent les sens qui doivent leur être le plus fréquemment attribués. L'algorithme 2 est l'algorithme que nous avons utilisé pour chercher les sens probables pour chaque terme sorti de notre lemmatiseur pour un document (ou une requête). Il permet une désambiguïsation simple, les sens évidents étant choisis prioritairement (pour une étude détaillée de la désambiguisation, voir l'article de Ide et Véronis [IV98]). Pour les autres cas, nous utilisons la fonction de Wu et Palmer pour rechercher le sens le plus proche d'un sens déjà présent.

Pour mesurer l'efficacité du système RIIO, nous avons utilisé la campagne Senseval2 [W3C], en particulier la piste (*track*, thème d'expérimentation) « English all-words » qui vise à choisir un synset parmi ceux de WordNet 1.7 pour désambiguïser 2500 mots d'un texte. Les résultats de notre indexation sont assez faibles : 26.2% en rappel, 15.7% en précision. Néanmoins, il faut noter que l'outil n'est pas paramétré pour la piste choisie. Leurs objectifs sont différents, en ce sens que Senseval cherche une désambiguïsation de chaque terme alors que nous désirons seulement obtenir l'ensemble des concepts importants d'un document ou d'une requête. Il faut néanmoins noter que RIIO n'est pas adapté à Senseval et cherche plutôt à trouver des concepts représentatifs d'un document qu'à désambiguïser des termes.

RIIO choisit en moyenne 5 concepts pour les requêtes et 23 pour les documents. Il est aussi à noter

```
<DOC>
<DOCNO>
1
</DOCNO>
<TITLE>
experimental investigation of the aerodynamics of a
wing in a slipstream .
</TITLE>
<AUTHOR>
brenckman,m.
</AUTHOR>
<BIBLIO>
j. ae. scs. 25, 1958, 324.
</BIBLIO>
<TEXT>
  an experimental study of a wing in a propeller slipstream was
made in order to determine the spanwise distribution of the lift
increase due to slipstream at different angles of attack of the wing
and at different free stream to slipstream velocity ratios .  the
results were intended in part as an evaluation basis for different
theoretical treatments of this problem .
  the comparative span loading curves, together with supporting
evidence, showed that a substantial part of the lift increment
produced by the slipstream was due to a /destalling/ or
boundary-layer-control effect .  the integrated remaining lift increment,
after subtracting this destalling lift, was found to agree
well with a potential flow theory .
  an empirical evaluation of the destalling effects was made for
the specific configuration of the experiment .
</TEXT>
</DOC>
```

Figure 3.1 – Document numéro un du corpus Cranfield.

```
<query>what similarity laws must be obeyed when constructing aeroelastic
models of heated high speed aircraft .</query>
```

Figure 3.2 – Requête numéro un du corpus Cranfield.

que seuls 765 concepts sont choisis pour indexer les documents. Cela provient pour partie de notre algorithme, mais lui-même utilise le fait que les documents appartiennent tous à un domaine très spécialisé.

Algorithme 2 : Désambiguïsation sémantique

entrées : Ensemble $\mathcal{W}$ des termes du document passés par un lemmatiseur; base de sens KS.
sortie : Ensemble de sens (synsets) représentatifs du document.
begin
  pour tous les $w_i \in \mathcal{W}$ **faire**
    pour tous les *sens possible* s_j **faire**
      si *il est unique* **alors**
        choisir(s_j);
      sinon si *il a déjà été relevé dans le document* **alors**
        augmenter(s_j);
      sinon si $s_j \in KS$ **alors**
        choisir(s_j);
      sinon si *un autre terme de* s_j *est dans le document* **alors**
        choisir(s_j);
  pour tous les *termes* w_i *du document sans sens choisi* **faire**
    pour tous les *sens possibles* s_j **faire**
      $poids_j \leftarrow Wu\&Palmer(s_j, sensChoisis + KS)$
    choisir(best($s_j, poids_j$));
end

3.2.3 Paramètres de la méthode ExSID choisis pour l'expérimentation

La mesure de similarité choisie pour nos expérimentations est celle que nous avons présenté précédemment.

En ce qui concerne la fonction de propagation, nous avons eu deux choix à faire : quelle « forme » doit-elle avoir ? quelle longueur totale doit-elle avoir ?

Nous avons besoin de pouvoir juger dans nos simulations de l'impact de la propagation par rapport au nombre de concepts impliqués dans la propagation. Nous voulons savoir ce qui se passe si x concepts sont pondérés par la propagation. La mesure de similarité sémantique ne nous donne pas directement cette valeur. Nous avons alors modifié notre fonction de propagation de sorte à ce qu'elle propage sur un ensemble de « classes d'équivalence » de concepts ayant une même valeur de similarité par rapport au concept central considéré. Ainsi il nous est possible de propager sur un certain nombre de classes, et d'obtenir des nombres de concepts pondérés dans chaque DSE à peu près équivalents pour chaque expérimentation.

En partant des classes de fonctions de propagation que nous avons définies en section 2.2.1, nous avons testé trois types différents de propagations :

- « carrée » (cf. figure 3.4 (a)) : du type $f_{l_1,0}$, où seul un ensemble de concepts sont pondérés, avec la valeur du concept central : ceux dont la valeur de similarité avec le concept central de la propagation est supérieure à l_1 ;
- « pentue » (cf. figure 3.4 (b)) : du type f_{0,l_2}, où les concepts dont la valeur de similarité est située au-delà de l_2 ont une pondération inférieure à 1 et décroissante avec une similarité décroissante ;
- « hybride » (cf. figure 3.4 (c)): du type f_{l_1,l_2} avec $l_1 = l_2$.

Nous avons testé ces trois types de fonctions de propagation sur une longueur de propagation moyenne de 5, 10 et 20 concepts (cf. figures 6.2 (a) et (b)). Nous pouvons noter que les fonctions de propagation

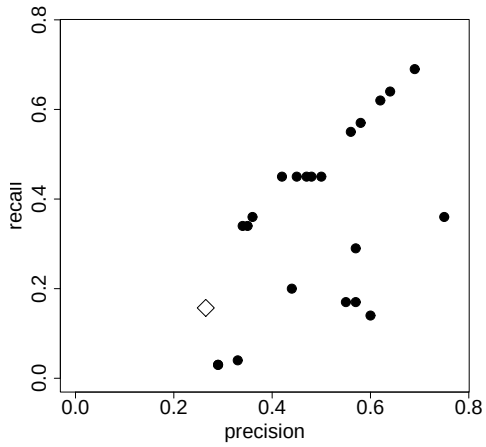


Figure 3.3 – Rappel et précision pour les différents systèmes ayant participé officiellement à la piste « English all-word » de la campagne SENSEVAL2 (rond noirs) et le système que nous avons choisi (losange).

pentues ont des résultats légèrement meilleurs que les autres, même si les différences sont faibles. C'est donc celle-ci que nous allons choisir.

3.2.4 Résultats d'expérimentation

Les figures 3.6 (a) et (b) présentent les résultats de l'expansion et d'EXSID en ratio de précision et de rappel par rapport au cosinus suivant le nombre de concepts ajoutés pour chaque concept central de la requête. Nous voyons que notre solution fait gagner un peu en précision (1%) et en rappel (3%) par rapport au cosinus, et qu'elle décroit moins vite que l'expansion lorsque des concepts sont ajoutés aux dimensions d'origine de la requête.

3.2.5 Discussion

Tout d'abord, il nous faut noter le manque de jeu de test classique pour les solutions de RI sémantiques. Il n'existe pas à notre connaissance de corpus de test indexé sémantiquement sur une ontologie de référence. Nous l'avons donc créé nous-même en utilisant un jeu de test de RI, un indexeur sémantique et une ontologie. Cela nous a permis de tester notre solution. Au cours des tests, il nous est apparu que la mesure de similarité sémantique et la fonction de propagation sont des paramètres essentiels pour obtenir de bons résultats. Nous ne sommes pas certains d'avoir opté pour les meilleurs valeurs. Néanmoins, notre solution a l'avantage d'être générique, et d'accepter une grande variété de paramètre, qui peuvent être modulés suivant le contexte, l'application, etc. Ainsi la propagation peut être plus ou moins forte, toucher plus ou moins de concepts, suivant que l'utilisateur cherche des documents précis sur sa requête ou généralistes, etc. Notre approche nécessiterait plus de tests, avec d'autres similarités sémantiques, d'autres fonctions de propagation, d'autres jeux de test, sans doute aussi d'autres indexations.

Mais le propos n'était pas ici de mettre en place un système de RI sémantique très efficace. Il s'agissait juste de montrer que EXSID n'a pas d'impact négatif sur les résultats. Or nous apercevons un léger

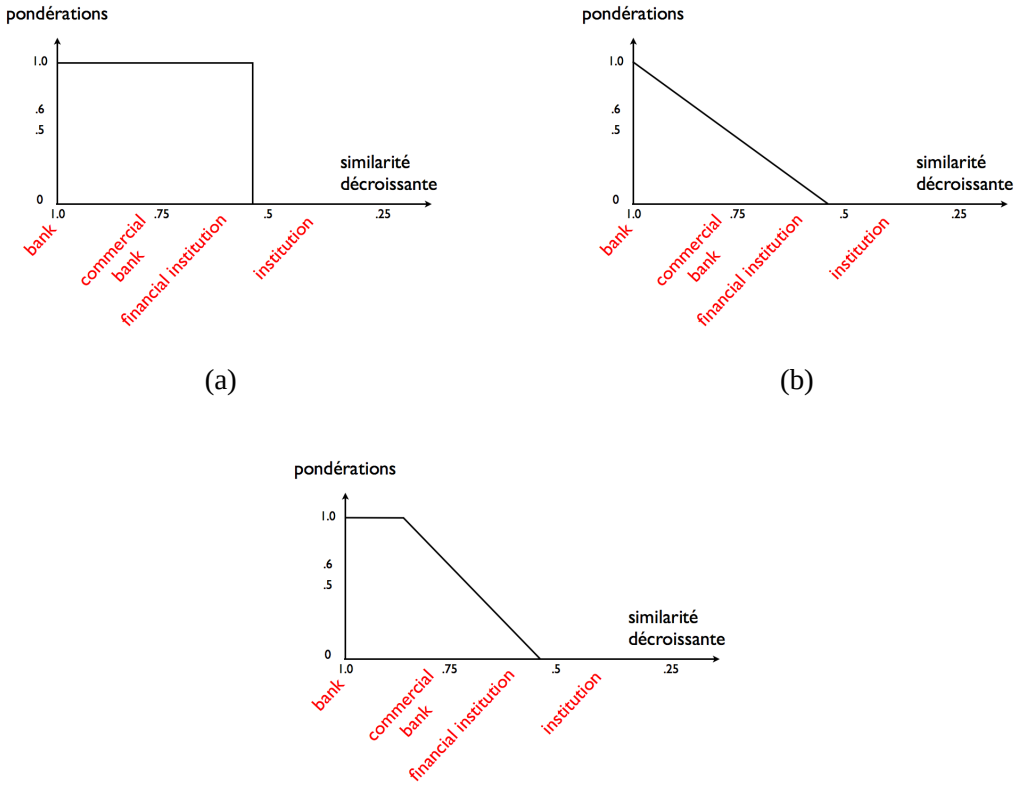
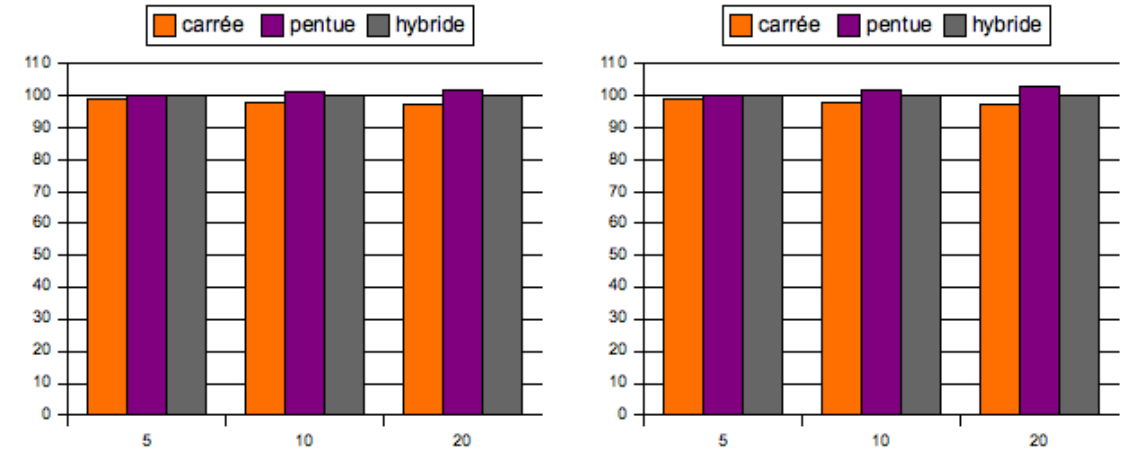


Figure 3.4 – Fonctions de propagation « carrée » (a), « pentue » (b) et « hybride » (c).

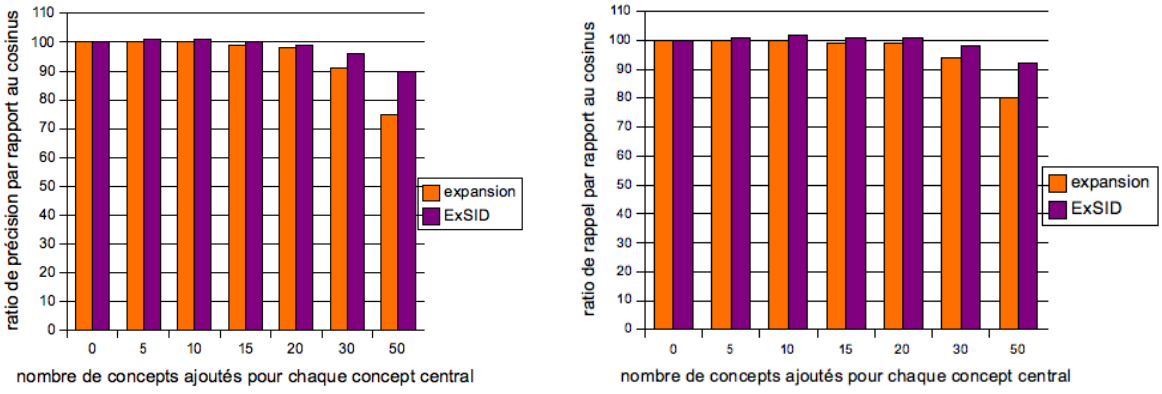
gain qui mériterait certes plus de tests pour voir s'il est systématique, mais qui contraste avec l'effet nul ou pénalisant de l'expansion. Il faut d'ailleurs noter [Voo94, MM00b] que l'expansion peut obtenir de bons résultats, si elle est effectuée de manière bien dirigée, par exemple par un utilisateur. Notre solution réussit à obtenir un petit gain par rapport au cosinus, même lorsqu'elle est automatique.

Notons de plus que notre approche peut s'appliquer à tous les processus utilisant les vecteurs sémantiques sans modifier ni leur indexation ni leur calcul de pertinence. Il est par exemple envisageable d'intégrer EXSID à un moteur de recherche.



(a) (b)

Figure 3.5 – Effets de la forme de la fonction de propagation sur le ratio de précision (a) et de rappel (b) par rapport à la propagation hybride.



(a) (b)

Figure 3.6 – (a) ratio de précision par rapport au cosinus suivant le nombre de concepts ajoutés pour chaque concept central de la requête ; (b) ratio de rappel par rapport au cosinus suivant le nombre de concepts ajoutés pour chaque concept central de la requête.

PARTIE II

Contexte d'hétérogénéité sémantique : apports de l'interprétation

CHAPITRE 4

Des connaissances diverses obligent à une intégration sémantique

Dans ce chapitre, nous soulignons rapidement qu'il est très difficile de faire l'hypothèse d'une ontologie universelle, ainsi que d'une ontologie unique convenant à l'ensemble des participants d'un système largement distribué. Ainsi, il est plus réaliste de considérer la présence de différentes ontologies. Ceci pose le problème de l'hétérogénéité sémantique. Dans un deuxième temps, nous montrons que la découverte de correspondances entre ontologies permet dans une certaine mesure d'assurer une intégration sémantique.

4.1 Le rêve d'une ontologie universelle

4.1.1 Que signifie « ontologie universelle » ?

L'ontologie est définie en philosophie comme la science de « l'être en tant qu'être ». C'est-à-dire une science consacrée à la question très générale : « qu'est-ce que l'être ? », qui est transversale à de nombreux champs philosophiques, mais qui intéresse principalement la métaphysique. Le champ des recherches sur l'être est immense et complexe. C'est en effet un donné immédiat, mais qui se définit beaucoup plus mal qu'il ne se ressent.

Les informaticiens ont repris cette idée de réflexion sur l'existant, et l'utilisent pour formaliser des connaissances. La différence essentielle entre les deux est que les informaticiens construisent des ontologies, sans parfois se poser toutes les questions qui intéressent les philosophes. Par exemple, la question des entités premières n'est pas très débattue en informatique, alors qu'elle est essentielle en philosophie. Finalement, les informaticiens ont prouvé qu'il est possible de construire de grandes ontologies (WordNet, CYC, etc.) en laissant de côté certaines questions abordées dans les discussions philosophiques.

Les deux définitions (philosophique, informatique) ont en commun de représenter entités, idées, événements, ainsi que leurs propriétés et relations. Certaines questions sont par ailleurs transversales aux deux champs : relativité ontologique (Husserl [Hus84], Quine [Qui77], Kripke [Eng85] en philosophie [Ros95], Sowa [Sow]) ou les débats sur une ontologie unique.

Cette question de l'ontologie universelle, de haut niveau, est commune aux informaticiens et aux philosophes. Il s'agit d'une formalisation commune à tous, qui représenterait « la réalité ». Cette idée repose sur un rationalisme qui donnerait à la raison humaine, également partagée, les mêmes connaissances de base ; ou en tout cas la capacité de comprendre et de partager ces connaissances de base : si un Inuit

et un Zoulou ne semblent pas avoir la même compréhension du monde, ils sont en fait structurellement semblables, et il est possible d'arriver à un consensus raisonnable entre leurs connaissances.

4.1.2 Utilité et viabilité d'une ontologie universelle

Une ontologie universelle est le meilleur moyen de pouvoir baser toutes les connaissances et tous les raisonnements humains sur la même formalisation. Cela permettrait de plus de décrire (indexer) les informations sur le même référentiel. Une ontologie unique et universelle est donc un gage de simplicité dans les recherches d'information. Au nom du principe d'Occam (dit « rasoir d'Occam »), qui demande de ne pas multiplier les entités, il paraît raisonnable de n'utiliser qu'une seule ontologie. De plus, une ontologie universelle est une solution simple à l'interopérabilité sémantique. Il s'agit en effet d'un référentiel unique sur lequel les différentes ontologies peuvent se lier.

Il existe différents projets de création d'ontologies universelles, parmi lesquels :

- Cyc [LGP+90] est un projet visant à reproduire le raisonnement humain. Pour ce faire, ils utilisent une « [...] ontologie de haut niveau dont le domaine est tout ce qui fait humainement consensus dans la réalité[1] ». Ce projet est souvent critiqué pour sa complexité, son prix, les efforts déployés, et beaucoup doutent de son évolutivité [ww12] ;
- Generalized Upper Model [BHR95, ww13] est une initiative pour développer une ontologie multilingue pour la linguistique. Elle devrait permettre d'analyser et de générer des documents en langage naturel, d'effectuer des traductions, etc. ;
- SENSUS [KL94] est une autre ontologie de très haut-niveau construite pour le traitement du langage naturel, puis pour le traitement de données en langage naturel ;
- WordNet [Fel98, MBF+90] a la même origine que les deux projets précédents, mais est surtout devenu un projet essentiel pour le traitement de la langue anglaise en RI, utilisant la sémantique [RS95a, GVCC98]. Il est cependant critiqué pour ne pas toujours clairement séparer le niveau conceptuel et le niveau terminologique [UG96]. Des versions de WordNet sont en développement dans d'autres langues, par exemple EuroWordNet [ww2] pour les langues de la communauté européenne ou BalkaNet pour certaines langues d'Europe orientale [SOP+02].

Dans tous les cas, il s'agit de projets lourds développés depuis plusieurs années, dont l'évolution est parfois compliquée, coûtant cher, etc. Ils ne connaissent finalement pas un franc succès et seule WordNet, la plus connue de ces ontologies universelles, est assez couramment utilisée dans des travaux de recherche en informatique.

4.1.3 Difficultés à concevoir une ontologie unique

L'ontologie n'est cependant pas la réalité : ce n'est qu'une représentation subjective de celle-ci. Comme l'indiquent Rousset [Rou02] et Hendler [Hen02], différents créateurs d'ontologies peuvent avoir différents points de vue sur différentes ontologisations. Ainsi, il n'est pas possible de supposer que tout le monde soit d'accord sur la façon de formaliser tel ou tel concept, telle ou telle relation.

Il existe de nombreuses méthodologies de construction d'une ontologie pour faire face à ces problèmes. Quelques-unes traitent l'ensemble du développement d'une ontologie [FGJ97] en s'inspirant parfois même d'une longue expérience de tel développement. Par exemple, dans le domaine de la gestion d'entreprise [GF95, SSS02, UK95], pour lequel la construction de bases de connaissances est un

[1] « The entire Cyc ontology containing hundreds of thousands of terms, along with millions of assertions relating the terms to each other, forming an upper ontology whose domain is all of human consensus reality ».

sujet de recherche et de développement déjà ancien. D'autres sont plus spécifiques à une ou plusieurs étapes du développement : comme la conceptualisation [AGBS00] ou l'ontologisation [Kas02]. Pour plus de détails, voir Corcho *et al.* [CFLGP03].

Dans tous les cas le processus apparaît comme long et collaboratif : il fait intervenir des experts du domaine de connaissance, des utilisateurs, des ingénieurs de la connaissance, etc. Comme nous le voyons, il s'agit d'un projet complexe, comparable au développement de n'importe quel autre projet informatique, à la différence près que le résultat de la création d'une ontologie ne peut pas se mesurer aussi « facilement » qu'avec un projet informatique classique (réussite/échec, bugge/fonctionne, etc.). Ici, c'est l'adéquation de la conception à la perception des utilisateurs qui est recherchée.

Une autre conclusion de ces études sur la sémantique, qui est un peu ancienne, est que le processus de développement des ontologies ne peut pas être ouvert aux utilisateurs finaux facilement, sous peine de générer des ontologies mal faites, voir inutilisables, incohérentes, etc. L'idée d'une ontologie universelle n'est qu'un prolongement de ce raisonnement. Cependant, ce n'est pas le paradigme qui émerge de l'utilisation récente de la sémantique sur le Web. Il s'agit plutôt, comme l'expriment les contributeurs à Staab [Sta02], de s'approcher du modèle classique du Web : celui de HTTP/HTML. Différents utilisateurs définissent des ontologies, les réutilisent, y font référence, etc. Les Webmasters de demain connaissent certaines ontologies, les utilisent pour indexer leurs contenus, en développent de nouvelles pour leurs besoins, étendent les existantes, etc.

Dans cette vision, l'usage à large échelle et de manière distribuée des ontologies implique qu'il n'est pas possible d'assurer certains principes des méthodologies actuelles : consistance, cohérence, accès assuré, etc. Les problèmes sont nombreux, mais la liberté de création et d'évolution doit être laissée aux développeurs pour qu'ils s'approprient ces outils. Et il n'est pas possible d'imposer une ontologie universelle. Le travail porte principalement sur la définition d'un langage adéquat pour décrire les ontologies et y accéder.

4.2 Correspondances entre ontologies

L'*hétérogénéité sémantique* dans un système largement distribué se caractérise par l'utilisation de différentes ontologies par les différents acteurs du système. De ce fait, pour un même concept les définitions peuvent varier. Par exemple, considérons une recherche sur le concept *author* dans Swoogle [www21] (cf. figure 4.1). Swoogle est un moteur de recherche pour le Web sémantique. Il permet de faire des recherches sur les ontologies. La recherche renvoie 3 137 résultats qui montrent que la modélisation de ce concept peut varier. Ces résultats sont tous représentatifs de différents points de vue, différentes définitions. Évidemment, nous retrouvons dans tous les cas des points communs. Une autre conséquence de l'hétérogénéité sémantique est qu'il n'est pas possible pour un pair fournisseur d'information de toujours comprendre les requêtes qui lui sont passées. Par exemple, si une requête porte sur le concept *chat siamois* qui est inconnu du fournisseur, il ne lui est pas possible de répondre.

Le travail consistant à « combler » les différences entre les ontologies est appelé *intégration sémantique*. Il a pour but de permettre :

- de poser des requêtes sur différentes sources : par exemple, en interrogeant une ontologie, il doit pouvoir être possible d'obtenir des réponses d'autres ontologies « proches »;
- de transformer les ontologies ou les documents sémantiquement indexés dans une autre ontologie ;
- de raisonner au travers de correspondances entre ontologies.

Pour faciliter l'intégration sémantique, une approche consiste à identifier en quoi les ontologies diffèrent et à établir des correspondances entre elles.

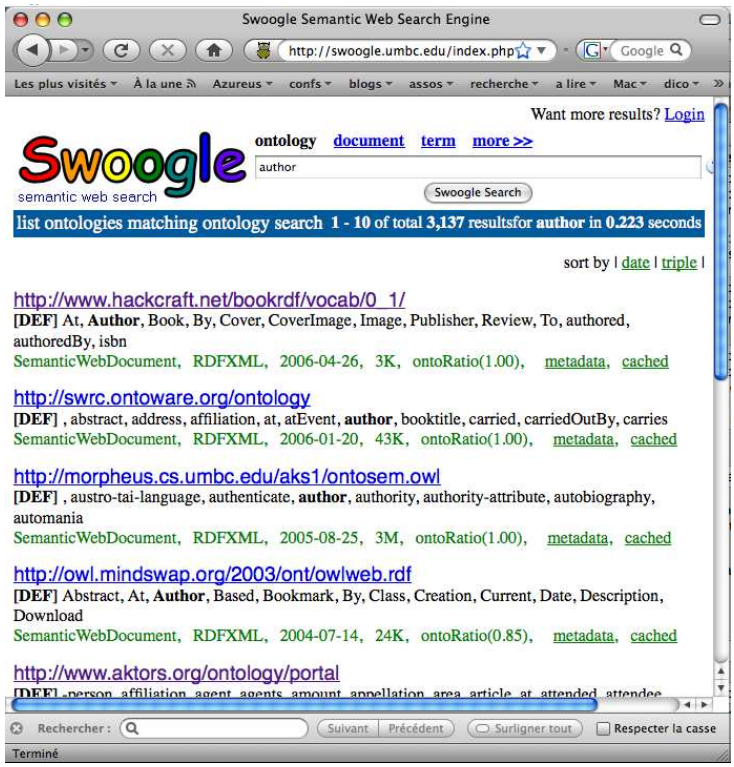


Figure 4.1 – Résultat d'une recherche sur le concept *author* dans Swoogle, moteur de recherche sur le Web sémantique, parmi les ontologie qu'il connaît.

4.2.1 Différences entre ontologies

Du point de vue des différents langages de description des ontologies, le manque de consensus entraîne inévitablement des différences importantes :

- syntaxique : les langages ne sont pas identiques et leur façon de décrire classes, relations, etc., ne sont pas identiques ;
- sémantique : le sens de certaines primitives (union, intersection, etc.) n'est pas toujours identique ;
- expressivité : les langages ne permettent pas non plus d'exprimer les même choses : la négation, les listes, les valeurs par défaut, la quantification, les intersections, les métaclasses, etc. ne sont pas présentes dans tous les langages. Il est alors évidemment difficile de pouvoir comprendre certaines notions d'une ontologie décrite dans le langage $\mathcal{L}_1$ avec une autre décrite dans un langage plus ou moins expressif $\mathcal{L}_2$.

Pour pallier le problème des différences de langage, la communauté du Web sémantique cherche à diffuser des standards W3C [W3C] tels que RDF/RDFS, OWL, SPARQL.

Au niveau ontologique, il peut y avoir d'importantes différences de modélisation liées aux constatations suivantes [Noy04] :

- des mêmes termes peuvent décrire des concepts différents : par exemple, le terme *avocat* peut désigner le fruit dans une ontologie et l'homme de loi dans une autre ;
- des termes différents peuvent décrire le même concept : par exemple, *automobile* et *voiture* peuvent désigner le même concept dans deux ontologies différentes ;

- des conventions de modélisation différentes peuvent être adoptées : *adresse* peut être considéré comme un concept dans une ontologie et comme une propriété (par exemple, du concept *personne*) dans une autre ;
- des niveaux de granularité différents peuvent être utilisés : une ontologie peut utiliser le concept *employé*, alors qu'une autre détaillera tout de suite en terme de *professeur*, *administratif*, etc.
- etc.

Cette liste n'est pas exhaustive, mais illustre l'éventail des points sur lesquels deux ontologies peuvent différer.

4.2.2 Mappings entre ontologies

Même si deux ontologies peuvent différer, il est tout de même possible de trouver des points communs, en particulier pour des ontologies qui décrivent le même domaine d'application, du même point de vue. Les notions principales de la mise en correspondance d'ontologies sont *matching*, *merging*, *alignment*, *mapping*, pour lesquels nous adoptons les définitions suivantes [ES07] :

Definition 10. *Correspondance*
Soient deux ontologies Ω_1 et Ω_2. Une correspondance *entre Ω_1 et Ω_2 est une quintuplet $\langle id, e_1, e_2, R, n\rangle$ tel que :*

- *id est l'identifiant unique de la correspondance ;*
- *e_1 et e_2 sont des entités de Ω_1 et Ω_2 respectivement ;*
- *R est une relation (l'équivalence, la généralité, le caractère disjoint, etc.) ;*
- *n est une mesure de confiance, souvent exprimée dans $[0..1]$.*

Definition 11. *Alignement*
*L'*alignement *est un ensemble de correspondances entre deux ou plusieurs (dans le cas de plusieurs matching) ontologies. Nous pouvons le définir formellement par :*
Soient deux ontologies Ω_1 et Ω_2. Un alignement *entre Ω_1 et Ω_2 est :*

- *un ensemble de correspondances entre Ω_1 et Ω_2 ;*
- *avec une cardinalité : 1-1, 1-*, etc. ;*
- *éventuellement des méta-données supplémentaires.*

Definition 12. *Mapping*
C'est la version orientée d'un alignement : il fait correspondre des entités d'une ontologie à au plus une entité d'une autre ontologie.

Definition 13. *Matching*
Le matching *est le processus permettant de trouver un alignement ou un mapping entre ontologies.*

Definition 14. *Merging*
Il s'agit de la création d'une nouvelle ontologie à partir de deux ontologies source. Les ontologies initiales demeurent inchangées. L'ontologie fusionnée est supposée contenir les connaissances des ontologies initiales : par exemple les conséquences de chaque ontologie sont les conséquences de la fusion.

Nous utiliserons principalement le terme *mapping* dans la suite de l'ouvrage, sachant que la connaissance d'un mapping d'une ontologie Ω_1 vers une ontologie Ω_2 et d'un mapping de Ω_2 vers Ω_1 permet d'obtenir un alignement. Souvent, lorsqu'il n'y a pas besoin de préciser, nous indiquerons juste qu'un concept est *partagé* lorsqu'il peut être aligné ou mappé avec le concept d'une autre ontologie. Nous notons $\Omega_1 \triangleleft \Omega_2$ le mapping de Ω_1 vers Ω_2.

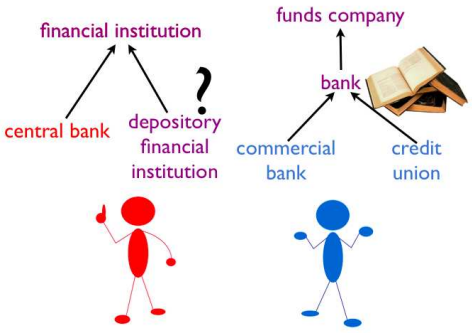


Figure 4.2 – Requête d'un utilisateur mettant en jeu des concepts mappés.

4.2.2.1 *Découverte de mappings*

Plusieurs techniques peuvent être mises en œuvre conjointement ou non pour découvrir des mappings [RB01a, KS03, Noy04, ES07]. Elle peuvent être purement manuelles, semi-automatiques ou automatiques. Certaines méthodes utilisent des ressources extérieures, telles que des ontologies de référence, la connaissance de certains objets spécifiques, des catalogues de mappings pré-établis, etc. D'autres ne considèrent que les éléments relatifs aux ontologies elles-mêmes, tels que les informations lexicales, la structure de l'ontologie, les propriétés des concepts, etc.

Les travaux sur la découverte ou le mapping d'ontologies sont nombreux. Le travail à effectuer est souvent très conséquent dès que les ontologies sont d'une taille importante. Avec le développement du Web sémantique, entre autres, cette tâche devient cependant cruciale et fait l'objet de plusieurs initiatives telles que celles concernant l'évaluation des algorithmes de matching (Ontology Alignement Evaluation Initiative [WWW14]). Tous les ans OAEI propose différents contextes d'évaluation. Par exemple, en 2007 il a été proposé une tâche de matching de thésaurus néerlandais [IMvdM+08], l'un contenant contenant un vocabulaire très vaste de 35 000 concepts généraux, l'autre contenant un grand ensemble de plus de 5 000 catégories. Les deux thésaurus ont des couvertures similaires, plus de 2 000 concepts ayant exactement le même label, mais diffèrent en granularité. Bien évidemment, un tel contexte nécessite de spécialiser les algorithmes.

4.2.2.2 *Utilisation de mappings*

L'apport de mappings pour améliorer l'interopérabilité est indéniable. En effet, la requête d'un utilisateur peut être totalement ou partiellement reformulée à l'aide des mappings de façon à ce que le fournisseur puisse comprendre au mieux la demande . Par exemple dans la figure 4.2, l'utilisateur qui pose une requête concernant le concept *depository financial institution* réussit à obtenir les documents du fournisseur relatifs à ce concept car il existe un mapping entre *depository financial institution* et *bank*, ainsi qu'entre *financial institution* et *funds company*.

Les mappings sont très utilisés dans les systèmes distribués, tels que les systèmes P2P, sémantiquement hétérogènes pour assurer une certaine interopérabilité [Rou04, HIMT03, NWQ+02a]. Ces travaux se basent sur l'existence de mappings entre deux pairs et une reformulation de la requête. Ainsi si le nombre de nombre de mappings inter-pairs est suffisamment important, l'interopérabilité sur l'ensemble du système est assurée. D'autres travaux proposent une étude théorique des conditions qui assurent l'interopérabilité de tout le système [CA04].

Les mappings permettent d'assurer un certain degré d'interopérabilité. Ainsi la solution proposée dans la suite de cette partie suppose leur existence. Néanmoins, les processus actuels ne permettent pas toujours d'obtenir un ensemble de mappings complet sur les ontologies considérées. D'autre part, dans certains domaines d'application collaboratifs (médecine, pharmacie, etc.), les participants s'accordent à ne partager qu'une partie de leurs ontologies, gardant le reste privé. Dans les deux cas, certains concepts ne seront pas mappés, ce qui implique un problème dans la reformulation de requêtes basée seulement sur des mappings.

CHAPITRE 5

Interpréter pour mieux répondre

Dans ce chapitre, nous présentons notre approche pour comparer documents et requêtes dans un cadre hétérogène sémantiquement. Nous considérons un utilisateur, initiateur de requêtes, et un fournisseur d'information, utilisant respectivement des ontologies Ω_1 et Ω_2. Dans un premier temps nous montrons que lorsque des mappings font défaut, le fournisseur d'information peut avoir des difficultés à fournir des documents pertinents (section 5.1). Pour pallier ces problèmes, le fournisseur *interprète* les DSEs de l'expansion structurante d'une requête (section 5.2). Cette approche permet d'appliquer EXSID dans le cadre hétérogène, qui devient EXSI^{2}D. La figure 5.1 illustre le positionnement du module d'interprétation dans le processus permettant de rechercher des documents pertinents par rapport à une requête. Il transforme les DSEs de l'expansion structurante en *dimensions sémantiquement interprétées*, qui sont utilisées pour calculer l'image de chaque document.

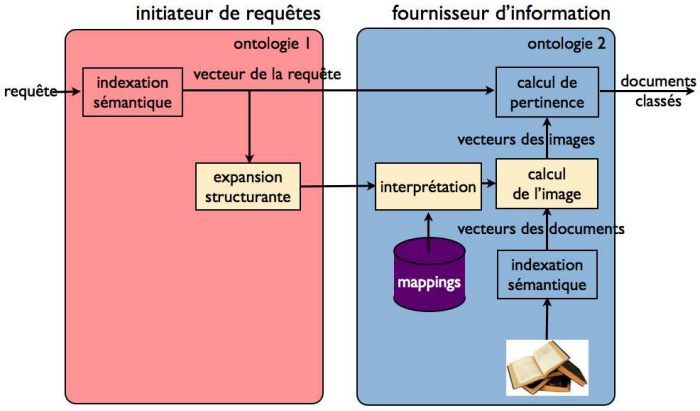


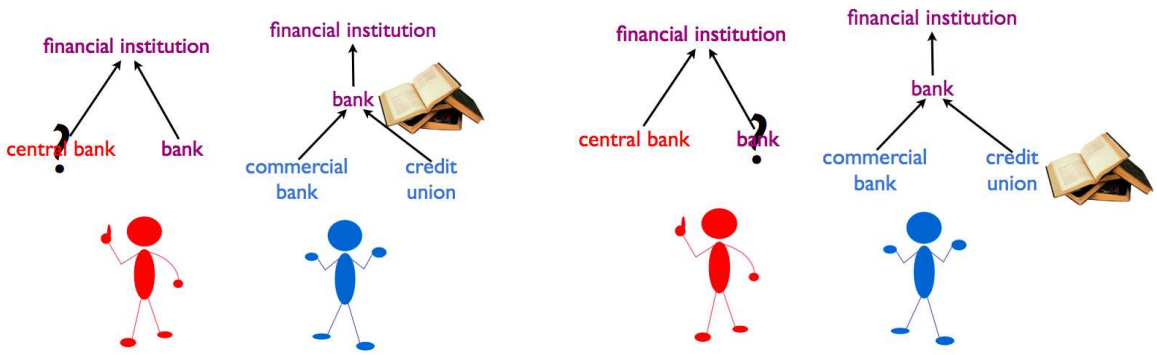
Figure 5.1 – Le cadre d'EXSI^{2}D. Notons qu'il est nécessaire de disposer de mappings entre concepts des deux ontologies.

5.1 Interopérabilité sémantique par les mappings : limites

Puisqu'il existe différentes ontologies, il est nécessaire d'opérer des mappings entre ces ontologies pour pouvoir faire de l'intégration de données, de la réécriture de requêtes, etc. Néanmoins, les mappings ne sont pas toujours complets sur les ontologies, soit parce qu'il n'a pas été possible de les mettre en place, soit parce qu'un des participants n'a pas voulu partager toute son ontologie. Ainsi, dans le cas des vecteurs sémantiques, si les ontologies Ω_1 et Ω_2 de deux pairs ne partagent pas tous les concepts, le calcul de la pertinence sera le plus souvent effectué dans l'espace $\Omega_1 \cap \Omega_2$. Donc toute requête exprimée sur des concepts non partagés ne pourra pas être traitée.

Dans l'exemple de la figure 5.2 (a) les deux pairs, le fournisseur d'information (à droite, en bleu) et l'utilisateur (à gauche, en rouge) ne partagent pas tous leurs concepts. Ainsi une requête portant sur le concept *central bank* ne peut être directement comprise par le fournisseur qui n'a pas de concept correspondant. Dans ce cas, une expansion de requête pourrait être suffisante pour que le concept *bank* soit pondéré et que le fournisseur puisse proposer ses documents.

L'exemple de la figure 5.2 (b) montre que certains cas posent toujours problème. L'utilisateur pose une requête sur le concept *bank* qui est partagé, mais le fournisseur dispose seulement de documents indexés sur un concept plus spécifique qui n'est pas connu de l'utilisateur. Ce dernier ne pouvant étendre sur un concept qu'il ne connaît pas, c'est au fournisseur de comprendre la requête dans sa propre ontologie. En particulier, le concept *credit union*, qui est similaire au concept partagé *bank* sur lequel est posée la requête, pourrait être pondéré dans la requête si l'initiateur le connaissait.



(a) (b)

Figure 5.2 – Limites des mappings entre ontologies. Les deux pairs ne partagent qu'une partie de leurs ontologies respectives (les parties communes sont en violet, les parties propres en bleu et en rouge). Les requêtes peuvent difficilement toucher les documents pertinents.

Une solution au problème précédent pourrait consister à effectuer une expansion de la requête initiale chez le fournisseur d'information. Nous avons écarté cette piste pour deux raisons. D'une part notre philosophie a toujours été de privilégier l'expansion côté utilisateur, car nous estimons que c'est lui le plus à même d'exprimer ses besoins. D'autre part, l'ensemble de concepts partagés de la requête initiale peut s'avérer trop réduit pour que le fournisseur puisse réaliser une expansion pertinente. Aussi nous nous sommes concentré sur une solution qui utilise les DSEs issus de l'expansion réalisée par l'utilisateur.

En effet, lorsque la requête est structurellement étendue et que le fournisseur reçoit un ensemble de

DSEs, certaines dimensions sont compréhensibles par ce dernier. Par exemple la figure 5.3 illustre une requête étendue, avec ses deux DSEs. Pour la DSE issue du concept *university* tous les concepts sont partagés. Le fournisseur d'information peut donc comprendre cette partie de la requête. Pour la DSE issue du concept *bank*, tous les concepts ne sont pas partagés. La dimension $\overrightarrow{dse}_{bank}$ n'est ainsi pas complètement compréhensible par le fournisseur d'information. Dans ce cas précis, c'est d'autant plus difficile que le concept central de la DSE n'est pas partagé.

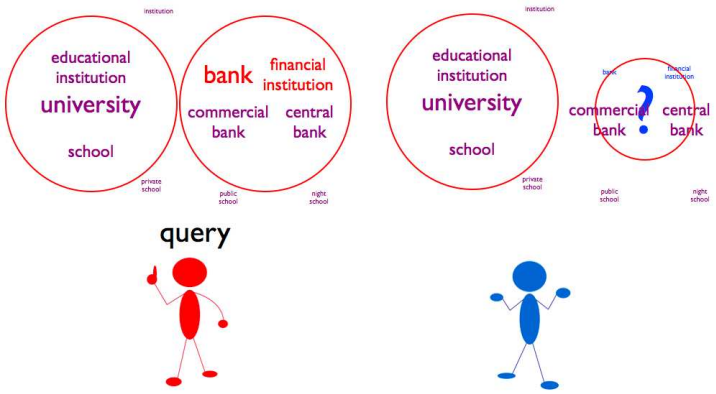


Figure 5.3 – Une requête structurellement étendue et une tentative de compréhension côté fournisseur d'information. La dimension relative des concepts correspond à leur pondération dans les DSEs. La plus petite police indique les concepts de valeur nulle. En particulier les concepts *bank* et *financial institution* chez le fournisseur d'information, non partagés, n'ont pas de pondération pour lui. L'ontologie utilisée est celle de la figure 1.11, page 21.

Néanmoins, certains concepts de cette DSE sont partagés. Sachant cela, en utilisant son ontologie et sa mesure de similarité sémantique, le fournisseur d'information peut déduire les concepts propres à son ontologie qui concernent la DSE. Dans la figure 5.3, la dimension $\overrightarrow{dse}_{bank}$ n'est pas à proprement parler compréhensible par le fournisseur d'information. Cependant, deux concepts sont pondérés, *commercial bank* et *central bank* et le fournisseur sait :

- que ces pondérations proviennent d'une DSE de l'émetteur de la requête ;
- que les deux concepts pondérés sont liés dans son ontologie à d'autres concepts ;
- qu'il existe des similarités sémantiques entre certains de ses concepts, et que les concepts pondérés dans la DSE qu'il reçoit sont proches de certains autres concepts non partagés.

Il lui est donc possible de pondérer avec pertinence des concepts non partagés de sa propre ontologie. Dans l'exemple, il pondérerait le concept *bank*, et dans une moindre mesure, *financial institution*. Intuitivement, le fournisseur d'information réalise une *interprétation* de la DSE. L'idée est donc de laisser au fournisseur d'information la liberté d'interpréter les différentes DSEs d'une requête en utilisant ses propres connaissances. Le processus ExSID s'enrichit donc d'un module supplémentaire au niveau du fournisseur (cf. figure 5.1). Ce module d'interprétation considère séparément chacune des DSE issu de l'expansion structurante. Il fournit un ensemble de *dimensions sémantiquement interprétées* qui sont utilisées pour calculer l'image du document. Le calcul de la pertinence des documents reste inchangé.

5.2 Interprétation

Nous commençons par définir une dimension sémantiquement interprétée.

Notation 1 (Mapping d'ontologies)**.**
$\Omega_1 \triangleright \Omega_2$ est le mapping d'entités de Ω_1 vers Ω_2.

Notation 2 (Mapping de concepts)**.**
$\mathcal{C}_{\Omega_1} \triangleright \mathcal{C}_{\Omega_2}$ est l'ensemble des concepts de $\mathcal{C}_{\Omega_1}$ en correspondance avec des concepts de $\mathcal{C}_{\Omega_2}$.

Notation 3 (Mapping de concept)**.**
Si un concept de $\mathcal{C}_{\Omega_1}$ a un correspondant dans $\mathcal{C}_{\Omega_2}$, nous utilisons le même nom pour les deux. Nous disons donc que $c \in \mathcal{C}_{\Omega_1}$ a pour correspondant $c \in \mathcal{C}_{\Omega_2}$.

Étant donnée une DSE dans une ontologie Ω_1, nous lui associons un autre vecteur dans Ω_2, qui correspond à l'interprétation de la DSE. Nous l'appelons *dimension sémantiquement interprétée* (DSI).

Definition 15 (Dimension sémantiquement interprétée)**.**
Soient Ω_1 et Ω_2 deux ontologies.
Soit c un concept appartenant à $\mathcal{C}_{\Omega_1}$ et $\overrightarrow{dse}_c$ une DSE à partir de c.
Un vecteur sémantique sur Ω_2 , $\overrightarrow{dsi}^{\overrightarrow{dse}_c}$ est une dimension sémantiquement interprétée *si et seulement si $\forall c' \in \mathcal{C}_{\Omega_1} \triangleright \mathcal{C}_{\Omega_2}, \overrightarrow{dsi}^{\overrightarrow{dse}_c}[c'] = \overrightarrow{dse}_c[c']$ et $\forall c' \in \mathcal{C}_{\Omega_2} \setminus \mathcal{C}_{\Omega_1} \triangleright \mathcal{C}_{\Omega_2}, \overrightarrow{dsi}^{\overrightarrow{dse}_c}[c'] \leq \overrightarrow{dse}_c[c]$.*

Cette définition demande uniquement que tous les concepts partagés par Ω_1 et Ω_2 conservent la même valeur dans la DSI et que les concepts non partagés aient une valeur inférieure à celle du concept central de la DSE. C'est une demande essentielle selon nous, car nous ne voulons pas que l'interprétation puisse modifier les valeurs que l'utilisateur donne aux concepts qu'il connaît, ni qu'il introduise des dimensions dont l'importance est supérieure à celle des concepts centraux de la requête.

Théorème 2. *En cadre sémantiquement homogène, une DSI est identique à sa DSE: $\forall c' \overrightarrow{dse}_c[c'] = \overrightarrow{dsi}^{\overrightarrow{dse}_c}[c']$.*

Démonstration.
Soient Ω_1 et Ω_2 deux ontologies, avec $\Omega_1 = \Omega_2$.
Il semble naturel dans ce cas que le mapping soit la fonction identité, donc $\forall c' \in \mathcal{C}_{\Omega_1} \rightarrow c' \in \mathcal{C}_{\Omega_1} \triangleright \mathcal{C}_{\Omega_2}$.
Donc par définition d'une DSI, $\forall c', c' \in \mathcal{C}_{\Omega_2} \overrightarrow{dse}_c[c'] = \overrightarrow{dsi}^{\overrightarrow{dse}_c}[c']$. □

Il est à noter que ce résultat est valable quelle que soit la mesure de similarité utilisée par les deux pairs.

Definition 16 (Interprétation d'une expansion structurante)**.**
Soient Ω_1 et Ω_2 deux ontologies, $\mathcal{C}_{\Omega_1}$ et $\mathcal{C}_{\Omega_1}$ les ensembles de concepts de ces ontologies.
Soit $\overrightarrow{q}$ le vecteur sémantique d'une requête sur Ω_1, $\mathcal{C}_{\overrightarrow{q}}$ l'ensemble de ses concepts centraux et $\mathcal{E}_{\overrightarrow{q}}$ son expansion structurante.
Une interprétation *de $\mathcal{E}_{\overrightarrow{q}}$, notée $\mathcal{I}_{\overrightarrow{q}}$ est un ensemble défini sur $\mathcal{C}_{\Omega_2}$ par : $\mathcal{I}_{\overrightarrow{q}} = \{\overrightarrow{dsi}^{\overrightarrow{dse}_c} : c \in \mathcal{C}_{\overrightarrow{q}}\}$.*

Théorème 3. *En cadre sémantiquement homogène, l'interprétation ne modifie pas l'expansion structurante : $\mathcal{I}_{\overrightarrow{q}} = \mathcal{E}_{\overrightarrow{q}}$.*

Démonstration. Dans ce cadre, toutes les DSIs sont équivalentes à leurs DSEs, donc l'ensemble des DSIs est équivalent à l'ensemble des DSEs: $\{\overrightarrow{dsi}^{\overrightarrow{dse_c}}\} = \{\overrightarrow{dse}_c\}$ et donc $\mathcal{I}_{\overrightarrow{q}} = \mathcal{E}_{\overrightarrow{q}}$. □

L'interprétation est une fonction identité dans le cadre sémantiquement homogène. L'avantage de ce résultat est qu'il est possible d'intégrer le module d'interprétation dans un système sans craindre une modification des résultats si les deux ontologies sont identiques. Le module d'interprétation ne sert que si cela est nécessaire, c'est-à-dire dans le cas d'une hétérogénéité.

Les définitions précédentes ont laissé de côté un problème qu'il nous faut maintenant adresser : pondérer les concepts non partagés de l'ontologie Ω_2 dans chaque DSI. Nous connaissons la requête et son expansion structurante sur Ω_1. Pour chaque DSE cela signifie que nous disposons de son concept central ainsi que des pondérations des concepts partagés. C'est à partir de ces données que nous devons obtenir une interprétation sur Ω_2. Pour ce faire, rappelons qu'une DSE est calculée à partir de son concept central et que les pondérations des autres concepts sont obtenues en fonction de leur similarité par rapport au concept central. C'est ce que nous allons exploiter en appliquant une démarche similaire pour la construction d'une interprétation. D'abord, nous allons rechercher sur Ω_2 un concept $\tilde{c}$ pouvant jouer le rôle de concept central. Puis, dans une deuxième étape, nous allons pondérer les concepts non partagés en fonction de leur similarité au concept $\tilde{c}$ et des valeurs connues pour les concepts partagés (fonction d'interprétation affine par morceaux).

Les deux sections suivantes répondent donc aux deux sous-problèmes :

1. choisir un concept central ;
2. pondérer les concepts non partagés en fonction du choix précédent.

5.2.1 Recherche du concept central de l'interprétation

Le concept central de l'interprétation sert à classer les concepts de Ω_2. Cet ordre doit indiquer la pertinence de chaque concept pour la dimension de la requête dans l'ontologie Ω_2. Les pondérations des concepts partagés servent de base à la pondération des concepts de Ω_2 non partagés.

Notation 4 (Concept correspondant au concept central d'une DSE)**.**
Soit $\overrightarrow{dse}_c$ une DSE sur une ontologie Ω_1 et c son concept central. Nous notons $\tilde{c}$ le concept correspondant à c dans une autre ontologie Ω_2.

Le concept correspondant au concept central d'une DSE est le concept central pour une interprétation. Nous sommes dans le cas hétérogène. Le concept correspondant au concept central d'une DSE n'est donc pas toujours évident. Il peut y avoir plusieurs candidats à être concept correspondant au concept central d'une intreprétation. En fait, c'est le cas de tous les concepts de $\otimes_{\in}$. Nous avons alors autant d'interprétations possibles qu'il y a de concepts candidats à être concepts centraux de cette interprétation. Il s'agit de différents points de vue d'une DSE à partir de l'ontologie du fournisseur d'information. En un sens, si une DSE a pour concept central c, il est tout à fait possible qu'il corresponde à plusieurs concepts dans l'ontologie du fournisseur d'information. Chacune de ces interprétations serait donc une façon de comprendre la DSE. Néanmoins, introduire de nouvelles entités doit être fait avec parcimonie. Et les multiplier ne doit être fait que si le gain le justifie. Or, dans ce cas, il est possible de définir une fonction de pertinence dans le choix du concept candidat à l'interprétation. Le but de ce processus est de rendre compte de la façon dont l'initiateur de requête aurait exprimé sa requête dans l'ontologie du fournisseur d'information. C'est pourquoi plusieurs cas peuvent se présenter :

- le concept central de la DSE a un correspondant dans l'ontologie du fournisseur d'information : dans ce cas il paraît opportun de choisir ce concept de l'ontologie du fournisseur d'information comme concept central de l'interprétation ;
- le concept central de la DSE n'a pas de correspondant dans l'ontologie du fournisseur d'information : il est alors possible de choisir un autre concept de l'ontologie du fournisseur, ou bien de faire intervenir un nouveau concept dans cette ontologie :
  - dans le premier cas, comme nous l'avons précisé, nous devons définir une fonction de pertinence qui nous fera préférer un concept central de l'interprétation d'une DSE à un autre ;
  - le deuxième cas est une hypothèse intéressante à plusieurs titres que nous n'avons cependant pas étudiée dans ce travail car cela semblait nous éloigner un peu trop de ce que nous cherchions.

Nous appelons *concept candidat à l'interprétation* un concept qui peut être central pour l'interprétation.

Nous proposons de définir formellement la notion de *fonction d'interprétation* (cf. définition 17), qui est relative à une DSE $\overrightarrow{dse}_c$ et à un concept candidat c', et qui attribue une pondération à toute valeur de similarité par rapport à c'. Elle consiste en quatre points. Le premier demande qu'elle attribue la valeur de $\overrightarrow{dse}_c[c]$ à la valeur de similarité 1, qui correspond à c'. Dans le deuxième point, nous utilisons les poids attribués par $\overrightarrow{dse}_c$ aux concepts partagés (*commercial bank*, *central bank* et *institution* dans les figures 5.4 (a) et (b)) et le classement des concepts suivant $sim_{c'}$. Cependant, il peut y avoir différents concepts partagés qui ont la même valeur de similarité par rapport à c', mais des poids différents dans $\overrightarrow{dse}_c$. Donc, nous demandons que la fonction $f_i^{\overrightarrow{dse}_c,c'}$ attribue la valeur minimum. C'est un choix que nous pouvons qualifier de pessimiste et il est aussi possible de prendre le maximum, une moyenne, etc. Pour le troisième point, nous introduisons c_{min}, qui est le concept partagé ayant la plus faible valeur de similarité (*institution* dans les deux exemples des figures 5.4 (a) et (b)). Nous considérons que nous n'avons pas assez d'information pour pondérer les valeurs de similarité plus faibles que $sim_{c'}(c_{min})$. Nous leur attribuons donc la valeur 0. Le quatrième point assure que les segments de la fonction affine par morceaux ne peuvent être que ceux définis par les trois points précédents.

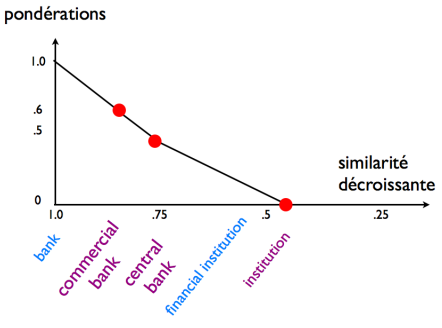
Definition 17 (Fonction d'interprétation)**.**
Soit une dimension sémantiquement enrichie $\overrightarrow{dse}_c$ et un concept c', $f_i^{\overrightarrow{dse}_c,c'} : [0..1] \rightarrow [0..1]$, notée f_i s'il n'y a pas d'ambiguïté, est une fonction d'interprétation *si et seulement si c'est une fonction affine par morceaux et :*

- $f_i(1) = \overrightarrow{dse}_c[c]$*;*
- $\forall c'' \in \mathcal{C}_{\Omega_2} \triangleright \mathcal{C}_{\Omega_1} \setminus \{c'' : sim_{c'}(c'') = 1\},\ f_i(sim_{c'}(c'')) = \min_{\substack{c''' \in \mathcal{C}_{\Omega_1} \cap \mathcal{C}_{\Omega_2} \\ sim_{\tilde{c}}(c') = sim_{\tilde{c}}(c'')}} (\overrightarrow{dse}_c[c'''])$*;*
- $\forall x \in [0..1],\ x < sim_{c'}(c_{min}) \ \Rightarrow\ f_i(x) = 0$*;*
- $Seg = \|\{x : \exists c'' \in \mathcal{C}_{\Omega_2} \triangleright \mathcal{C}_{\Omega_1}, c'' \neq c' \ and \ sim_{c'}(c'') = x\}\| + 1$ *où Seg est le nombre de segments de f_i.*

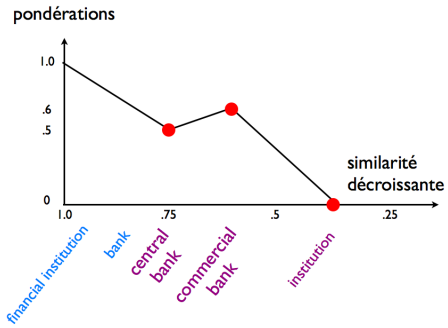
Intuitivement, les critères permettant de choisir un concept correspondant parmi tous les concepts candidats peuvent être exprimés en terme de propriétés de la fonction d'interprétation affine par morceaux. Bien entendu, il y a autant de fonctions d'interprétation que de concepts candidats. Mais l'idée générale est de choisir la fonction f_i qui ressemble le plus à une fonction de propagation. Considérons l'exemple des figures 5.4 (a), (b) et (c). Trois concepts (en rouge) sont partagés : *commercial bank*, *central bank* et *institution* et deux (en bleu) ne le sont pas : *bank* et *financial institution*. La fonction d'interprétation de la figure (a) est obtenue en considérant le concept candidat *bank* et en classant tous les concepts à partir de ce concept. La fonction d'interprétation de la figure (b) est obtenue de la même façon en considérant le concept *central bank* comme candidat. La fonction d'interprétation de la figure

(c) utilise le concept *institution* comme concept central. Notons sur ces figures que $f_i(1) = \overrightarrow{dse}_c[c]$, qui était pondéré par 1 dans la DSE $\overrightarrow{dse}_{bank}$. Si nous devions choisir entre *bank*, *central bank* et *institution*, nous préférerions *bank*, parce que sa fonction d'interprétation décroît de manière monotone, alors que celles de *central bank* et de *institution* ont un « désordre » plus important en comparaison d'une fonction de propagation.

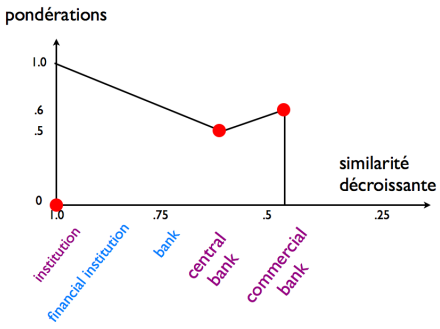
Le désordre induit par une fonction d'interprétation peut être considéré selon plusieurs caractéristiques. Par exemple, nous pouvons choisir la fonction qui minimise le nombre de minima locaux, c'est-à-dire le nombre de fois que le signe de la dérivée de la fonction change. Un autre exemple peut être de choisir la fonction qui minimise les variations de poids entre les minima locaux et le maximum local qui les suit, pour pénaliser les fonctions qui ne sont pas décroissantes de manière monotone. Un troisième exemple peut combiner ces critères. Etc.



(a)



(b)



(c)

Figure 5.4 – Trois fonctions d'interprétation centrées sur trois concepts candidats : *bank* (a), *central bank* (b) ou *institution (c)*.

L'algorithme 3 résume ces idées. La fonction *estPartagé(*c*)* renvoie vrai si le concept c est partagé (il appartient à $\mathcal{C}_{\Omega_2} \triangleright \mathcal{C}_{\Omega_1}$). La fonction *calculDuDésordre(*f_i*)* calcule une évaluation du désordre généré par f_i selon un critère considéré (par exemple : le nombre de minima locaux). Plus cette évaluation est faible, plus le concept c' est intéressant. Notons qu'il est possible que plusieurs concepts aient la même valeur

Algorithme 3 : Recherche du meilleur concept pour l'interprétation d'une dimension sémantiquement enrichie.

input : Une DSE $\overrightarrow{dse_c}$ sur une ontologie Ω_1, une ontologie Ω_2
output : un concept central pour l'interprétation c̃
begin
  if *estPartagé(c)* **then**
    c̃← c;
  else
    candidats← $\emptyset$;
    plusPetitDésordre ← $+\infty$;
    forall $c' \in \mathcal{C}_{\Omega_2}$ **do**
      désordre ← calculDuDésordre($f_i^{\overrightarrow{dse}_c,c'}$);
      if *désordre=plusPetitDésordre* **then**
        candidats ← candidats∪ $\{c'\}$;
      else
        if *désordre < plusPetitDésordre* **then**
          plusPetitDésordre ← désordre ;
          candidats ← $\{c'\}$
    c̃← similaritéMaximum(candidats)
  retourne c̃;
end

de désordre pour leur fonction d'interprétation. C'est par exemple le cas dans une structure arborescente, où les ascendants d'un concept quelconque peuvent avoir la même fonction que lui. C'est pourquoi l'algorithme utilise une liste de concepts candidats. Ensuite, l'algorithme choisit un concept dans cette liste pour lequel la similarité avec les concepts partagés est la plus forte. Dans l'exemple précédent c' serait choisi de préférence à ses ascendants avec un tel critère. Si plusieurs concepts de la liste ont la même similarité avec les concepts partagés, un concept est choisi au hasard. En effet, nous ne voulons qu'une DSI pour chaque DSE. Il nous faut donc un unique concept central de l'interprétation.

5.2.2 Pondération des concepts de l'ontologie

Une DSE $\overrightarrow{dse_c}$ est donc interprétée en une DSI, avec $\tilde{c}$ pour concept central de l'interprétation. En ce qui concerne la pondération, nous laissons leur poids d'origine aux concepts partagés, et les concepts non partagés obtiennent une valeur en utilisant la fonction d'interprétation.

Definition 18 (Interprétation d'une DSE)**.**
Soit $\overrightarrow{dse_c}$ une DSE sur $\mathcal{C}_{\Omega_1}$ et soit $\tilde{c}$ le concept correspondant à c pour $\mathcal{C}_{\Omega_2}$. Soit $sim_{\tilde{c}}$ une fonction de similarité et soit $f_i^{\overrightarrow{dse}_c,\tilde{c}}$, notée aussi f_i, une fonction d'interprétation. Alors la DSI $\overrightarrow{dsi}^{\overrightarrow{dse}_c}$ est une interprétation de $\overrightarrow{dse}_c$ si et seulement si :

- $\overrightarrow{dsi}^{\overrightarrow{dse}_c}[\tilde{c}] = f_i(1)$;
- $\forall c' \in \mathcal{C}_{\Omega_2} \triangleright \mathcal{C}_{\Omega_1} \setminus \{c'' : sim_{c'}(c'') = 1\},\ \overrightarrow{dsi}^{\overrightarrow{dse}_c}[c'] = \overrightarrow{dse}_c[c']$;
- $\forall c' \in \mathcal{C}_{\Omega_2} \setminus \mathcal{C}_{\Omega_2} \triangleright \mathcal{C}_{\Omega_1},\ \overrightarrow{dsi}^{\overrightarrow{dse}_c}[c'] = f_i(sim_{\tilde{c}}(c'))$;

Dans la figure 5.5, nous pouvons voir que le fournisseur de documents a classé ses concepts grâce au concept correspondant au concept central de la DSE $\overrightarrow{dse}_{bank}$. Les concepts partagés (en rouge) conservent leur poids. C'est le cas pour les concepts *commercial bank*, *central bank* et *institution*. L'interprétation attribue un poids aux concepts non partagés (en bleu) : *bank* et *financial institution*.

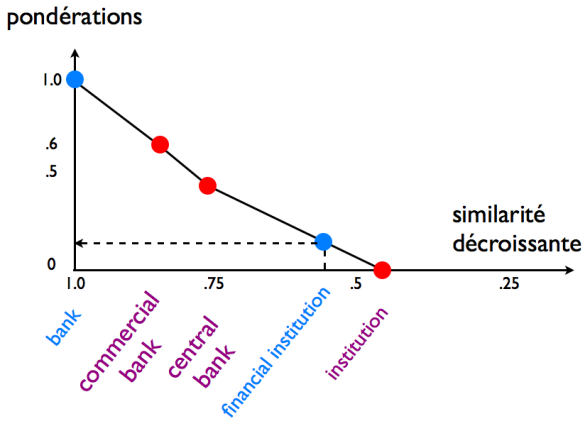


Figure 5.5 – Pondération des concepts non partagés (en bleu) *bank* et *financial institution* grâce à une fonction d'interprétation.

Dans ce chapitre nous avons présenté la solution EXSI^{2}D, qui consiste à ajouter à l'expansion structurante un module d'interprétation avant de créer l'image des documents. Il permet au fournisseur d'information de décrire les DSEs dans sa propre ontologie Ω_2 en utilisant des mappings entre l'ontologie de l'initiateur de la requête Ω_1 et Ω_2. Pour ce faire, il est nécessaire de trouver un concept correspondant au concept central de l'expansion dans Ω_2. Il sert de concept central d'une interprétation qui déduit des pondérations des concepts partagés entre Ω_1 et Ω_2 une pondération des concepts non partagés de Ω_2, et qui génère une DSI à partir d'une DSE. La recherche de ce concept correspondant n'est pas efficace dans la solution que nous proposons car elle considère comme candidats tous les concepts de Ω_2. Des solutions peuvent être trouvées pour réduire l'ensemble des candidats, en éliminant certains concepts : par exemple en tenant compte des propriétés de la similarité sémantique utilisée. D'autre part, l'introduction de nouveaux concepts n'a pas été retenue dans notre solution, mais semble intéressante. En effet il est possible d'utiliser ce qu'une DSE indique des similarités entre concepts par les pondérations des concepts partagés pour introduire un nouveau concept dans Ω_2. L'idée de l'interprétation de requête a donné lieu à trois publications [VCLV07, VCLV08a, VCLV08c].

L'importance de la mesure de similarité dans la mise en place de l'interprétation nous amène à penser qu'elle a une place presque aussi grande que les mappings dans l'interopérabilité offerte par EXSI^{2}D. Pour que l'initiateur de la requête et le fournisseur d'information se comprennent, il est important que les mesures de similarité sémantique soient *compatibles*, c'est-à-dire qu'elles conservent le classement des concepts. Il s'agit d'un accord sans doute plus profond que les mappings. En effet, les similarités sémantiques ne sont peut être pas toujours liées qu'aux ontologies et peuvent représenter une forme de référent culturel, générationnel, etc. [KlRJ07]. C'est sans aucun doute un point sur lequel nous pourrions

apporter quelques idées, car il a une incidence sur la façon d'évaluer des mappings, des proximités d'ontologies, la découverte de mappings, etc.

CHAPITRE 6

Evaluations et discussions

Nous voulons dans ce chapitre évaluer l'approche EXSI^{2}D. Le problème auquel nous sommes confrontés est celui de la définition du jeu de test que nous voulons utiliser. Mesurer l'efficacité d'une méthode de RI dans un cadre sémantiquement hétérogène n'est en effet pas très aisé.

Nous commençons donc par une discussion et des propositions en ce qui concerne le contexte d'évaluation. Ensuite nous présentons les méthodes de référence pour nos évaluations, les résultats, puis nous discutons le processus choisi et les résultats.

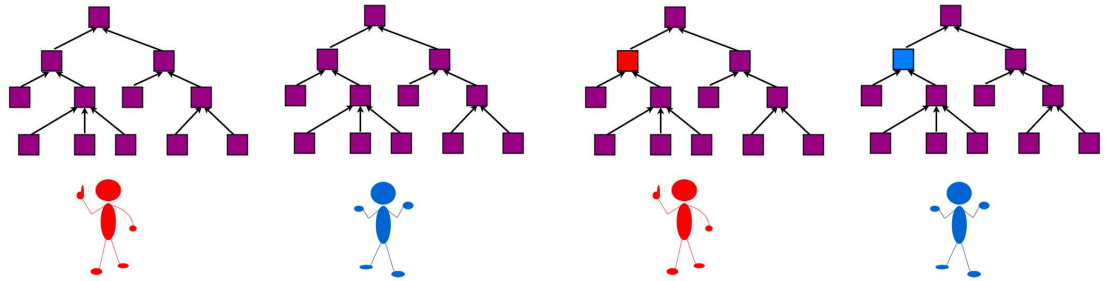
6.1 Contexte des évaluations

Le cadre du travail de cette deuxième partie est celui d'une hétérogénéité sémantique entre l'utilisateur posant une requête et le fournisseur d'information. Nous voulons montrer que'EXSI^{2}D fonctionne lorsque les ontologies utilisées pour indexer documents et requêtes ne partagent que certains concepts. Les travaux sur l'hétérogénéité sémantique entre ontologies ont donné lieu à des recherches sur des jeux de test permettant d'évaluer les algorithmes de mappings. Ainsi le projet OAEI [WW14] propose différentes pistes d'évaluation pour différents points de vue, différentes applications, etc. du domaine. Nous ne cherchons pas à faire des mappings entre ontologies. Ces jeux de test ne s'appliquent donc pas directement à nous. Nous voulons montrer que l'interopérabilité peut être maintenue même si les mappings sont partiels entre deux ontologies. D'autre part, nous voulons pouvoir maîtriser finement le degré d'hétérogénéité pour pouvoir juger précisément l'efficacité de notre solution en fonction de ce paramètre.

L'idée générale est donc de générer nous-même l'hétérogénéité. Cela nous permettra de maîtriser ce qui est hétérogène et à quel degré d'hétérogénéité. Nous proposons donc qu'initiateur de requêtes et fournisseur d'information partagent la même ontologie. Ensuite, nous pouvons supprimer les correspondances sur certains concepts, pour faire varier l'hétérogénéité entre ces ontologies. Par exemple, dans la figure 6.1 (a), la même ontologie est utilisée par deux pairs, l'un en rouge (à gauche), l'autre en bleu (à droite). Tous les concepts sont mappés, puisque ce sont les mêmes ontologies. Enlever la correspondance sur un concept, comme dans la figure 6.1 (b) fait augmenter l'hétérogénéité entre les deux ontologies.

6.1.1 Variation du degré d'hétérogénéité

Notre première intuition est de faire varier le nombre de mappings, afin d'observer l'évolution des systèmes face à un accroissement de l'hétérogénéité. Dans ce type de test, nous supprimons donc aléatoirement des mappings entre un initateur de requête et un fournisseur d'information. Il est donc par exemple possible de faire évoluer l'hétérogénéité par niveaux successifs : 0%, 10%, 20%, …, 90%, 100%. Chaque niveau à n signifiant que n% des mappings sont supprimés.



(a) (b)

Figure 6.1 – Deux pairs partageant la même ontologie (a) et pour lesquels un concept n'est plus mappé (b).

6.1.2 Suppression dirigée des mappings

Un autre type d'hétérogénéité peut être envisagé. Lorsque les connaissances entre deux acteurs sont globalement les mêmes mais qu'ils n'utilisent pour dialoguer que des connaissances spécifiques, on arrive généralement à une impasse dans la communication. Il est facile avec notre approche de modéliser ce scénario : il suffit de supprimer les mappings sur les concepts importants de la discussion. Pour notre protocole de test, cela signifie supprimer uniquement les mappings sur les concepts centraux de chaque requête. Les participants se comprennent globalement, sauf sur les concepts principaux.

6.1.3 Corpus, ontologie et indexation

Nous avons utilisé pour nos évaluations les mêmes outils que pour l'évaluation de la partie précédente (cf. section 3.2.2, page 45), à savoir le corpus Cranfield, indexé par RIIO sur WordNet. C'est donc sur WordNet que nous allons « supprimer » des mappings selon l'évaluation considérée. Il s'agit en fait d'indiquer au fournisseur d'information au cas par cas quels concepts ne sont pas en correspondance avec ceux de la requête. Cela ne modifie pas l'indexation.

6.2 Méthodes de référence

Les méthodes de référence auxquelles nous nous comparons sont le cosinus et l'expansion (classique). Comme nous l'avons exprimé plus haut (3.2.1, page 44), le cosinus est à la fois la méthode la plus utilisée avec le modèle vectoriel en RI, et aussi la solution que nous avons choisie pour le module de pertinence. Nous voulons donc comparer ExSI^{2}D au cosinus pour connaître l'effet de notre méthode par rapport à la mesure de pertinence classique et pour connaître l'effet d'ExSI^{2}D dans un système de RI utilisant les vecteurs sémantiques et le cosinus. L'expansion est une amélioration essentielle, y compris lors de problèmes d'hétérogénéité, quand les vocabulaires d'indexation ne sont pas semblables.

6.3 Paramètres des évaluations

En ce qui concerne ExSI^{2}D, nous n'avons pas de solution simple pour trouver le meilleur concept candidat pour l'interprétation s'il n'appartient pas à l'ensemble des concepts mappés. L'algorithme 3 (cf.

section 5.2.1, page 67) a en effet une trop grande complexité. C'est pourquoi nous utilisons un algorithme spécifique dans nos expérimentations, prenant en compte la hiérarchie de WordNet : l'algorithme 4. Pour chaque dimension sémantiquement enrichie, si nous ne connaissons pas le concept central, nous cherchons le plus petit généralisant commun aux concepts apparaissants dans la dimension, c'est-à-dire l'ancêtre de tous les concepts de la dimension sémantiquement enrichie le plus éloigné de la racine. Notre similarité sémantique tenant compte principalement de la structure de la hiérarchie de concepts et étant la même pour l'initiateur de la requête et le fournisseur d'information, elle nous permet souvent de trouver de « bons » concepts. Néanmoins, cette heuristique ne fonctionne bien que lorsque la propagation affecte les descendants du concept central. Dans le cas contraire, lorsque la propagation provoque une remontée des propagations dans la hiérarchie, cette technique ne permet pas de trouver une solution optimale. En particulier, ceci est le le cas lorsque le concept central est une feuille, ou qu'il est suffisamment proche des feuilles.

Algorithme 4 : Obtention du concept central d'une interprétation à partir d'une dimension sémantiquement enrichie.

entrée : une dimension sémantiquement enrichie $\overrightarrow{dse_c}$.
sortie : le concept central d'une interprétation de la dimension sémantiquement enrichie $\overrightarrow{dse_c}$.
begin
  $hyponymes \leftarrow \emptyset$;
  forall $c' \in \overrightarrow{dse_c}$ **do**
    // $hyponymes(c)$ contient tous les hyponymes de c
    $hyponymes \leftarrow hyponymes(c')$;
  $profondeurMax \leftarrow 0$;
  $central \leftarrow \emptyset$;
  forall $c' \in hyponymes$ **do**
    if $profondeur(c') < profondeurMax$ **then**
      $central \leftarrow c'$;
      $profondeurMax \leftarrow profondeur(c')$;
  retourne $central$;
end

Les paramètres que nous utilisons pour ExSI^{2}D et pour l'expansion sont les mêmes que pour le cas homogène (cf. 3.2.3 page 48). Nous prenons donc une propagation « pentue », qui étend sur une dizaine de concepts par dimension initiale de la requête, en utilisant la similarité que nous avons décrite par ailleurs (cf. 2.4.2.3 page 40). En ce qui concerne les suppressions de mappings, il s'agit de cacher chez le pair fournisseur d'information le fait que certains concepts sont les mêmes dans la requête qu'il reçoit et dans ses documents. Dans l'expérimentation utilisant une variation du degré d'hétérogénéité, à partir d'un pourcentages défini, nous supprimons au hasard des mappings sur des concepts de WordNet. Remarquons qu'à 0% de mappings supprimés nous nous trouvons finalement dans le cas sémantiquement homogène, identique à la section 3.2.4, page 49. Dans l'évaluation avec une suppression de mappings dirigée, nous supprimons uniquement les mappings sur ses concepts centraux à l'évaluation de chaque requête. La première expérimentation a été effectuée deux fois sur les 225 requêtes de Cranfield. La seconde une fois sur les 225 requêtes.

6.4 Résultats des évaluations

6.4.1 Variation du degré d'hétérogénéité

Les figures 6.2 (a) et (b) présentent les résultats obtenus par notre expérimentation. La mesure de référence est celle obtenue par le système utilisant le cosinus quand aucun mapping n'est supprimé. Les valeurs dans les deux figures sont des ratios entre les différentes méthodes et cette mesure de référence. Cela nous est apparu plus parlant que des valeurs brutes. Quand le pourcentage de mappings supprimés augmente, le ratio de précision et de rappel diminuent. C'est-à-dire que comparé au cosinus dans le cadre sémantiquement homogène, les systèmes ont des résultats qui décroissent. Cependant, notre système a de bien meilleurs résultats. Quand le pourcentage de mappings supprimés est inférieur à 70%, nous obtenons toujours un ratio de 80% de précision et de rappel : quand les ontologies divergent à 70% nous continuons à être proches des valeurs de précision et de rappel du cosinus en cas homogène à 80%.

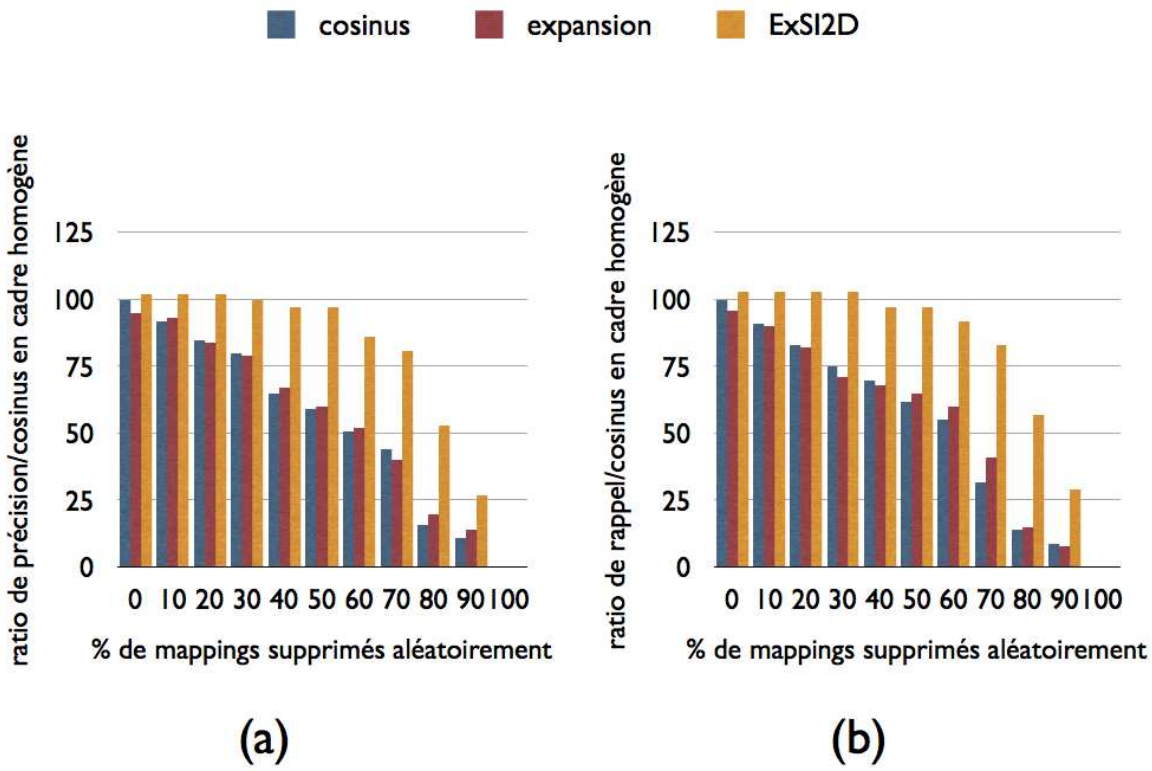


Figure 6.2 – Evolution du ratio de précision (a) et de rappel (b) entre les différentes mesures et le cosinus dans le cadre homogène, en fonction du pourcentage de mappings supprimés aléatoirement.

6.4.2 Hétérogénéité dirigée

Les figures 6.3 (a) et (b) présentent les résultats de notre évaluation. La mesure de référence est celle obtenue par la méthode utilisant le cosinus quand aucun mapping n'est supprimé, comme pour l'évaluation précédente. Les valeurs dans les deux figures sont des ratios entre les différentes méthodes et cette mesure de référence. Avec le cosinus seul, il n'y a plus de comparaison possible entre les requêtes et les documents, car les dimensions centrales des requêtes ne sont plus mappées. Il n'est donc plus possible de sélectionner de document pertinent. L'adjonction de l'expansion au cosinus permet de retrouver quelques documents pertinents en atteignant certains concepts proches des concepts centraux de la requête. Cela donne des ratios de précision et rappel proche de 10% . ExSI^{2}D quant à elle dépasse légèrement les 90% . Elle pourrait être encore meilleure si la solution que nous avons choisie pour calculer le concept central de l'interprétation était plus efficace. En effet, cette heuristique n'est pas une bonne approximation quand les concepts centraux des requêtes sont des feuilles, ou sont bas dans la hiérarchie des concepts.

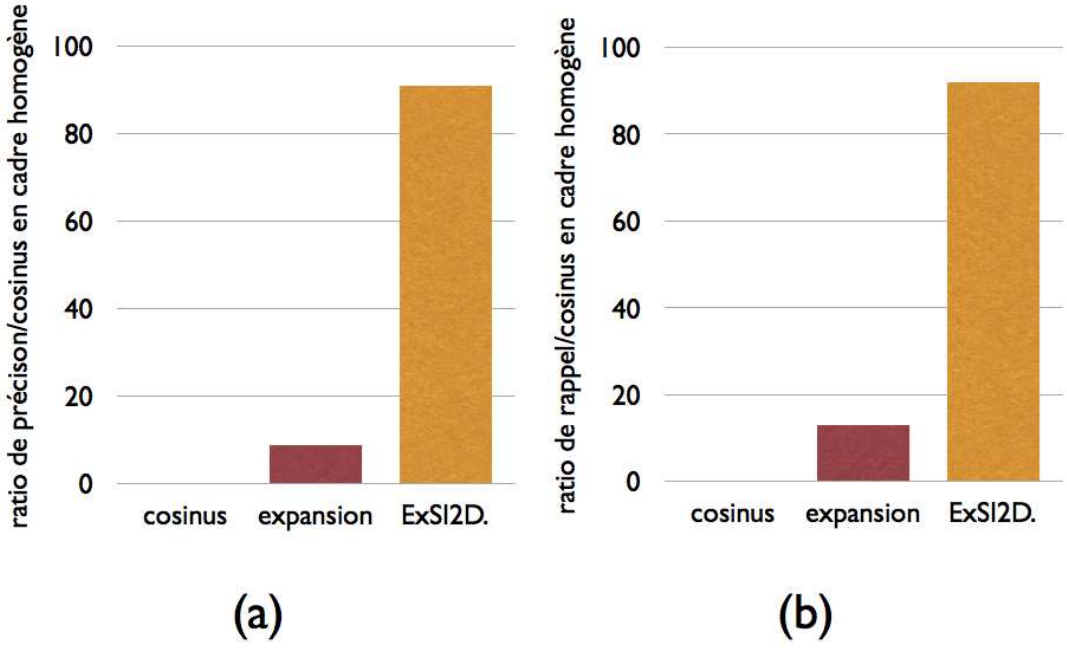


Figure 6.3 – Ratios de précision (a) et de rappel (b) entre les différentes mesures et le cosinus dans le cadre homogène. Le mapping sur les concepts centraux de la requête est supprimé.

6.5 Discussion

Nous avons été confrontés à un manque de jeu d'essai pour tester notre approche. À notre connaissance, il n'existe pas de test en RI avec des requêtes et des documents indexés sur des ontologies proches, dans le but de tester des solutions à l'hétérogénéité sémantique. Il existe bien des jeux de test avec des données indexées sur plusieurs thésaurus (par exemple la travail d'Isaac *et al.* [IMvdM+08] pour l'OAEI), mais cela ne constitue pas un vrai corpus de test pour une solution RI (pas de requête). Les solutions que nous avons vues permettent principalement de juger de la précision de méthodes et outils d'alignement d'ontologies. Bien évidemment, ExSI^{2}D mérite une évaluation plus large mettant en œuvre d'autres corpus de test. Nous pouvons utiliser d'autres corpus de RI, avec la même ontologie et le même processus expérimental. Mais nous recherchons aussi un corpus extrait d'un cas d'utilisation réel avec des ontologies différentes, pour valider notre approche. D'ailleurs, nous avons commencé à étudier l'utilisation du corpus fourni dans l'*entreprise track* de TREC [W24].

D'autre part, notre solution utilise le fait que notre similarité sémantique est structurelle, et que notre mise en place de l'hétérogénéité ne modifie pas la structure. Ce sont des critères dont nous ne pouvions pas nous passer, puisque la seule ontologie que nous pouvions utiliser avec un corpus de test en langue anglaise est WordNet, qui est principalement une hiérarchie de concepts, et que la mesure de similarité sémantique que nous utilisons tient compte de la structure de l'ontologie. Néanmoins, cela a sans doute un impact sur les résultats.

Les ontologies peuvent évoluer de différentes manières : de manière conceptuelle, en ajoutant et supprimant des concepts, ou structurelle, en modifiant l'organisation de la hiérarchie de concepts [ES04]. Nous avons dans ces évaluations suivi la première hypothèse : en changeant les correspondances entre les concepts, nous ne faisons que modifier l'identification des concepts dans deux ontologies. Par exemple, telle ontologie pense que le père du concept *chat* est *félin*, et telle autre qu'il s'agit du concept *feline*. La structure n'est pas modifiée, la correspondance entre deux concepts est juste impossible. Mais qu'en est-il si pour une autre ontologie le père de *chat* est *végétal* ? La structure est là très modifiée.

Le cadre général de l'évaluation semble donc favorable pour notre solution. Mais il est à noter qu'il l'est aussi pour les méthodes de référence, en particulier pour l'expansion, qui tire aussi parti du fait que la structure des ontologies est identique dans tous les cas. De plus, il est à noter que l'utilisation de *similarités compatibles*, c'est-à-dire qui conservent l'ordre induit sur les concepts chez les deux pairs permet d'obtenir les mêmes résultats. Il n'est donc pas nécessaire que les deux mesures de similarité sémantique soient identiques.

Les précautions ci-dessus étant prises, il n'en demeure pas moins que les résultats obtenus sont très encourageants. Pour le moins, ils indiquent une potentialité importante qui est d'autant plus intéressante que, comme remarqué précédemment, l'adjonction de l'interprétation peut être systématique. En effet, si l'hétérogénéité sémantique n'est pas présente, les résultats obtenus sont identiques. En effet, en cas d'homogénéité sémantique, l'interprétation n'interfère pas et ne modifie pas les résultats. Rappelons encore que cette technique peut s'appliquer à d'autres processus utilisant les vecteurs sémantiques sans avoir à modifier ni l'indexation ni la méthode de calcul de pertinence. Cette généricité permet par exemple à un moteur de recherche d'utiliser ExSI^{2}D sans rien modifier de son architecture et en gagnant en interopérabilité sémantique.

De plus, comme nous l'avons indiqué au chapitre précédent (cf. 5.2.2 page 71) l'utilisation de similarités compatibles aide grandement au calcul du concept correspondant pour l'interprétation. Ici les similarité sont les mêmes. Or, c'est en partie ce qui permet d'avoir de bons résultats, même quand il y a peu de mappings. Si le fournisseur d'information réussit à classer les concepts de la même façon que l'initiateur de la requête, cela augmente les chances pour eux de se comprendre. Cela dénote aussi qu'ils

partagent une représentation du monde qui va plus loin que le mapping entre leurs concepts : en plus du mapping, ils partagent la même notion de proximité relative entre les concepts. L'étude de cette notion de compatibilité entre les fonctions de similarité nous semble pouvoir donner des résultats intéressants. Il est clair que cette notion joue un rôle important dans un processus de compréhension mutuelle. Cela pourrait avoir des répercussions sur l'évaluation de la pertinence, la qualité, de mappings, ou encore devenir un critère pour leur construction.

PARTIE III

La sémantique dans les réseaux P2P non-structurés : traitement de requêtes et interopérabilité

CHAPITRE 7

Traitement de requêtes et sémantique dans les systèmes P2P

Les systèmes pairs-à-pairs (P2P) sont une architecture distribuée permettant de s'affranchir du client/-serveur et visant à obtenir une meilleure tolérance aux pannes, un meilleur passage à l'échelle, etc.Les systèmes P2P ont prouvé leur efficacité dans :

- le calcul partagé : la plateforme BOINC [And04, www19] et son projet le plus célèbre, SETI@home [ACK+02, www17] ;
- l'échange de données, par exemple Gnutella [JAB01], KaZaA [www16] ou PeerDB [OST03] ;
- les télécommunications : Skype [www3] ;
- etc.

Ce qui distingue le paradigme P2P des solutions distribuées classiques concerne tout d'abord la dynamicité, c'est-à-dire que les pairs peuvent rejoindre ou quitter le système à tout moment ; d'autre part, il n'est pas possible de vouloir toutes les réponses à une requête, les pairs disposant des données concernées pouvant ne pas être présents au moment de la requête et l'interrogation de tout le réseau étant inefficiente ; il n'est pas non plus possible de maintenir un catalogue central indexant toutes les données dont disposent les pairs. Nous allons dans ce chapitre présenter les réseaux P2P : les principales architectures, celle qui nous convient le mieux, le routage de requêtes, et en particulier de certaines requêtes très importantes : les top-k. Ensuite nous présentons l'utilisation de sémantique dans les réseaux P2P : les différents modèles et les systèmes qui les utilisent.

7.1 Réseaux P2P

Les premières recherches sur les réseaux P2P ont concerné le routage de requêtes dans des réseaux non-structurés. Ces systèmes, popularisés par les premières applications de partage de données, sont très simples mais ont de gros problèmes de performance. Ces problèmes ont débouché sur la définition des solutions structurées et hybrides, mais aussi sur la recherche de moyens d'utiliser des données sémantiquement riches et des applications complexes : par exemple ActiveXML [ABC+03], Edutella [NWQ+02a], Piazza [HIMT03, IHMT03], PIER [HHL+03]. Nous allons cependant utiliser dans nos travaux un système P2P non structuré, pour les propriétés dont dispose ce type d'architecture. Nous allons ensuite étudier dans ce cadre les techniques de routage de requêtes et en particulier les requêtes top-k, parce qu'elles sont complexes mais d'une grande utilité pour les systèmes d'information.

7.1.1 Types de réseaux

Les systèmes P2P utilisent un réseau P2P, qui est lui-même une couche logique du réseau physique (par exemple, internet). La topologie du réseau et son degré de centralisation ont un impact sur les propriétés du système P2P, comme la tolérance aux fautes, la performance, le passage à l'échelle, etc. Trois classes de réseaux P2P sont généralement considérées : non-structuré, structuré et super-pair [APV07].

7.1.1.1 Non-structurés

Les réseaux non-structurés sont créés de manière aléatoire, un pair arrivant au hasard dans le système et créant ses voisinages *ex nihilo*. Le placement des données dans le réseau n'est pas déterminé par la topologie, les pairs disposant des données qu'ils souhaitent eux-même mettre à disposition. Le routage de requête est généralement *l'inondation* (*flooding*), qui consiste à envoyer le message à tout le réseau.

Le réseau ne dépendant d'aucun pair, la tolérance aux pannes est très importante. Cependant, l'inondation n'est pas un processus qui permet de passer à l'échelle facilement : le nombre de messages nécessaires pour cette manière de communiquer est trop important. D'autre part, il n'est pas possible de garantir de trouver tous les résultats d'une requête : pairs disposant des ressources inactifs, trop loin, etc. Dans les deux cas, cela diminue l'efficacité du système. Gnutella [JAB01] est un réseau non-structuré.

7.1.1.2 Structurés

Face aux problèmes de passage à l'échelle des réseaux non-structurés, une solution a été de proposer le modèle des réseaux structurés. Ils contrôlent directement la topologie et le placement des données pour arriver à diminuer fortement le nombre de messages échangés : les données étant placées à des positions précises, il est plus facile de les retrouver.

Les réseaux structurés prennent très souvent la forme de tables de hachage distribuées (*distributed hash table*, DHT). Ces tables permettent d'associer une clé à tout objet à indexer dans le système P2P. Les pairs sont responsables de données qui leurs sont attribuées suivant leur position dans le réseau. Pour trouver un objet, il suffit donc de connaître la clé à laquelle il correspond, ce qui donne la position où il se trouve, et donc le pair qui en est responsable.

Le routage des requêtes est très efficace : trouver le pair responsable d'un objet se fait en $O(\log n)$ décisions de routages, n étant le nombre de pairs dans le réseau. L'autonomie est cependant drastiquement limitée, autant dans les données mises à disposition par les pairs que par leur position dans le réseau. D'autre part, l'expressivité des requêtes est limitée (par l'utilisation de clés de hachage). L'extension à d'autres types de requêtes est un champ de recherche en cours [APV06a].

Chord [SMK+01], CAN [RFH+01], Pastry [RD01] sont des réseaux structurés.

7.1.1.3 Super-pairs

Ils sont appelés hybrides en référence au fait qu'ils sont un compromis entre les réseaux P2P « purs », où tous les pairs sont égaux, fournissent les mêmes fonctionnalités, et les réseaux clients-serveurs. Dans ces réseaux, certains pairs sont choisis comme *super-pairs* et ont le rôle de serveurs pour d'autres pairs. Les super-pairs eux-même sont aussi en relation de voisinage entre eux, et peuvent utiliser n'importe quel système P2P pour s'organiser. Les super-pairs peuvent être élus par rapport à leurs capacités (bande passante, capacité de calcul) et remplacés dynamiquement.

Les réseaux super-pairs sont très efficients, le temps pris pour trouver une donnée parmi les super-pairs (qui indexent les pairs dont ils sont responsables) est très court, comparé à l'inondation. La qualité

de service (l'efficience perçue par l'utilisateur : complétude du résultat, temps mis pour obtenir une réponse) est élevée, pour les mêmes raisons. Les réseaux super-pairs sont cependant très dépendants des super-pairs qui les composent : ces derniers regroupant tous les problèmes des serveurs dans les architectures clients/serveurs ; ils peuvent ainsi être surchargés, le réseau dépend de leur bon fonctionnement, etc. Edutella [NWQ+02a], JXTA [www.7] sont des réseaux super-pairs.

7.1.1.4 Comparaison

La table 7.1 donne un aperçu de propriétés de chacun des trois modèles présentés précédemment. Chacun des symboles, +, - et = correspond à une capacité forte, faible ou moyenne dans la propriété considérée.

	non-structuré	structuré	super-pair
autonomie	+	-	=
efficience	-	+	+
qualité de service	-	+	+
tolérance aux pannes	+	+	-

Table 7.1 – Propriétés des principales architectures de réseaux P2P [APV07].

L'originalité des systèmes P2P provient principalement selon nous de l'autonomie des sources de données et de la tolérance aux pannes. C'est pourquoi ce sont les propriétés que nous mettons en avant et que nous allons dans la suite de cette thèse nous intéresser principalement aux réseaux non-structurés.

7.1.2 Routage de requêtes dans des systèmes non-structurés

Amener une requête jusqu'aux pairs qui possèdent les données pertinentes, le *routage de requêtes* est une tâche très difficile et importante dans les systèmes P2P [APV07]. Dans cette section, nous allons présenter les approches permettant de faire du routage de requêtes dans des systèmes non-structurés. Toutes ces approches sont gourmandes en nombre de messages envoyés sur le réseau. Elles essaient donc toutes de minimiser cette quantité. Nous allons en présenter trois principales, ainsi que leurs extensions [TR06] : BFS, random walks, indexation du voisinage.

7.1.2.1 BFS

Dans sa version de base, comme Gnutella [JAB01] l'implémente par exemple, la recherche par parcours en profondeur (*breath-first search*, BFS) inonde le réseau P2P à une distance TTL (*time to live*) de l'initiateur de la requête. Chaque pair recevant la requête décroît la valeur du TTL, exécute la requête et la transmet à tous ses voisins, si le TTL a une valeur supérieure un. Cela permet donc d'atteindre tous les pairs situés à une distance TTL de l'initiateur.

Les améliorations principales de BFS visent à diminuer le nombre de messages envoyés, en sélectionnant un sous-ensemble des voisins. BFS « modifié » [KGZY02] n'envoie qu'à un sous-ensemble des voisins, diminuant bien le nombre des messages mais perdant des résultats. Pour dépasser ce problème, dans BFS intelligent [KGZY02], chaque pair conserve une trace de la pertinence des réponses renvoyées par un chemin (cette information lui arrivant de ses voisins) pour chacune des requêtes qu'il traite. À l'arrivée d'une nouvelle requête, il la route selon les nœuds qui ont été les plus intéressants pour une

requête similaire. BFS « intelligent » nécessite un grand nombre de messages de routage (les pairs renvoyant les informations par le chemin d'où vient la requête), et est dépendant de la volatilité du réseau, mais permet d'obtenir plus de réponses que la version modifiée de BFS.

Lorsque la requête demande un ensemble réduit de réponses, il est possible d'utiliser la recherche par profondeurs successives (*iterative deepening* [LCC+02, YGM02]). Cela consiste à utiliser BFS avec des degrés de TTL de plus en plus grands, en partant de $TTL = 1$, jusqu'à obtenir toutes les réponses souhaitées. Si les réponses sont situées dans l'environnement de l'initiateur de la requête, les performances sont très bonnes.

7.1.2.2 Random walks

L'algorithme de marche au hasard (*random walks* [LCC+02]) génère k « marcheurs » qui partent de k différents voisins du pair initiateur de requête et parcourent le réseau. A chaque fois qu'un marcheur atteint un pair, il exécute la requête, interroge le pair initiateur pour savoir si la condition de terminaison est remplie, et si non est transféré à un voisin du pair actuel. Il est aussi possible de n'interroger que régulièrement le pair d'origine. En tout cas, dès que la terminaison est détectée par un marcheur, il s'arrête.

L'efficience de cet algorithme est assez bonne au sens où le nombre total de messages ($k \times TTL$ dans le pire des cas) n'est pas dépendant de la topologie. Par contre les résultats dépendent eux de la topologie et des choix de routage (aléatoires).

Par défaut l'algorithme ne permet pas d'apprentissage. Cependant la recherche probabiliste adaptative (*adaptative probabilistic search* [TR03]) permet d'estimer la pertinence de chacun des noeuds lors de l'envoi d'un marcheur. Cette probabilité évoluant avec le temps et tenant compte des bonnes réponses (qui sont renvoyées aux pairs sur tout le chemin parcouru par le marcheur). Les performances de ce système sont très bonnes, mais sont dépendantes de la dynamicité du réseau.

7.1.2.3 Indexation du voisinage

Diverses solutions préconisent d'utiliser une indexation des pairs voisins dans le but de savoir à qui envoyer : à un pair éloigné (chaque pair connaissant son voisinage, il faut accéder à des pairs lointains pour avoir des données non atteintes), les voisins qui ont eux-même des voisins intéressants, etc.

L'approche basée sur les index locaux [CGM02, YGM02] demande que chaque pair indexe ses voisins dans un rayon r, c'est-à-dire tous les pairs situés à moins de r bonds. Les requêtes sont alors transmises à des pairs suffisament éloignés pour qu'il n'y ait pas de recouvrement des données indexées : la « zone » couverte par un pair est $2 \times r + 1$; les requêtes sont donc exécutées par des pairs situés à $m \times (2 \times r + 1)$ de l'initiateur de la requête. Bien entendu, l'exécution d'une requête dans le système est plus faible que BFS (moins de pairs sont contactés). Cependant la mise en place de cet algorithme (routage, création de index, etc.) peut être coûteuse, particulièrement dans des réseaux dynamiques.

En utilisant des filtres de Bloom [RK02], c'est-à-dire des résumés de données, il est possible d'indexer les données accessibles au travers de chacun des voisins d'un pair, ainsi que leur distance, en nombre de bonds. En recevant une requête, il suffit alors de la transférer au voisin qui dispose de la donnée intéressante la plus proche de lui. L'avantage des filtres de Bloom est leur compacité. Il n'est cependant pas possible de les modifier, et pour des réseaux dynamiques, cette solution n'est pas optimale.

7.1.3 Requêtes top-k dans des réseaux non-structurés

Les techniques de routage présentées précédemment concernent principalement les requêtes qui cherchent un sous-ensemble de solutions, dont la condition est unique. Il existe cependant d'autres types de requêtes plus complexes qui ne peuvent pas être exécutées simplement dans les réseaux P2P. C'est le cas par exemple des requêtes de jointure, des requêtes nécessitant une fraîcheur des résultats, etc. Dans cette section, nous présentons les requêtes *k-meilleurs*, ou requêtes *top-k*, type de requêtes pour lesquelles nous proposerons un algorithme dans la partie suivante.

7.1.3.1 Quelques précisions

Les requêtes top-k indiquent un nombre k de résultats dont la pertinence avec la requête est la plus forte. Cette valeur k est définie par le pair initiateur de la requête. Ce type de requêtes a une grande importance dans différents champs de l'informatique, comme la surveillance système et réseau [BO03], la RI [MTW05, BNST05, PZSD96], les bases de données multimédia [GM04, dVMNK02], etc. Akbarinia *et al.* [APV06b] montrent en particulier leur utilité pour la gestion de données, particulièrement lorsque le nombre de réponses est très élevé. Le système ne peut pas se permettre de renvoyer tous les résultats, et l'initiateur de la requête n'a généralement pas besoin de tous les résultats.

7.1.3.2 Solution classique

La solution naïve consiste à envoyer la requête à tous les pairs accessibles, l'exécuter et retourner les réponses correctes à l'initiateur qui fait le classement. Il est évident que cette solution n'est pas acceptable en coût de communication et temps de traitement [APV06b].

Dans PlanetP [CAPMN03] chaque pair maintient un index global du contenu de ses voisins en utilisant un algorithme de propagation de rumeurs (*gossiping*). L'initiateur d'une requête crée une liste de pertinence des pairs qu'il connaît et envoie à chacun son tour, dans l'ordre de la liste, la requête en demandant un nombre n de ses meilleurs documents. L'utilisation de l'index global, répliqué sur les pairs, est le point le plus critiqué de ce système, la mise à jour étant difficile à assurer.

7.1.3.3 Algorithme FD

Akbarinia *et al.* [APV06b] proposent une solution totalement distribuée (*fully distributed*, FD) pour exécuter des requêtes top-k dans des réseaux structurés. L'idée générale est assez simple mais très performante. Il s'agit d'effectuer une fusion des résultats à chaque niveau. Un pair envoie une requête à chacun de ses descendants, et attend leur réponse. Un fois qu'il les a, il les fusionne et les renvoie au pair qui lui avait envoyé la requête. Le nombre de messages est largement inférieur aux solutions précédentes, puisque les pairs n'envoient pas tous leurs réponses directement au pair initiateur de la requête, et le temps d'exécution est assez limité. Des stratégies servent à gérer le départ des pairs du système, pour ne pas bloquer l'algorithme FD.

7.2 La sémantique dans les réseaux P2P

Les systèmes P2P ont pour objectif principal de résoudre les problème de tolérance aux pannes et de passage à l'échelle. Cependant, ils n'ont pas souvent de notion de sémantique et ont des difficulté à échanger des connaissances. D'autres communautés, celle du web sémantique par exemple, proposent

des solutions pour l'échange de connaissances. Mais elles ont tendance à utiliser des solutions centralisées. Le rapprochement entre les deux communauté paraît donc aller de soi, et beaucoup d'efforts récents sont effectués pour les rapprocher [Agn07].

Dans la suite de cette section, nous supposons que les systèmes travaillent avec des mappings (d'ontologies ou de schémas de manière indifférente). Il y a deux manières de gérer la sémantique dans les systèmes P2P : la première consiste à utiliser une ontologie générale, abstraite des ontologies locales, à la manière de l'intégration de données traditionnelle ; la seconde utilise des mappings entre les ontologies des différents pairs.

7.2.1 Coopération des fournisseurs de données

7.2.1.1 Modèles classiques de l'intégration de données

Les systèmes d'intégration de données proviennent du champ de la gestion de données. Le développement des bases de données a logiquement amené à chercher à les intégrer : la base des pompiers et celle des renseignements téléphoniques, développées de manière indépendantes ont pu avec le temps nécessiter une *intégration* pour permettre aux pompiers de connaître l'adresse d'un appel au secours, etc. Mais ces deux bases ont été développées de manière séparée, et il n'est pas évident de faire correspondre tous les éléments de l'une avec des éléments de l'autre. Les solutions d'entrepôts de données (*data warehouse*) permettent de stocker des données de différentes sources hétérogènes après les avoir « extraites, transformées et chargées » (*extracted, transformed, and loaded* : ETL). Ces solutions peuvent être efficaces mais, le processus d'ETL a quelques limitations. Entre autres celui de la fraîcheur des données : quand les sources de données d'origine sont modifiées, l'entrepôt continue pendant un certain temps a stocker les anciennes valeurs.

Des recherches récentes [TRV98, BCF$^+$01] ont promu un nouveau paradigme pour l'intégration de données, où il s'agit de définir un schéma médiateur global, avec des mappings entre les schémas locaux des bases de données et le schéma global [Wie92]. Les utilisateurs expriment leurs requêtes en utilisant le schéma global, et le médiateur les transforme pour qu'elles correspondent aux schémas locaux des sources de données. Des adaptateurs (*wrappers*) permettent pour chaque source de données de traduire le schémas global dans le langage d'interrogation local. Les deux principales approches sont le *global-as-view* (GAV) et le *local-as-view* (LAV) [HIMT03]. La première définit le schéma médiateur comme comme une vue sur les schémas locaux, la seconde les schémas locaux comme des vues sur le schémas global. Dans l'approche GAV les sources peuvent décrire leurs données comme elles veulent. Cependant cette forte autonomie des sources s'accompagne d'un effort important pour maintenir le schéma global, qui est très fortement dépendant de l'évolution des sources et de l'ajout de nouvelles sources. De l'autre côté, les sources sont contraintes dans le modèle LAV, mais il est plus facile pour le médiateur de gérer la maintenance des sources : celles-ci devant se décrire sur le schéma global.

7.2.1.2 Application des solutions classiques à la sémantique dans les réseaux P2P

Ces différentes solutions d'intégration de données ont été reprises pour la gestion de descriptions sémantiques dans les systèmes P2P. Par exemple dans Edutella [NWQ$^+$02a] chaque pair gère localement des données par rapport à une ontologie de référence. Des super-pairs servent de médiateurs sur les données des pairs qui leurs sont liés. Quand on leur transmet une requête, les super-pairs commencent par regarder si la requête concerne leur schémas ; si oui ils transmettent aux pairs qu'ils gèrent, sinon ils transmettent à un de leurs voisins dans le niveau des super-pairs. APPA [AM07] propose une solution

utilisant une description de schéma commune (*common schema description*, CSD) sur laquelle les pairs passent un accord pour un temps donné (une expérimentation, etc.). STyX [ABFS02a, ABFS02b] met en place un langage de mapping pour des données XML avec une ontologie globale pour référence. Cette ontologie globale regroupe les connaissances d'un champ d'intérêt pour l'utilisateur qui choisit cette ontologie globale pour décrire ses données. Ces solutions sont toutes plus ou moins proches du paradigme LAV, au sens où une description a priori ou consensuelle sert aux pairs pour se décrire.

D'autres solutions sont plus proches du modèle GAV. XPeer [SMGC04] traite du partage de données XML. Il consiste en une architecture super-pairs, ceux-ci gérant un ensemble de pairs de schémas similaires. Les super-pairs sont organisés en une hiérarchie et stockent l'union des schémas de leurs fils. Elle sert dans la hiérarchie pour trouver le super-pair qui peut traiter une requête. MediaPeer [DGY05] est un projet très proche qui utilise un réseau P2P de médiateurs pour traiter des requêtes Xquery. Dans Bibster [HJBM+04] chaque pair gère son ontologie (bibliographique) propre et le réseau est sémantiquement organisé en fonction des expertises de chaque pair. DBGlobe [PAP+03] gère deux couches de super-pairs caractérisant des zones géographiques et des communauté d'intérêt. Les super-pairs stockent les informations sur les pairs qui leur sont attachés sont tous interconnectés, de façon à pouvoir s'échanger toutes les informations (grâce à des filtres de Bloom).

Des solutions hybrides permettent de tirer partie des deux paradigmes, par exemple BAV (*both as view* [MP03]), et GLAV (*generalized local as view* [CGLR04]) en proposant des vues aux niveaux local et global pour adresser le problème de l'évolutivité de GAV tout en maintenant l'autonomie des sources que ne garantit pas LAV. CoDB [FKLZ04] est un exemple d'une architecture GLAV.

7.2.1.3 SONS

Le regroupement de pairs en réseaux couvrant sémantiques (*semantic overlay networks*, SON [TXD02, CG02, ACMHP04]) est une piste de recherche importante depuis plusieurs années. Que ce soit dans des réseaux non structurés, où des algorithmes de gossiping permettent de connaître l'intérêt du voisinage et donc de créer des communautés d'intérêt, ou dans des solutions structurées comme les tables de hachage distribuées, les SONs s'imposent. Dans les deux cas, ils permettent d'enrichir le modèle des données et les langages de requêtes (données relationnelles, semi-structurées, ou basées sur des triplets). Ils permettent surtout de regrouper les pairs qui sont sémantiquement similaires et de ce fait de transmettre les requêtes aux SONs dont l'intérêt est proches de celui de la requête, ce qui fait diminuer le coût de routage et augmente l'efficacité du système.

7.2.2 Mise en place de correspondances entre schémas

Une alternative à la mise en place d'un schéma global comme le propose l'intégration de données classique est de créer des correspondances sémantiques (mappings) entre pairs. Il y a trois types principaux de procédés différents pour la création de ces mappings : par coordination statique, par découverte dynamique, ou de manière semi-automatique [Lum05].

7.2.2.1 Définition statique des correspondances

SomeWhere [ACG+04, Rou04] présente une solution totalement distribuée, dans laquelle il n'y a ni super-pairs ni serveur central. Les pairs créent des mappings entre les schémas de leurs données et réécrivent les requêtes quand ils les transmettent. La pertinence d'un voisin plutôt qu'un autre se fait en considérant les mappings entre leurs ontologies et la requête. Piazza [HIMT03, IHMT03] aussi propose

que les pairs se décrivent avec leur propre ontologie et qu'ils déclarent des correspondances avec des pairs qui deviennent leurs voisins. Ce qui différencie les deux approche est le langage de représentation des données et de mappings. Dans SomeWhere les données sont décrites avec des classes et des relations simples sur ces classes, et le raisonnement se fait à partir de la logique des propositions. Dans la dernière version de Piazza, les données sont décrites en XML et les mappings sont des inclusions et des équivalences entre des chemins XML.

7.2.2.2 Définition semi-automatique de mappings sémantiques

Les solutions de cette catégorie utilisent des règles pour générer automatiquement les mappings entre ontologies. Dans le projet Hyperion [RGKG+05, WWW4], les pairs recherchent des pairs « connaissances » (*acquaintance*) avec lesquels ils échangent leurs schémas. Des règles permettent de générer des tables de mappings chez les pairs concernés. Le système GLUE [DHA03] utilise une approche par apprentissage automatique afin de mettre les concepts de deux ontologie en correspondance de manière quasi-automatique. Pour mesurer la similarité entre deux concepts $c_i \in \mathcal{C}_{\Omega_1}$ et $c_j \in \mathcal{C}_{\Omega_2}$, le système GLUE fait une classifiaction croisée : les données utilisant le concept c_i sont classifiées en utilisant le classifieur sur le concept c_j et inversement. HepToX [BCL+05] permet, comme Hyperion, a un pair de se lier de manière plus au moins automatique à des connaissances qui partagent le même thème. Des règles, couplées aux annotations des schémas, permettent de générer des mappings qu'un utilisateur peut compléter ou préciser.

7.2.2.3 Découverte dynamique des mappings

PeerDB [OST03] utilise des techniques de RI : fréquence d'apparition des mots-clés dans les données indexées par certains concepts, etc. Des agents coopérants portent les requêtes dans le réseau et recherchent des pairs ayant des schémas comparables à celui de la requête. Ils renvoient ensuite les schémas ou les parties de schémas resultants à l'utilisateur qui décide ou non d'exécuter la requête chez les pairs concernés.

Le Chatty Web [ACMH03] utilise une propagation de rumeur pour que les pairs puissent connaître les ontologies des autres pairs du réseau. Ensuite les relations de voisinage sont créées automatiquement si les pairs ont la même ontologie ou des ontologies similaires (valeur attribuée par une fonction qui utilise des règles fournies par le système). La requête est réécrite à chaque bond, en vérifiant que les modifications ne font pas diverger trop les requêtes de la requête initiale.

Les systèmes P2P sont un paradigme distribué potentiellement très intéressant : tolérant aux pannes, décentralisé, etc. Il existe cependant un certain nombre de problèmes, dont le routage de requêtes et l'expressivité des données et des requêtes. La sémantique est une solution couramment admise pour résoudre certains des problèmes considérés. Nous allons dans la suite nous situer dans le cadre d'un système d'information P2P non structuré hétérogène sémantiquement. Pour gérer l'hétérogénéité nous proposons de ne pas utiliser d'abstraction globale, mais de gérer des correspondances entre ontologies. Parmi les points que nous allons devoir adresser se trouvent : la représentation d'un pair de manière sémantique à partir de ses données et le routage dans un cadre hétérogène.

CHAPITRE 8

Vers l'utilisation d'ExSI²D dans un système P2P sémantique

Ce chapitre est consacré à la présentation de solutions pour intégrer ExSI²D dans un système d'information distribué et hétérogène. Nous avons choisi de nous situer dans le cadre d'une architecture P2P totalement distribuée, car c'est celle qui laisse le plus d'autonomie aux pairs et qui assure la meilleure tolérance aux pannes. Notons tout de même qu'ExSI²D n'est pas limité à ce type d'architecture, et qu'il est possible de l'utiliser dans un réseau structuré ou hybride.

Nous utilisons donc un système d'information documentaire, les documents étant décrits par des vecteurs sémantiques. Le premier objectif est de définir les caractérisations de pairs et de communautés par le même type de représentation. Car dans ce cas il est possible d'utiliser les mêmes types de processus, qu'ils concernent des documents, des pairs ou des communautés. En particulier, le calcul de pertinence d'une requête et d'un pair peut être identique à celui d'une requête et d'un document. Nous présentons dans la première section la caractérisation d'ensembles de documents par un seul vecteur sémantique, de façon à ce qu'elle soit utilisée dans un système distribué.

Dans la seconde section nous présentons des propositions pour le routage de requêtes dans un cadre sémantiquement hétérogène.

8.1 AGR4QUERY: une agrégation optimiste dépendante des requêtes

Dans cette première section, nous présentons la caractérisation d'un pair à partir de ses documents, ou d'un regroupement de pairs à partir de leurs caractérisations. Nous cherchons à construire une *agrégation* à partir d'un ensemble d'éléments. Ces éléments peuvent être les documents d'un pair : l'agrégation consiste alors en une représentation de l'information qu'il stocke. Ils peuvent aussi être des agrégations, permettant alors de décrire des communautés de pairs ou des communautés de communautés, etc. Nous voulons garder un cadre uniforme pour la recherche, qu'il s'agisse de mesurer la pertinence d'un document, d'un pair ou d'une communauté de pairs. C'est pourquoi les agrégation sont des vecteurs sémantiques et que la mesure de pertinence est le cosinus. L'agrégation n'est pour nous que le reflet des documents qu'un pair contient contient. Elle est unique. Tous les points évoqués : unicité de la description, vecteur sémantique, description par le contenu, peuvent être discutés : par exemple, nous aurions pu envisager d'utiliser les solutions classiques consistant à produire un certain nombre de classes de documents. Mais cette démarche fait tomber l'hypothèse de représentation par un vecteur sémantique.

Nous renvoyons le lecteur aux ouvrages de Brucker et Barthélémy [FJP07] et Tufféry [Tuf07] pour plus de détails sur le sujet.

Nous pouvons définir une fonction d'agrégation dans un cadre très général comme une fonction qui prend en entrée un ensemble d'éléments (par exemple des documents) et génère un vecteur sémantique. Par souci de clarté dans l'écriture, nous identifions par la suite un document et sa représentation. Par exemple, $\overrightarrow{d}$ indiquera aussi bien le document d que sa représentation dans un modèle vectoriel. Nous adoptons la définition suivante :

Definition 19 (Fonction d'agrégation)**.**
Soient Ω une ontologie et $\mathcal{C}_\Omega$ l'ensemble de ses concepts.
Soient $\mathcal{D}$ l'ensemble de documents sur $\mathcal{C}_\Omega$, et $\mathcal{P}^\mathcal{D}$ l'ensemble des parties de cet ensembles (classes de documents). Soit $\mathcal{V}$ l'ensemble des vecteurs sémantiques sur $\mathcal{C}_\Omega$.
Une fonction, notée f_{agr}, est une fonction d'agrégation *si et seulement si elle a la signature suivante :*

$$f_{agr} : \left| \begin{array}{lcl} \mathcal{P}^\mathcal{D} & \mapsto & \mathcal{V} \\ \{\overrightarrow{d_1}, \ldots, \overrightarrow{d_n}\} & \rightarrow & f_{agr}(\{\overrightarrow{d_1}, \ldots, \overrightarrow{d_n}\}) \end{array} \right.$$

Une agrégation est *optimiste* par rapport à une mesure si elle ne sous-estime pas les valeurs de ses composantes par rapport à cette mesure.

Definition 20 (Agrégation optimiste (par rapport au cosinus))**.**
Soient Ω une ontologie et $\mathcal{C}_\Omega$ l'ensemble de ses concepts.
Soient $\{\overrightarrow{d_1}, \ldots, \overrightarrow{d_n}\},\ n \in \mathbb{N}$ un ensemble de documents et $\overrightarrow{q}$ une requête sur $\mathcal{C}_\Omega$.
f_{agr} est une agrégation optimiste *si et seulement si :*
$\forall i,\ 1 \leq i \leq n,\ \cos(\overrightarrow{q}, f_{agr}(\{\overrightarrow{d_1}, \ldots, \overrightarrow{d_n}\})) \geq \cos(\overrightarrow{q}, \overrightarrow{d_i})$.

Cette définition est utile lorsque nous nous basons sur l'agrégation pour décider d'interroger un pair ou non. En effet, si nous décidons de ne pas interroger le pair, on est sûrs qu'il n'est pas possible de manquer un document. C'est-à-dire que la pertinence d'une caractérisation est toujours plus forte ou identique à celle des documents de la collection.

Dans la suite nous présentons notre solution pour obtenir une agrégation de manière générale dans le cadre de vecteurs sémantique. Nous montrons ensuite qu'ExSI^{2}D est applicable à notre solution d'agrégation. La définition suivante prend en compte tous les éléments d'une collection.

Definition 21 (ALL@1)**.**
Soient Ω une ontologie et $\mathcal{C}_\Omega$ l'ensemble de ses concepts.
Soit $\{\overrightarrow{d_1}, \ldots, \overrightarrow{d_n}\},\ n \in \mathbb{N}$ un ensemble de documents sur $\mathcal{C}_\Omega$.
ALL@1 *est une fonction d'agrégation telle que :*

$$\forall c \in \mathcal{C}_\Omega\ \textit{all@1}(\{\overrightarrow{d_1}, \ldots, \overrightarrow{d_n}\})[c] = \left\{ \begin{array}{ll} 1 & \textit{si } \exists \overrightarrow{d_i},\ i \in [0..1],\ \overrightarrow{d_i}[c] > 0 \\ 0 & \textit{sinon} \end{array} \right.$$

ALL@1 est donc une fonction qui donne à la caractérisation d'un ensemble de documents la valeur 1 à chaque dimension pour laquelle il existe un document qui la pondère de manière non nulle.

Théorème 4 (ALL@1 est cumulative)**.**
L'agrégation ALL@1 *est* cumulative.
Soient $\{\overrightarrow{d_1}, \ldots, \overrightarrow{d_n}\},\ n \in \mathbb{N}$ un ensemble de documents.
$\forall\, \overrightarrow{d} \notin \{\overrightarrow{d_1}, \ldots, \overrightarrow{d_n}\}\ \forall c \in \mathcal{C}_\Omega\ \textit{all@1}(\{\textit{all@1}(\{\overrightarrow{d_1}, \ldots, \overrightarrow{d_n}\}), \overrightarrow{d}\}) = \textit{all@1}(\{\overrightarrow{d_1}, \ldots, \overrightarrow{d_n}, \overrightarrow{d}\})$

Cette propriété permet à un pair ou à une communauté de rajouter un élément à son agrégation sans avoir à recalculer toute l'agrégation.

Démonstration. Ajouter un document ne peut pas supprimer de concept à 1, mais juste en ajouter. □

Théorème 5 (ALL@1 est composable)**.**
L'agrégation est composable*.*
Soient $\{\overrightarrow{d_1^1}, \ldots, \overrightarrow{d_{n_1}^1}\}$ *et* $\{\overrightarrow{d_1^2}, \ldots, \overrightarrow{d_{n_2}^2}\}$, n_1 *et* $n_2 \in \mathbb{N}$ *deux ensembles de documents.*
all@1$(all@1(\{\overrightarrow{d_1^1}, \ldots, \overrightarrow{d_n^1}\}), all@1(\{\overrightarrow{d_1^2}, \ldots, \overrightarrow{d_n^2}\})) = all@1(\{\overrightarrow{d_1^1}, \ldots, \overrightarrow{d_n^1}, \overrightarrow{d_1^2}, \ldots, \overrightarrow{d_n^2}\})$

Cette propriété permet d'obtenir simplement une description de plusieurs pairs : soient p_1 et p_2, ayant les ensembles de documents $\{\overrightarrow{d_1^1}, \ldots, \overrightarrow{d_{n_1}^1}\}$ et $\{\overrightarrow{d_1^2}, \ldots, \overrightarrow{d_{n_2}^2}\}$ n_1 *et* $n_2 \in \mathbb{N}$ respectivement, alors leur agrégation commune est la composition de leurs deux agrégations respectives, sans qu'il soit nécessaire de repasser par les ensembles de documents.

Démonstration. Les concepts à 1 des deux vecteurs sémantiques représentant les agrégations vont se combiner. Il n'est pas possible d'en enlever. □

Nous proposons la définition d'une fonction d'agrégation relative à une requête : AGR4QUERY.

Definition 22 (AGR4QUERY)**.**
Soient Ω *une ontologie et* $\mathcal{C}_\Omega$ *l'ensemble de ses concepts.*
Soit $\{\overrightarrow{d_1}, \ldots, \overrightarrow{d_n}\}$, $n \in \mathbb{N}$ *un ensemble de documents sur* $\mathcal{C}_\Omega$*. Soit* $\overrightarrow{q}$ *une requête sur* $\mathcal{C}_\Omega$*.*
AGR4QUERY *, notée A4Q*$_{\overrightarrow{q}}$*, est une fonction d'agrégation telle que :*

$$\forall c \in \mathcal{C}_\Omega \; A4Q_{\overrightarrow{q}}(\{\overrightarrow{d_1}, \ldots, \overrightarrow{d_n}\}) = \begin{cases} \overrightarrow{q}[c] & \text{si } all@1(\{\overrightarrow{d_1}, \ldots, \overrightarrow{d_n}\})[c] = 1 \\ 0 & \text{sinon} \end{cases}$$

Cette définition attribue à chaque dimension partagée entre la requête et l'ensemble de documents la pondération que lui confère la requête. ALL@1 est un masque booléen pour les vecteurs sémantiques des requêtes, qui permet de manière efficace de conserver les dimensions apparaissant dans une collection et de générer AGR4QUERY.

Théorème 6 (AGR4QUERY est une agrégation optimiste par rapport au cosinus)**.**
Soient Ω *une ontologie et* $\mathcal{C}_\Omega$ *l'ensemble de ses concepts.*
Soit $\{\overrightarrow{d_1}, \ldots, \overrightarrow{d_n}\}$, $n \in \mathbb{N}$ *un ensemble de documents quelconque sur* $\mathcal{C}_\Omega$*. Nous avons :* $\forall i,\ 1 \leq i \leq n$ $\cos(\overrightarrow{q}, A4Q_{\overrightarrow{q}}(\{\overrightarrow{d_1}, \ldots, \overrightarrow{d_n}\})) \geq \cos(\overrightarrow{q}, \overrightarrow{d_i})$

Démonstration.
Notons $\overrightarrow{a} = \text{A4Q}_{\overrightarrow{q}}(\{\overrightarrow{d_1}, \ldots, \overrightarrow{d_n}\})$.
Supposons qu'il existe un document $\overrightarrow{d_i}$, $i \in [1..n]$, dans l'agrégation qui ait un cosinus avec la requête supérieur à celui de l'agrégation avec la requête $\overrightarrow{q}$. Supposons que l'agrégation a k concepts en commun avec la requête : $|\mathcal{C}_{\overrightarrow{q}}| = l > k$. Supposons que $\overrightarrow{d_i}$ ait j concepts en commun avec la requête. Par définition, nous savons que $\{c_1, \ldots c_j\} \subseteq \{c_1, \ldots c_k\} \subseteq \{c_1, \ldots c_l\}$, car les concepts du documents pondèrent ALL@1, qui permet d'avoir une pondération dans AGR4QUERY, seulement si la requête en a

une aussi.

$$\begin{aligned}
\cos(\overrightarrow{q},\overrightarrow{a}) &< \cos(\overrightarrow{q},\overrightarrow{d_i}) \\
\frac{\overrightarrow{q}\cdot\overrightarrow{a}}{|\overrightarrow{q}|\times|\overrightarrow{a}|} &< \frac{\overrightarrow{q}\cdot\overrightarrow{d_i}}{|\overrightarrow{q}|\times|\overrightarrow{d_i}|} \\
\frac{\overrightarrow{q}\cdot\overrightarrow{a}}{|\overrightarrow{a}|} &< \frac{\overrightarrow{q}\cdot\overrightarrow{d_i}}{|\overrightarrow{d_i}|} \\
\frac{\sum_{c\in\{c_1,\dots c_k\}}\overrightarrow{a}[c]^2}{\sqrt{\sum_{c\in\{c_1,\dots c_k\}}\overrightarrow{a}[c]^2}} &< \frac{\overrightarrow{q}\cdot\overrightarrow{d_i}}{|\overrightarrow{d_i}|}
\end{aligned}$$

Or, $\forall x\in\mathbb{R}\ \frac{x}{\sqrt{x}}=\sqrt{x}$. Donc :

$$\begin{aligned}
\sqrt{\sum_{c\in\{c_1,\dots c_k\}}\overrightarrow{a}[c]^2} &< \frac{\overrightarrow{q}\cdot\overrightarrow{d_i}}{|\overrightarrow{d_i}|} \\
\sqrt{\sum_{c\in\{c_1,\dots c_k\}}\overrightarrow{a}[c]^2} &< \frac{\sum_{c\in\{c_1,\dots c_j\}}\overrightarrow{q}[c]\times\overrightarrow{d_i}[c]}{\sqrt{\sum_{c\in\{c_1,\dots c_j\}}\overrightarrow{d_i}[c]^2}}
\end{aligned}$$

Par définition, les valeurs de $\overrightarrow{a}$ et de $\overrightarrow{q}$ sont identiques pour tous les concepts qu'ils ont en commun, $\{c_1,\dots c_k\}$, et $\{c_1,\dots c_j\}\subseteq\{c_1,\dots c_k\}$:

$$\begin{aligned}
\sqrt{\sum_{c\in\{c_1,\dots c_k\}}\overrightarrow{a}[c]^2} &< \frac{\sum_{c\in\{c_1,\dots c_j\}}\overrightarrow{a}[c]\times\overrightarrow{d_i}[c]}{\sqrt{\sum_{c\in\{c_1,\dots c_j\}}\overrightarrow{d_i}[c]^2}} \\
1 &< \frac{\sum_{c\in\{c_1,\dots c_j\}}\overrightarrow{a}[c]\times\overrightarrow{d_i}[c]}{\sqrt{\sum_{c\in\{c_1,\dots c_k\}}\overrightarrow{a}[c]^2}\times\sqrt{\sum_{c\in\{c_1,\dots c_j\}}\overrightarrow{d_i}[c]^2}} \\
1 &< \cos(\overrightarrow{a},\overrightarrow{d_i})
\end{aligned}$$

Ce qui n'est pas possible, le cosinus prenant ses valeurs dans $[0,1]$. Il n'y a donc pas de document pouvant avoir une valeur de pertinence supérieure à celle de l'agrégation AGR4QUERY. AGR4QUERY est une fonction d'agrégation optimiste. □

Donc, AGR4QUERY ne peut pas indiquer une valeur de pertinence plus faible qu'un de ses documents par rapport à une requête.

Il est très facile d'intégrer ExSI^{2}D à une solution distribuée utilisant AGR4QUERY. AGR4QUERY devient donc :

Definition 23 (AGR4QUERY dans le cadre ExSI^{2}D)**.**
Soient Ω_1 et Ω_2 deux ontologies et $\mathcal{C}_{\Omega_1}$ $\mathcal{C}_{\Omega_2}$ l'ensemble de leurs concepts.
Soit $\{\overrightarrow{d_1},\dots,\overrightarrow{d_n}\},\ n\in\mathbb{N}$ un ensemble de documents sur $\mathcal{C}_{\Omega_2}$. Soit $\overrightarrow{q}$ une requête sur $\mathcal{C}_{\Omega_1}$ et $\mathcal{E}_{\overrightarrow{q}}$ son expansion structurante.

AGR4QUERY *, notée A4Q$_{\overrightarrow{q},\mathcal{E}_{\overrightarrow{q}}}$, est une fonction d'agrégation telle que :*

$$\forall c \in \mathcal{C}_\Omega \; A4Q_{\overrightarrow{q},\mathcal{E}_{\overrightarrow{q}}}(\{\overrightarrow{d_1},\ldots,\overrightarrow{d_n}\})[c] = \begin{cases} \overrightarrow{q}[c] & \textit{si } \overrightarrow{i}_{all@1(\{\overrightarrow{d_1},\ldots,\overrightarrow{d_n}\})}[c] > 0 \\ 0 & \textit{sinon} \end{cases}$$

où $\overrightarrow{i}_{all@1(\{\overrightarrow{d_1},\ldots,\overrightarrow{d_n}\})}$ *est l'image du vecteur* $all@1(\{\overrightarrow{d_1},\ldots,\overrightarrow{d_n}\})$ *par rapport à la requête (cf. définition 9 page 30).*

Par une démonstration assez similaire à la précédente, nous pouvons montrer que AGR4QUERY dans le cadre EXSID est optimiste.

8.2 Éléments pour le routage

Nous présentons dans cette section des propositions pour le routage de requêtes dans un cadre sémantiquement homogène puis hétérogène. Enfin, nous donnons quelques pistes pour l'évaluation d'un systèmes P2P utilisant nos solutions.

8.2.1 Un parcours en profondeur frugal : SPARTANBFS

Dans cette section nous présentons différentes propositions pour le routage de requêtes top-k dans des systèmes P2P non structurés. Toutes les propositions n'utilisent pas la sémantique, et nous le faisons dans un cadre sémantiquement homogène dans cette section. L'ensemble de ces propositions constitue un *algorithme frugal*, c'est-à-dire qui respecte les ressources réseau. Nous donnons le nom de SPARTANBFS à cet algorithme. Une solution très intéressante est le « fusion et retour » (*merge and backward* [APV06b]), noté M'N B. Nous l'utilisons comme base dans SPARTANBFS, en proposant quelques améliorations donc. Dans M'N B, chaque pair transmet ses réponses au pair qui lui a envoyé la requête. Ce dernier fusionne les résultats et les renvoie au pair qui lui avait envoyé la requête, etc. jusqu'à l'initiateur de la requête. La figure 8.1 décrit les différentes étapes du processus de M'N B. Chaque pair transmet sa requête puis attend la réponse de ses descendants pour la transmettre à son pair. Ce processus permet de diminuer de manière importante le nombre de messages sur le réseau et de ne pas surcharger le pair initiateur de requête. En effet, chaque pair ne fait pas transiter les réponses par tout le réseau, mais le renvoie à son ancêtre qui ne renvoie à son tour pas toutes les réponses, mais une fusion de celles-ci. D'autre part, cela permet de ne pas surcharger le pair initiateur de la requête : il ne reçoit des réponses que de ses descendants directs. M'N B permet de réduire les coûts de communications de 35% ou plus.

8.2.1.1 Respecter le TTL

La plupart des algorithmes d'inondation (*flooding*) utilisent un TTL, c'est-à-dire un compteur de durée de vie de tout message, qui se décrémente en passant de pair en pair, jusqu'à être nul, auquel cas le message est supprimé. L'intuition est que tous les pairs situés à une distance $TTL = n$ sont atteints par ces messages. Néanmoins, les algorithmes précisent aussi que les pairs ne traitent pas les requêtes qu'ils ont déjà traitées [APV07], en particulier ils ne les transmettent pas. Imaginons le réseau de la figure 8.2 : le pair A envoie une requête avec un $TTL = 3$ sur le réseau. Supposons que la latence est importante entre les pair A et B. La requête envoyée par A est donc transmise à C, qui la traite, décrémente le TTL, et la transmet à son voisin : B. Si ce dernier reçoit la requête de C en premier, avec un TTL de 2, il la traite et la transmet à D et E qui sont les derniers pairs à la traiter ($TTL = 0$). En recevant la requête du pair A, B se rend compte qu'il l'a déjà traitée et supprime le message. Cependant, les pairs F et G, non

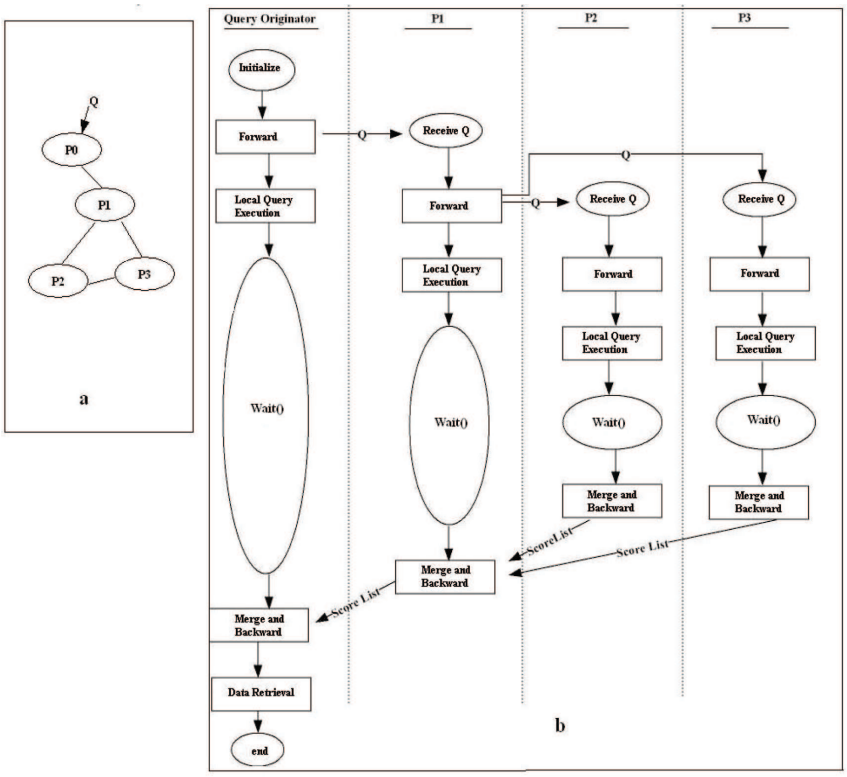


Figure 8.1 – Scénario de l'algorithme du merge and backward [APV06b].

touchés par la requête dans ce scénario, sont bien à une distance de 3 rebonds de A. Dans cette approche classique de l'inondation, il n'y a aucune garantie sur le nombre de pairs atteints par une requête.

Nous proposons dans SPARTANBFS une solution qui assure que tous les pairs situés à une distance donnée d'un initiateur de requête sont bien touchés par cette requête. L'algorithme 5 décrit cette solution. Le nombre de messages est dans le meilleur des cas le même, et en moyenne proche d'une inondation classique. La méthode n'augmente pas particulièrement le temps de calcul : il s'agit juste d'une vérification du TTL le plus fort que la requête courante a eu. Il permet par contre de garantir une propriété intéressante : tous les pairs actifs situés à une certaine distance sont touchés.

8.2.1.2 Retarder les envois pour éviter la surcharge

Comme nous l'avons vu au chapitre précédent (cf. 7.1.2.1 85) un des algorithmes les plus simples mais les moins efficaces pour le routage de requêtes dans un système P2P non structuré est le BFS, parcours en profondeur [JAB01]. Il est inefficace car il est gourmand en ressources. Des solutions permettent d'améliorer ses résultats : soit en choisissant au hasard un certain nombre de voisins ou en sélectionnant ces voisins parmi les derniers ayant bien répondu [KGZY02], soit en élargissant la recherche de plus en plus, en partant d'une interrogation des voisins directs seulement [LCC$^+$02, YGM02].

Dans le cadre de SPARTANBFS, nous proposons de n'utiliser qu'une partie des ressources réseau pour chaque requête. Pour ce faire, nous n'envoyons qu'à certains voisins, quitte à renvoyer plus tard la requête aux autres et en cherchant à envoyer d'abord aux plus « intéressants » des pairs. Le paradigme

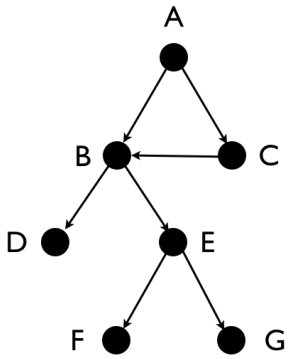


Figure 8.2 – Exemple de réseau P2P.

Algorithme 5 : Permet de faire transiter une requête à tous les pairs situés à une distance TTL d'un initiateur de requête.

```
input : une requête q⃗ avec un TTL.
begin
    if nouvelleRequête( q⃗ ) then
        // q⃗ n'a jamais été traitée
        archiver( q⃗ );
        if TTL > 0 then
            transférerRequête( q⃗ )
        traiterRequête( q⃗ )
    else
        // q⃗ a déjà été traitée
        if getTTL( q⃗ )>getArchiveTTL( q⃗ ) then
            transférerRequête( q⃗ );
            archiver( q⃗ )
end
```

que nous promouvons est d'envoyer en priorité aux pairs les plus intéressants, sans forcément négliger les autres. C'est-à-dire que nous acceptons que des réponses soient retardées, l'envoi à certain pairs étant lui-même retardé. Le but est d'avoir rapidement de bons (meilleurs) résultats, en ayant tous les résultats plus tard et sans charger le réseau. Accepter que tous les résultats n'arrivent pas d'un coup est dans l'esprit des requêtes continues. L'algorithme précise donc qu'à chaque pair, si le TTL n'est pas nul, ce dernier transfère la requête à m de ses n voisins. Ce paramètre peut dépendre de la capacité du nombre de connexions que peut traiter le pair, de la charge actuelle du réseau, de la priorité de la requête, etc.. Par exemple, un pair peu capable de gérer de multiples connexions prendra un m faible, d'autant plus

faible que le réseau est chargé, sauf si la requête est importante, etc. Lorsqu'il reçoit la réponse d'un de ses fils, le pair peut alors transférer la requête à un autre fils non encore sollicité, jusqu'à épuisement de sa liste de descendants. La figure 8.3 décrit le protocole du SPARTANBFS.

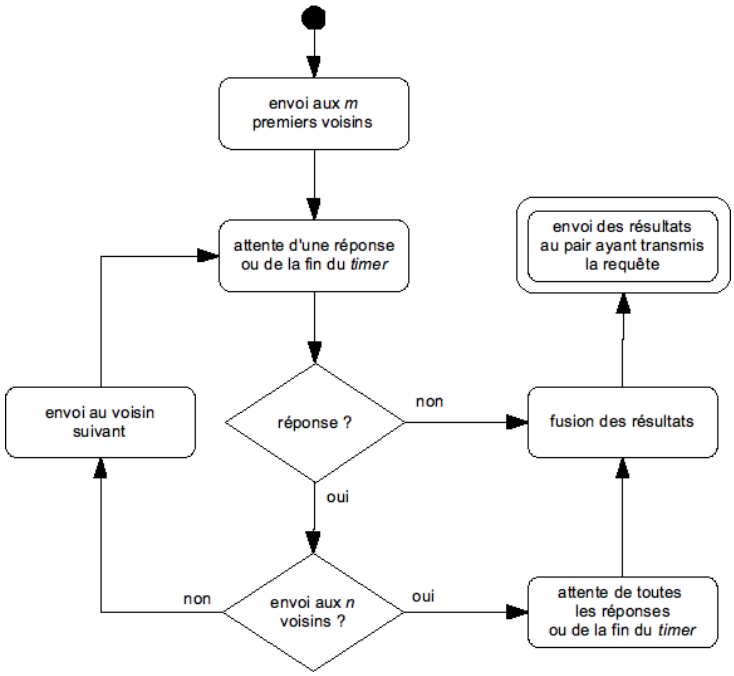


Figure 8.3 – Protocole de base du SPARTANBFS.

Le choix des voisins auxquels envoyer peut se faire de manière aléatoire. Néanmoins, mais il est possible d'utiliser la fonction d'agrégation que nous avons présentée à la section précédente : AGR4QUERY. A partir du vecteur d'agrégation de chacun des voisins, il est possible d'obtenir pour chaque pair p_i une valeur de pertinence en utilisant le cosinus : $\cos(\overrightarrow{q}, \text{A4Q}_{\overrightarrow{q},\mathcal{E}_{\overrightarrow{q}}}(\mathcal{D}_{p_i}))$. Il suffit alors de trier les voisins et d'utiliser cette liste dans l'algorithme de base de SPARTANBFS, pour envoyer en priorité aux voisins ayant la valeur de pertinence la plus importante.

Il est aussi possible d'utiliser des connaissances plus « lointaines » sur le voisinage d'un pair. Si un pair peut connaître l'agrégation de ses descendants directs, il peut aussi avoir celle de ses descendants plus lointains. ALL@1, l'agrégation des documents d'un pair, peut être échangée sur le réseau et aider au routage. Ainsi,un pair p pourrait savoir qu'à une distance de x rebonds logiques de trouve un pair p_i accessible via le pair p_j, dont la représentation semble bonne pour sa requête. L'obtention des représentations des pairs descendants peut se faire avec un algorithme de rumeur, comme dans PlanetP [CAPMN03] en utilisant des filtres de bloom [RK02], une indexation des pairs locaux [CGM02, YGM02], etc.Les pairs peuvent regrouper les informations de leur voisinage à une distance donnée. Par exemple un pair peut décrire ses successeurs à 3 rebonds logiques de distance. Ces descriptions sont des vecteurs sémantiques qui servent à savoir si un chemin est pertinent ou non. Ainsi la liste triée des voisins précédente est reprise en utilisant les descriptions à TTL rebonds logiques. Il suffit d'obtenir les valeurs de tous les pairs situés à moins de TTL rebonds logiques, et de les trier. Ensuite, l'envoi de la requête se fait aux meilleurs voisins dans l'ordre de la liste, en choisissant les voisins pour leur valeur de pertinence ou pour celle de leurs successeurs.

8.2.1.3 Éviter les envois inutiles

L'agrégation optimiste permet que la valeur de pertinence d'une agrégation soit plus forte ou égale à tous les documents de la collection : étant donnée une valeur de pertinence d'une agrégation par rapport à une requête, il n'est pas possible de trouver un document plus pertinent. Or, au cours des recherches top-k, nous disposons d'une liste triée de documents, avec leur valeur de pertinence par rapport à la requête. Nous avons donc la valeur du document avec la plus faible pertinence : le plus bas dans la liste des top-k. Au moment d'interroger un pair, via son agrégation, il est donc possible de savoir s'il a une chance d'avoir des documents s'intégrant dans la liste des top-k. Si $\cos(\overrightarrow{q}, \text{A4Q}_{\overrightarrow{q},\mathcal{E}_{\overrightarrow{q}}}(\mathcal{D}_{p_i})) \geq \cos(\overrightarrow{q}, \overrightarrow{d_{min}})$ avec d_{min} le document situé en bas de la liste top-k, alors il est possible qu'il ait un document intéressant. Sinon, il n'est pas possible qu'un de ses documents ait une valeur supérieure à d_{min}, et il n'est donc pas nécessaire de l'interroger.

Ainsi, si un pair reçoit une requête avec $TTL = 1$, c'est-à-dire que ses successeurs sont les derniers pairs contactés pour cette requête sur le chemin qui le traverse, et qu'il dispose de la valeur minimale pour qu'un document fasse partie de la liste top-k (la valeur la plus basse d'un document dans la liste), il peut savoir quels pairs contacter, et *couper* certains chemins. La figure 8.4 illustre ce processus. Le pair p_1 reçoit une requête avec un $TTL = 1$: ses successeurs sont donc des « feuilles » du graphe de recherche. Il connaît la valeur du document situé le plus bas dans la liste top-k actuelle. Il calcule alors le cosinus de l'agrégation de chacun de ses successeurs p_2, p_3 et p_4 et coupe le chemin menant à celui qui a une valeur inférieure à celle du document le plus faible de la liste top-k. p_3 ne peut en effet pas avoir de document pouvant s'intégrer dans la liste.

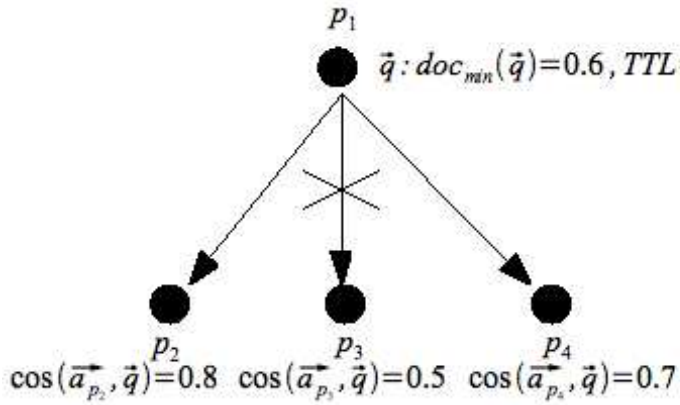


Figure 8.4 – Méthode de coupe pour les chemins vers des feuilles dans SPARTANBFS.

Le protocole de la figure 8.5 décrit le mécanisme mis en œuvre pour couper les chemins vers les successeurs en connaissant leurs agrégations et la valeur du document le plus bas dans la liste top-k.

Si les pairs disposent d'une connaissance du réseau à plus d'une distance de 1 rebond logique, il est possible que les coupes ne concernent pas uniquement les feuilles, mais tout un chemin. En effet, dans le processus de fusion de réponses et de remontées des réponses fusionnées (*merge and backward*) dans SPARTANBFS que nous avons décrit pour l'instant, les pairs attendent les réponses de leurs premiers voisins avant d'envoyer à des voisins suivants. En fusionnant les réponses courantes, le pair peut donc mettre à jour la valeur minimale pour entrer dans la liste top-k, et la soumettre au successeur suivant. Il peut aussi l'utiliser pour savoir si un pair (et tous ses descendants) sont intéressants ou non. Et donc

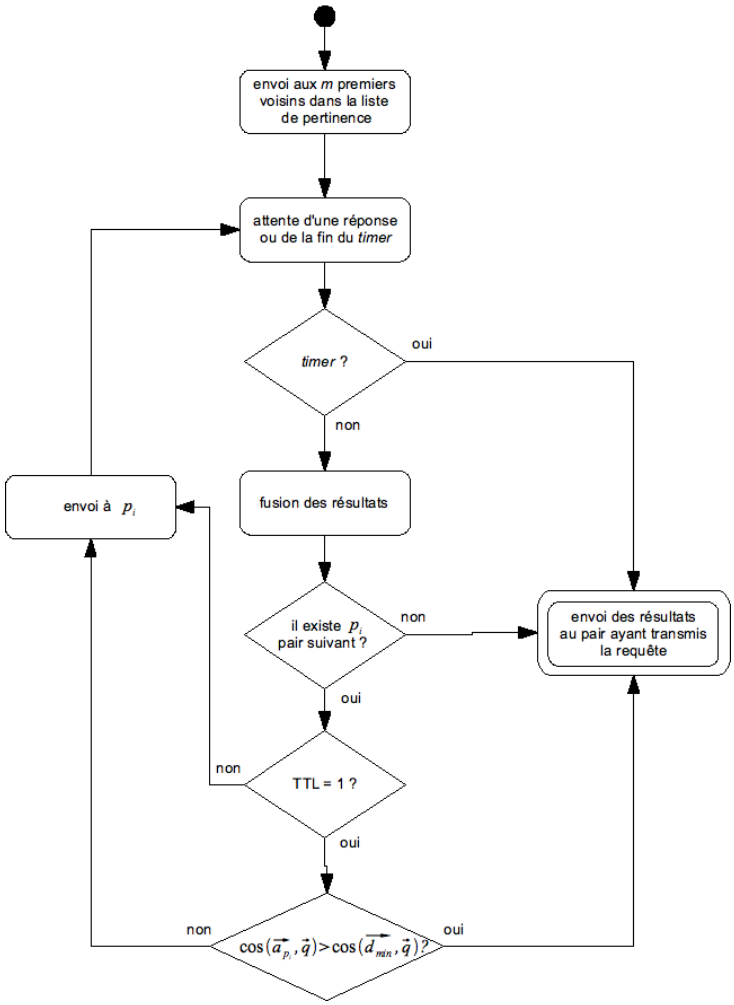


Figure 8.5 – Protocole de base du SPARTANBFS avec coupe dans les chemins vers des voisins.

couper plus que le chemin vers une feuille, mais celui vers toute une série de descendants à partir d'un successeur. Nous notons $descr(p_i)$ la « puissance descriptive » d'un pair, c'est-à-dire la profondeur de voisinage qu'il indexe dans son agrégation. Typiquement, un pair qui n'agrège que ses documents a une valeur de $descr(p_i) = 0$. Nous nous servons de cette valeur dans l'algorithme 8.6. Il diffère de l'algorithme 8.5 par le test sur le TTL et la profondeur d'agrégation : si un pair agrège ses descendants sur un nombre de rebonds au moins aussi grand que le TTL actuel et qu'il a une valeur de pertinence inférieure à d_{min}, alors il est possible de couper tout son sous-arbre de voisinage.

8.2.2 ExSI²D dans les systèmes distribués

Nous avons pour l'instant présenté SPARTANBFS dans un environnement sémantiquement homogène, utilisant $\overrightarrow{q}$ et $\mathcal{E}_{\overrightarrow{q}}$ plutôt que $\mathcal{I}_{\overrightarrow{q}}$. L'utilisation d'ExSI²D plutôt qu'ExSID dans un système distribué, et en particulier SPARTANBFS, n'est pas évidente. En effet, les parties partagées des ontolo-

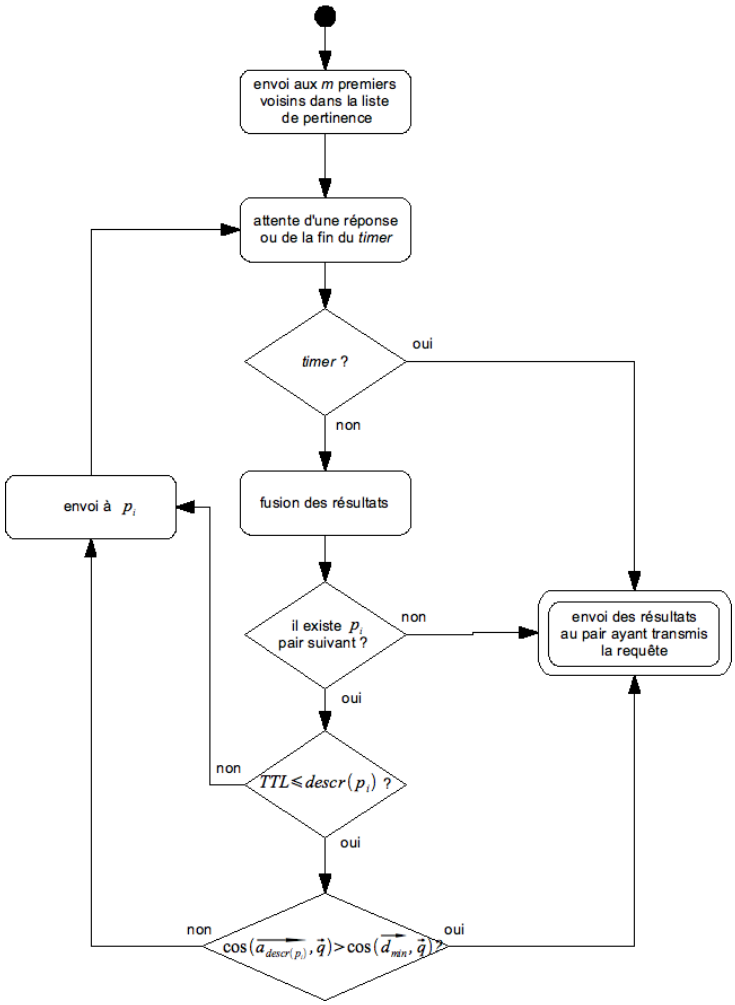


Figure 8.6 – Protocole de base du SPARTANBFS avec coupe dans les chemins vers des voisins ayant des descriptions de leur voisinage.

gies ne sont pas les mêmes sur tout le réseau, et après quelques envois de requêtes, il est possible qu'un fournisseur n'ayant aucune partie commune dans son ontologie avec celle de l'initiateur de requête soit interrogé.

Lorsqu'il n'y a pas beaucoup de mappings entre les ontologies, il n'y a pas beaucoup de raisons qu'ils vérifient la relation suivante : $(\mathcal{C}_{\Omega_1} \triangleright \mathcal{C}_{\Omega_2}) \triangleright \mathcal{C}_{\Omega_3} = \mathcal{C}_{\Omega_1} \triangleright \mathcal{C}_{\Omega_3}$. Une solution pourrait d'utiliser l'interprétation de l'expansion structurante plutôt que l'expansion structurante : $\mathcal{I}_{\overrightarrow{q}}$ plutôt que $\mathcal{E}_{\overrightarrow{q}}$. En effet, si p_1 a pour voisin p_2, qui a lui-même pour voisin p_3, nous pouvons supposer facilement qu'il existe des mappings entre $\mathcal{C}_{\Omega_1}$ et $\mathcal{C}_{\Omega_2}$ et entre $\mathcal{C}_{\Omega_2}$ et $\mathcal{C}_{\Omega_3}$. Si p_1 émet une requête $\overrightarrow{q}$, avec son expansion structurante $\mathcal{E}_{\overrightarrow{q}}$, alors p_2, un de ses voisins, peut la recevoir et l'interpréter. p_2 dispose maintenant de $\overrightarrow{q}$, $\mathcal{E}_{\overrightarrow{q}}$ et $\mathcal{I}_{\overrightarrow{q}}$. $\mathcal{I}_{\overrightarrow{q}}$ est exprimée sur $\mathcal{C}_{\Omega_2}$. Et puisque p_2 a pour voisin p_3, c'est qu'il existe des mappings entre les ontologies des deux pairs : $\mathcal{C}_{\Omega_2}$ et $\mathcal{C}_{\Omega_3}$. Il est donc possible que p_3 interprète à son tour $\mathcal{I}_{\overrightarrow{q}}$. Mais comme nous l'avons dit, cela peut générer des problèmes quand les interprétations dénaturent

finalement trop la requête d'origine.

Notre proposition est donc que les pairs fassent transiter $\overrightarrow{q}$, $\mathcal{E}_{\overrightarrow{q}}$ et la dernière $\mathcal{I}_{\overrightarrow{q}}$ à chaque rebond logique. $\mathcal{E}_{\overrightarrow{q}}$ peut permettre de mesurer le degré de cohérence, du point de vue de la requête, entre les ontologies Ω_1 et l'ontologie courante Ω_n dans un routage de requête. Par exemple, en mesurant le nombre de concepts apparaissant dans l'expansion et dans Ω_n. Un seuil, défini par le système ou par les utilisateurs, permet de savoir à partir de quel degré d'hétérogénéité il n'est pas possible d'exécuter la requête (mais il est toujours possible de la transférer). Dans la figure 8.7 nous pouvons voir quatre pairs avec leurs ontologies et les valeurs de cohérence entre leurs ontologies. Lorsque p_1 envoie une requête $\overrightarrow{q}$, il indique un seuil de cohérence en-dessous duquel la requête n'est pas transmise au pair suivant. Il peut transmettre la requête ($\overrightarrow{q}$ et $\mathcal{E}_{\overrightarrow{q}}$) à p_2 car la valeur de cohérence entre Ω_1 et Ω_2 est supérieure au seuil. p_2 interprète la requête par rapport à son ontologie, et la transmet à p_3 et p_4, mais seul p_4 l'exécute, car Ω_3 a une valeur de cohérence avec Ω_1 trop faible. p_4 peut interpréter l'interprétation de p_2 dans sa propre ontologie et transmettre à ses voisins, comme $p3$. Etc.

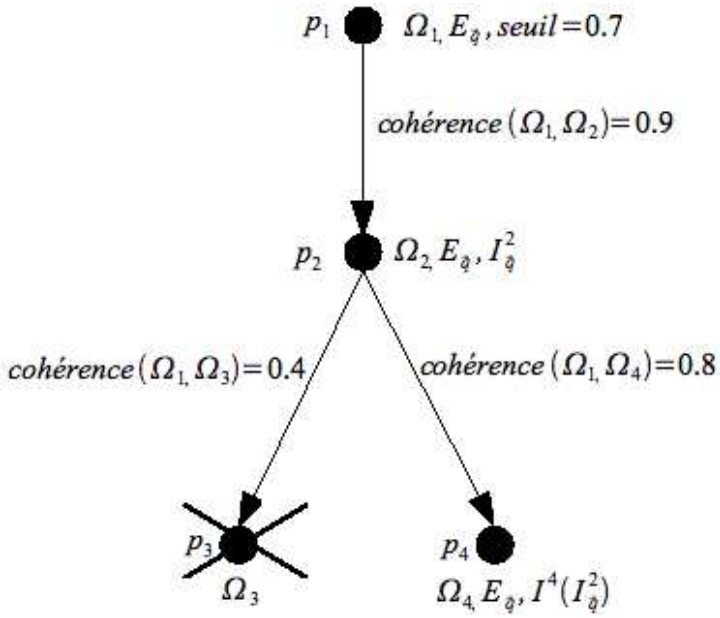


Figure 8.7 – Routage de requête avec vérification de la valeur de cohérence des ontologies avec celles de l'initiateur de requête.

La seule autre modification de SPARTANBFS est le calcul valeur de pertinence d'un pair par rapport à une requête. Il n'utilise pas $\cos(\overrightarrow{q}, \text{A4Q}_{\overrightarrow{q},\mathcal{E}_q}(\mathcal{D}_{p_i}))$ pour tout pair p_i, mais suivant les cas :

- $\cos(\overrightarrow{q}, \text{A4Q}_{\overrightarrow{q},\mathcal{E}_{\overrightarrow{q}}}(\mathcal{D}_{p_i}))$ pour les voisins de l'initiateur de requête ;
- $\cos(\overrightarrow{q}, \text{A4Q}_{\overrightarrow{q},\mathcal{I}^n(\mathcal{I}^{n-1}(\ldots(\mathcal{I}^1(\mathcal{I}_{\overrightarrow{q}}))\ldots))}(\mathcal{D}_{p_i}))$.

Avec $p_1, \ldots p_n$ le chemin de pairs dont l'ontologie est cohérente avec celle de l'initiateur de requête jusqu'à p_i.

8.2.3 Éléments pour l'évaluation

Nous n'avons pas eu le temps d'évaluer notre approche dans le cadre distribué. Notre idée était d'utiliser PeerSim [www], et nous avons pour cela fait une étude de cette plateforme de simulation de réseaux P2P. C'est ce que nous allons vous présenter.

PeerSim est un simulateur de réseaux dynamiques à grande échelle. Il fournit une architecture permettant de tester des *protocols* dans un contexte P2P : routage, topologie, etc. Il permet d'implémenter toutes sortes d'algorithmes de placement des pairs et de comportement lors de l'envoi de messages. Un certain nombre de réseaux classiques (Pastry, Chord, etc.) sont implémentés et peuvent être appelés de façon transparente. Il est très facile de développer sa propre classe, par exemple le comportement d'un pair lors de la réception d'un message, et de l'intégrer à PeerSim. Un fichier de configuration permet de lier tous les *protocols* (topologie, couche de transport, routage, etc.) et de mettre en place les données le test : nombre de pairs, dynamicité, nombre de requêtes, etc. Cependant, il n'existe pas de gestion de la charge des pairs et du réseau ! Nous avons donc proposé deux *protocols* : le premier permettant de gérer la charge des pairs et l'autre la charge du réseau. Il existe deux modes de simulation : événementiel ou par cycle. Nous avons choisi le premier mode parce qu'il est plus adapté à l'envoi de messages tel que nous le concevons.

8.2.3.1 Gérer la charge des pairs avec PeerSim

L'idée générale est d'interférer très peu avec les développements des *protocols*. Il suffit d'ajouter des méthode `work(délai)` avec le délai de travail du pair pour effectuer une certain nombre d'action. Ce délai correspond au temps logique de PeerSim. Il est possible de le considérer de différents points de vus : millisecondes, etc. Il peut y avoir plusieurs méthodes `work` par *protocol*. Les messages qui sont envoyés par le pair tiennent compte de ces délais. Ainsi, si dans le code d'un algorithme de routage un délai total de 30 est indiqué par deux méthodes `work` avant un envoie de message, ce dernier sera envoyé après 30. Nous proposons une classe abstraite dont le comportement est celui de la figure 8.8. La classe abstraite donne la main à l'algorithme proprement dit qui traite la requête et fait des appels à la méthode `work`. Toutes les requêtes qui doivent être envoyées le sont avec un délai suffisant, et le pair est dans un état endormi le temps que le traitement soit « effectué », c'est-à-dire que la totalité du délai soit épuisé. La classe abstraite est alors prête à traiter les requêtes qui seraient arrivées entre temps ou qui vont arriver.

Plus précisement, nous avons l'algorithme 6 qui définit le comportement de la classe abstraite à l'arrivée d'un message à chaque pair.

8.2.3.2 Gérer la charge du réseau

Nous considérons le réseau comme un sac dans lequel les messages sont envoyés et qui se remplit jusqu'à être surchargé. C'est une estimation un peu grossière, mais qui permet déjà de voir apparaître des phénomènes de surcharge réseau. Quand le réseau est surchargé, les messages ne peuvent pas être délivrés, et doivent attendre que la charge du réseau descende en dessous du seuil de surcharge. Nous utilisons une liste de cellule $(charge, tempsDeLivraison)$ de tous les messages actuellement sur le réseau. Cette liste est utilisée pour savoir la charge actuellement sur le réseau. À chaque nouveau message, cette liste est parcourue et les messages qui ont été délivrés par le simulateur sont enlevés. Les messages qui restent sont ceux qui sont actuellement dans le réseau et qui permettent de connaître la charge actuelle. Pour un envoi de message, il faut donc connaître la charge actuelle, et selon qu'elle soit inférieure ou supérieure au seuil, envoyer le message (en utilisant la latence entre les neouds et la taille du message pour savoir quand il peut être délivré) ou l'envoyer quand le réseau ne sera plus saturé (en parcourant les

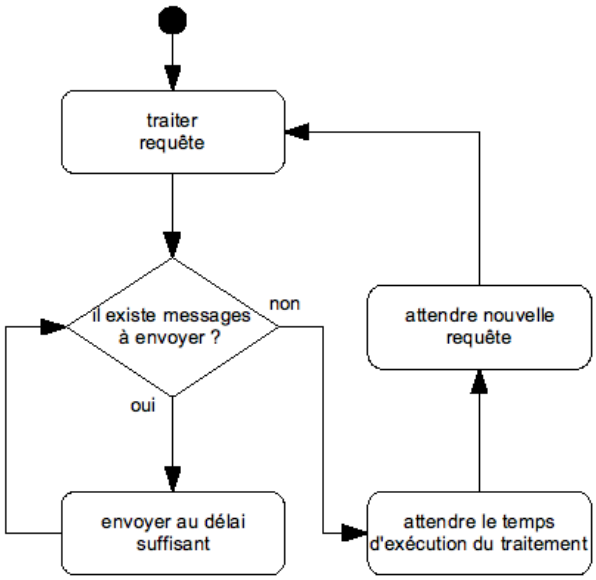


Figure 8.8 – *Protocol* permettant de gérer la charge d'un pair dans PeerSim.

Algorithme 6 : LMProtocol : gère la charge des pairs.

```
input  : m : un message reçu; p=le pair concerné
begin
    if m=message de réveil then
        setCharge(0);
        setRéveillé(1);
        if listeRequêtes≠ 0⃗ then
            utiliserAlgorithme(p,listeRequêtes.première());
            setRéveillé(0);
            envoyerRéveil(p,délai);
            supprimer(listeRequêtes.première());
    else if m=message à transférer then
        envoi(coucheTransport,m);
    else
        if p.getRéveillé()=1 then
            utiliserAlgorithme(p,m);
            setRéveillé(0);
            envoyerRéveil(p,délai);
        else
            listeRequêtes.ajouter(m);
end
```

premières cellules et regardant le moment où une livraison désaturera le réseau). C'est le simulateur qui se charge de passer le message au pair destination au temps qui lui est indiqué (cf. algorithme 7).

Algorithme 7 : LMTransport : gère la charge du réseau.

```
input  : m : un message reçu; seuil = le seuil de surcharge
begin
    chargeAcuelle ← 0;
    forall cellule c_i:(charge,tempsDeLivraison) ∈ listeChargesTriée do
        if c_i.getTempsDeLivraison()< tempsActuel then
            supprimer(c_i);
        else
            chargeActuelle←chargeActuelle + c_i.getCharge();
    délai←tempsPropagationEtEmission(m.source,m.destination);
    if chargeActuelle+m.getCharge>seuil then
        i←0;
        repeat
            chargeActuelle←chargeActuelle - c_i.getCharge();
            i←i++;
        until chargeActuelle+m.getCharge>seuil ;
        délai←délai+(c_i.getTempsDeLivraison()-tempsActuel);
    Simulator.envoyer(m,délai,m.source,m.destination);
    listeChargesTriée.ajouter(m.getCharge(),tempsActuel+délai);
end
```

La gestion de la charge des pairs et du réseau est une première étape essentielle aux simulations d'un système P2P. D'autant plus que notre solution accepte de retarder l'arrivée des résultats pour ne pas surcharger le réseau. Sans charge sur les pairs et le réseau, elle n'est pas mise en valeur.

Nous avons présenté dans ce chapitre des pistes pour l'utilisation d'EXSI^{2}D dans un système distribué et hétérogène. Les solutions que nous avons définies permettent de créer une agrégation optimiste d'un ensemble de documents. Que ce soit un vecteur sémantique permet d'utiliser les mêmes processus pour comparer documents et requêtes mais aussi pairs et requêtes. SPARTANBFS regroupe différentes propositions pour une utilisation d'EXSI^{2}D dans un système d'information distribué et hétérogène.

Conclusion

Dans cette thèse nous avons proposé une solution générale pour adresser l'hétérogénéité sémantique dans un système d'information distribué et hétérogène. Le problème de l'hétérogénéité sémantique est complexe, et nous avons pu montrer qu'il dépassait la mise en correspondance entre ontologies. En effet, les correspondances ne sont pas toujours complètes, et il existe des cas, pourtant simples du point de vue humain, où les approches classiques sont peu adaptées. Le raisonnement humain y arrive lui, et même s'il ne dispose pas de toutes les connaissances utilisées dans un document, il peut *interpréter* et comprendre (au sens étymologique : *cum-prendere*, mettre ensemble) ce qui est dit. Un exemple simple est celui des idiomatismes, ces concepts, tournures ou expressions particuliers à une langue et intraduisibles. Ce n'est pas parce qu'il est impossible de traduire directement « faire d'une pierre deux coups » qu'un anglais ne pourra pas comprendre l'expression, ni parce que le concept de *polis* ($\pi\acute{o}\lambda\iota\varsigma$) est très particulier qu'il n'est pas possible d'imaginer ce qu'il représentait pour les Grecs de l'antiquité. Lors de la « controverse de Valladolid », où entre autres se posait la question de savoir si les Amérindiens étaient humains, Sepúlveda, s'appuyant sur une remarque d'Aristote (« le rire est le propre de l'homme » [Ari03]) propose d'offrir un spectacle comique à un groupe d'Amérindiens. Devant leur impassibilité face aux bouffons européens, il se lève pour prononcer un argumentaire contre leur condition humaine, se prend les pieds dans son manteau et tombe. Les Amérindiens rient (et se sauvent par la même occasion). Bien qu'ils n'aient pas les mêmes connaissances, les mêmes référents culturels, les êtres humains peuvent se comprendre. Il leur faut apprendre la manière de penser des autres, se mettre à leur place, interpréter leur pensée. C'est l'idée générale de notre travail : comment permettre à des utilisateurs disposant de connaissances proches mais différentes de se comprendre.

Dès la première partie de notre travail, nous nous sommes placés dans le cadre des vecteurs sémantiques, un des mieux à même selon nous de représenter les documents. Cependant, l'espace ainsi généré ne rend pas compte des similarités entre dimensions. Nous avons proposé EXSID, une solution basée sur (1) une nouvelle notion d'expansion structurante et (2) la définition de l'image de documents au travers de cette expansion de requête. Au contraire des solutions classiques pour corriger l'indépendance des dimensions, elle ne propose pas d'ajouter de nouvelles dimensions à la requête mais de regrouper plusieurs dimensions. Il est important de noter que les mesures de pertinence classiques, le cosinus par exemple, restent applicables. Nous avons mis en place un processus d'évaluation pour des solutions sémantiques en recherche d'information. C'est à notre connaissance un travail unique. Pour ce faire, nous avons dû étudier des solutions de similarité sémantique entre concepts d'une même ontologie, de propagation de l'intérêt d'un concept à l'autre. Pour mettre en œuvre les tests nous avons aussi dû travailler sur une méthode d'indexation sémantique ainsi que sur le choix d'un corpus de test. Avec EXSID nous avons obtenu des résultats qui ne sont pas dégradés par rapport aux solutions classiques, et les améliore plutôt quand les paramètres sont choisis judicieusement. Il est donc tout à fait possible d'intégrer EXSID dans un moteur de recherche pour permettre aux utilisateurs d'étendre certaines dimensions de leurs requêtes.

Dans la deuxième partie nous avons montré qu'il n'est pas possible d'obtenir une seule ontologie de référence dans un environnement ouvert et distribué. Les solutions proposant des correspondances entre ontologies ont des limites, techniques (impossibilité de réaliser certaines correspondances), « déontiques » (interdiction de réaliser certaines correspondances), ou autres. L'hétérogénéité sémantique résiduelle n'est pas adressée par les solutions classiques. Selon nous, il est possible de laisser les pairs décrire leurs données et leurs requêtes avec leurs propres ontologies, qui ne sont que partiellement par-

tagées avec les autres participants d'un système d'information distribué, et de les faire se comprendre. L'expansion structurante génère en effet des dimensions sémantiquement enrichies qui sont autant d'explications sur la façon dont l'initiateur de requêtes conçoit les dimensions principales de sa requête. Nous avons proposé donc la solution ExSI^{2}D qui complète ExSID avec un module d'interprétation des dimensions sémantiquement enrichies de la requête. Il s'agit pour les fournisseurs d'information d'exprimer les dimensions de la requête avec leurs propres connaissances, en tenant compte de ce que l'initiateur de la requête a indiqué. Nous n'avons pas trouvé de « benchmark » dans la littérature pour juger de l'impact de l'hétérogénéité sémantique dans les systèmes distribués hétérogène. Nous avons donc mis en place un protocole d'évaluation pour ce cadre. Nous avons proposé deux séries d'expérimentations, l'une faisant varier le degré de non-correspondance entre ontologies, de 0% à 100% ; la seconde n'éliminant les correspondances que sur les concepts principaux des requêtes. Dans les deux cas ExSI^{2}D a eu d'excellents résultats : jusqu'à 70% de mappings manquant entre les ontologies de l'initiateur de la requête et du fournisseur d'information, ExSI^{2}D a toujours plus de 80% d'efficacité dans les recherches, alors que les autres méthodes s'écroulent (aux alentours de 40%) pour la première évaluation ; et pour la seconde ExSI^{2}D a plus de 90% d'efficacité alors que les autres méthodes ont entre 0 et 15% d'efficacité. Il faut noter que dans nos tests nous avons utilisé la même mesure de similarité sémantique pour l'initiateur de la requête et pour le fournisseur d'information (une similarité sémantique plutôt « structurelle »), et que les structures de leurs ontologies sont identiques. Néanmoins, nous avons pu montrer que cela n'induisait pas toujours de meilleurs résultats (en particulier pour la seconde évaluation). Dans tous les cas, il nous a fallu inventer un processus expérimental et faire des choix parmi toutes les possibilités qui s'offraient à nous. Par exemple, nous ne savons pas vraiment comment évoluent les ontologies, si c'est plutôt la structure qui est modifiée ou plutôt les concepts entre deux ontologies de domaines. Nos choix sont discutables, mais permettent de mettre en évidence qu'ExSI^{2}D est une solution très prometteuse. D'autre part, nous avons remarqué dans cette partie toute l'importance de la mesure de similarité sémantique pour l'interopérabilité sémantique. À notre connaissance c'est un point peu étudié dans la littérature, mais qui mériterait plus d'attention, et qui est pleinement pris en compte dans ExSI^{2}D. Notre solution s'appuie sur ExSID et nous avons montré qu'elle ne la modifiait pas dans le cadre homogène sémantiquement. Cette modularité permet aussi à ExSI^{2}D de s'intégrer facilement dans un système d'information existant utilisant des vecteurs sémantiques.

La troisième partie s'est attachée à proposer des solutions pour l'utilisation de ExSI^{2}D dans un cadre P2P. L'avantage de notre solution est qu'elle peut s'adapter à toutes sortes d'architectures P2P et d'approches d'intégration sémantique. Nous avons montré que les architectures P2P non structurées sont celles qui laissent la plus grande autonomie aux pairs, tout en les laissant égaux et assurant une tolérance aux pannes complète. Le passage à l'échelle et l'efficacité sont cependant compromis avec les algorithmes de base de cette solution. Parmi les solutions d'intégration sémantique, celles ne demandant pas d'accord entre les pairs pour la création d'une ontologie (abstraite) globale mais créant des correspondances entre les ontologies laissent la plus grande autonomie aux pairs mais ne sont limitées par les correspondances (qui ne sont pas totales). Nous avons proposé SPARTANBFS, un premier algorithme pour la gestion sémantique dans un réseau non structuré avec des correspondances entre les ontologies des pairs. Il a pour objectif de diminuer les messages échangés pour le routage de requête afin d'augmenter l'efficacité et le passage à l'échelle d'un système d'information fortement distribué, et d'améliorer les correspondances entre pairs en utilisant ExSI^{2}D.

Toutes nos solutions ont un objectif commun, qui est de rendre plus interopérables les systèmes d'information distribués et hétérogènes. Elles proposent de laisser les participants au système d'information libres d'utiliser tous leurs concepts, y compris ceux qu'ils ne partagent pas avec les autres participants. Les initiateurs de requêtes décrivent dans leur propre ontologie, avec leur propre mesure de similarité et

leur propre fonction de propagation les dimensions de leurs requêtes. Ce processus d'enrichissement des dimensions permet de lier d'autres concepts de l'ontologie au concepts centraux de la requête. Les fournisseurs d'information peuvent alors interpréter les dimensions enrichies dans leur ontologie, avec leur mesure de similarité sémantique, en essayant de trouver ce que l'initiateur de requête aurait pondéré dans leur propre ontologie. Finalement l'image des documents indique la présence des concepts principaux de la requête ou de concepts qui leurs sont proches dans les documents.

Parmi les pistes de recherche ouvertes par notre travail, nous trouvons la personnalisation dans les systèmes de RI. Bordogna et Pasi [BP05] présentent un modèle qui permet d'indexer différemment un document suivant différents utilisateurs. Notre notion d'image est proche de cette idée d'une représentation différente suivant les besoins ou les souhaits d'un utilisateur. Dans notre approche, nous ne modifions pas l'indexation des documents à proprement parler, mais nous créons une nouvelle indexation, dépendante de l'expansion structurante de la requête. Deux utilisateurs mettant en place des expansions différentes ont des images différentes pour les documents, et donc des classements de pertinence différents. À bien y réfléchir, EXSI^{2}D semble définir lui aussi un modèle de personnalisation de l'information, bien que nous ne l'avons pas défini dans cet objectif. L'utilisateur peut définir dans ses DSEs le point de vue qu'il a sur la collection de documents.

De nouvelles expérimentations sont à mener pour mesurer la robustesse de notre système. Nous sommes en train d'utiliser un corpus important, le corpus W3C de TREC, pour tester EXSID sur une grande quantité de documents. De même, nous sommes en court de validation de notre approche SPARTANBFS sur la plate-forme de simulation PeerSim. Après avoir proposé quelques classes pour améliorer la gestion de la charge des pairs et la charge du réseau dans PeerSim, nous sommes en train de définir les paramètres de nos évaluations. Sans doute distribuerons-nous le corpus TREC sur un réseau P2P non structuré. Toutes ces expérimentations devraient être effectuées à court terme. L'autre voie que nous voulons explorer est l'impact des différentes hétérogénéités sémantiques. La première idée est d'étudier des modifications plus en profondeur les ontologies, en ne supprimant pas uniquement les mappings. C'est un problème difficile, sur lequel nous nous posons de nombreuses questions. Évidemment, nous voulons tester l'impact que cela a sur l'interopérabilité et sur les résultats d'EXSI^{2}D. Ce qui a émergé de notre travail est aussi que l'interopérabilité sémantique est accrue par l'utilisation de similarités sémantiques comparables. Nous voulons donc étudier en général l'importance de la similarité sémantique pour l'interopérabilité, en particulier nous pensons qu'EXSI^{2}D peut servir de pierre de touche pour mesurer la proximité sémantique entre acteurs. Par exemple, il serait intéressant de valider les mappings en effectuant des expansions structurantes/interprétations croisées de concepts en correspondance, pour vérifier qu'il s'agit ou non de mappings corrects. Etc. A plus long terme, il serait bon d'utiliser d'autres représentations que les vecteurs sémantiques pour notre approche.

Bibliographie

[ABC+03] Serge Abiteboul, Angela Bonifati, Grégory Cobéna, Ioana Manolescu, and Tova Milo. Dynamic xml documents with distribution and replication. In *SIGMOD '03: Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, pages 527–538, New York, NY, USA, 2003. ACM.

[ABFS02a] Bernd Amann, Catriel Beeri, Irini Fundulaki, and Michel Scholl. Ontology-based integration of xml web resources. In *International Semantic Web Conference*, pages 117–131, 2002.

[ABFS02b] Bernd Amann, Catriel Beeri, Irini Fundulaki, and Michel Scholl. Querying xml sources using an ontology-based mediator. In *CoopIS/DOA/ODBASE*, pages 429–448, 2002.

[ACG+04] Philippe Adjiman, Philippe Chatalic, François Goasdoue, Marie-Christine Rousset, and Laurent Simon. Somewhere in the semantic web. Technical report, LRI, 2004.

[ACK+02] David P. Anderson, Jeff Cobb, Eric Korpela, Matt Lebofsky, and Dan Werthimer. Seti@home: an experiment in public-resource computing. *Communications of the ACM*, 45(11):56–61, 2002.

[ACMH03] Karl Aberer, Philippe Cudré-Mauroux, and Manfred Hauswirth. The chatty web: emergent semantics through gossiping. In *WWW '03: Proceedings of the 12th international conference on World Wide Web*, pages 197–206, New York, NY, USA, 2003. ACM.

[ACMHP04] Karl Aberer, Philippe Cudré-Mauroux, Manfred Hauswirth, and Tim Van Pelt. Gridvine: Building internet-scale semantic overlay networks. In *International Semantic Web Conference*, pages 107–121, 2004.

[AE03] M. Andrea Rodríguez and Max J. Egenhofer. Determining semantic similarity among entity classes from different ontologies, 2003.

[AG08] Nathalie Aussenac-Gilles. Le web sémantique, quel renouvellement pour la recherche d'information ? In Mohand Boughanem and Jacques Savoy, editors, *Recherche d'information : état des lieux et perspectives*, pages 231–266, Paris, 2008. Lavoisier.

[AGBS00] Nathalie Aussenac-Gilles, Brigitte Biebow, and Sylvie Szulman. Revisiting ontology design: A methodology based on corpus analysis. In *EKAW*, pages 172–188, 2000.

[Agn07] Vijay Srinivas Agneeswaran. A Survey of Semantic Based Peer-to-Peer Systems. Technical report, LSIR, 2007. communicated to International Journal of Computer Science and Software Technology.

[AM07] Reza Akbarinia and Vidal Martins. Data management in the appa system. *Journal of Grid Computing*, 5(3):303–317, 2007.

[And04] David P. Anderson. Boinc: A system for public-resource computing and storage. In *GRID '04: Proceedings of the Fifth IEEE/ACM International Workshop on Grid Computing*, pages 4–10, Washington, DC, USA, 2004. IEEE Computer Society.

[APV06a] Reza Akbarinia, Esther Pacitti, and Patrick Valduriez. An efficient mechanism for processing top-k queries in dhts. In *BDA*, 2006.

[APV06b] Reza Akbarinia, Esther Pacitti, and Patrick Valduriez. Reducing network traffic in unstructured p2p systems using top-k queries. *Journal of Distributed and Parallel Databases*, 19(2-3):67–86, 2006.

[APV07] Reza Akbarinia, Esther Pacitti, and Patrick Valduriez. Query processing in p2p systems. Technical Report 6112, INRIA, France, 2007.

[Ari03] Aristote. *Les parties des animaux*. Belles Lettres, 2003.

[Aus82] Geoffrey Austrian. *Herman Hollerith: Forgotten Giant of Information Processing*. Columbia University Press, 1982.

[BBA05] Mustapha Baziz, Mohand Boughanem, and Nathalie Aussenac-Gilles. Conceptual indexing based on document content representation. In *CoLIS*, pages 171–186, 2005.

[BBM02] Holger Billhardt, Daniel Borrajo, and Victor Maojo. A context vector model for information retrieval. *JASIST*, 53(3):236–249, 2002.

[BCF+01] Luc Bouganim, Maria Claudia Cavalcanti, Françoise Fabret, Maria Luiza Campos, François Llirbat, Marta Mattoso, Rubens Melo, Ana Maria Moura, Esther Pacitti, Fabio Porto, Margareth Simoes, Eric Simon, Asterio Tanaka, and Patrick Valduriez. The ecobase project: database and web technologies for environmental information systems. *SIGMOD Rec.*, 30(3):70–75, 2001.

[BCL+05] Angela Bonifati, Elaine Qing Chang, Aks V. S. Lakshmanan, Terence Ho, and Rachel Pottinger. Heptox: marrying xml and heterogeneity in your p2p databases. In *VLDB '05: Proceedings of the 31st international conference on Very large data bases*, pages 1267–1270. VLDB Endowment, 2005.

[BDJ99] Michael W. Berry, Zlatko Drmac, and Elizabeth R. Jessup. Matrices, vector spaces, and information retrieval. *SIAM Rev.*, 41(2):335–362, 1999.

[BHR95] John A. Bateman, Renate Henschel, and Fabio Rinaldi. Generalized Upper Model 2.0: documentation. Technical report, GMD/Institut für Integrierte Publikations- und Informationssysteme, Darmstadt, Germany, 1995.

[Bid02] Alain Bidault. *Affinement de requêtes posées à un médiateur*. PhD thesis, University Paris XI, Orsay, Paris, France, july 2002.

[BK03] J. Becker and D. Kuropka. Topic-based vector space model. In *Proceedings of the 6th International Conference on Business Information Systems*, pages 7–12, Colorado Springs, July 2003.

[Bla84] David C. Blair. The data-document distinction in information retrieval. *Communication of the ACM*, 27(4):369–374, 1984.

[Bla06] David C. Blair. The data-document distinction revisited. *SIGMIS Database*, 37(1):77–96, 2006.

[BNST05] Wolf-Tilo Balke, Wolfgang Nejdl, Wolf Siberski, and Uwe Thaden. Progressive distributed top-k retrieval in peer-to-peer networks. In *ICDE '05: Proceedings of the 21st International Conference on Data Engineering*, pages 174–185, Washington, DC, USA, 2005. IEEE Computer Society.

[BO03] Brian Babcock and Chris Olston. Distributed top-k monitoring. In *SIGMOD '03: Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, pages 28–39, New York, NY, USA, 2003. ACM.

[BP98] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1–7):107–117, 1998.

[BP05] Gloria Bordogna and Gabriella Pasi. Personalised indexing and retrieval of heterogeneous structured documents. *Information Retrieval*, 8(2):301–318, 2005.

[Bri95] Eric Brill. Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. *Computational Linguistics*, 21(4):543–565, 1995.

[BS08] Mohand Boughanem and Jacques Savoy, editors. *Recherche d'information : état des lieux et perspectives*. Lavoisier, Paris, 2008.

[CA04] Philippe Cudré-Mauroux and Karl Aberer. A necessary condition for semantic interoperability in the large. In *CoopIS/DOA/ODBASE*, 2004.

[CAPMN03] Francisco Matias Cuenca-Acuna, Christopher Peery, Richard P. Martin, and Thu D. Nguyen. PlanetP: Using Gossiping to Build Content Addressable Peer-to-Peer Information Sharing Communities. In *12th IEEE International Symposium on High Performance Distributed Computing (HPDC-12)*. IEEE Press, 2003.

[CFLGP03] Oscar Corcho, Mariano Fernández-López, and Asunción Gómez-Pérez. Methodologies, tools and languages for building ontologies: where is their meeting point? *Data Knowl. Eng.*, 46(1):41–64, 2003.

[CG02] Arturo Crespo and Hector Garcia-Molina. Semantic overlay networks for p2p systems, 2002.

[CGLR04] Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Riccardo Rosati. Logical foundations of peer-to-peer data integration. In *PODS '04: Proceedings of the twenty-third ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 241–251, New York, NY, USA, 2004. ACM.

[CGM02] Arturo Crespo and Hector Garcia-Molina. Routing indices for peer-to-peer systems. In *ICDCS '02: Proceedings of the 22 nd International Conference on Distributed Computing Systems (ICDCS'02)*, page 23, Washington, DC, USA, 2002. IEEE Computer Society.

[DDL+90] Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

[DGY05] Florin Dragan, Georges Gardarin, and Laurent Yeh. Mediapeer: A safe, scalable p2p architecture for xml query processing. In *DEXA Workshops*, pages 368–373, 2005.

[DHA03] Anwitaman Datta, Manfred Hauswirth, and Karl Aberer. Updates in highly unreliable, replicated peer-to-peer systems. In *ICDCS '03: Proceedings of the 23rd International Conference on Distributed Computing Systems*, page 76, Washington, DC, USA, 2003. IEEE Computer Society.

[DJ02] Emmanuel Desmontils and Christine Jacquin. *The Emerging Semantic Web*, volume 75 of *Frontiers in Artificial Intelligence and Applications*, chapter Indexing a web site with a terminology oriented ontology., pages 181–197. IOS press, 2002.

[Dub04] David Dubin. The most influential paper gerard salton never wrote. *Library Trends*, 52(4):748–764, 2004.

[dVMNK02] Arjen P. de Vries, Nikos Mamoulis, Niels Nes, and Martin Kersten. Efficient k-nn search on vertically decomposed data. In *SIGMOD '02: Proceedings of the 2002 ACM SIGMOD international conference on Management of data*, pages 322–333, New York, NY, USA, 2002. ACM.

[Eng85] Pascal Engel. *Identité et référence. La théorie des noms propres chez Frege et Kripke*. Presses de l'École Normale Supérieure, 1985.

[ES04] Marc Ehrig and Steffen Staab. Qom - quick ontology mapping. In *International Semantic Web Conference*, pages 683–697, Hiroshima, Japan, 2004.

[ES07] Jérôme Euzenat and Pavel Shvaiko. *Ontology matching*. Springer-Verlag, Heidelberg (DE), 2007.

[Fel98] Christiane Felbaum. *WordNet : an electronic lexical database*. Bradford Books, March 1998.

[FGJ97] Mariano Fernandez, Asuncion Gomez-Perez, and Natalia Juristo. Methontology: from ontological art towards ontological engineering. In *Proceedings of the AAAI97 Spring Symposium Series on Ontological Engineering*, pages 33–40, Stanford, USA, March 1997.

[FGM+01] Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. Placing search in context: the concept revisited. In *The Tenth International World Wide Web Conference (WWW10)*, pages 406–414, 2001.

[FJP07] Brucker François and Barthélémy Jean-Pierre. *Éléments de classification: aspects combinatoires et algorithmiques*. méthodes stochastiques appliquées. Lavoisier, PAris, 2007.

[FKLZ04] Enrico Franconi, Gabriel Kuper, Andrei Lopatenko, and Ilya Zaihrayeu. Queries and updates in the codb peer to peer database system. In *VLDB '04: Proceedings of the Thirtieth international conference on Very large data bases*, pages 1277–1280. VLDB Endowment, 2004.

[FLGD83] G. W. Furnas, T. K. Landauer, L. M. Gomez, and S. T. Dumais. Statistical semantics: analysis of the potential performance of key-word information systems. *The Bell System Technical Journal*, 1983.

[FLGD87] G. W. Furnas, T. K. Landauer, L. M. Gomez, and S. T. Dumais. The vocabulary problem in human-system communication. In *Communications of the Association for Computing Machinery*, volume 30, pages 964–971, 1987.

[Für04] Frédéric Fürst. *Contribution à l'ingénierie des ontologies, une méthode et un outil d'opérationalisation*. PhD thesis, Université de Nantes, France, 2004.

[Gar08] Delphine Gardey. *Ecrire, calculer, classer. Comment une révolution de papier a transformé les sociétés contemporaines (1800-1940)*. Textes á l'appui. La Découverte, 2008.

[GF95] Michael Grüninger and Mark S. Fox. Methodology for the design and evaluation of ontologies. In *Workshop on Basic Ontological Issues in Knowledge Sharing, held in conduction with IJCAI-95*, Montreal, Canada, 1995.

[GGM92] Terry Gaasterland, Parke Godfrey, and Jack Minker. An overview of cooperative answering. *J. of Intelligent Information Systems*, 1(2):123–157, 1992.

[GKZ08] Fausto Giunchiglia, Uladzimir Kharkevich, and Ilya Zaihrayeu. Concept search: Semantics enabled syntactic search. In *SemSearch*, pages 109–123, 2008.

[GM04] Luis Gravano and Amelie Marian. Optimizing top-k selection queries over multimedia repositories. *IEEE Trans. on Knowl. and Data Eng.*, 16(8):992–1009, 2004. Member-Surajit Chaudhuri.

[Goe08] Lorraine Goeuriot. *Découverte et caractérisation des corpus comparables*. PhD thesis, Université de Nantes, Nantes, 2008.

[GPFC04] A. Gómez-Pérez, M. Fernández, and O. Corcho. *Ontological Engineering*. Springer-Verlag, London, 2004.

[Gru95] Thomas R. Gruber. Toward principles for the design of ontologies used for knowledge sharing. *International Journal of Human-Computer Studies*, 43(5-6):907–928, 1995.

[GVCC98] Julio Gonzalo, Felisa Verdejo , Irina Chugur, and Juan Cigarran. Indexing with wordnet synsets can improve text retrieval. In *Proceedings of the COLING/ACL '98 Workshop on Usage of WordNet for NLP*, pages 38–44, Montreal, Canada, 1998.

[Hab00] Benoît Habert. Des corpus représentatifs : de quoi, pour quoi, comment ? In Mireille Bilger, editor, *Linguistique sur corpus. Études et réflexions*, number 31 in Cahiers de l'université de Perpignan, pages 11–58. Presses Universitaires de Perpignan, Perpignan, 2000.

[Hen02] James Hendler. Ontologies on the Semantic Web. *IEEE Intelligent Systems*, 17(2), 2002.

[HHL+03] Ryan Huebsch, Joseph M. Hellerstein, Nick Lanham, Boon Thau Loo, Scott Shenker, and Ion Stoica. Querying the internet with pier. In *vldb'2003: Proceedings of the 29th international conference on Very large data bases*, pages 321–332. VLDB Endowment, 2003.

[HIMT03] Alon Y. Halevy, Zachary G. Ives, Peter Mork, and Igor Tatarinov. Piazza: data management infrastructure for semantic web applications. In *WWW '03: Proceedings of the 12th international conference on World Wide Web*, pages 556–567, New York, NY, USA, 2003. ACM.

[HJBM+04] Peter Haase, Marc Ehrig Jeen Broekstra, Maarten Menken, Peter Mika, Michal Plechawski, Pawel Pyszlak, Björn Schnizler, Ronny Siebes, Steffen Staab, and

Christoph Tempich. Bibster - a semantics-based bibliographic peer-to-peer system. In *International Semantic Web Conference (ISWC2004)*, pages 349–363, 2004.

[Hod88] Andrew Hodges. *Alan Turin ou l'énigme de l'intelligence*. Payot, Paris, 1988.

[Hus84] Edmund Husserl. *La Logique Formelle Et Transcendantale*. Presses Universitaires de France, Paris, 1984.

[IHMT03] Zachary G. Ives, Alon Y. Halevy, Peter Mork, and Igor Tatarinov. Piazza: mediation and integration infrastructure for semantic web data. *Journal of Web Semantics*, 2003.

[IMvdM+08] Antoine Isaac, Henk Matthezing, Lourens van der Meij, Stefan Schlobach, Shenghui Wang, and Claus Zinn. Putting ontology alignment in context: Usage scenarios, deployment and evaluation in a library case. In *ESWC*, pages 402–417, 2008.

[IV98] N. Ide and J. Véronis. Introduction to the special issue on word sense disambiguation: the state of the art. *Computational Linguistics*, 24(1):1–40, 1998.

[JAB01] M. Jovanovic, F. Annexstein, and K. Berman. Scalability issues in large peer-to-peer networks - a case study of gnutella. Technical report, University of Cincinnati, 2001.

[JC97] J. Jiang and D. Conrath. Semantic similarity based on corpus statistics. In *International Conference on Research in Computational Linguistics*, 1997.

[Jon81] Karen Sparck Jones. *Information Retrieval Experiment*. Butterworth-Heinemann, Newton, MA, USA, 1981.

[Kas02] G Kassel. Ontospec : une méthode de spécification semi-informelle d'ontologies. In *Actes des 13ème journées francophones d'Ingénierie des Connaissances,*, pages 75–87, Rouen, May 2002.

[KB83] Donald H. Kraft and Duncan A. Buell. Fuzzy sets and generalized boolean retrieval systems. *International Journal of Man-Machine Studies*, 19(1):45–56, 1983.

[KC92] Robert Krovetz and W. Bruce Croft. Lexical ambiguity and information retrieval. *Information Systems*, 10(2):115–141, 1992.

[KGZY02] Vana Kalogeraki, Dimitrios Gunopulos, and D. Zeinalipour-Yazti. A local search mechanism for peer-to-peer networks. In *CIKM '02: Proceedings of the eleventh international conference on Information and knowledge management*, pages 300–307, New York, NY, USA, 2002. ACM.

[KL94] Kevin Knight and Steve K. Luk. Building a large-scale knowledge base for machine translation. In *AAAI '94: Proceedings of the twelfth national conference on Artificial intelligence (vol. 1)*, pages 773–778, Menlo Park, CA, USA, 1994. American Association for Artificial Intelligence.

[KlRJ07] C. Keß ler, M. Raubal, and K. Janowicz. The effect of context on semantic similarity measurement. In *3rd International IFIP Workshop On Semantic Web & Web Semantics (SWWS'07)*, 2007.

[KS03] Yannis Kalfoglou and Marco Schorlemmer. Ontology mapping: the state of the art. *Knowledge Engineering Review*, 18(1):1–31, 2003.

[Lan68] F.Wilfrid Lancaster. *Information Retrieval Systems: Characteristics, Testing and Evaluation*. John Wiley & Sons, 1968.

[LC98] Claudia Leacok and Martin Chodorow. Combining local context and and wordnet similarity for word sense identification. In Christiane Fellbaum, editor, *WordNet: An electronic lexical database and some of its applications*. The MIT Press, 1998.

[LCC+02] Qin Lv, Pei Cao, Edith Cohen, Kai Li, and Scott Shenker. Search and replication in unstructured peer-to-peer networks. In *ICS '02: Proceedings of the 16th international conference on Supercomputing*, pages 84–95, New York, NY, USA, 2002. ACM.

[LGP+90] Douglas B. Lenat, R. V. Guha, Karen Pittman, Dexter Pratt, and Mary Shepherd. Cyc: toward programs with common sense. *Commun. ACM*, 33(8):30–49, 1990.

[Lin98] Dekang Lin. An information-theoretic definition of similarity. In *Proc. 15th International Conf. on Machine Learning*, pages 296–304. Morgan Kaufmann, San Francisco, CA, 1998.

[Low87] Graham Lowe. *Women in the Administrative Revolution*. Polity Press, 1987.

[Lum05] Nicolas Lumineau. *Organisation et Localisation de données hétérogènes et réparties sur un réseau pair-à-pair*. PhD thesis, Université Pierre et Marie Curie, Paris VI, 2005.

[LYL04] David D. Lewis, Y. Yang, and F. Li. Rcv1: A new benchmark collection for text categorization researchl. *Journal of Machine Learning Research*, 5:361–397, 2004.

[MBF+90] Georges A. Miller, Richard Beckwith, Christiane Felbaum, Derek Gross, and Katherine Miller. Introduction to wordnet : an on-line lexical database. *International journal of lexicography*, 3(4):235–244, 1990.

[MC91] George A. Miller and Walter G. Charles. Contextual correlates of semantic similarity. *Language and Cognitive Processes*, 6(1):1–28, 1991.

[MM00a] Rada Mihalcea and Dan Moldovan. Semantic indexing using wordnet senses. In *Proceedings of the ACL-2000 workshop on Recent advances in natural language processing and information retrieval*, pages 35–45, Morristown, NJ, USA, 2000. Association for Computational Linguistics.

[MM00b] Rada Mihalcea and Dan I. Moldovan. An iterative approach to word sense disambiguation. In *Proceedings of the Thirteenth International Florida Artificial Intelligence Research Society Conference*, pages 219–223, 2000.

[MP03] Peter McBrien and Alexandra Poulovassilis. Data integration by bi-directional schema transformation rules. In *ICDE*, pages 227–238, 2003.

[MRS08] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, Cambridge, July 2008.

[MTW05] Sebastian Michel, Peter Triantafillou, and Gerhard Weikum. Klee: a framework for distributed top-k query algorithms. In *VLDB '05: Proceedings of the 31st international conference on Very large data bases*, pages 637–648. VLDB Endowment, 2005.

[Nam00] Fiammetta Namer. Flemm : un analyseur flexionnel du français à base de règles. *Traitement Automatique des Langues (TAL)*, 41(2):523–549, 2000.

[NJ02] Jian-Yung Nie and Fuman Jin. Integrating logical operators in query expansion invector space model. In *SIGIR workshop on Mathematical and Formal methods in Information Retrieval*, 2002.

[Noy04] Natalya F. Noy. Semantic integration: a survey of ontology-based approaches. *SIGMOD Rec.*, 33(4):65–70, 2004.

[NWQ+02a] Wolfgang Nejdl, Boris Wolf, Changtao Qu, Stefan Decker, Michael Sintek, Ambjörn Naeve, Mikael Nilsson, Matthias Palmér, and Tore Risch. Edutella: a p2p networking infrastructure based on rdf. In *WWW '02: Proceedings of the 11th international conference on World Wide Web*, pages 604–615, New York, NY, USA, 2002. ACM.

[OST03] Beng Chin Ooi, Yanfeng Shu, and Kian-Lee Tan. Relational data sharing in peer-based data management systems. *SIGMOD Rec.*, 32(3):59–64, 2003.

[PAP+03] Evaggelia Pitoura, Serge Abiteboul, Dieter Pfoser, George Samaras, and Michalis Vazirgiannis. Dbglobe: a service-oriented p2p system for global computing. *SIGMOD Record*, 32(3):77–82, 2003.

[PBP03] Siddarth Patwardhan, Satanjeev Banerjee, and Ted Pedersen. Using measure of semantic relatedness for word sense disambiguation. In *Proceedings of the Fourth International Conference on Intelligent Text Processing and Computational Linguistics*, Mexico, fevrier 2003.

[PK07] Artem Polyvyanyy and Dominik Kuropka. A quantitative evaluation of the enhanced topic-based vector space model. Technical Report 19, Hasso-Plattner-Institute, Postdam, 2007.

[Por80] Martin Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[PZSD96] Michael Persin, Justin Zobel, and Ron Sacks-Davis. Filtered document retrieval with frequency-sorted indexes. *Journal of the American Society for Information Science*, 47(10):749–764, 1996.

[QF93] Y. Qiu and H. P. Frei. Concept based query expansion. In *Research and Development in Information Retrieval, ACM-SIGIR*, pages 160–169, 1993.

[Qui68] M. Quillian. Semantic memory. In M. Minsky, editor, *Semantic Information Processing*, pages 227–270. MIT Press, Cambridge, 1968.

[Qui77] Willard V. O. Quine. *Relativité de l'ontologie et autres essais*. Aubier, Paris, 1977.

[RB01a] Erhard Rahm and Philip A. Bernstein. A survey of approaches to automatic schema matching. *The VLDB Journal*, 10(4):334–350, 2001.

[RD01] Antony I. T. Rowstron and Peter Druschel. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In *Middleware '01: Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms Heidelberg*, pages 329–350, London, UK, 2001. Springer-Verlag.

[Res95] Philip Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *IJCAI*, pages 448–453, 1995.

[RFH+01] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, and Scott Schenker. A scalable content-addressable network. In *SIGCOMM '01: Proceedings*

of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications, pages 161–172, New York, NY, USA, 2001. ACM.

[RG65] Herbert Rubenstein and John B. Goodenough. Contextual correlates of synonymy. *Commun. ACM*, 8(10):627–633, 1965.

[RGKG+05] Patricia Rodríguez-Gianolli, Anastasios Kementsietsidis, Maddalena Garzetti, Iluju Kiringa, Lei Jiang, Mehedi Masud, Renée J. Miller, and John Mylopoulos. Data sharing in the hyperion peer database system. In *VLDB '05: Proceedings of the 31st international conference on Very large data bases*, pages 1291–1294. VLDB Endowment, 2005.

[RK02] Sean C. Rhea and John Kubiatowicz. Probabilistic location and routing. In *Proceedings of INFOCOM 2002*, 2002.

[RMBB89] R. Rada, H. Mili, E. Bicknell, and M. Blettner. Development and application of a metric on semantic nets. *IEEE Transaction on Systems, Man, and Cybernetics*, 19(1):17–30, jan–feb 1989.

[Ros95] Jean-Gérard Rossi. *Le problème ontologique dans la philosophie analytique*. Kimé, Paris, 1995.

[Rou02] Marie-Christine Rousset. The Semantic Web Needs Languages for Representing (Complex) Mappings Between (Simple) Ontologies . *IEEE Intelligent Systems*, 17(2), 2002.

[Rou04] Marie-Christine Rousset. Small can be beautiful in the semantic web. In *International Semantic Web Conference*, pages 6–16, 2004.

[RS95a] Ray Richardson and Alan F. Smeaton. Using WordNet in a knowledge-based approach to information retrieval. Technical Report CA-0395, Dublin City University, Dublin, Ireland, 1995.

[RSM94] R. Richardson, A. F. Smeaton, and J. Murphy. Using WordNet as a knowledge base for measuring semantic similarity between words. Technical Report CA-1294, Dublin City University, Dublin, Ireland, 1994.

[RWHBG95] Stephen E. Robertson, Steve Walker, Micheline Hancock-Beaulieu, and M. Gatford. Okapi at trec-3. In *Text REtrieval Conference (TREC-3)*, pages 109–126, 1995.

[San00] Mark Sanderson. Retrieving with good sense. *Information Retrieval*, 2(1):49–69, 2000.

[Sav05] Jacques Savoy. Indexation manuelle et automatique: une évaluation comparative basée sur un corpus en langue française. In *CORIA*, pages 9–24, Grenoble, France, 2005.

[Seb02] Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.

[SM83] Gerard Salton and M.J. MacGill. *Introduction to Modern Information Retrieval*. MacGraw-Hill, 1983.

[SMGC04] Carlo Sartiani, Paolo Manghi, Giorgio Ghelli, and Giovanni Conforti. Xpeer: A self-organizing xml p2p database system. In *EDBT Workshops*, pages 456–465, 2004.

[SMK+01] Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, and Hari Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *SIGCOMM '01: Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 149–160, New York, NY, USA, 2001. ACM.

[SOP+02] Sofia Stamou, Kemal Oflazer, Karel Pala, Dimitris Christoudoulakis, Dan Cristea, Dan Tufis, Svetla Koeva, George Totkov, Dominique Dutoit, and Maria Grigoriadou. Balkanet: A multilingual semantic network for the balkan languages. In *Proceedings of the 1st Global WordNet Association conference*, 2002.

[SOT03] Christopher Stokoe, Michael P. Oakes, and John Tait. Word sense disambiguation in information retrieval revisited. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 159–166, New York, NY, USA, 2003. ACM Press.

[Sow76] John F. Sowa. Conceptual graphs for a database interface. *IBM Journal of Research and Development*, 20(4):336–357, 1976.

[SSS02] York Sure, Steffen Staab, and Rudi Studer. Methodology for development and employment of ontology based knowledge management applications. *SIGMOD Rec.*, 31(4):18–23, 2002.

[Sta02] Steffen Staab. Ontologies' kisses in standardization. *IEEE Intelligent Systems*, 17(2):70–79, 2002.

[SVH04] Nuno Seco, Tony Veale, and Jer Hayes. An intrinsic information content metric for semantic similarity in wordnet. In *Proceedings of ECAI'2004, the 16th European Conference on Artificial Intelligence*, 2004.

[SW49] Claude E. Shannon and Warren Weaver. *The Mathematical Theory of Communication*. University of Illinois Press, 1949.

[SWY75] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Commun. ACM*, 18(11):613–620, 1975.

[TR03] Dimitrios Tsoumakos and Nick Roussopoulos. Adaptive probabilistic search for peer-to-peer networks. In *P2P '03: Proceedings of the 3rd International Conference on Peer-to-Peer Computing*, pages 102–109, Washington, DC, USA, 2003. IEEE Computer Society.

[TR06] Dimitrios Tsoumakos and Nick Roussopoulos. Analysis and comparison of p2p search methods. In *InfoScale '06: Proceedings of the 1st international conference on Scalable information systems*, page 25, New York, NY, USA, 2006. ACM.

[TRV98] Anthony Tomasic, Louiqa Raschid, and Patrick Valduriez. Scaling access to heterogeneous data sources with disco. *IEEE Transactions on Knowledge and Data Engineering*, 10(5):808–823, 1998.

[Tuf07] Stéphane Tufféry. *Data Mining et statistique décisionnelle*. Technip, 2007.

[Tur94] James Turner. Le choix spontané de termes pour l'indexation des images: résultats de recherche. l'industrie de l'information en transition. In *Actes de la 22e conférence annuelle de l'ACSI (Association canadienne des sciences de l'information)*, pages 376–393, Université McGill, Toronto, 1994.

[Tur95] James Turner. Comparing user-assigned terms with indexer-assigned terms for storage and retrieval of moving images. In *Actes du congrès de l'ASIS*, Chicago, 1995.

[Tve77] Amos Tversky. Features of similarity. *Psychological Review*, 84(4):327–352, July 1977.

[TXD02] C. Tang, Z. Xu, and S. Dwarkadas. Peer-to-peer information retrieval using self-organizing semantic overlay networks, 2002.

[TZC+06] Michael Taylor, Hugo Zaragoza, Nick Craswell, Stephen Robertson, and Chris Burges. Optimisation methods for ranking functions with multiple parameters. In *CIKM '06: Proceedings of the 15th ACM international conference on Information and knowledge management*, pages 585–593, New York, NY, USA, 2006. ACM.

[UG96] Mike Uschold and Michael Grüninger. Ontologies: principles, methods, and applications. *Knowledge Engineering Review*, 11(2):93–155, 1996.

[UK95] Mike Uschold and Martin King. Towards a methodology for building ontologies. In *Workshop on Basic Ontological Issues in Knowledge Sharing, held in conduction with IJCAI-95*, Montreal, Canada, 1995.

[Van79] C.J. Van Rijsbergen. *Information Retrieval, 2nd edition*. Dept. of Computer Science, University of Glasgow, 1979.

[VCLV07] Anthony Ventresque, Sylvie Cazalens, Philippe Lamarre, and Patrick Valduriez. Query expansion and interpretation to go beyond semantic interoperability. In *OTM Conferences (1)*, pages 870–877, Vilamoura, Portugal, 2007. short paper.

[VCLV08a] Anthony Ventresque, Sylvie Cazalens, Philippe Lamarre, and Patrick Valduriez. Dealing with p2p semantic heterogeneity through query expansion and interpretation. In *Proceedings of the 2008 International Workshop on Data Management in Peer-to-Peer Systems*, pages 3–10, Nantes, France, 2008.

[VCLV08b] Anthony Ventresque, Sylvie Cazalens, Philippe Lamarre, and Patrick Valduriez. Enrichissement sémantique de requête utilisant un ordre sur les concepts. In *Workshop "Similarité Sémantique", associé à EGC'08*, Sophia-Antipolis, France, 2008.

[VCLV08c] Anthony Ventresque, Sylvie Cazalens, Philippe Lamarre, and Patrick Valduriez. Improving interoperability using query interpretation in semantic vector spaces. In *ESWC'08, European Semantic Web Conference*, pages 539–553, 2008. nominee for best paper award (4/51 accepted papers).

[Ven06b] Anthony Ventresque. Une mesure de similarité sémantique utilisant des résultats de psychologie. In *COnférence francophone en Recherche d'Information et Applications, CORIA*, pages 371–376, Lyon, 2006.

[Voo94] E. M. Voorhees. Query expansion using lexical-semantic relations. In *Research and Development on Information Retrieval - ACM-SIGIR*, pages 61–70, Dublin, 1994.

[Voo05] Ellen M. Voorhees. Trec: Improving information access through evaluation. *Bulletin of the American Society for Information Science and Technology*, 32(1), October/November 2005.

[Voo07] Ellen M. Voorhees. Trec: Continuing information retrieval's tradition of experimentation. *Commun. ACM*, 50(11):51–54, 2007.

[Wie92] Gio Wiederhold. Mediators in the architecture of future information systems. *Computer*, 25(3):38–49, 1992.

[Woo97] W. Woods. Conceptual indexing: A better way to organize knowledge. Technical report, Sun Microsystems Laboratories, April 1997.

[WP94] Zhibiao Wu and Martha Palmer. Verb semantics and lexical selection. In *32nd. Annual Meeting of the Association for Computational Linguistics*, pages 133–138, New Mexico State University, Las Cruces, New Mexico, 1994.

[WZW85] S. K. M. Wong, Wojciech Ziarko, and Patrick C. N. Wong. Generalized vector spaces model in information retrieval. In *SIGIR '85: Proceedings of the 8th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 18–25, New York, NY, USA, 1985. ACM Press.

[YGM02] Beverly Yang and Hector Garcia-Molina. Improving search in peer-to-peer networks. In *ICDCS '02: Proceedings of the 22 nd International Conference on Distributed Computing Systems (ICDCS'02)*, page 5, Washington, DC, USA, 2002. IEEE Computer Society.

[Zad65] Lofti A. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.

Références hypertextes

[WWW 1] *Conceptual Graphs for a Data Base Interface, by John F. Sowa.*
http://domino.research.ibm.com/tchjr/journalindex.nsf/a3807c5b4823c53f85256561006324be/621e3d26272d01ad85256bfa0067f7b3?OpenDocument

[WWW 2] *EuroWordNet : Building a multilingual database with wordnets for several European languages.*
http://www.illc.uva.nl/EuroWordNet/

[WWW 3] *Skype.*
http://www.skype.com/intl/fr/

[WWW 4] *The Hyperion Project: Share your data, anywhere ... anytime ...*
http://www.cs.toronto.edu/db/p2p/index.html

[WWW 5] *PeerSim : a Peer-to-Peer Simulator.*
http://peersim.sourceforge.net/

[WWW 7] *JXTA Community.*
https://jxta.dev.java.net/

[WWW 8] *WordNet::Similarity.*
http://wn-similarity.sourceforge.net/

[WWW 9] *Ontology, by John Sowa.*
http://www.jfsowa.com/ontology/

[WWW 10] *The WordSimilarity-353 Test Collection.*
http://www.cs.technion.ac.il/~gabr/resources/data/wordsim353/

[WWW 11] *Java WordNet Library.*
http://jwordnet.sourceforge.net/

[WWW 12] *Common Knowledge or Superior Ignorance?*
http://www.panix.com/~clocke/ieee.html

[WWW13] *Generalized Upper Model*.
..http://www.fb10.uni-bremen.de/anglistik/langpro/webspace/jb/gum/index.htm

[WWW14] *Ontology Alignement Evaluation Initiative*.
.....http://oaei.ontologymatching.org/

[WWW15] *World Wide Web Consortium*.
.....http://w3.org

[WWW16] *KaZaA home page*.
.....http://www.kazaa.com

[WWW17] *SETI@home*.
.....http://setiathome.ssl.berkeley.edu/

[WWW18] *Conceptual Graphs home page*.
.....http://conceptualgraphs.org/

[WWW19] *Logiciel ouvert de calcul bénévole et de calcul distribué*.
.....http://boinc.berkeley.edu/

[WWW20] *Senseval : Evaluation Exercises for the Semantic Analysis of Text*.
.....http://www.senseval.org/

[WWW21] *Swoogle: Semantic Web Search*.
.....http://swoogle.umbc.edu/

[WWW22] *Herman Hollerith: Data Processing Pioneer*.
.....http://www-03.ibm.com/ibm/history/exhibits/builders/builders_hollerith.html

[WWW23] *Reuters Test Collection*.
.....http://www.jmlr.org/papers/volume5/lewis04a/lyrl2004_rcv1v2_README.htm

[WWW24] *Text REtrieval Conference home page*.
.....http://trec.nist.gov/

[WWW25] *The Snowball Project*.

.....http://snowball.tartarus.org/

[www 26] *WordNet : a lexical database for the English language.*

.....http://wordnet.princeton.edu/

[www 27] *The internet and daily life.*

.....www.pewinternet.org/pdfs/PIP_Internet_and_Daily_Life.pdf

[www 28] *Ellen Voorhees defends Cranfield.*

.....http://thenoisychannel.blogspot.com/2008/04/ellen-voorhees-defends-cranfield.html

[www 29] *Ellen Voorhees defends Cranfield (TREC) evaluation.*

.....http://www.searchenginecaffe.com/2008/04/ellen-voorhees-defends-cranfield-trec.html

[www 30] *Trésor de la Langue Française Informatisé.*

.....http://atilf.atilf.fr/

[www 31] *SIGIR'08 call for papers.*

.....http://www.sigir2008.org/callforpapers.html

[www 32] *ComScore: the internet and daily life.*

.....http://www.comscore.com/press/release.asp?press=2209

[www 33] *Introduction to Information Retrieval.*

.....http://nlp.stanford.edu/IR-book/html/htmledition/irbook.html

[www 34] *Rapport financier de Google.*

.....http://investor.google.com/fin_data.html

[www 35] *How much information 2003?*

.....http://www.sims.berkeley.edu/research/projects/how-much-info-2003/execsum.htm

Table des figures

Partie I — Expansion structurante et image d'un document dans un cadre homogène sémantiquement

Partie II — Contexte d'hétérogénéité sémantique : apports de l'interprétation

Partie III — La sémantique dans les réseaux P2P non-structurés : traitement de requêtes et interopérabilité

Table des matières

Partie II — Contexte d'hétérogénéité sémantique : apports de l'interprétation

Partie III — La sémantique dans les réseaux P2P non-structurés : traitement de requêtes et interopérabilité

Espaces vectoriels sémantiques : enrichissement et interprétation de requêtes dans un système d'information distribué et hétérogène

Anthony VENTRESQUE

Résumé

Les systèmes d'information font face à un problème de pertinence dans les recherches dû à l'augmentation considérable des données accessibles. De plus, le nombre d'appareils communicants ne cesse de croître et de menacer le modèle client/serveur. Une nouvelle architecture distribuée tend donc à s'imposer : les réseaux pair-à-pair (P2P). Mais ils sont peu économes en ressource réseau (une requête inonde le réseau) et offrent des fonctionnalités limitées (recherche par mots-clés). Dans les deux communautés, RI et systèmes P2P, les recherches penchent vers l'utilisation de sémantique. En informatique, les approches basées sur la sémantique nécessitent souvent de définir des ontologies. Le développement important et distribué des ontologies génère une hétérogénéité sémantique. La solution classique est d'utiliser des correspondances entre parties de deux ontologies. Mais c'est une solution qui est difficile à obtenir et qui n'est pas toujours complète. Souvent les parties non-partagées de deux ontologies ne sont pas gérées, ce qui entraîne une perte d'information. Notre solution : EXSI^{2}D, utilise une expansion particulière, appelée expansion structurante, du côté de l'initiateur de requêtes. Cela lui permet de préciser les dimensions de sa requête sans modifier la requête elle-même. EXSI^{2}D offre aussi la possibilité au fournisseur d'information d'interpréter l'expansion structurante dans sa propre ontologie. Ainsi, il est possible à chaque participant d'un système d'information sémantiquement hétérogène d'utiliser toute son ontologie, y compris les parties non partagées. Nous montrons aussi l'utilisation d'EXSI^{2}D dans un système P2P, grâce à SPARTANBFS, un protocole « frugal » pour systèmes P2P non structurés.

Mots-clés : Sémantique, représentation de documents et requêtes, modèle vectoriel sémantique, pertinence et classement de documents, expansion et interprétation de requête, systèmes distribués pair-à-pair, interopérabilité sémantique.

Abstract

Information systems face a relevance problem in retrieval due to the huge increase of available data. Moreover, the number of networking devices grows up and jeopardizes the client/server architecture model. A new architecture is then emerging: peer-to-peer networks (P2P). But they are greedy in network resources (queries flood the network) and offer limited functionalities (key word search). In both fields, IR and P2P systems, research are going deeper on the use of semantics. In computer science, semantics based approaches generally relies on the definition of ontologies. Huge and distributed development of ontologies leads to a semantic heterogeneity. A classical solution relies on the use of mappings between parts of two ontologies. But this solution is difficult to obtain and not always complete. Unshared parts of two ontologies are often not managed, which leads to a loss of information. Our solution, EXSI^{2}D, uses a special query expansion, called structuring expansion, on query initiator's side. Then she can specify the dimensions of her query without any modification of the query itself. Information provider is also allowed to interpret the structuring expansion within her own ontologies. Thus each participant of a semantic heterogeneous information system is able to use all her ontology, including the unshared parts. We also present a solution to the use of EXSI^{2}D in a P2P system, thanks to SPARTANBFS, a "frugal"protocol for unstructured P2P systems.

Keywords: Semantics, Documents and Queries Representation, Semantic Vector Space Model, Relevance and Ranking of Documents, Query Expansion and Interpretation, Distributed P2P Systems, Semantic Interoperability.

Classification ACM

Catégories et descripteurs de sujets : H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Query formulation, Retrieval models, Selection process*; C.2.4 [**Computer-communication Networks**]: Distributed Systems—*Distributed databases*; M.7 [**Knowledge Management**]: Knowledge Retrieval