N° d'ordre

Université Paris XI
UFR scientifique d'Orsay

Thèse présentée pour obtenir

Le grade de docteur en sciences de l'université Paris XI Orsay

par Haïfa Zargayouna

Sujet : Indexation sémantique de documents XML

Soutenue publiquement le 15 Décembre 2005 devant la commission d'examen :

Mohand Boughanem	Rapporteur
Jérôme Euzenat	Rapporteur
Gerard Sabah	Directeur
Sylvie Salotti	Co-directrice
Marie-Christine Rousset	Examinateur

Remerciements

Me voilà encore devant une page blanche que je ne vais pas tarder à noircir. La liste de personnes à remercier me donne le tournis, que de chemin parcouru depuis ce jour où je me suis perdue dans les bois avant de retrouver mon chemin et arriver à bout de souffle au LIMSI.

Cette thèse n'aurait pas été possible sans mes encadrants. Je les en remercie vivement.
Gérard Sabah pour avoir accepté la direction de ma thèse et pour la confiance qu'il n'a cessée de me renouveler.
Sylvie Salotti pour avoir tenu bon tout au long de ses années, pour la grande liberté qu'elle m'a laissée et pour sa patience à toute épreuve et son soutien dans les moments difficiles.

Je remercie en premier lieu les membres du jury pour avoir accepté de rapporter et examiner ce travail.
Mohand Boughanem pour ses remarques pertinentes et sa grande générosité. Il fut parmi les premiers à m'accueillir dans la communauté RI.
Jérôme Euzenat qui malgré un emploi du temps chargé a pris le temps d'annoter ce manuscrit page par page.
Marie-Christine Rousset en tant que directrice de l'école doctorale, elle a suivi année après année le déroulement de ma thèse et pour avoir accepté de présider le jury.

Ma famille a été pour moi le principal moteur, chacun à sa manière de près ou de loin a su m'insuffler l'énergie nécessaire.
P'Ma merci pour les allers-retours incessants que je vous ai fait faire, pour les grands coup de pouce, pour vous être fait du soucis pour moi sans jamais me le montrer et pour me procurer ce sentiment d'être "unique" pour vous.
Sof pour avoir essayé et essaye encore de m'insuffler un souffle de spiritualité dans une vie jalonnée de stress et de deadlines.
Sony pour les cafés, le théâtre, le brin de folie, les soirées déprime, les biberons à 3h du mat', les cloppes, tout en vrac, pas dans l'ordre et j'en oublie.
Chapi pour l'image que tu me renvoie de moi qui me donne l'impression que je suis une *super woman*. Ines j'aurais pu voler rien que pour rester à la hauteur.
Last but not least, doudou qui a beaucoup grandi et même devenu un collègue sans que j'ai eu le temps de m'en rendre compte. Par tes critiques, tes encouragements, tes relectures tu m'as été d'une grande aide ! Mahdi merci aussi pour le baby sitting, les biberons, les couches, ton canapé, le chocolat et les longues veillées à discuter...

B.B. que dire ? poki poki poki poops ... Tu fus et reste mon plus grand contradicteur, merci de m'avoir poussé dans mes retranchements, ta rigueur scientifique et tes remarques -qui des fois me faisaient douter de tout- m'ont permis de mieux formaliser les bribes d'idées qui m'assaillaient à pas d'heure. Merci aussi pour tes encouragements, pour m'avoir suppléé auprès de Luna et avoir été le papa, la maman et la nounou avec autant de talent ! tu fais quoi les cinquante prochaines année ? je t'engage ! ;)

Dernier astre de mon univers : ma Luna. Comment un si petit être peut donner autant d'amour et d'énergie ? Luna merci de m'avoir écoutée toujours avec le sourire et de m'avoir réconfortée avec si peu de vocabulaire .. t'assures vraiment en tant que bébé ;)

Les amis, les copains, les collègues, qu'auraient été ces années de thèse sans vous.
Nico, Elsa, Guille, Sarrah, Sandra, Jérôme, Nouari, Marjorie, Xavier : Merci de votre soutien, de ne pas m'avoir oublié pendant ma longue période d'hibernation, d'avoir toléré l'absence de nouvelles, les faux plans, les sauts d'humeur, les moments de doutes ... Les moments que j'ai passés avec vous furent les meilleurs, j'espère vous garder encore longtemps.

Le LIMSI a été pour moi, surtout les derniers mois, une seconde résidence. Sans mes colocataires (bien plus que des collègues) l'ambiance n'aurait pas été la même. Un grand merci à Nédé, Léo, Jamal, Nicolas, Stéphanie, Gilles, Amandine, Estelle, Anne-Laure, Jean-Pascal, Alexandre, Sylvain pour tous ces agréables moments passés en votre compagnie ..

Je tiens également à remercier les membres du groupe AMI et à leur tête Jean-Paul Sansonnet d'avoir tout fait pour que cette thèse se passe au mieux. Je remercie également au delà des groupes, Christian Jacquemin, Michael Zock, Bill Turner, Michèle Jardino, Anne Vilnat, Brigitte Grau, Patrick Paroubek, Gabriel Illouz, Frédéric Vernier, Faiza El Kateb.

J'ai eu la chance d'être membre associé au LIPN dans l'équipe RCLN. Je remercie ses membres de leur accueil et de l'intérêt qu'ils portent à mon travail et leurs conseils qui m'ont permis d'améliorer ma soutenance. Je remercie particulièrement Adeline Nazarenko pour sa disponibilité et ses précieux conseils, Sylvie Szulman pour avoir mis à ma disposition tous les outils logiciels dont j'avais besoin, Daniel Kayser pour avoir eu la patience de me faire répéter ma soutenance et vaincre ma peur d'être en face de lui :).
Les conférences, journées de recherche, écoles d'été sont l'occasion de faire des rencontres intéressantes. C'était le cas pour moi, merci à ceux qui ont rendu festive l'ambiance studieuse ! La liste est longue .. un merci particulier à Asma, Karen et Fleur.

La thèse a été égaillée par les enseignements que j'ai pu donner. Je remercie vivement mes collègues enseignants à l'Université Paris Sud (IUT d'Orsay, FIIFO, fac) et à l'IUT de Villetaneuse. Les organisateurs du CIES de Versailles ont été formidables, les stages proposés n'ont peut-être pas révolutionné ma conception de l'enseignement mais m'ont aidée dans ma remise en question perpétuelle.

Je remercie surtout tous mes étudiants[1], vous avez été mon rayon de soleil, mes petits "bout de chou" à moi. Vous êtes ma véritable source de jouvence, merci pour toute cette énergie, je rêve d'entendre encore "MADAME ça ne marche pas !" ..

La rumeur court qu'il y a une vie après la thèse .. Encore une hypothèse à formaliser, tester, expérimenter et prouver ou réfuter .. tout un programme.

[1] Même Julien au fond à gauche à côté de la fenêtre.

À B.B.

Table des matières

Liste des tableaux

Table des figures

Introduction

Pour qu'un savoir puisse se transmettre, il faut d'abord pouvoir le reproduire et le stocker. Dans ce contexte, l'Humanité a fait des pas de géants de la glyptique au document numérique, en passant par l'imprimerie de Gutenberg, et ce quel que soit le support utilisé (le rouleau, le codex, le numérique), ainsi que les divers agencements du texte par rapport au support. Mais ensuite et surtout, il faut pouvoir accéder aux informations stockées. Ce besoin d'accès demeure intact aujourd'hui, l'accroissement des masses d'information disponibles ne faisant qu'accentuer ce besoin ancestral, qui devient bien plus compliqué à gérer. En effet, à quoi servirait le stockage d'une information si on ne peut y accéder ? Créer une information est un travail souvent onéreux et si on ne peut y accéder, ce même travail est à refournir. Il est d'ailleurs révélateur que peu après l'invention des ordinateurs au début des années 1950, le domaine de la Recherche d'Information ait vu le jour, démontrant que le stockage et le traitement de l'information vont de pair avec les techniques d'accès qui leurs sont associées.

Aujourd'hui, toutes les données méritant publication sont destinées -à terme- à être numérisées, le problème du stockage et de la pérennisation des informations tend ainsi à être résolu. Nous le voyons bien, nous assistons à une époque, où le savoir individuel est théoriquement, dès sa publication, universel. Notre civilisation peut désormais prétendre à la capitalisation synchronisée du savoir, à l'accélération des avancées technologiques en mutualisant les efforts et en évitant le gaspillage et la redondance. Mais aujourd'hui, la masse de connaissances stockées est tellement immense, que nous assistons au phénomène inverse : l'information n'est désormais plus une denrée rare et l'instantanéité de sa disponibilité est assurée grâce à Internet. C'est désormais au niveau individuel que nous nous posons des questions. En effet, pour un individu, un système ou une organisation, rechercher une information précise dans l'amas des données en croissance exponentielle sur le Net serait comme chercher une aiguille dans une botte de foin.

A la naissance du domaine de la Recherche d'Information, les chercheurs s'enthousiasmaient à l'idée d'utiliser les ordinateurs, pour la recherche des informations dont la taille dépassait les capacités calculatoires humaines. Dès les premiers Systèmes de Recherche d'Information (SRI), les modèles de RI sont construits autour du triplet <document, besoin, correspondance>, ces modèles constituent encore aujourd'hui la base autour de laquelle sont développés les moteurs de recherche sur le Web. Ainsi, un SRI est un système qui stocke un ensemble de *documents* sous une forme électronique (corpus ou base documentaire), dans le but de permettre aux utilisateurs de retrouver ceux dont le contenu correspond le mieux à leur *besoin d'information*. Une phase d'indexation permet de stocker une abstraction des contenus des documents. Ces abstractions sont ensuite comparées à la représentation des besoins de l'utilisateur (la requête) à la phase

d'interrogation (ou de recherche) grâce à une fonction de *correspondance*.
Très vite les problèmes inhérents à la richesse des langues, se sont imposés. Les SRI doivent traiter les problèmes de synonymie et de polysémie des termes.

Contexte de travail

Notre travail se situe dans le contexte de la *Recherche d'Information (RI)* et de *l'Ingénierie des Connaissances (IC)*.
La RI propose de retrouver parmi une masse volumineuse de documents, ceux qui correspondent au besoin informationnel d'un utilisateur généralement formulé par une requête en langage naturel en RI classique. Les documents sont considérés comme des unités atomiques et indépendantes. Dans la pratique, les documents possèdent une structure interne et sont parfois inter-reliés (c'est le cas des pages HTML sur le Web). Notre travail se situe dans la perspective d'une RI semi-structurée. L'objectif d'un modèle de **Recherche d'Information Semi-structurée (RIS)** est de prendre en compte la structure des documents, ce qui nécessite de s'interroger sur la sémantique d'une telle structure, et donc des relations entre les différents éléments, pour pouvoir comprendre son impact sur la description de l'information. En d'autres termes, il faut se demander comment l'auteur d'un document utilise la structuration pour décrire le message qu'il veut faire passer. La représentation de la structure n'est pas une fin en soi. Il faut savoir l'utiliser pour répondre au mieux aux besoins des utilisateurs.
L'objectif de l'**Ingénierie des Connaissances (IC)** est de proposer des concepts, des méthodes et des techniques permettant d'acquérir, de modéliser et de formaliser des connaissances pour les mobiliser dans une activité individuelle ou collective au sein d'une organisation ou d'une communauté. Dans le cadre d'une recherche d'information, l'IC propose la construction de bases de connaissances qui permettront d'accéder aux contenus des documents. La difficulté de la mise en place d'une telle base de connaissance a freiné l'adaptation des outils de l'IC dans un cadre général, tel que le Web.
Avec la vision du **Web Sémantique (WS)** de Tim Berners-Lee[2] [Berners-Lee *et al.*, 2001], le Web de demain sera un web de "sens" à la place du web de "liens" actuel. Le Web sémantique est une infrastructure qui permet une utilisation de connaissances formalisées en plus du contenu informel actuel du Web. L'IC retrouve de ce fait toute sa place au sein du Web et les ontologies formelles[3] deviennent le concept pivot. La structuration des documents en XML constitue la brique de base de l'infrastructure du Web Sémantique.
C'est dans ce contexte d'émergence d'un Web sémantique que nous explorons la prise en compte de la sémantique et de la structure des documents dans une tâche de recherche d'information.

Problématique

Notre problématique de recherche a comme élément pivot le **texte**. Nous nous intéressons à la prise en compte des informations explicites autour du texte, à savoir la **structure**, ainsi qu'aux

[2] fondateur du W3C.
[3] Une ontologie décrite dans un langage avec une syntaxe et une sémantique bien définies.

informations implicites[4], à savoir la **sémantique**.

Texte et structure

Les besoins d'information d'un utilisateur peuvent être variés :
– L'utilisateur peut chercher une information précise en ayant une bonne connaissance du domaine.
– ou chercher une information générale en ayant une connaissance vague du domaine
Les SRI doivent permettre à l'utilisateur de définir son besoin pour identifier le type de réponse attendue. Dans le cadre de documents structurés, les requêtes de l'utilisateur peuvent être exprimées de deux manières différentes :

1. avec une connaissance *a priori* de la structure.
   – Les requêtes peuvent comporter des conditions sur les éléments de structure. Ces conditions peuvent être strictes (si la structure est complètement spécifiée) ou vague (si la structure est partiellement représentée, par fragments).
   – ou bien les requêtes comportent un élément à retourner.
   – ou bien les requêtes ne comportent pas d'élément à retourner.
2. sans connaissance de la structure. Les éléments à retourner sont à définir par le système, la difficulté est de trouver les unités d'information pertinentes mais qui soient aussi assez générales pour être retournées toutes seules à l'utilisateur.

Texte et sémantique

La vision implicite du Web Sémantique repose sur les hypothèses suivantes :
– Il existe des ontologies formelles pour décrire objectivement les connaissances d'un domaine.
– Il est possible de décrire les ressources du Web en utilisant les concepts de ces ontologies.
– Il est possible pour l'utilisateur de rechercher l'information en utilisant ces mêmes concepts.
Ces hypothèses font émerger plusieurs problématiques :
– Comment faire coïncider le point de vue du concepteur d'un document avec celui de l'utilisateur ainsi qu'avec celui qui a créé l'ontologie ?
– Comment permettre à l'utilisateur de profiter de la richesse sémantique des ontologies en posant sa requête de manière habituelle, en d'autres termes comment intégrer l'ontologie dans le processus d'indexation de manière transparente pour l'utilisateur ?
Dans ce travail, nous apportons des éléments de réponse à ces questions.

Principales contributions

Notre contribution concerne les deux domaines : RI et Web Sémantique.
Notre contribution dans le cadre de la RI structurée se situe à plusieurs niveaux :

[4]Nous considérons dans notre travail que les informations sémantiques sur le texte sont implicites dans la mesure où le contenu n'est pas explicitement annoté par une représentation de son sens.

- L'indexation et le stockage des documents : nous proposons un modèle d'indexation qui étend le modèle vectoriel de Salton aux documents XML, en représentant les unités textuelles par des vecteurs de termes. Ce qui nous a amené à redéfinir les notions suivantes par rapport au modèle classique :
  - *Les limites d'un document* : Il est clair que les entités textuelles d'un document représentent maintenant les entités atomiques à la place du document. Mais leur agencement au sein d'un même document doit être pris en compte. Nous nous démarquons des travaux de recherche qui définissent la notion de document logique qui serait plutôt la plus petite information dans le document.
  - *L'unité d'information* à représenter : nous définissons les unités de contexte pertinentes à indexer sans avoir une redondance dans l'index.
  - *Une mesure de pondération* adéquate pour traduire la représentativité des termes au sein des entités et des documents.
  - *Une mesure de propagation* de ces pondérations à la hiérarchie de structure.
- L'interrogation des documents : nous proposons un langage qui repose sur une syntaxe XML. Contrairement à la plupart des langages, il ne nécessite pas une connaissance du langage. En effet, XQuery par exemple, possède une syntaxe difficile d'accès pour beaucoup d'utilisateurs potentiels. Or il est probable que le succès d'un langage d'interrogation dépende certes de son expressivité (il doit prendre en compte les besoins de nombreux utilisateurs différents) mais aussi de sa simplicité d'utilisation et sa facilité d'accès à de nombreux utilisateurs potentiels. Notre langage de requête permet la formulation de requêtes à base de simples mots clés, ainsi que la formulation de contraintes sur la structure. Ces contraintes peuvent être strictes quand l'utilisateur a une bonne connaissance de la structure des documents dans le corpus ou vagues quand l'utilisateur ne veut (ou peut) pas donner de précision en chemin absolu.
- La présentation des résultats à l'utilisateur par ordre de pertinence : nous proposons une mesure de concordance entre les requêtes et les éléments des documents, qui prend en considération les distances structurelles et sémantiques.

Notre contribution dans le cadre du Web sémantique se situe à plusieurs niveaux :

- L'indexation : nous proposons de relier les concepts de l'ontologie aussi bien aux termes, qu'à la structure. Nous effectuons une projection lexicale des corpus dans l'ontologie, et proposons une procédure de désambiguïsation sémantique.
- Nous proposons une mesure de similarité numérique qui capture le sens des termes par rapport à leur contexte d'apparition et leur voisinage sémantique.

Afin de vérifier la faisabilité de ces propositions, nous avons développé un prototype en Java (**SemIndex : Semantic Indexing** et **SemIR : Semantic Information Retrieval**) et les propositions ont été évaluées sur une collection de test.

Plan de la thèse

Cette thèse se décompose en deux parties : l'état de l'art (chapitre de 1 à 4) et une partie où nous présentons nos propositions et nos expérimentations (chapitre 5 à 8)

Partie 1 Structure et Texte : L'objectif cette partie est de présenter l'état de l'art des approches proposées pour la recherche d'information classique dans des textes "plats" ainsi que dans des textes semi-structurés.

Chapitre 1 : Systèmes de Recherche d'Information Dans ce chapitre nous présentons les briques de base d'un système de recherche d'information. Nous commençons d'abord par décrire l'architecture d'un SRI, puis nous discutons les choix suivants :
- la plus petite unité à indexer ;
- la pondération d'une telle unité ;
- les modèles qui existent ;
- et les évaluations possibles de tels systèmes.

Chapitre 2 : Systèmes de Recherche d'Information Semi-structurée L'objectif de ce chapitre est d'introduire la notion de structure explicite et plus spécifiquement XML. Après un bref aperçu des technologies XML, nous présentons les approches de RI semi-structurée proposées dans la littérature, en mettant l'accent sur celles qui adaptent le modèle vectoriel et se rapprochent de ce fait de nos travaux.

Partie 2 Sémantique et Texte Cette deuxième partie traite de l'état de l'art des ontologies et de leur utilisation en RI, nous présentons également le Web Sémantique avec la place prépondérante des ontologies pour formaliser le contenu du Web actuel. Nous nous intéressons particulièrement à l'utilisation des ontologies dans le cadre d'une RI.

Chapitre 3 : Ontologies Dans ce chapitre nous nous positionnons par rapport aux différentes terminologies utilisées, puis nous énonçons le besoin de formalisation en présentant les différents langages de représentation de connaissances existants. Enfin, nous détaillons les différents travaux de RI qui utilisent les ontologies (formelles ou non) afin de montrer l'apport de la sémantique par des ressources externes.

Chapitre 4 : Web sémantique Après avoir présenté la vision du Web Sémantique selon Berners-Lee [1999], nous détaillons les problématiques émergentes dues à la nature même du web : dynamisme, hétérogénéité, etc. Certains de ces problèmes sont déjà bien connus en IC.

Partie 3 Structure Sémantique et Texte **Chapitre 5 : Similarité sémantique** Dans ce chapitre nous proposons une mesure de similarité sémantique que nous situons par rapport aux mesures existantes. Nous pointons les problèmes que posent de telles mesures et notre adaptation de la mesure de Wu & Palmer [1994] pour appliquer la mesure de similarité dans le cadre de la RI. Nous présentons les évaluations effectuées pour tester les performances de telles mesures et dressons les limites actuelles dues à la procédure expérimentale ainsi qu'à la structuration même de WordNet. Nous présentons l'application de notre mesure dans le cadre d'ontologies formelles, en introduisant la fonction PPG (Plus Petit Généralisant) qui calcule le concept le plus spécifique qui subsume deux concepts donnés.

Chapitre 6 : SemIndex Ce chapitre présente SemIndex (Semantic Indexing) qui est le modèle d'indexation que nous proposons pour répondre aux différentes problématiques de la recherche d'information semi-structurée. Nous introduisons les différentes fonctions de pondération qui tiennent compte de la répartition des termes dans les balises et les documents. Nous exposons également comment s'intègre le calcul de similarité sémantique dans notre processus d'indexation.

Chapitre 7 : SemIR Ce chapitre présente notre langage de requêtes SemIR (Semantic Information Retrieval) qui à la différence des langages de requêtes qui existent est

plus intuitif et repose sur une syntaxe XML. Notre langage permet de poser des requêtes vagues sur la structure et sur le contenu ainsi que des requêtes booléennes.

Chapitre 8 : Expérimentations et résultats Ce chapitre présente les deux prototypes (SemIndex et SemIR) qui ont été réalisés ainsi que les résultats d'expérimentation de la mesure de similarité et de tout le système sur un corpus de test.

Première partie

Texte et Structure

Chapitre 1

Systèmes de Recherche d'Information (SRI)

Sommaire

1.1 Introduction

Face à la quantité croissante des données sur les réseaux internes et mondiaux (intranets, internet, etc.) la question de l'accès à l'information est un des plus grands enjeux d'actualité mais aussi un des plus délicats. Dans ce contexte, il est nécessaire de pouvoir accéder au contenu des documents par des moyens rapides et efficaces.
La **Recherche d'Information (RI)** ou recherche documentaire, propose de retrouver parmi une masse volumineuse de documents textuels[5], ceux qui correspondent au besoin informationnel d'un utilisateur généralement formulé par une requête en langage naturel.
Les premiers modèles de RI ont établi la notion de triplet (document, besoin, correspondance).
Un **Système de Recherche d'Information (SRI)** est un système qui stocke un ensemble de *documents* sous forme électronique (corpus ou base documentaire), dans le but de permettre aux utilisateurs de retrouver ceux dont le contenu *correspond* le mieux à leur *besoin d'information*.
Le contenu des documents est extrait à la phase d'indexation et stocké dans des structures appelées index, qui sont ensuite comparées à la représentation des besoins de l'utilisateur (la requête) pendant la phase d'interrogation (ou de recherche).
Ce chapitre a pour objectif de présenter les briques de base de la RI. Il est organisé comme suit. Nous commençons tout d'abord par décrire l'architecture globale d'un SRI (section 1.2). Cette architecture met en place un processus d'indexation et un processus d'interrogation. Nous détaillons les différents types d'indexation (section 1.3) qui peuvent être automatiques ou manuelles (section 1.3.1), en langage libre ou contrôlé (section 1.3.2). Nous présentons les différentes étapes nécessaires pour l'indexation, à savoir l'identification des unités d'index (section 1.4) ainsi que leur pondération (section 1.6). Nous passons ensuite en revue les modèles classiques de RI (section 1.7), qui sont (i) le modèle booléen (section 1.7.1), (ii) le modèle vectoriel (section 1.7.2) et (iii) le modèle probabiliste (section 2.5.2). Nous présentons également des modèles qui étendent le modèle booléen (section 1.8.1 et section 1.8.2) et le modèle vectoriel (section 1.8.3 et section 1.8.4). Enfin, nous présentons les principales mesures d'évaluation proposées pour évaluer les différents modèles et systèmes (section 5.8).

1.2 Architecture d'un SRI

Le but d'un SRI est de présenter à l'utilisateur des documents répondant à ses besoins. Selon [Baeza-Yates & Ribeiro-Neto, 1999], de manière globale, un SRI se modélise par le quadruplet SRI=<D,Q,M,P> (voir figure 1.1) où :
– D est l'ensemble des documents du corpus ;
– Q est un langage de requête destiné à représenter les besoins d'information de l'utilisateur. Ce langage définit l'ensemble des requêtes que peut formuler directement ou indirectement un utilisateur d'un SRI ;
– M est un modèle de RI qui sert à décrire les documents de D et à exprimer les requêtes de Q.
– P est une fonction qui associe une valeur de pertinence entre toute requête q_i de Q et tout document d de D. Cette fonction peut fournir un ordonnancement des documents par rapport à la requête q_i.

[5]Nous parlons dans ce travail de documents textuels, sachant que dans un cadre général, les documents peuvent être multimédia (texte, image, vidéo, etc.)

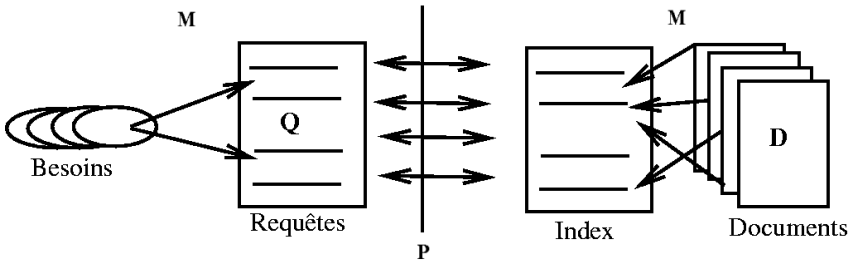

FIG. 1.1 – Exemple d'architecture d'un SRI

Le SRI se décompose essentiellement en deux processus de base :
- **Le processus d'indexation** qui consiste à identifier dans un document certains éléments significatifs qui serviront de clés pour retrouver ce document au sein d'une collection[6]. Il produit une représentation des documents (l'index). Pour faciliter le processus d'interrogation, le même processus d'indexation s'applique généralement aux requêtes.
- **Le processus d'interrogation** (ou de recherche) qui consiste à comparer les représentations des requêtes avec celles des documents.

Cette architecture peut être enrichie par un retour arrière sur pertinence[7] (*relevance feedback*) qui affine la recherche et améliore la qualité des résultats en tenant compte de l'évaluation de l'utilisateur qui classe les documents en pertinents et non pertinents. Il est aussi possible d'avoir recours à l'expansion de requêtes qui permet d'étendre la requête (en rajoutant des termes) ou la ré-écrire, nous donnons un exemple d'expansion de requêtes dans le chapitre 3.
Les processus d'indexation et de recherche sont dépendants l'un de l'autre. En effet, la phase d'indexation est une phase importante qui a un impact direct sur la recherche. Si un document est mal indexé, il risque de ne plus être retrouvé et donc perdu. La norme AFNOR NF Z 47-102 1996, définit l'indexation de la manière suivante :

Définition

> L'indexation est l'opération qui consiste à décrire et à caractériser un document à l'aide de représentations des concepts contenus dans ce document, c'est-à-dire à transcrire en langage documentaire les concepts après les avoir extraits du document par une analyse.

Le but de l'indexation est donc d'extraire le contenu des documents et de les représenter de manière à identifier le document. Le choix des unités de représentation (appelées aussi unités d'index ou descripteurs) du document est crucial, il influence la qualité de l'indexation. L'ensemble de ces unités constitue le vocabulaire d'indexation, ce vocabulaire peut être libre ou contrôlé (cf section 1.3). Ces unités varient d'une chaîne de caractères (n-gramme) à des groupes nominaux ou des unités linguistiques complexes (cf section 1.4). La structure de l'index est reliée au choix du modèle de recherche utilisé (cf section 1.7 pour les modèles classiques).

[6] Définition de Encyclopedia Universalis, "http ://universalis.edu.com".
[7] Appelé aussi bouclage de pertinence.

L'interrogation est l'opération qui consiste à formuler les besoins des utilisateurs et à les représenter suivant le modèle de recherche en vue d'une comparaison avec les représentations des documents dans l'index.

Nous présentons dans ce qui suit les concepts de base pour un SRI à savoir le choix du type d'indexation, l'extraction des unités d'index ainsi que le choix du modèle de recherche.

1.3 Type d'indexation

Le premier problème de l'indexation est de déterminer les unités d'information qui doivent être retenues dans l'index. Il faut déterminer le type d'indexation : manuelle (par des sujets humains) ou automatique (par le système), ainsi que choisir le vocabulaire : fixé *a priori* (contrôlé) ou libre. Ces choix dépendent de plusieurs paramètres, à savoir la taille du corpus, le domaine et l'application.

1.3.1 Indexation manuelle versus indexation automatique

L'indexation manuelle se différencie de l'indexation automatique par les acteurs qui entrent en jeu pour la phase d'indexation. *L'indexation manuelle* (appelée aussi indexation humaine) fait intervenir des agents humains considérés comme experts (généralement des **documentalistes**). Ces derniers ont pour rôle d'identifier les unités d'index pertinentes qui caractérisent le mieux les documents. L'indexation manuelle est très souvent critiquée pour son coût élevé ainsi qu'un manque d'homogénéité dans le choix des représentations des documents. En effet, les agents doivent avoir une connaissance minimale des contenus des documents pour pouvoir choisir les informations qui les caractérisent sinon l'indexation risque d'être erronée. La variabilité des choix des unités d'index (pour un même individu et *a fortiori* d'un individu à un autre) peut diminuer les performances du système car les index obtenus peuvent manquer de cohérence.

L'indexation automatique présente l'avantage d'une régularité de l'indexation, le document aura toujours le même index. Une telle indexation s'est imposée avec l'avènement du Web où il est difficilement envisageable de traiter les documents à la main. Une manière automatique de caractériser le contenu des documents est donc primordiale. Elle peut être faite en temps réel (lorsque le système est interrogé) ou en différé. Les systèmes d'indexation automatique sont donc dans l'obligation de manipuler le contenu des textes. Cette phase n'est pas triviale et le système aura à résoudre des problèmes classiques que pose la richesse de la langue naturelle (ambiguïté, etc). Les outils de Traitement Automatique de la Langue (TAL) proposent différentes approches pour extraire les unités d'index, ils sont plus ou moins sophistiqués suivant le besoin des systèmes. Nous présentons dans la section1.4 ces techniques ainsi que leur apport dans le cadre d'un SRI.

L'introduction d'outils sophistiqués demande cependant une validation humaine pour choisir les unités d'index pertinentes, on parle alors *d'indexation semi-automatique*, cela représente un frein à leur utilisation dans les systèmes qui traitent de gros corpus tels que le Web.

1.3.2 Langage contrôlé versus langage libre

Le choix entre l'indexation en langage contrôlé et l'indexation en langage libre se pose aussi bien pour l'indexation humaine que pour l'indexation automatique.
Un *langage contrôlé* est un langage normalisé, c'est-à-dire, qu'une liste des descripteurs pour l'indexation est définie *a priori*. Elle est appelée *liste d'autorité*. Elle sert à éviter les problèmes d'ambiguïté (dûs à l'homonymie et à la polysémie de certains termes) ainsi que les problèmes de redondance (synonymie, etc.). Ainsi, un terme d'indexation possède un seul sens défini et chaque sens donné n'est décrit que par un seul terme. Les descripteurs retenus seront les seuls mots clés acceptés lors de la requête. Une liste de termes interdits est aussi explicitement définie, ces termes ne doivent pas faire partie des descripteurs. Des relations peuvent être définies entre les termes retenus et d'autres termes (les non descripteurs). Le vocabulaire est généralement organisé dans un thésaurus[8] où des relations entre descripteurs peuvent être représentées (e.g. relations d'équivalence, d'association, etc.) et permettent une représentation riche du document. L'indexation en langage contrôlé réduit le nombre de représentations possibles d'un document. Cela n'empêche pas l'indexation d'être subjective si elle est réalisée par un sujet humain, même si les sens et les termes sont bien délimités, l'interprétation humaine du contenu textuel est sujette à variation.
Le *langage libre* est dit libre parce qu'il n'est pas contraint par un contrôle. Les descripteurs sont choisis librement et aucune contrainte n'est fixée *a priori*. La représentation des documents est donc plus souple et permet une couverture large du contenu. Cette approche est néanmoins sujette aux problèmes d'ambiguïté sémantique de la langue naturelle. Le langage libre pose aussi le problème d'adéquation avec la requête, les mots clés recherchés doivent figurer parmi les mots clés choisis pour décrire le document. Le vocabulaire étant ouvert et non prédéfini, comment un utilisateur peut-il formuler une requête sans connaître les mots clés qui ont servi à indexer le texte ? Si l'indexation est manuelle, le risque de variabilité des descriptions est d'autant plus fort.

1.4 Unité d'index

Les unités d'index définissent les unités de base de l'espace d'indexation. Elles peuvent varier d'un groupe de caractères (n-gramme) à un ensemble de termes (les syntagmes). Les unités d'index constituent les descripteurs des documents et servent d'entrée à l'index. L'attribution des unités d'index n'est pas une tâche triviale et influence directement le résultat de la requête. Une entrée absente de l'index constitue une perte d'information.
Nous décrivons dans ce qui suit les différentes techniques de TAL qui peuvent être utilisées dans un SRI. Jacquemin & Zweigenbaum [2000] différencient trois paliers d'analyse : le palier morphologique, le palier syntaxique et le palier sémantique et pragmatique. Chaque palier représente un degré supplémentaire d'analyse. La plupart des SRI s'arrêtent au niveau de l'analyse morphologique et syntaxique, l'analyse sémantique est souvent utilisée pour l'enrichissement des requêtes où la taille de la requête permet de faire des traitements assez fins.

[8] Nous définissons à ce niveau un thésaurus comme un vocabulaire d'un langage d'indexation contrôlé, organisé de façon à expliciter les relations à priori entre les notions (par exemple les relations générique/spécifique) [norme ISO 59641] (Voir chapitre 3 pour une présentation détaillée.).

Le choix le plus courant aussi, consiste à ne sélectionner que des termes simples, c'est-à-dire des noms (N), verbes (V) et adjectifs (A) non composés. La plupart des systèmes, à langage contrôlé ou non, retiennent d'ailleurs ce choix. Nous présentons dans ce qui suit brièvement ces trois paliers d'analyse ainsi que leur apport en RI. Le lecteur intéressé peut se référer à [Moreau & Sébillot, 2005] pour une étude approfondie des contributions des techniques de TAL à la RI.

1.4.1 Palier morphologique

La morphologie concerne l'étude de la formation des mots et de leurs variations de forme. L'exploitation en RI d'informations morphologiques a pour objectif de permettre de reconnaître au sein des documents et requêtes les différentes formes d'un même mot et de pouvoir les apparier.

La première étape concerne la **segmentation** d'un texte en unités linguistiques. Elle consiste à trouver les unités élémentaires (« tokens ») qui correspondraient à des mots ainsi qu'à la délimitation des phrases. Cette partie nécessite le repérage des séparateurs, tant pour trouver les fins de phrases que les limites des mots, ce qui est souvent moins trivial que l'on pourrait penser. Ainsi l'apostrophe dans un mot comme « aujourd'hui » ne sépare pas deux unités lexicales, et un point dans un sigle n'indique pas forcément la fin d'une phrase. Des automates à états finis permettent de désambiguïser ces différentes formes.

Le découpage en unités élémentaires se fait généralement selon les espaces entre les chaînes de caractères. La difficulté consiste à reconnaître des unités qui correspondent vraiment à des mots. On s'aide généralement de lexique ou de modèles de mots (automates à états finis). En général, la recomposition succède à la segmentation. Mais, des ambiguïtés demeurent toujours. Dans la chaîne « pomme de terre cuite », a-t-on affaire à « pomme de terre » ou à « terre cuite » ? Ces ambiguïtés peuvent être propagées aux phases d'analyse supérieures qui peuvent disposer de plus d'information pour trancher.

La deuxième étape est l'**analyse lexicale** qui consiste à ramener les mots à une forme morphologique de base. La morphologie comporte deux aspects : la morphologie *flexionnelle*, qui concerne tout ce qui a trait à la conjugaison, au genre et au nombre des mots, et la morphologie *dérivationnelle*, qui s'intéresse à la façon dont les mots sont dérivés d'autres mots (par exemple l'adjectif dangereux est dérivé du nom danger) [Sébillot, 2002].

Cette analyse procède donc à une normalisation, c'est-à-dire un recodage des différentes variantes du mot par une forme unique, et vise à reconnaître la forme de base des mots et les différents affixes qui leur sont associés. Cette forme de base est soit le *lemme*, c'est-à-dire l'infinitif pour un verbe, la forme masculin singulier pour un nom (e.g. manger pour mangent, porte pour portes), etc, soit le *radical* (e.g. mang(e) pour manger, mangeoire, mangeables), soit la *racine* (e.g. nation pour nationalité)[9].

Cette procédure peut être effectuée par une racinisation (*stemming*) qui peut se faire par approximation de phénomènes linguistiques, en désuffixant et en recodant. Le désuffixage supprime les suffixes qui différencient les flexions des mots (les formes conjuguées d'un verbe par exemple).

[9]La différence entre racine et radical est souvent ambiguë. La racine est la forme abstraite servant de base de représentation à tous les radicaux qui en sont les manifestations [Sébillot, 2002].

Le recodage permet de regrouper suivant les allomorphes[10] et ajoute une terminaison prédéfinie. Ces deux phases peuvent être successives, comme dans le racineur Lovins [Lovins, 1968] ou simultanées comme dans celui de Porter [Porter, 1980].
La lemmatisation est une autre manière de normaliser, elle consiste à regrouper les mots qui ont une même origine si leur morphologie flexionnelle est différente en un même *lexème* ou *lemme*. Elle est dépendante du type grammatical de base (e.g. parti(verbe) : partir ; parti (nom) : parti). Cette procédure nécessite l'utilisation de dictionnaire.
La reconnaissance des mots est très importante en recherche d'information et constitue l'opération de base à l'indexation. Cependant certains systèmes s'en passent en procédant à une extraction des caractéristiques fondée sur les modèles de n-gramme. C'est-à-dire que toutes les chaînes de longueur fixe de n caractères sont extraites. Par exemple pour découper le texte « le terme » en n-grammes (n=5) on obtient le découpage suivant : le-terme ; le-te ; e-ter ; -term ; terme[11].
Des expériences ont montré que la racinisation et la lemmatisation améliorent significativement les performances pour les langues riches morphologiquement (e.g. le français, l'italien, etc.) [Jacquemin & Tzoukermann, 1997; Gaussier *et al.*, 1997; Gaussier *et al.*, 2000].

1.4.2 Palier syntaxique

L'exploitation en RI d'informations syntaxiques a pour objectif de permettre aux systèmes d'extraire de nouvelles entités linguistiques (des termes complexes, structurés, etc.) qui peuvent permettre de réduire l'ambiguïté, ces termes étant généralement moins polysémiques que les termes simples. L'analyse syntaxique s'intéresse à l'étude de la structure des phrases et des syntagmes. Elle permet de reconnaître la catégorie grammaticale d'un mot (verbe, nom, adverbe, etc.) et permet d'éliminer les problèmes d'homographie (e.g. normale est un nom ou un adjectif ?). Le problème de polysémie est propagé à la phase d'analyse sémantique.
Cette phase permet notamment de repérer les termes complexes, constitués d'au moins deux termes pleins (il s'agit des noms, des verbes, des adjectifs et parfois des adverbes). Les termes complexes ont l'avantage d'être moins ambigus que les simples. Ils peuvent être extraits soit par une analyse statistique, fondée sur la fréquence de co-occurrence des termes, soit par une analyse syntaxique, fondée sur une analyse de la structure en exploitant des patrons prédéfinis, soit les deux.
Les termes structurés sont des ensembles de mots qui ont des relations de dépendance, ce sont des structures composés d'un terme "tête" (élément central) qui régit un ensemble de termes modifieurs[12]. Il s'agit donc de repérer les groupes syntagmatiques (groupes nominaux, verbaux, etc.) qui déterminent la structure syntaxique et permettent d'établir les relations de dépendances.
L'analyse syntaxique permet également d'extraire des entités nommées. Ces dernières comprennent les organisations (entreprises, administrations, etc.), les lieux (ville, région, etc.), les personnes (hommes politiques, chefs d'entreprise, etc.) et les numériques (poids, valeurs monétaires, etc.). La reconnaissance d'entités nommées peut contribuer à la constitution d'index très discriminants et permet de répondre à des questions du type "Quelle est la ville organisatrice

[10] Variantes d'une même racine.
[11] - signifie un espace.
[12] Un terme composé peut être vu comme un sous-ensemble particulier de termes structurés.

des jeux olympiques 2012 ?".
L'analyse syntaxique peut conduire à une analyse profonde des documents, ce qui n'est pas toujours réalisable dans le cadre d'une RI où les temps de traitement sont un facteur important à prendre en compte. De plus, la détermination de termes complexes ou structurés doit être suivie par une validation humaine. La stratégie de pondération de ces termes est aussi importante.

1.4.3 Palier sémantique et pragmatique

Cette analyse est de loin la plus complexe, cela est dû essentiellement à l'ambiguïté sémantique. Elle permet d'étiqueter les termes selon des classes sémantiques. Ces classes peuvent être extraites automatiquement des documents (par apprentissage) ou extraites de ressources sémantiques externes.
L'étiquetage sémantique vise donc à associer à chaque mot en contexte une étiquette sémantique qui peut être une catégorie sémantique générale (e.g. événement, mouvement, etc.) ou un sens (e.g. souris-animal, souris-périphérique). Cette phase peut profiter de la phase de désambiguïsation syntaxique (e.g. un livre [nom masculin] vs une livre [nom féminin]). Elle nécessite un traitement linguistique assez fin telle que la résolution d'anaphores qui consiste à relier entre elles les références à une même entité au sein d'un texte. La résolution d'anaphores est typiquement utile dans plusieurs applications telles que l'extraction d'information, les systèmes question réponse, résumé automatique (voir section 1.10).
Nous détaillons plus en détail cette phase et présentons des SRI qui intègrent des connaissances sémantiques dans le chapitre 3.

1.5 Discussion

Les potentiels des outils de TAL pour la RI sont loins d'être totalement exploités. Cependant, on se trouve confronté au dilemme de la finesse des traitements (jusqu'où pousser l'analyse ?), par rapport à l'automatisation de la tâche d'indexation. Une analyse fine demande toujours une validation que les systèmes de RI (notamment pour le Web) ne peuvent se permettre. De plus, leurs apports dans le cadre de RI sont encore mitigés et dépendent de la langue, de la taille des corpus, du type de texte et du modèle de RI. La plupart des SRI procèdent à un étiquetage du texte suivant les catégories morpho-syntaxiques. A chaque mot est associé sa ou ses catégories grammaticales. Les étiqueteurs réalisent conjointement l'analyse morphologique des mots et l'analyse flexionnelle en les ramenant aux lemmes. Nous citons parmi ces étiqueteurs Brill [Brill, 1992] et le Treetagger que nous présentons brièvement.

Treetagger : analyseur morpho-syntaxique

TreeTagger est un étiqueteur morpho-syntaxique et un lemmatiseur développé par l'*Institute for Computational Linguistics de l'Université de Stuttgart*. Il est distribué librement à des fins d'évaluation, de recherche ou d'enseignement. Pour l'étiquetage, il implémente une méthode probabiliste (arbres de décision) nécessitant une phase d'entraînement ; il est donc possible de développer une version spécifique selon la langue pour laquelle on souhaite l'utiliser. Une ver-

sion dédiée au français est ainsi disponible sur la page d'accueil du TreeTagger[13] ; seul le fichier de paramètres varie : le moteur probabiliste reste inchangé. Trente-six étiquettes sont utilisées pour l'anglais contre soixante-deux pour le français. Par exemple, le texte suivant :

Le 29 mai 2005, les Français ont rejeté à 54,67% le projet de Constitution européenne
est étiqueté par le TreeTagger comme présenté dans le tableau 1.1[14].

Le	DET :ART	le
29	NUM	card
mai	NOM	mai
2005	NUM	card
,	PUN	,
les	DET :ART	le
Français	NAM	Français
ont	VER :aux :pres	avoir
rejeté	VER :pper	rejeter
à	PRP	à
54,67%	NUM	card
le	DET :ART	le
projet	NOM	projet
de	PRP	de
Constitution	NOM	constitution
européenne	ADJ	européen

TAB. 1.1 – Exemple de sortie du Treetagger

La plupart des étiqueteurs sont à performance égale et atteignent des précisions avoisinant les 95%.
Une fois l'ensemble des termes extraits, on sélectionne généralement les lemmes des mots pleins, (Français avoir rejeter projet constitution européen) pour l'exemple précédent. Le plus souvent, on utilise d'abord un **dictionnaire d'arrêt** (*stop words*) pour supprimer les mots peu informatifs qui sont généralement des mots courts, communs qui n'ont pas de signification du point de vue de la recherche, comme "le" "un" "de" et "pour".
Cette analyse ne donne aucune indication sur l'importance d'un terme dans le document. Des calculs de fréquence sont donc appliqués aux termes pour évaluer leur *pertinence* dans un document, c'est le rôle des mesures de pondération.

1.6 Pondération des unités d'index

Les méthodes statistiques furent exploitées dès le début de la RI pour la pondération des termes. L'objectif est de trouver les termes qui caractérisent le mieux le contenu d'un document. La manière la plus simple pour calculer le poids d'un terme est de calculer sa fréquence d'apparition, un terme qui apparaît souvent dans un document peut bien caractériser son contenu. Cependant, "l'informativité" d'un terme, c'est-à-dire, l'information sur le document que véhicule le terme, dépend aussi de sa répartition dans toute la base, on définit ainsi le pouvoir discriminatoire d'un terme, c'est-à-dire, le degré avec lequel un terme distingue un document d'un autre. Plusieurs fonctions de pondération de termes ont été proposées. Nous présentons le tf-idf (*term frequency - inverse document frequency*) utilisé dans le modèle vectoriel (cf section 1.7.2) et que nous adaptons dans notre travail.
Les dimensions retenues pour calculer le poids d'un terme sont :

[13] http ://www.ims.uni-stuttgart.de/projekte/corplex/
[14] Les mots qui ne sont pas reconnus par le TreeTagger se voient attribuer l'étiquette "UNKNOWN".

– Une pondération locale qui détermine l'importance d'un terme dans un document. Elle est, généralement, représentée par sa fréquence (*tf*).
– Une pondération globale qui détermine la distribution du terme dans la base documentaire. Elle est, généralement, représentée par l'inverse de la fréquence des documents qui contiennent le terme (*idf*).

Plusieurs formules sont proposées pour le calcul du tf et du idf. Nous en présentons quelques unes -dont la plupart sont implémentées dans le système SMART[15]- dans le tableau suivant [Beigbeder & Mercier, 2003] avec leurs images (intervalle des valeurs) :

Les différentes fonctions tf		Les différentes fonctions idf	
Formule	image	Formule	image
$f(d,t)$	$[0,+\infty]$	$\frac{1}{df(t)}$	$[\frac{1}{\|D\|},1]$
$\frac{f(d,t)}{max_{t'}f(d,t')}$	$[0,1]$	$log(1+\frac{max_{d,t'}f(d,t')}{df(t)})$	$[log(2),log(1+cste)]$
$\frac{1}{2}+\frac{1}{2}\frac{f(d,t)}{max_{t'}f(d,t')}$	$[\frac{1}{2},1]$	$log(1+\frac{\|D\|}{df(t)})$	$[log(2),log(\|D\|+1)]$
$1+log(f(d,t))$	$[1,+\infty]$	$log(\frac{\|D\|}{df(t)})$	$[0,log(\|D\|)]$

TAB. 1.2 – Les différentes fonctions tf et idf

Les données de base de ces formules sont $f(d,t)$ qui est la fréquence du terme dans le document et $df(t)$ qui est le nombre de documents ayant au moins une occurrence du terme t. Les fonctions tf dénotent une monotonie croissante et df une monotonie décroissante.
Les meilleurs termes étant ceux qui apparaissent fréquemment dans certains documents mais rarement dans le reste de la collection, un document pourra alors être représenté par ses seuls termes ayant un tf-idf suffisant.
Une lacune de la pondération par le $tf-idf$ est que ce dernier ne tient pas compte de la longueur du document. Des techniques de normalisation ont été proposé (eg. BM25 proposée par Robertson & Walker [Robertson & Walker, 1994] ; [S.E. Robertson & Payne, 1995] ; [Robertson & Jones, 1997]) pour ne pas donner plus d'importance à un document long qu'à un document court.

1.7 Modèles de Recherche d'Information

Plusieurs modèles sont implémentés dans les systèmes de recherche d'information. Nous présentons les plus connus à savoir le modèle booléen, le modèle vectoriel et le modèle probabiliste. Ces modèles ont été adapté par plusieurs travaux pour prendre en considération l'aspect structuré des documents XML, nous les présentons dans le chapitre 2.

1.7.1 Le modèle booléen

Ce modèle est le plus simple des modèles de RI, il repose sur l'algèbre de Boole. Un document est représenté par une conjonction de termes (non pondérés) : $d = t_1 \wedge t_2 \wedge \ldots \wedge t_n$. Une requête est une expression logique de termes en utilisant les opérateurs AND ($\wedge$), OR ($\vee$) et NOT ($\neg$),

[15] ftp ://ftp.cs.cornell.edu/pub/smart/smart.11.0.tar.Z

par exemple $q = (t_1 \wedge t_2) \vee \neg t_3$[Salton & McGill, 1983]. Pour qu'un document corresponde à une requête, il faut que l'implication $d \rightarrow q$ soit valide. La correspondance entre le terme et la requête est déterminée de la manière suivante :

$corr(d, q_i) = 1$ si $q_i \in d$
$corr(d, q_1 \wedge q_2) = 1$ si $corr(d, q_1) = 1$ **ET** $corr(d, q_2) = 1$; 0 sinon.
$corr(d, q_1 \vee q_2) = 1$ si $corr(d, q_1) = 1$ **OU** $corr(d, q_2) = 1$; 0 sinon.
$corr(d, \neg q_1) = 1$ si $corr(d, q_1) = 0$; 0 sinon.

Les documents retournés par le système sont considérés à pertinence égale. La conjonction est très contraignante (un document qui contient quelques uns des termes recherchés est rejeté au même titre qu'un document qui ne contient aucun terme) et la disjonction très permissive (les documents contenant seulement un terme sont aussi bons qu'un document qui satisfait tous les termes). Les termes dans le document ou la requête ont une pondération binaire (1 si présent et 0 si absent), il n'est pas possible d'exprimer qu'un terme est plus important qu'un autre. De plus, la formulation booléenne des requêtes complexes n'est pas évidente pour des utilisateurs non expérimentés.
Toutes ces raisons font que le modèle booléen standard est rarement utilisé de nos jours, mais plutôt les extensions proposées pour corriger ses lacunes. Nous présentons brièvement le modèle booléen pondéré ainsi que le modèle p-norme (qui combine le modèle vectoriel et le modèle booléen) dans les sections 1.8.1 et 1.8.2.

1.7.2 Le modèle vectoriel (*VSM : Vector Space Model*)

Le modèle vectoriel (*VSM : Vector Space Model*) est le fondement de très nombreux SRI. L'implémentation la plus connue est le système SMART présenté par Salton [1971]. Les documents ainsi que les requêtes sont représentés comme des vecteurs de mots qui le caractérisent le plus avec leurs pondérations [Salton *et al.*, 1975; Salton & McGill, 1983]. La recherche d'information revient à une recherche dans l'espace vectoriel fondée sur une fonction d'appariement entre vecteurs. La fonction d'appariement implémente les mesures de similarité entre vecteurs.
Un document d contient un ensemble de descripteurs $t_1..t_n$ (les descripteurs peuvent être des mots simples ou complexes (cf. section 1.4)). A chaque descripteur est assignée une pondération w_i [Salton & Buckley, 1988]. Le document est alors représenté par un vecteur de poids des descripteurs :

$$\overrightarrow{d} = (w_1, w_2, .., w_n)$$

La requête est aussi représentée par un vecteur de poids des termes recherchés :

$$\overrightarrow{Q} = (q_1, q_2, .., q_m)$$

où q_i représente le poids des termes recherchés par la requête. Ces termes recherchés doivent correspondre à des descripteurs d'indexation.
Le poids des termes peut être calculé en fonction de leur fréquence dans le document ainsi que leur distribution dans l'ensemble des documents (cf section 1.6). Pour chaque vecteur document un score est calculé en utilisant une mesure de correspondance avec la requête, les documents sont ordonnés par rapport à ce score et sont retournés par ordre décroissant de leur similarité.

L'évaluation de la correspondance entre un document et une requête, dans le modèle vectoriel, revient à un calcul dans l'espace géométrique. Plusieurs mesures sont proposées et elles sont plus ou moins équivalentes. Cette mesure peut être un simple calcul de produit scalaire :

$$produit = \sum_{i=1}^{m} q_i * w_i$$

La mesure la plus connue est le cosinus[16] de l'angle des vecteurs qui donne une valeur normalisée entre [0, 1][17] :

$$cosinus(Q, d) = \frac{\overrightarrow{Q} * \overrightarrow{d}}{||\overrightarrow{Q}||*||\overrightarrow{d}||} = \frac{\sum_{i=1}^{m} q_i * w_i}{\sqrt{\sum_{i=1}^{m} q_i^2} \cdot \sqrt{\sum_{i=1}^{m} w_i^2}}$$

Où q_i et w_i représentent respectivement les poids des termes indexés dans la requête et le document. $\sum_{i=1}^{m} q_i^2$ et $\sum_{i=1}^{m} w_i^2$ sont les normes euclidiennes des vecteurs $\overrightarrow{d}$ et $\overrightarrow{Q}$ ($||\overrightarrow{d}||$ et $||\overrightarrow{Q}||$)
Les avantages d'un tel modèle est le calcul de pertinence d'un document par rapport aux poids de ses termes et ceux de la requête. Si un terme n'apparaît pas dans un document, la pertinence de ce dernier par rapport à la requête en sera amoindrie. La principale limite est que l'ajout d'un document oblige à recalculer tous les poids de ses termes.

1.7.3 Le modèle probabiliste

Le but de ce modèle est de calculer la probabilité qu'un document soit pertinent par rapport à la requête. En utilisant le théorème de Bayes [VanRijsbergen, 1979] les probabilités de pertinence ($P(R|d)$) et de non pertinence ($P(NR|d)$) d'un document sont calculées par :

$$P(R|d) = \frac{P(d|R) * P(R)}{P(d)}$$
$$P(NR|d) = \frac{P(d|NR) * P(NR)}{P(d)}$$

Où
- $P(d|R)$ est la probabilité que d fasse partie de l'ensemble de documents pertinents.
- $P(d|NR)$ est la probabilité que d fasse partie de l'ensemble de documents non pertinents.
- $P(R)$ est la probabilité de pertinence d'un document quelconque du corpus.
- $P(NR)$ est la probabilité de non pertinence d'un document quelconque du corpus.
- $P(d)$ est la probabilité que le document d soit choisi.

La fonction de correspondance évalue la pertinence d'un document par rapport à la requête :

$$P = \frac{P(R|d)}{P(NR|d)}$$
$$\approx \frac{P(d|R)}{P(d|NR)} \text{ si on considère que } P(R) \text{ et } P(NR) \text{ sont des constantes.}$$

Cela nous ramène à l'estimation de $P(d|R)$ et $P(d|NR)$. Si on considère que les termes sont indépendants :

$$P(d|R) = \prod_{t_i \in d} P(t_i|R)$$
$$P(d|NR) = \prod_{t_i \in d} P(t_i|NR)$$

[16] Si les poids des termes sont normalisés, la formule du $cosinus$ revient à calculer le produit scalaire.
[17] D'autres mesures de correspondance existent parmi lesquelles on peut citer les mesures de Jaccard et Dice.

Avec $P(t_i|R)$ (respectivement $P(t_i|NR)$) la probabilité de l'apparition du terme t_i dans un document pertinent (respectivement non pertinent).
Pour calculer ces probabilités, on peut procéder par échantillonnage, en choisissant deux ensembles de documents pertinents ou non pertinents. Ils peuvent aussi être calculés pour chaque terme en utilisant les retours arrière sur pertinence. Les poids sont modifiés par rapport à la distribution du terme dans les documents pertinents ou non pertinents. Dans la pratique, la plupart des expérimentations se fondent sur des méthodes de tf-idf (voir section 1.6) [Jones *et al.*, 1998]. L'avantage du modèle probabiliste est l'amélioration dynamique des performances du système, mais la mise à jour du système est très coûteuse. Parmi les systèmes fondés sur le modèle probabiliste, nous pouvons citer le moteur de recherche OKAPI [S.E. Robertson & Payne, 1995] ; [Robertson & Jones, 1997].

1.8 Modèles étendus

Nous présentons dans cette section le modèle booléen basé sur des ensembles flous et le modèle p-norme ainsi que les modèles LSI (*Latent Semantic Indexing*) et DSIR (*Distributional Semantics based Information Retrieval*) qui se proposent d'améliorer le modèle vectoriel et se rapprochent de ce fait de nos travaux.

1.8.1 Le modèle booléen basé sur des ensembles flous

Cette extension du modèle booléen tient compte de la pondération des termes dans les documents [Salton *et al.*, 1983]. Le document est donc représenté par un ensemble de termes pondérés. La requête reste toujours une expression booléenne. L'évaluation de l'appariement d'un terme à une requête repose sur une des évaluations classiques proposées par Zadeh [1965] dans le cadre des ensembles flous :

$$
\begin{aligned}
&corr(d, q_i) = w_{t_i} \\
&corr(d, q_1 \wedge q_2) = min(corr(d, q_1), corr(d, q_2)) \\
&corr(d, q_1 \vee q_2) = max(corr(d, q_1), corr(d, q_2)) \\
&corr(d, \neg q_1) = 1 - corr(d, q_1)
\end{aligned}
$$

où w_{t_i} est le poids du terme t_i de la requête q_i dans le document d. Ces évaluations permettent de présenter les résultats par ordre de pertinence. Cependant, elles ne reflètent pas parfaitement le sens des opérateurs booléens. Ainsi par exemple $corr(d, q \wedge \neg q)$ n'est pas égale à 0 et $corr(d, q \vee \neg q)$ n'est pas égale à 1.

1.8.2 Le modèle p-norme

Le but de ce modèle est d'attribuer une pondération aux termes des documents et des requêtes ainsi qu'aux opérateurs booléens (AND et OR). Il permet de prendre en considération les poids des termes et ceux de la requête et relâche les contraintes des opérateurs booléens [Salton, 1984].

$$corr(d, q_{and}) = 1 - \sqrt[p]{\frac{[1-x_1]^p + [1-x_2]^p + \ldots + [1-x_m]^p}{m}}$$

$$corr(d, q_{or}) = \sqrt[p]{\frac{x_1^p + x_2^p + ... + x_3^p}{m}}$$

Cette évaluation correspond au modèle vectoriel[18] quand p est égale à 1 et au modèle booléen fondé sur les ensembles flous quand p est égale à ∞.

1.8.3 Le modèle LSI (*Latent Semantic Indexing*)

Le modèle LSI (*Latent Semantic Indexing* proposée par Deerwester *et al.* [1990], est un modèle algébrique de RI fondé sur la décomposition en valeurs singulières (*SVD : Singular Value Decomposition*) de la matrice (terme-document) qui représente l'espace d'indexation du modèle vectoriel.
Cette matrice est projetée dans un espace de dimension plus faible où les descripteurs considérés ne sont plus de simples termes. Avec cette méthode, les termes apparaissant ensemble sont projetés sur la même dimension. Cette représentation est censée résoudre partiellement le problème des synonymie et de polysémie. Elle permet de trouver des documents pertinents pour une requête même s'ils ne partagent aucun mot avec cette requête. Grâce à une analyse statistique de grands corpus, le sens de chaque mot est caractérisé par un vecteur dans un espace de grandes dimensions, la proximité entre deux vecteurs correspondant à la proximité de sens de ces mots. Cette analyse statistique consiste à construire une matrice d'occurrences qui sera réduite afin de faire ressortir les relations sémantiques *« latentes »* entre mots ou entre textes. En effet, deux mots peuvent être considérés sémantiquement proches s'ils sont utilisés dans des contextes similaires. Le contexte d'un mot est ici défini comme l'ensemble des mots qui apparaissent conjointement avec lui. Cette notion de co-occurrence est évidemment statistique : la méthode fonctionne si un nombre suffisant de textes est utilisé.
Cette approche permet donc de représenter les termes de la collection suivant la structure sémantique latente.
Le LSI utilise une matrice X (terme-document) qui est composée des vecteurs de termes et de documents (comme pour le VSM). Elle utilise la technique de décomposition à valeur singulière afin d'approximer la matrice terme-document par des combinaisons linéaires et permet donc de créer un nouvel espace vectoriel :

$$X_{t*d} = T_{0_{t*m}} * S_{0_{m*m}} * D'_{0_{m*d}}$$

Où
- T_0 est la matrice orthogonale des vecteurs singuliers de gauche, $(T_0 T_0^{'} = I)$
- D_0 est la matrice contenant les colonnes orthogonales des vecteurs singuliers de droite, $(D_0 D_0^{'} = I)$
- $D_0^{'}$ est le transposé de la matrice D_0,
- S_0 est la matrice diagonale (triée) des valeurs singulières.
- t est le nombre de lignes dans X, d est le nombre de colonnes dans X et m est le rang de X tel que $(m \leq min(t, d))$.

Il est prouvé qu'il existe une seule décomposition de cette manière [Golub & Loan, 1996].

[18] La mesure de correspondance est dans ce cas équivalente au produit scalaire

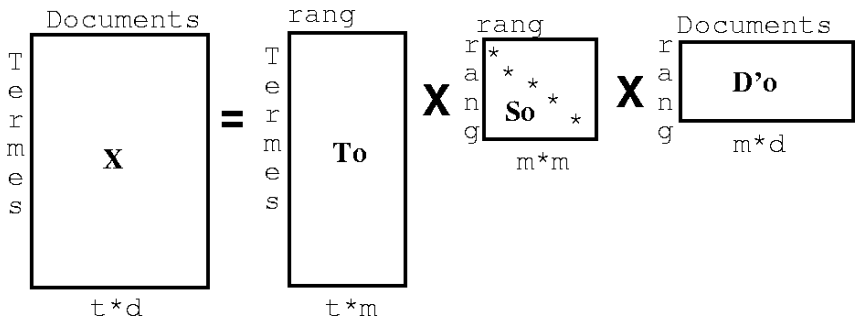

FIG. 1.2 – Représentation de la décomposition en valeurs singulières de la matrice X

Cette matrice est par la suite réduite par la matrice Xh contenant les plus grandes valeurs singulières k ($k \leq m$)[19].

$$Xh_{k*d} = T_{t*k} * S_{k*k} * D'_{k*d}$$

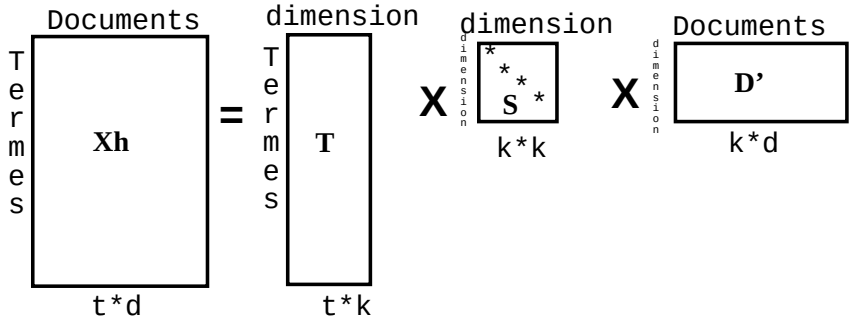

FIG. 1.3 – Réduction de la SVD de la matrice X

Xh ne garde que les k premières valeurs et permet donc de représenter les documents dans un espace de dimension k.
L'espace sémantique étant construit, la proximité sémantique entre deux mots est déterminée par le cosinus de leur angle. La requête est aussi traduite dans ce nouvel espace, elle est transformée en pseudo-document (il est à noter que les nouveaux documents à indexer suivent aussi ce même processus). La requête est traduite en

$$D_q = X'_q * T * S^{-1}$$

Où X_q est le vecteur de mots clés de la requête.
Le pseudo-document est rajouté à la matrice D (comme un nouveau document) et le calcul de similarité se fait par :

[19] Les valeurs les plus faibles de S_0 sont remplacées par la valeur 0

$$Xh^{'} * Xh = D * S^2 * D^{'}$$

La recherche d'information du modèle LSI peut se faire à trois niveaux : on peut comparer deux termes (quelle similarité entre le terme i et le terme j), deux documents (quelle similarité entre un document i et un document j) ou un terme par rapport à un document (quelle association entre le terme i et le document j). L'approche est la même que pour le modèle vectoriel, en calculant la similarité entre les vecteurs de termes, les vecteurs de documents ou la fréquence d'un terme dans un document. Elle utilise pour cela la matrice Xh qui représente les unités utiles de X.

– Comparer deux termes
  $Xh = T * S^2 * T^{'}$
– Comparer deux documents
  $Xh^{'} * Xh = D * S^2 * D^{'}$
– Comparer un terme et un document
  $Xh = T * S * D^{'}$

Une phase d'apprentissage permet de calculer la matrice X. Les documents qui n'ont pas servis à la phase d'apprentissage sont ajoutés à cet espace réduit en approximant leur position suivant le vecteur contenant le vocabulaire qui le caractérise. Ce qui suppose que l'espace LSI créé au départ caractérise bien les dimensions importantes de similarité pour pouvoir approximer un nouveau terme ou un nouveau document dans la collection. Ce genre d'approche suppose que l'échantillon utilisé pour l'apprentissage est réellement représentatif de la collection de documents.

Le paramètre k est important à définir car une réduction à un espace de trop grande dimension ne ferait pas suffisamment émerger les liaisons sémantiques entre mots, et un trop petit nombre de dimensions conduirait à une trop grande perte d'informations. Le nombre adéquat de dimensions ne peut pas être actuellement déterminé théoriquement ; seuls des tests empiriques ont permis de situer cette valeur entre 100 et 300 dans le cas de l'anglais [Deerwester *et al.*, 1990]. De plus, les valeurs de la matrice après réduction ne sont pas interprétables (par les êtres humains).

Les associations sémantiques extraites par le LSI ne se réduisent pas uniquement à des relations de synonymie, puisqu'elles proviennent d'un traitement de la co-occurrence des termes avec leurs contextes. Ces associations sémantiques ont l'avantage d'être relativement générales mais possèdent l'inconvénient de ne pouvoir être caractérisées plus finement. Ce modèle a montré des performances très intéressantes. Pour un corpus de petit ou moyenne taille, la performance est très supérieure au modèle vectoriel classique, et est un des meilleurs modèles. Quand la taille de corpus augmente, la différence avec les autres modèles classiques semble diminuer [G.W. Furnas & Lochbaum, 1988].

Notre proposition se différencie du modèle LSI par l'utilisation d'une ressource externe (l'ontologie). Cependant nous partageons les mêmes intuitions sur l'utilisation d'une approche distributionnelle qui nous permet d'évaluer la similarité sémantique d'un terme avec les termes qui sont utilisés dans des contextes similaires.

1.8.4 Le modèle DSIR (*Distributional Semantics based Information Retrieval*)

Le modèle DSIR (*Distributional Semantics based Information Retrieval*) propose d'étendre le modèle vectoriel en introduisant une représentation des documents qui intégre une **information sémantique**, en utilisant une représentation distributionnelle, fréquentielle et co-fréquentielle de la sémantique d'un terme [Rajman *et al.*, 2000].

Ce modèle repose sur la notion de sémantique distributionnelle [Rajman & Bonnet, 1992] qui s'inscrit dans la continuité des approches distributionnalistes du langage [Harris *et al.*, 1989].

Le modèle DSIR est un modèle vectoriel permettant d'intégrer des informations sémantiques supplémentaires par l'utilisation de co-occurrences [Besançon, 2002]. Cette approche suppose l'existence d'une corrélation forte entre la co-occurrence des mots et leurs sens. Les contextes dans lesquels apparaissent les mots apportent suffisamment d'informations pour identifier leur sens. Besançon *et al.* [2003] présentent l'exemple suivant :

(1) "Certains X, par exemple, attaquent naturellement les rats." (*Darwin, l'origine des espèces)*
(2) "Quelque X sur les toits, marchant lentement, bombait son dos aux rayons pâles du soleil." *(Flaubert, Madame Bovary)*
(3) "Il entendait au loin dans la forêt les miaulements des X." *(Anatole France, L'étui de nacre)*

L'hypothèse centrale est que ces contextes apportent suffisament d'information pour identifier X comme une *sorte de* chat.

La définition d'un contexte d'un mot est alors une étape cruciale puisqu'elle influence directement le sens qui peut être attribué à un terme. Selon le type de relations sémantiques à identifier, trois sortes de contexte peuvent être définis [Habert *et al.*, 1997] :

Contexte positionnel : les contextes sont définis par des fenêtres de n mots, la valeur de n est choisie selon le type de relation contextuelle (n est petit pour favoriser les relations de composition ; n est grand pour favoriser les mots du même champs sémantique [Church & Hanks, 1989]).

Contexte documentaire : les contextes sont définis par les unités textuelles à l'intérieur d'un document (paragraphe, section, chapitre). La taille de l'unité choisie influence aussi le type de relation contextuelle.

Contexte syntaxique : les contextes dépendent de la structure syntaxique, c'est-à-dire des mots dans le même groupe syntaxique ou en relation de dépendance syntaxique. C'est ce type de contexte qui a été choisi dans le modèle DSIR.

Voici un exemple de relations de co-occurrences représentées dans un **graphe de co-occurrence**[20], pour le contexte représenté par la phrase "L'acteur porte un masque grimaçant de théâtre antique".

[20] Les noeuds représentent les lemmes et les arcs représentent les relations de co-occurrence.

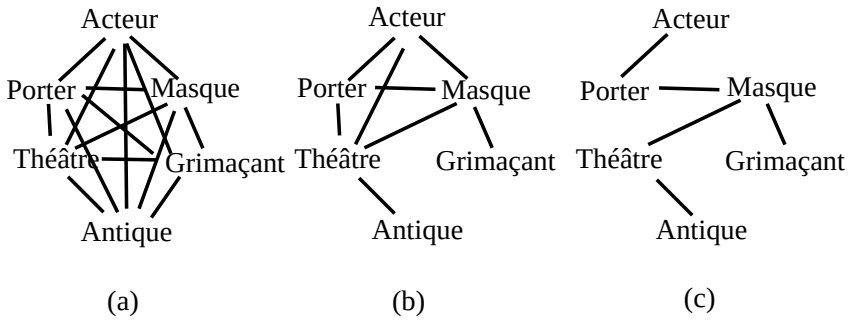

FIG. 1.4 – Exemple de graphes de co-occurrences pour la phrase "L'acteur porte un masque grimaçant de théâtre antique", (a) sans filtrage syntaxique, (b) avec filtrage sur les groupes syntaxiques, (c) avec filtrage sur les relations syntaxiques [Besançon *et al.*, 2003]

Les expérimentations ont montré qu'un contexte syntaxique avec filtrage sur les groupes syntaxiques est le plus adéquat dans le cadre d'une RI, contrairement à un contexte positionnel (qui rajoute des relations non suggérées dans la phrase) ou un contexte syntaxique avec filtrage sur les relations syntaxiques [Besançon, 2002].
Le modèle DSIR est aussi fondé sur la notion d'unités linguistiques par rapport aux termes retenus pour l'indexation. Les unités linguistiques sont les lemmes des noms, verbes et adjectifs, les termes d'indexation sont les unités linguistiques retenues pour leur "représentativité" [21] des documents.
Dans le cadre de ce modèle, les unités linguistiques u_i considérées sont représentées par un vecteur $c_i = (c_{i1}, .., c_{in})$, appelé profil de co-occurrence, dont chaque composante c_{ij} est la fréquence de co-occurrence de l'unité linguistique u_i avec un terme d'indexation t_j, n étant le nombre de termes retenus pour l'indexation. L'ensemble des unités linguistiques est donc représenté par une matrice de co-occurrences de dimension $m * n$ (où m est le nombre d'unités linguistiques choisies)

$$C = \begin{Bmatrix} c_{11} & \dots & c_{1n} \\ c_{21} & \dots & c_{2n} \\ \dots & \ddots & \dots \\ c_{m1} & \dots & c_{mn} \end{Bmatrix}$$

Un document d est alors représenté comme la somme pondérée des profils de co-occurrence des unités linguistiques qu'il contient, c'est-à-dire par un vecteur

$$d = (d_1, .., d_n)$$

où chaque d_j est défini par :

$$d_j = \sum_{u_i \in d} w_i * c_{ij}$$

où la pondération w_i est définie de la même manière que le modèle vectoriel, le $tf - idf$ dans ce modèle.
La collection de document est alors représentée par le produit matriciel :

[21] Généralement en fonction de leur fréquence.

$$D = F * C$$

Où C est la matrice de co-occurrences déjà présentée et F est une matrice qui représente les documents et les unités linguistiques.
Les termes explicitement présents dans le document ne sont pris en compte que par le biais de leur profil de co-occurrence car généralement c_{ii} étant nul, l'information concernant la présence d'un terme est perdue. Pour tenir compte de la présence d'un terme dans un document, un modèle DSIR hybride est proposé pour intégrer à la fois les occurrences et les co-occurrences des termes dans le document. Dans ce modèle, un document est toujours représenté par $(d_1, .., d_n)$ où d_j) :

$$\mathrm{d}_j = \alpha\ \mathrm{w}_j + (1 - \alpha) \sum_{u_i \in d} w_i * c_{ij}$$

où α ($0 \leq \alpha \leq 1$) représente le facteur d'hybridation permettant de contrôler l'importance relative dans l'hybridation du modèle DSIR par rapport au modèle VSM standard.
L'ensemble des documents est alors représenté par

$$D = \alpha F' + (1 - \alpha) F * C$$

Où F' est la matrice F de dimension réduite au nombre de termes. Elle correspond donc au modèle vectoriel standard.
La similarité entre les documents est calculée par les mesures de similarité usuelles du modèle vectoriel (cf section 1.7.2). L'intégration des co-occurrences a été testée sur les données de la campagne Amaryllis[22]. Le modèle DSIR a montré des améliorations par rapport au modèle vectoriel (Rajman, 2000). Ce qui indique que la prise en compte des co-occurrences améliore la représentation des documents. Nous proposons une approche similaire au modèle DSIR puisque nous nous fondons sur la thèse de similarité distributionnelle. Le modèle DSIR est fondé sur la notion d'unités linguistiques par rapport aux termes retenus pour l'indexation. Les unités linguistiques sont les lemmes des noms, verbes et adjectifs, les termes d'indexation sont les unités linguistiques retenues pour leur représentativité des documents. La définition du contexte influence la qualité des résultats obtenus. La matrice de co-occurrence donne les fréquences des co-occurrences des unités linguistiques par rapport aux termes d'indexation et risque donc d'avoir beaucoup de valeurs nulles. Nous montrons dans le chapitre 6, que notre contexte est plus large que celui du modèle DSIR mais que les calculs de similarité sont limités au vocabulaire de ce contexte.

1.9 Evaluation d'un SRI

L'évaluation sert à juger la performance ou la valeur d'un système. L'évaluation en RI est apparue avec les premiers prototypes, le but étant d'expérimenter les potentiels des systèmes entièrement automatiques. Kent *et al.* [1955] furent les premiers à proposer les critères de pertinence et les mesures de précision et de pertinence (appelée plus tard rappel) pour l'évaluation des systèmes RI. L'évaluation d'un système repose sur les éléments suivants :

[22] Amaryllis est une Action de Recherche Concertée cofinancée par l'Aupelf-Uref et par le MENRT. Ce projet cherche à promouvoir l'élaboration de corpus et de procédures d'évaluation concernant le français.

(i) Définition du (ou des) critère(s) représentant l'objectif de l'évaluation ;
(ii) Définition des mesures fondées sur un tel critère ;
(iii) Définition d'un instrument de mesure (eg. Les jugements des utilisateurs) ;
(iv) Définition d'une méthodologie pour obtenir de telles mesures et conduire de telles évaluations.

Les propriétés -proposées déjà en 1967 par Swets- nécessaires pour une mesure d'évaluation sont les suivantes :

– Elle doit exprimer la capacité d'un système RI à distinguer entre les documents pertinents et les documents non pertinents.
– Elle ne doit pas être influencée par le nombre de documents retournés.
– Elle doit être immédiatement appréhendée, dans le sens où on peut se représenter la performance du système à la seule vue de la mesure (une mesure réelle de préférence).
– Elle doit permettre d'établir une relation d'ordre total des différentes performances des systèmes et doit avoir un vrai zéro et une valeur maximale.

D'après Saracevic [1995], évaluer un système revient à poser des questions sur ses performances en fonction des objectifs fixés : à quel point le système (ou un de ses composants) exécute-t-il ce pourquoi il a été conçu ? Quel objectif du système RI doit être pris en compte ? Il a proposé six classes ou paliers d'objectifs pour un SRI :

– Le palier *ingénierie* où les questions matérielles et logicielles sont posées, comme les questions de maintenance, de cohérence, de flexibilité, etc.
– Le palier *entrées du système* comme le degré de couverture du domaine.
– Le palier *traitements* faits par le système ce qui regroupe l'évaluation des algorithmes, approches, etc.
– Le palier *sorties du système* où on s'intéresse aux interactions avec le système et les résultats fournis ce qui inclut l'évaluation de la recherche, les interactions, les bouclages, etc.
– Le palier *utilisation et utilisateur* ce qui comprend l'application à certains problèmes et tâches.
– Le palier *aspect social* qui évalue l'impact du système dans l'environnement de l'utilisateur (Comment la RI affecte la productivité, la prise de décision, etc.).

Ces paliers ne sont pas mutuellement exclusifs. La plupart des évaluations portent sur le palier concernant les traitements, les mesures les plus répandues sont la précision et le rappel (voir figure 1.5) où les réponses du système pour toutes les requêtes de la collection de test sont comparées aux jugements de pertinence des utilisateurs.

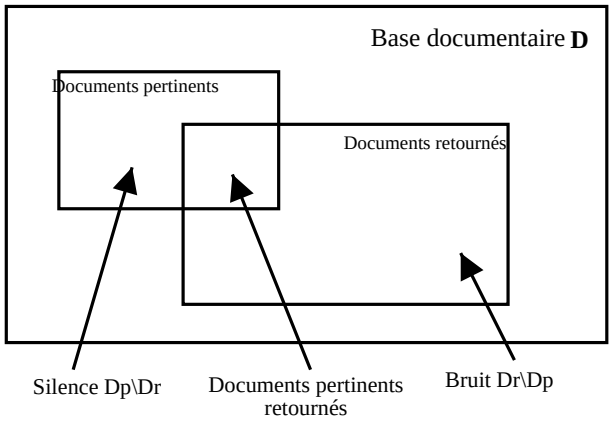

FIG. 1.5 – Evaluation dans un SRI

Soient D_r l'ensemble de documents retournés par un système pour une requête donnée. D_p l'ensemble de documents pertinents à cette même requête. La précision est la proportion des documents retournés qui sont pertinents.

$$\text{Précision} = \frac{|D_r \cap Dp|}{|D_r|}$$

Le rappel est la proportion des documents pertinents qui sont retournés.

$$\text{Rappel} = \frac{|D_r \cap D_p|}{|D_p|}$$

Le résidu (fallout) proposé par VanRijsbergen [1979] donne une indication de la proportion de documents non pertinents renvoyés par le système.

$$\text{Résidu} = \frac{|D_r \cap \neg D_p|}{|\neg D_p|}$$

La précision est grande (proche de 1) quand la quantité de bruit est faible et le rappel est grand (proche de 1) quand la quantité de silence est faible (voir figure 1.5). Les systèmes tendent à améliorer le taux de précision et de rappel, le système « idéal » devrait avoir pour objectif un rappel et une précision élevés. Cependant ces deux taux ont souvent tendance à fonctionner en opposition, et chaque système cherche généralement un équilibre entre ces deux valeurs, favorisant parfois l'une aux dépens de l'autre selon le but visé.
La précision peut être directement calculée par les jugements de pertinence, le rappel est un peu plus compliqué parce qu'il ne dépend pas seulement de ce qui a été retournés mais aussi des documents qui sont pertinents et qui n'ont pas été retrouvé par le système. Dans le cas des corpus « jouet » ou de petite base documentaire, tous les documents peuvent être parcourus pour juger de leur pertinence, cela n'est pas possible pour les grands corpus (ceux de la campagne d'évaluation TREC[23] par exemple). Une des méthodes consiste à parcourir uniquement les documents qui ont été retournés par les systèmes en compétition. Dans des systèmes ouverts tels que le Web, la notion de rappel n'est plus très pertinente puisqu'il est impossible de connaître l'intégralité des documents satisfaisant une requête.

[23] La campagne d'évaluation TREC (*Text REtrieval Conference*) est une plate-forme internationale d'évaluation de Systèmes de Recherche d'Information

Le rappel et la précision sont calculés pour chaque requête mais il est possible de synthétiser les évaluations et les présenter par rapport à toutes les requêtes posées. Une moyenne est calculée et les points rappels/précision sont interpolés pour calculer la précision à des points standards (on obtient ainsi une mesure scalaire de la qualité du système), généralement on calcule la précision aux 11 points standards de rappel de 0 à 1 (par pas de 0.1). On peut aussi calculer la *R-précision* qui est la précision obtenue pour un nombre de documents retournés correspondant au nombre de documents pertinents présents dans la base (donc pour ce cas la précision est égale au rappel).
Le jugement par des utilisateurs humains, bien que nécessaire, pose les problèmes d'objectivité. De plus il est difficile de caractériser la notion de pertinence d'un document. Elle peut différer d'un individu à un autre, pour une même requête posée les attentes des utilisateurs peuvent êtres différentes. Les jugements peuvent aussi varier pour un même utilisateur suivant ce qu'il a consulté avant. Parmi les hypothèses posées pour de telles évaluations figure celle du jugement absolu, on suppose ainsi que le jugement sur la pertinence ne dépend pas de l'ordre des documents consultés, ce qui, dans la pratique, n'est pas évident.

1.10 Conclusion

Nous venons de voir un aperçu de ce que peut être un Système de Recherche d'Information. Notre but premier est de rappeler les fondements de base d'un tel système. Nous avons discuté l'importance de la détermination des unités à indexer, leur pondération ainsi que le choix d'un modèle de RI. Nous avons souligné l'importance des campagnes d'évaluation pour promouvoir les recherches dans ce domaine, elles ont largement contribué à l'amélioration des systèmes de recherche actuels.
La RI permet d'accéder aux contenus des documents. D'autres méthodes d'accès à l'information existent et présentent des problématiques proches de celles traités en RI. Parmi ces méthodes, nous distinguons les systèmes Question/réponse, de résumé automatique et d'aide à la navigation que nous présentons ci dessous.

Les systèmes de Question/réponse

Ils visent à répondre à des besoins précis en matière d'information. Le système QALC [Ferret *et al.*, 2001] commence par analyser les questions afin d'en extraire une requête contenant les termes significatifs et le type de réponse attendu, les documents sélectionnés par un moteur de recherche sont re-indexés afin de n'en retenir qu'un sous-ensemble. Il s'agit ensuite de repérer les éléments correspondant au type de réponse attendu.

Les systèmes de résumé automatique

Ils visent à produire une version digérée d'un document, en sélectionnant l'information pertinente en fonction des besoins des utilisateurs, en synthétisant et en la générant en langue naturelle. En raison de la difficulté des deux derniers processus, la plupart des systèmes se focalisent sur le problème d'extraction du contenu ainsi que le problème de cohérence globale.
Le système Pertinence Summarizer [Lehmam, 2005] en est un exemple et permet de s'interfacer avec des moteurs de recherche en ligne.

Les systèmes d'aide à la navigation

Ils se situent à mi-chemin entre la recherche d'information et les systèmes Question/réponse [Nazarenko, 2005]. Ils ont pour but d'aider l'utilisateur à trouver une réponse par lui même, ou bien à simplement découvrir la base documentaire. Le système peut s'amorcer par une requête de l'utilisateur, il présente alors les documents résultats comme base pour débuter la navigation. Un exemple d'aide à la navigation à l'intérieur d'un même document est InDoc [Ait-El-Mekki, 2004] qui identifie les descripteurs d'un document et segmente le document relativement à ces descripteurs pour identifier les passages qui figureront dans l'index du document.
Les systèmes d'accès au contenu partagent les mêmes problèmes d'identification et d'extraction d'unités pertinentes dans les textes et ont souvent recours à des outils de traitement automatique de la langue naturelle (TAL).

La RI a un long historique et on peut considérer que c'est un domaine de recherche qui est arrivé à maturité. Il reste bien sûr beaucoup de questions encore sans réponse :

- Comment recentrer la RI autour des utilisateurs et des usages ? Nous avons présenté dans la section 5.8, la difficulté d'avoir des jugements objectifs pour évaluer les performances des Systèmes de RI. Les recherches sur le "modèle cognitif" de l'utilisateur peuvent permettre la mise en place de systèmes flexibles et évolutifs en fonction des utilisateurs.
- Comment capturer la sémantique des **textes** ? Nous avons montré la difficulté de mettre en place des traitements assez fins pour traiter la sémantique des textes, cela est néanmoins nécessaire pour pallier les problèmes de silence et de bruit. Nous présentons les méthodes de RI qui proposent une recherche "sémantique" de documents, dans le chapitre 3. Ces méthodes se trouvent encouragés par l'émergence du Web Sémantique (cf. chapitre 4).
- Comment prendre en considération la structure logique des documents pour pouvoir retourner des parties du documents plus précises à une requête ? L'avènement des documents XML (documents qui présentent une structure interne explicite) a permis de reconsidérer les limites du document (il n'est de ce fait plus une unité atomique) et pose de nouveaux défis à la RI. il devient nécessaire de concevoir des modèles qui traitent ce type de document. La RI semi-structurée est une thématique de recherche assez récente, la première campagne d'évaluation INEX (*Initiative for the Evaluation of XML Retrieval*) a eu lieu en 2002. Nous présentons dans le prochain chapitre les nouvelles problématiques soulevées et les différents travaux émergents.

Chapitre 2

Systèmes de Recherche d'Information Semi-structurée (SRIS)

Sommaire

2.1 Introduction

Nous avons vu dans le chapitre précédent que la RI a un long historique et on peut considérer que c'est un champ de recherche bien stabilisé. Les enjeux actuels sont la prise en compte de la sémantique des textes pour pallier les problèmes de silence et de bruit ainsi que la prise en compte de la structure logique des documents pour avoir un accès plus ciblé à l'information et dépasser les limites des documents.
Des travaux comme ceux de Moffat *et al.* [1993] ; Salton *et al.* [1996] ; Hearst [1997] se sont intéressés à la segmentation des documents pour détecter les passages homogènes dans un document. Le but du découpage des documents en segments thématiques est d'apporter une réponse plus précise en proposant comme réponse un ensemble de segments [Moffat *et al.*, 1993]. Le segment est considéré comme l'unité pertinente à retourner à l'utilisateur. Sa détermination repose sur l'identification d'unités d'information élémentaires. Hearst [1997] considère que l'unité élémentaire est la pseudo-phrase (fenêtre de mots de taille fixe) et arrive ainsi à détecter les variations de thématique par un calcul de similarité entre phrases. Salton *et al.* [1996] considèrent que l'unité de base est le paragraphe et s'intéressent comme Hearst à regrouper ces unités en des groupes homogènes. Ces travaux mettent l'accent sur la nécessité de dépasser les limites d'un document et de prendre en considération sa structuration thématique.
D'autre part, des travaux concernant la RI sur le Web essaient d'extraire des informations plus riches que celles véhiculées par les mots, en prenant en considération la structure interne des documents (les balises HTML ainsi que les méta-données). Même si l'information contenue dans les balises peuvent s'avérer intéressantes (on peut supposer qu'un texte balisé en gras peut être considéré comme important), ces informations restent liées aux aspects visuels. Géry [1997] étudie la sémantique des liens hypertextes et propose un typage des liens intra et inter-documents. L'auteur parle de recherche d'information structurée par les hyperliens. Les liens peuvent être de composition ou de cheminement.
Le besoin de structure dans les documents trouve une réponse avec le langage XML (*eXtensible Markup Language*) qui permet une parfaite séparation entre la structure, le contenu et les styles d'affichage. XML est de plus en plus reconnu comme un format standard de documents et l'on peut penser que dans un futur proche, un nombre important de documents et de données seront disponibles dans ce format.
La prise en compte de la structuration explicite des documents XML est depuis peu considérée. Deux communautés, RI et BD (Base de Données), se sont intéressées à l'élaboration de moteurs de recherche pour les documents XML en essayant d'y intégrer leurs domaines d'expertises respectifs.
Il est intéressant de noter que le type de documents XML est différent suivant la communauté (BD ou RI). En effet, les langages de requête issus de la communauté BD sont plus adaptés à une utilisation des documents XML pour un échange de données dans une forme structurée, comme les EDI [24] classiques. Les documents sont **orientés «données»** et par ce fait présentent un contenu textuel assez faible. Le traitement sur le texte ne dépasse généralement pas le simple

[24] *Electronic Data Interchange* : Échange électronique de données

appariement de mots clés. Les documents XML sont qualifiés de semi-structurés parce que la structure est flexible et peut varier à la différence des structures classiques dans les modèles de BD. Les langages issus de la communauté RI s'intéressent plus au contenu textuel, les documents sont donc **orientés «texte»**. Les documents XML sont généralement qualifiés de structurés par rapport aux documents contenant seulement du texte, aussi appelés plats.
Dans le présent document, nous considérons les documents XML comme semi-structurés parce que (i) d'une part la structure est flexible (ii) et d'autre part ils combinent informations sur la structure avec des données textuelles non structurées.

Nous présentons dans ce chapitre quelques systèmes qui se sont intéressés à la recherche d'information dans les documents XML[25] en mettant l'accent sur ceux qui étendent le modèle vectoriel et sont de ce fait proches de nos travaux. Nous les classons suivant qu'ils sont inspirés de modèles existants ou spécialement conçus pour XML (Natif XML). Nous présentons, également, la campagne d'évaluation INEX (*Initiative for the Evaluation of XML Retrieval*) [INEX, 2002] ; [INEX, 2003] ; [INEX, 2004]. INEX est à ce jour la seule campagne d'évaluation des différents SRI sur les documents XML. Nous présentons les différentes tâches de recherche proposées (orientées contenu, orientées structure et contenu) dans le cadre de cette campagne. Nous présentons aussi les évaluations actuelles dans INEX ainsi que leurs limites. Nous commençons par présenter le langage XML ainsi que quelques technologies de la famille des spécifications XML.

2.2 XML et sa famille de spécifications

Le langage XML partage avec le langage HTML un ancêtre commun : le langage SGML (*Standard Generalized Markup Language*). La complexité de la mise en oeuvre de ce dernier en avait limité l'usage à de très grands systèmes documentaires. En 1996 une équipe de Sun Microsystems décide de créer une version de SGML plus simple et mieux adaptée aux besoins d'échange de données sur le Web. Le langage XML est né[26].
Le document XML est structuré en éléments à l'aide de balises[27] qui marquent le début et la fin de chaque élément. Les éléments peuvent contenir du texte et éventuellement d'autres éléments (ils ont une structure arborescente). L'ensemble des données du document XML est contenu dans un élément unique appelé racine, élément qui contient tous les autres éléments. Les éléments peuvent avoir des attributs (définis par leur nom et la donnée associée) qui doivent être uniques pour un même élément (la syntaxe XML n'autorise pas d'avoir des attributs de même nom). Les documents XML doivent être **bien formés**, l'enchevêtrement des éléments doit être correct (<a> est interdit). L'enchaînement hiérarchique des éléments constitue un chemin. Plusieurs relations sont possibles :

[25] Les lecteurs intéressés par un état de l'art complet peuvent se référer à [Sauvagnat & Boughanem, 2004a] [Weigel, 2002].

[26] L'utilisation du standard Unicode, un système d'encodage des caractères qui permet de mélanger des textes dans la plupart des alphabets a largement facilité sa propagation.

[27] Une balise est une chaîne de caractères encadrés par "<" et ">".

– *Père* : un élément e est père d'un élément e' s'il est placé directement au dessus dans la hiérarchie.
– *Ancêtre* : un élément e est ancêtre d'un élément e' s'il est placé au dessus dans la hiérarchie.
– *Fils* : un élément e est fils d'un élément e' s'il est placé directement en dessous dans la hiérarchie.
– *Descendant* : un élément e est descendant d'un élément e' s'il est placé en dessous dans la hiérarchie.
– *Frère* : un élément e est frère d'un élément e' s'il est placé au même niveau dans la hiérarchie et qu'ils ont le même père.

Un élément est défini par une étiquette et le chemin qui y mène de la racine. Voici un exemple de document XML structuré où la racine est l'élément <article> ayant comme attribut $anne$ et comme fils <titre>, <auteur>, etc.

```
<article année="1995">
    <titre>Recherche d'Information</titre>
    <auteur>Gerard Salton </auteur>
    <section>
       <titre>Modèle vectoriel</titre>
       <paragraphe>La recherche d'information ..</paragraphe>
       <paragraphe>Les limites ..</paragraphe>
    </section>
</article>
```

XML fait partie des normes fondamentales pour la réalisation du Web Sémantique. Avec RDF et URI (cf. chapitre 4), ils constituent les briques de base de l'architecture du Web de l'avenir tel que défini par le W3C (*World Wide Web Consortium*). Nous présentons ces derniers dans le chapitre 4, dans ce qui suit nous présentons brièvement les technologies les plus importantes qui sont apparues autour des spécifications XML (voir figure 2.1)

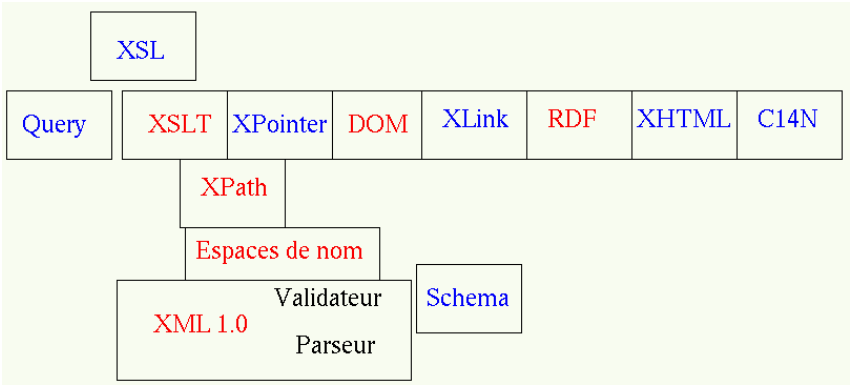

FIG. 2.1 – Famille de spécifications XML (Veillard [2000])

Les espaces de nom

Les espaces de nom (*namespaces*) [Consortium, 1999] permettent de disposer, dans un document XML, de balises provenant de différentes sources. Il se peut que deux balises ayant le

même nom aient des significations différentes. Les espaces de nom résolvent ce problème en qualifiant de manière unique un objet (élément ou attribut), caractérisé par un nom et un domaine de définition (URI)[28] pour lequel il a telle signification. On préfixera alors l'objet de l'espace de nom correspondant. Voici un exemple d'utilisation d'espace de nom

```
<section xmlns s="urn:com:books-r-us">
   <s:title>titre de la section</title>
</section>
```

qui utilise l'élément $title$ tel qu'il est défini dans l'espace de nom auquel fait référence s.

DTD et XML Schema

Les DTD et XML Schema ont pour but de formaliser l'ordonnancement d'un document XML par la description de règles de structuration qui précisent ce qui doit être utilisé ou non au sein dudit document. Si un document XML est bien formé et qu'il est conforme à la DTD ou au Schéma auquel il est associé, ce document est qualifié de **valide**. Une DTD ne décrit cependant que la structure du document (hiérarchie des champs, paramètres, type des données, etc.) et non, par exemple, les valeurs autorisées des champs ou paramètres. Contrairement aux DTDs, les schémas XML sont des documents XML, leur principale caractéristique est qu'ils présentent de plus grandes possibilités de structuration. Ils permettent de spécifier des contraintes supplémentaires sur les données en associant au document XML un ensemble de types de données (*datatypes*) plus complet. XML Schema permet aussi l'utilisation des espaces de noms : on sait alors à quel domaine de définition se rapporte un objet et comment il doit être interprété, selon sa spécification. Voici un exemple de XML Schema qui spécifie que l'élément personne est composé de nom, prénom et date de naissance comme définis dans l'espace de noms référencé par xs.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="<http://www.w3.org/2001/XMLSchema>">
  <xs:element name="personne">
     <xs:complexType>
        <xs:sequence>
           <xs:element name="nom" type="xs:string"/>
           <xs:element name="prenom" type="xs:string"/>
           <xs:element name="date_naissance" type="xs:date"/>
        </xs:sequence>
     </xs:complexType>
  </xs:element>
</xs:schema>
```

SAX (*Simple API for Xml*) et DOM (*Document Object Model*)

Il existe deux types d'analyseurs (*parser*) pour analyser syntaxiquement les documents XML : SAX (*Simple API for Xml*) et DOM (*Document Object Model*). Ce sont des APIs[29] qui per-

[28] Une URI (*Uniform Resource Identifier*), permet d'identifier une ressource de manière unique.
[29] API : *Applications Programming Interface*

mettent d'analyser des documents XML et de naviguer dans l'arborescence pour avoir accès au contenu et attribut des éléments. SAX parcourt linéairement les documents et produit des événements à chaque rencontre de balise, attribut ou texte qui sont alors gérés ou non selon l'application. DOM construit une représentation objet de la structure et bien que beaucoup plus lourd en mémoire permet la navigation dans l'arborescence des documents XML.

XPath

XPath est une recommandation du W3C [November, 1999], une proposition de XPath2 est encore en révision (working draft [April, 2005a]). C'est un langage d'expression destiné à la recherche de nœuds (ou sous-arbres) dans un arbre XML. Chaque expression XPath est composée d'un axe, d'un test et d'un ou plusieurs prédicats (on peut le schématiser par $axe :: test[predicat]$). L'axe permet de définir une dimension pour l'adressage, en d'autres termes la direction du parcours (e.g. les axes fils, descendant, parent, ancêtre, etc.). Le test permet de faire une sélection de nœuds parmi l'axe qui vérifient le test (par exemple $descendant :: paragraphe$ utilise l'axe descendant pour ne sélectionner que les noms de balise paragraphe). Le prédicat permet de raffiner plus la localisation (e.g. $descendant :: paragraphe[position() = 1]$ sélectionne le premier paragraphe). Chaque prédicat peut utiliser des fonctions de positionnement, des opérateurs relationnels, etc. Une syntaxe abrégée permet de raccourcir la plupart des expressions, l'exemple précédent s'écrit alors $//paragraphe[1]$.

XSL (eXtensible Stylesheet Language)

XSL est une recommandation du W3C ([October, 2001]). C'est un langage de feuilles de style, il est composé de deux parties, XSLT et XSL. XSLT est un langage de transformation et XSL est un jeu d'instruction de formatage en XML destiné à la présentation (par des modèles de formatage tels que les boites, positionnement, etc. et des propriétés d'affichage). XSL utilise XPath pour localiser les nœuds à substituer. Par exemple *<xsl :for-each select="//paragraphe[1]"> –corps traitement </xsl :for-each>* signifie pour chaque premier paragraphe exécuter corps traitement.

XPointer

XPointer est une recommandation du W3C [March, 2003], il définit une base commune aux langages d'adressage sur des objets contenus dans un document. Il est construit au dessus de XPath et définit de nouvelles fonctionnalités telle que la possibilité de spécifier le document dans lequel réaliser l'adressage, de définir un ensemble de mots sur lesquels doit être faite la relation (sans avoir besoin d'ancre comme pour le HTML), des types de données (un nœud XML, un point de caractère ou de nœud par exemple ou un intervalle de texte ou d'éléments). Exemple $\#xpointer(/chapitre[2]/p[3])$ crée un pointeur vers le troisième paragraphe du deuxième chapitre.

XLink

XLink fournit les constructions hypertextes de XML. Il définit un espace de nom et une liste d'attributs, des liens simples et étendus[30] ainsi que des annotations. L'objectif est de définir les mécanismes nécessaires à la mise en relation d'informations contenues dans des documents. Voici un exemple de lien simple

```
<bibliographie xlink:href="biblio.xml">
La liste des références bibliographiques.
</bibliographie>
```

XML Base et XInclude constituent des fonctionnalités supplémentaires à XLink. XML Base définit les mécanismes de base pour les références depuis un document XML et XInclude définit un mécanisme d'inclusion pour un document XML.

XML Signature

XML Signature vise à garantir l'intégrité d'un document XML et à l'authentifier. Il utilise la syntaxe XML et possède un mécanisme de signature.

La plupart des travaux actuels en RIS s'intéressent à l'aspect structurel des documents et ne prennent pas en compte, à notre connaissance, les liens possibles entre les documents (XLink[31], XPointer), ni les feuilles de styles (XSL). Ils utilisent généralement un parseur XML et développent des outils qui s'approchent de XPath pour avoir accès aux éléments.

L'existence d'une structure explicite dans les documents XML permet aux SRI d'avoir accès à des parties du document. Désormais les documents ne constituent plus une unité atomique mais peuvent être traités à niveau de granularité plus fin (à savoir les fragments délimités par des balises).
Le but d'un SRIS est alors de tirer parti de cette structuration pour retourner à l'utilisateur des *unités d'information* qui répondent à sa requête. Cela pose de nouveaux défis que nous discutons dans la section suivante.

2.3 Les défis de la RIS

La Recherche d'Information Semi-structurée (RIS) est différente en plusieurs aspects de la RI classique :

– L'information est organisée en unités plus fines que le document. Ces unités (les éléments) sont reliées entre elles (au sein d'un même document) à l'inverse des documents qui sont considérés en RI classique comme indépendants les uns des autres.

[30] Un lien étendu associe au lien un nombre arbitraire de ressources qui peuvent être locales ou distantes.

[31] Quelques systèmes prennent en considération les liens XLink, les documents sont alors représentés en graphe, le lecteur intéressé peut se référer à [Schenkel *et al.*, 2005a] pour plus de précisions.

– Chaque élément dans la hiérarchie du document, ainsi que le document lui même, peuvent être retournés comme réponse à une requête.
– Les utilisateurs peuvent spécifier leur besoin en combinant les informations sur le contenu et la structure.
– L'évaluation des réponses des systèmes tient aussi compte de l'aspect structuré des documents, et permet de juger les différents éléments.

Afin de prendre en compte ces aspects, les techniques de RI doivent repenser de nouvelles méthodes d'indexation et de recherche.
En ce qui concerne l'indexation, il s'agit de prendre en compte l'information structurelle dans la représentation du document. Plusieurs questions se posent alors : tous les éléments de structure sont-ils pertinents à indexer ? Quelle est la relation entre les différents éléments de structure : disjoints ou imbriqués ? Comment indexer le texte par rapport à la structure ? Quelle dimension (élément, document, corpus) doit être prise en compte dans la pondération des termes d'indexation ?
En ce qui concerne la recherche, il s'agit de prendre en compte la notion de structure dans la fonction d'appariement. L'information structurelle permet de poser d'autres types de requêtes que la RI classique. Ainsi l'utilisateur peut poser des **requêtes orientées structure et contenu** et spécifier des conditions sur la structure des documents à retourner ainsi que préciser l'élément à retourner. La recherche classique par mots clés reste la plus intuitive pour les utilisateurs et doit de ce fait rester possible. Les utilisateurs peuvent ne pas connaître la structure des documents recherchés, ils posent alors des **requêtes orientées contenu**. Le système doit alors décider de l'unité d'information à retourner.
Nous présentons dans ce qui suit les différents travaux qui apportent une réponse à ces questions. Nous commençons par exposer les différents travaux qui se sont intéressés à l'adaptation du modèle vectoriel à une recherche d'information semi-structurée. Le modèle vectoriel a largement démontré son efficacité dans les différentes campagnes d'évaluation TREC dans le cadre d'une recherche d'information classique (chapitre 1), ces travaux, comme nous, se sont intéressés à tester son efficacité dans le cadre d'un SRIS.

2.4 Adaptation du Modèle vectoriel

Une extension naturelle du modèle vectoriel est de séparer l'information structurelle de l'information sur le contenu. Le premier modèle qui prend en compte la structure d'un document est le ESVM (*Extended Vector Space Model*) [Fox, 1983], il représente différentes classes d'information appelées *identifieurs objectifs* (ou *concept termes* et dénotées par *c-type*) d'un document, comme l'auteur, le titre, les informations bibliographiques, etc. par des vecteurs. Un document est alors représenté par un vecteur étendu. La similarité entre un document D et une requête Q est calculée par une mesure de similarité (équation 2.1) qui est une combinaison linéaire entre les similarités des différents sous-vecteurs.

$$Sim(D,Q) = \alpha Sim(d,q) + \beta Sim(o_D, o_Q) + \delta sim(c_D, c_Q) \tag{2.1}$$

Où α, β, δ sont des coefficients de pondération des similarités de d, o et c. d étant le vecteur des termes, o un vecteur des identifieurs objectifs et c un vecteur de références bibliographiques. Les vecteurs étendus du document et de la requête sont :

$$D = (d_1, \ldots, d_n, o_1, \ldots, o_m, c_1, \ldots, c_k)$$
$$Q = (q_1, \ldots, q_n, o_{q1}, \ldots, o_{qm}, c_{q1}, \ldots, c_{qk})$$

Nous présentons dans ce qui suit quatre systèmes qui étendent le modèle vectoriel pour prendre en compte la structure et le contenu. Les problèmes soulevés sont :
– la détermination des poids des termes par rapport à leur contexte d'appariement (l'élément).
– la détermination d'une mesure de similarité qui ne considère plus le document comme l'unité de base pour le calcul.

2.4.1 JuruXML : Travaux d'IBM

Carmel *et al.* [2002] proposent un système de recherche de documents XML par fragments XML. Un fragment XML est un texte délimité par une structure. Le but du système est de ne pas contraindre l'utilisateur à poser une requête fortement structurée. Le système retrouve les documents qui ont une structure proche. Les requêtes sont fractionnées en fragments XML, ainsi par exemple la requête structurée :
<chapter><title> XML Tutorials </chapter></title>
devient
(XML, chapter/title), (Tutorial, chapter/title)
Les requêtes peuvent aussi être sans structure comme par exemple : (XML Tutorials) qui est traduite par (XML, NULL), (Tutorial, NULL)
Carmel *et al.* ont adapté leur système Juru pour prendre en considération la structure des documents. Une entrée d'index n'est plus un terme mais un couple (t, c) du terme t avec son contexte c d'apparition, le contexte est assimilé à l'élément (avec son chemin). Chaque couple (t, c) constitue une entrée unique dans l'index t#c. L'apport majeur du système réside dans le calcul de similarité entre la requête et le document. Rappelons que, dans le modèle vectoriel la similarité entre un document et une requête peut se calculer par le cosinus (équation 2.2).

$$Cosinus(D, Q) = \frac{\sum_{i=1}^{m} w_Q(t_i) * w_D(t_i)}{||Q|| * ||D||} \tag{2.2}$$

Carmel *et al.* introduisent la notion de ressemblance de contexte *cr*, la similarité entre un document et la requête devient :

$$S(D, Q) = \frac{\sum_{ti,ci \in Q} \sum_{ti,ci \in D} w_Q(t_i, c_i) * w_D(t_i, c_k) * cr(c_i, c_k)}{||Q|| * ||D||} \tag{2.3}$$

Tels que $w_X(t_i)$ est le poids du terme t_i dans X (X peut être D ou Q), $cr(c_i, c_k)$ prend une valeur entre 0 et 1 et vaut 1 quand les deux contextes sont identiques. w_X et $cr(Q, A)$ sont calculés par les formules suivantes :

1.
$$w_X(t_i) = tf_X(t, c) * idf(t, c) \tag{2.4}$$

où

- tf_X calcule le nombre d'occurrences du couple (t,c) dans X.
- $idf(t,c)$ = $log(\frac{|N|}{|N(t,c)|})$ où |N| est le nombre total de documents dans la collection et |N(t,c)| est le nombre de documents contenant le couple (t,c).

2.

$$cr(Q,A) = \alpha LCS(Q,A) + \beta POS(Q,A) - \gamma GAPS(Q,A) - \delta LD(QA) \qquad (2.5)$$

où

- $\alpha, \beta, \gamma, \delta$ sont des valeurs positives à déterminer empiriquement telle que la somme de α et β doit être égale à 1 pour s'assurer que cr vaut 1 dans le cas de contextes identiques. Carmel *et al.* [2002] donnent les valeurs 0.75, 0.25, 0.25, 0.2 respectivement pour $\alpha, \beta, \gamma, \delta$.
- LCS (*Longest Common Subsequence*) calcule la taille de la plus grande sous-séquence (enchaînement d'éléments) commune normalisée par la taille du chemin de la requête. Elle prend des valeurs entre 0 et 1 et a la forme suivante :
  $$\text{LCS(Q,A)}=\frac{lcs(Q,A)}{|Q|}$$
  où lcs(Q,A) calcule la taille de la plus grande sous-séquence.
- POS calcule par le lcs la position moyenne d'un appariement optimal entre Q et A (noté *AverOptimalPosition*). Le LCS calcule la position moyenne actuelle (noté *AP pour Average Position*). Il s'agit ensuite de comparer ces deux valeurs, le but est de favoriser les chemins qui ont les éléments en commun en tête parce qu'ils sont plus discriminants. POS prend aussi la valeur entre 0 et 1, elle est calculée de la manière suivante :
  $$\text{POS(Q,A)}=1-\frac{((AP-AverOptimalPosition))}{(|A|-2*AverOptimalPosition+1))}$$
- GAPS calcule le nombre d'éléments qui interfèrent dans le chemin, le but est de favoriser les chemins qui s'approchent le plus du chemin de la requête. Dans le cas d'un appariement exact (deux chemins identiques), GAP prend 0 comme valeur. Elle est calculée comme suit :
  $$\text{GAPS(Q,A)}=\frac{gaps}{(gaps+lcs(Q,A))}$$
- LD (*Lenght difference*) a pour but de calculer la différence des distances dans les chemins, dans le but de favoriser les chemins qui ont la même taille. Elle a une valeur entre 0 et 1, elle est calculée par :
  $$\text{LD(Q,A)}=\frac{(|A|-lcs(Q,A))}{|A|}$$

Lors de la campagne INEX 2002, Mass *et al.* [2002] remplacent cette fonction par :

$$cr(c_i,c_k) = \begin{cases} \frac{1+|c_i|}{1+|c_k|} & \text{si c}_i \text{ est une sous-séquence de c}_k \\ O & \text{sinon} \end{cases}$$

Où $|c_i|$ est le nombre d'éléments dans la requête et $|c_k|$ est le nombre d'éléments dans le contexte des documents. Mass *et al.* [2002] n'expliquent pas le choix de cette fonction plutôt que la fonction *cr* (équation 2.5) qui est plus riche. Ils ont néanmoins obtenu de bons résultats et ont été classé 4ème avec une précision moyenne de 0.320 pour les requêtes orientées structure et contenu (CAS). Une autre méthode a été testée et a donné de meilleurs résultats (classé 2ème) en remplaçant $cr(c_i,c_k)$ par $w(c_i)$ où $w(c_i)$ est le poids du contexte c_i.

$$w(c_i) = 1 + |c_i| \tag{2.6}$$

La limite de cette proposition est qu'elle retourne le document entier. Pour résoudre ce problème, Mass & Mandelbrot ont calculé des statistiques sur les requêtes posées à INEX 2002 et en ont déduit un ensemble d'éléments qui sont pertinents (par rapport aux jugements des utilisateurs) à retourner et donc à indexer. Ils proposent ainsi six index (un pour chaque élément jugé pertinent e.g *article, section, paragraphe, sous-section*, etc.). Les scores calculés pour chaque index sont normalisés et les éléments sont filtrés avant d'être retournés à l'utilisateur. Le filtrage évite de retourner des éléments avec une information redondante à l'utilisateur. Ainsi, par exemple, si un élément a un score plus grand que ses fils, il est le seul à être retourné.
Même si Mass & Mandelbrot [2003] précisent qu'une telle méthode peut s'appliquer à n'importe quel corpus, elle reste quand même liée au fait de disposer de statistiques sur les requêtes des utilisateurs. Cela est difficile à envisager dans le cadre réel d'un SRIS. De plus la redondance des index nous paraît inutile (l'index article contient toute la base documentaire et reprend donc les mêmes termes indexés dans les autres index) et impose la phase de filtrage lors de la recherche.

2.4.2 Travaux de l'université de Monte Carlo

Azevedo *et al.* [2004] adaptent aussi le modèle vectoriel et proposent une mesure de similarité fondée sur le cosinus (cf. équation 2.7).

$$S(D,Q) = \frac{\sum_{t_i \in Q \cap D} w_Q(t_i) * w_D(t_i,e) * fxml(t_i,e)}{||Q|| * ||D||} \tag{2.7}$$

tels que $w_D(t_i,e)$ attribue un poids au terme t_i dans l'élément e (cf. équation 2.8)et $fxml(t_i,e)$ attribue des poids aux termes en fonction de leur occurrence dans l'arbre XML (cf. équation 2.9).

1. $$w_D(t_i,e) = log(tf(t_i,e)) * log(N * idf(t_i,e)) \tag{2.8}$$

   où
   – $tf(t_i,e)$ calcule le nombre d'occurrences du terme t_i dans e.
   – $idf(t_i,e)$ est l'inverse du nombre d'éléments contenant t_i.
   – N est le nombre total d'éléments dans la collection.
   Chaque élément constitue un index (un vecteur de termes), un document avec deux éléments, par exemple, donnera lieu à trois index. Chaque terme sera donc indexé dans l'index de l'élément où il apparaît ainsi que dans l'index de ses ancêtres. La fonction $fxml$ a pour rôle de régulariser le poids des termes (pour éviter qu'ils augmentent en montant dans la hiérarchie).

2. $$fxml(t_i,e) = fnh(t_i,e) * fstr(t_i,e) * focr(t_i,e) \tag{2.9}$$

   – *fnh (nesting factor)* exprime la pertinence d'un terme en fonction de sa position dans l'arbre XML.

$$fnh(t_i, e) = \frac{1}{1+nl}$$

où nl est la longueur du chemin entre un élément e et un de ses fils contenant t_i.
Ce facteur réduit la contribution des termes en bas de l'arbre.

– *fstr (structure factor)* exprime la relation entre la structure de la requête et celle du chemin d'un élément du document.

$$fstr(t_i, e) = \frac{(common\_markups+1)}{(nr\_qmarkups+1)}$$

où *common_markups* est le nombre d'éléments communs entre la requête et l'élément qui contient le terme t_i et *nr_qmarkups* est le nombre d'éléments de la requête (pour un même chemin).
Ce facteur valorise les éléments ayant un chemin qui se rapproche de celui de la requête.

– *focr (co-occurrence factor)* exprime la relation entre l'élément et son contenu.

$$focr(t_i, e) = cf(t_i, e) * idf(t_i, e) * N * icf(e)$$

où $cf(t_i, e)$ est le nombre d'éléments qui contiennent le terme t_i. $Idf(t_i, e)$ est l'inverse du nombre d'éléments e qui contiennent le terme t_i. $icf(e)$ est l'inverse du nombre d'occurrences de l'élément dans le corpus. N est le nombre total de documents de la collection.
Ce facteur prend en considération la répartition des éléments dans le corpus ainsi que le nombre de co-occurrences du couple (terme, élément).

Le système permet d'indexer des corpus de schémas différents. Les auteurs affirment qu'ils peuvent dés lors traiter les corpus hétérogènes (cf. section 2.8.2). Néanmoins, ils ne précisent pas comment leur système gère ces corpus pour répondre aux requêtes. Car même si le système indexe, par exemple, la balise *auteur* et *author*, il les considère comme deux éléments différents et ne pourra pas retourner les éléments auteur quand la requête porte sur author par exemple.

2.4.3 Travaux de l'université du Minnesota

Les travaux de Crouch *et al.* [2002] ; Crouch *et al.* [2003] ; Crouch *et al.* [2004] reposent sur le modèle vectoriel étendu de Fox (cf. section 2.4) et reprennent la notion de *c-type* (il y a 18 classes d'information qui correspondent aux différents éléments dans INEX), cependant ils font la différenciation entre identifieurs objectifs et identifieurs subjectifs.

– Identifieurs subjectifs : ce sont les éléments qui contiennent un texte qui requiert un appariement partiel. Les auteurs stipulent que le contenu de ces éléments ne peut pas être comparé objectivement. Ils donnent l'exemple des éléments body (*bdy*), abstract (*abs*), etc.
– Identifieurs objectifs : ce sont les éléments qui contiennent un texte qui requiert un appariement exact et sont de ce fait caractérisés comme objectifs, comme par exemple les éléments auteur (*au*), année de publication (*pub_yr*).

Les identifieurs objectifs servent de filtrage pour le retour des résultats, si on cherche par exemple à retrouver les articles qui contiennent "recherche par le contenu" et qui ont été publiés en 1999. Le système commence par retrouver tous les articles qui contiennent "recherche par le contenu" et ne présentent que ceux qui sont publiés en 1999.

Dans une application directe du EVSM, les auteurs fixent empiriquement les coefficients des sous-vecteurs d'identifieurs subjectifs comme suit (bdy,2), (bibl_atl,2) , (abs,2) ,(kwd, 1), (atl,

1), (bibl_ti,0), (ed_intro, 0), (ack, 0) [32][Crouch *et al.*, 2003]. La limite de cette méthode est qu'elle retourne seulement les documents. Pour pouvoir retourner des éléments, les auteurs introduisent une mesure de propagation des poids des éléments suivant une approche ascendante. Ils commencent par lancer la requête sur les éléments de base (paragraphe par exemple) à partir des éléments retournés, ils construisent un arbre par document avec comme feuille les éléments retournés. Ils assignent deux valeurs à chaque nœud retourné : (i) une valeur représentant l'exhaustivité des éléments noté *e-value*, et (ii) une valeur représentant la spécificité d'un nœud *s-value*.

L'exhaustivité tient compte de la présence ou de l'absence de l'information recherchée et la spécificité s'intéresse au degré avec lequel l'élément traite toute l'information recherchée. Ils s'inspirent ici des directives concernant les jugements de pertinence dans INEX (voir section 2.7). La valeur d'exhaustivité est celle retournée directement par le ESVM, la valeur de spécificité est plus difficile à retrouver mais Crouch *et al.* lui donnent la même valeur que celle de l'exhaustivité pour les nœuds feuilles. Les valeurs sont propagées aux parents de la manière suivante :

– Pour les e-values : *e-value(père)*=$\sum e - value(fils)$. Puisque l'exhaustivité de l'élément père doit avoir une valeur maximale par rapport aux éléments fils.
– Pour les s-values : *e-value(père)*=$moyenne(s - value(fils))$. Puisque la spécificité de l'élément père peut diminuer en fonction du nombre de fils.

Le score final d'un nœud est le produit de e-value et s-value (voir figure 2.2).

score(e) = e-value * s-value

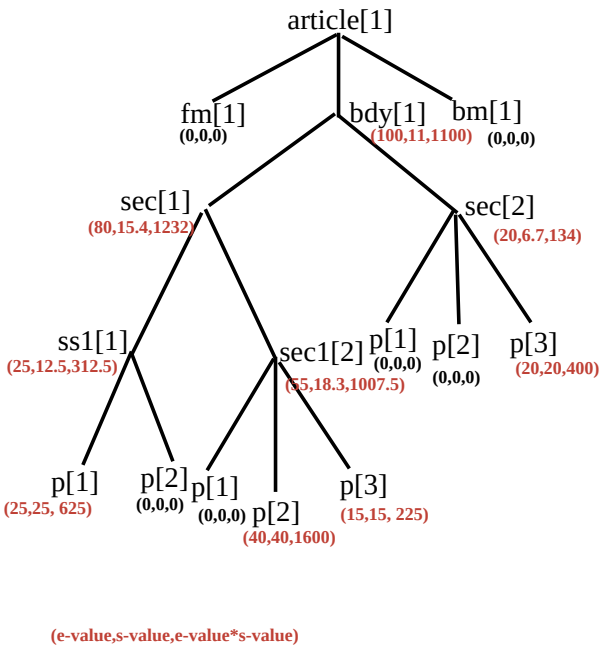

FIG. 2.2 – Propagation de pertinence dans l'arbre XML [Mahajan, 2004]

[32]Ces éléments correspondent à bdy : body, bibl_atl :bibliography article title, abs : abstract, kwd : keywords, atl : article title, bibl_ti : bibliography, ed_intro : editor introduction, ack : acknowledgements

Le choix arbitraires des c-types ainsi que la distinction entre identifieurs objectifs et identifieurs subjectifs ne nous paraît pas triviale. Elle est arbitraire et essentiellement fondée sur le corpus d'INEX. Ce qui rend cette approche difficilement généralisable.

2.4.4 Travaux de l'université de Munich

Schlieder & Meuss [2002] introduisent la notion de *s-terme*. Un s-terme (*structured term*) est un arbre étiqueté dont les nœuds peuvent être des éléments, des attributs ou des termes. Les éléments au même titre que les termes sont considérés comme s-termes (voir figure 2.3). Le corpus des documents est aussi représenté par un arbre étiqueté, chaque sous-arbre est considéré comme *document logique* (par opposition à document physique) et peut être retourné comme réponse. Des types sont associés aux arbres, ils correspondent à l'étiquette de la racine. Ainsi un document logique et une requête sont de même type s'ils ont la même racine.

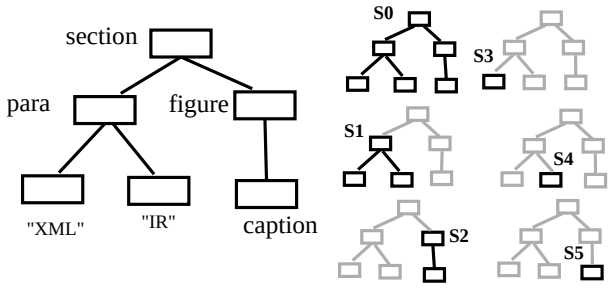

FIG. 2.3 – Arbre de requête constitué de six s-termes [Weigel *et al.*, 2004b]

Répondre à une requête revient donc à placer l'arbre de la requête dans l'arbre du corpus. Les auteurs adaptent la méthode de Kilpeläinen [1992] pour l'inclusion d'arbre et définissent une fonction d'appariement. Une requête a un appariement partiel si au moins un sous-arbre de la requête apparaît dans le document logique. Le calcul du poids d'un s-terme s'inspire aussi du $tf - idf$ et est calculé de la manière suivante :

$$w^t_{s,d} = tf_{s,d} * idf^t_s \tag{2.10}$$

Où

- $\text{tf}_{s,d}$ indique la fréquence d'un s-terme s dans un document d, elle est égale à $\frac{freq_{s,d}}{maxfreq_d}$. La fréquence du s-terme est ainsi normalisée par la fréquence maximale de n'importe quel s-terme dans le document.
- idf^t_s est calculé par $log\frac{|D^t|}{df_s} + 1$. Où $|D^t|$ est le nombre de documents de type t, ce calcul est donc fait pour les documents d'un type t (toutes les sections par exemple).

La similarité entre une requête et un document est le produit scalaire des vecteurs. L'originalité de cette approche est dans le calcul dynamique des poids des termes. Ce poids est calculé lors de la phase de recherche, parce que la notion d'arbre logique est définie en fonction de la structure de la requête. Des techniques avancées (CADG : *Content Aware Data-Guide* [Weigel *et al.*, 2004a]) permettent une navigation intelligente dans les nœuds et évite de devoir parcourir toute la hiérarchie [Weigel *et al.*, 2004b] ; [Weigel *et al.*, 2004a].

Ce type d'approche ne fonctionne pas bien pour les requêtes sur le contenu, comme c'est la requête qui définit la structure, les éléments retournés pour une requête non structurée seront tous de type article.

2.4.5 Discussion

Les travaux présentés montrent l'intérêt d'utiliser le modèle vectoriel. Néanmoins, notre revue de ces systèmes met à jour les problématiques suivantes :
– La définition de l'unité élémentaire à indexer.
– La définition des poids des termes en fonction de leur contexte d'apparition.
– La définition de fonction de propagation de poids dans l'arbre XML.
– La définition d'une mesure de concordance entre les requêtes et les unités indexés.
– L'agrégation de la mesure de concordance au niveau de l'arbre XML.
Dans notre travail, nous posons toutes ses problématiques et proposons des mesures pour prendre en compte la structure et le contenu.

2.5 Adaptation des autres modèles classiques

D'autres modèles de RI classiques ont été adapté pour prendre en considération l'aspect structurel. Nous présentons dans ce qui suit le modèle booléen pondéré et le modèle probabiliste.

2.5.1 Modèle booléen pondéré

Dans [Hayashi *et al.*, 2000], les auteurs présentent un système qui indexe le contenu textuel par rapport à son élément d'apparition. Les éléments sont dénotés par un nom et un chemin de la forme *X/Y* si X est un fils de Y ou *X//Y* si X est un ancêtre de Y. La spécification du chemin n'est pas aussi stricte que XPath. Si le document contient les éléments *X/Y[1]* et *X/Y[2]*, leur contenu textuel sera indexé en dessous de la même structure *X/Y*. La pondération des termes d'indexation est calculée par une variante du $TF - IDF$. Le système indexe les documents indépendamment de leur DTD. Les éléments indexables (qui serviront à la recherche) sont spécifiés a priori.
Le moteur de recherche analyse la requête et la représente en arbre tels que les nœuds non feuilles représentent les opérateurs booléens. La requête est évaluée récursivement en parcourant l'arbre. Le moteur de recherche commence par calculer les scores de pertinence des nœuds feuilles, il parcourt ensuite l'arbre s'il rencontre un nœud OR, le score résultat est la $somme$, si le nœud est un AND, le score résultat est le $minimum$.
Le système permet ainsi de :
– Prendre en considération les opérateurs booléens.
– Associer des poids aux termes de la requêtes et des document.
– Faire un appariement partiel.
La requête suivante par exemple :
(UNIX 0.8 and title=(network 0.3 or TCP/IP 0.8) and topic_area=(IT or "computer systems"))

cherche à retrouver les documents qui contiennent le terme "UNIX" (avec un poids de 0.8) n'importe où dans le document et le terme "network" (avec un poids de 0.3) ou "TCP/IP" (avec un poids de 0.8) dans l'élément "title" et les termes "IT" ou "computer systems" dans l'élément "topic_area".

2.5.2 Modèle probabiliste

Nous présentons dans cette section un modèle probabiliste fondé sur les réseaux bayésiens [Piwowarski *et al.*, 2002]. Un réseau bayésien est associé à un document XML. Chaque nœud du réseau bayésien du document est une variable booléenne qui indique si l'information associée au nœud est pertinente ou non par rapport à la requête. La structure du réseau est directement dérivée de la structure du document. Différents réseaux peuvent représenter le même document. Les figure 2.4b et c montrent deux modèles différents du document de la figure 2.4a [Piwowarski & Gallinari, 2003].

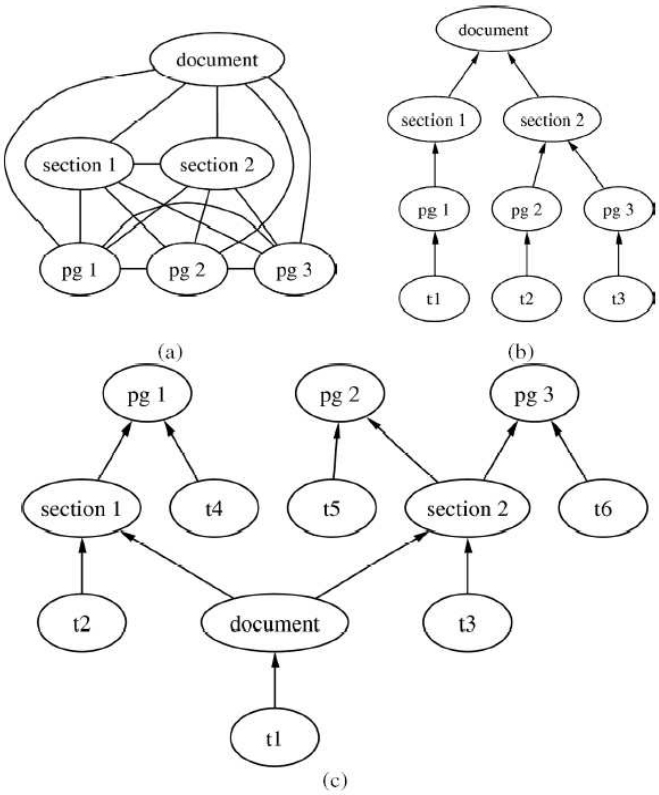

FIG. 2.4 – Trois différents modèles du même document. (a) Toutes les parties sont dépendantes, (b) et (c) deux différents modèles avec des indépendances conditionnelles dans le réseau. [Piwowarski & Gallinari, 2003]

Deux variables sont donc prises en compte, celle qui est associée à la structure et celle associée aux textes. Les deux variables sont binaires et prennent en compte l'ensemble {R = documents pertinents pour la requête, ¬R = documents non pertinent pour la requête}. La première est calculée par des inférences du réseau bayésien et la deuxième peut être calculée par n'importe

quel modèle probabiliste (Piwowarski *et al.* utilisent OKAPI [Robertson & Jones, 1997]). Le réseau bayesien propage les probabilités de pertinence d'un nœud à ses descendants.
Soit T la variable associée au contenu textuel, les probabilités sont calculés par le modèle OKAPI par :

$$P(T) = cosinus(T, q) \tag{2.11}$$

OKAPI donne des scores de 0 à 1 qui sont utilisés comme probabilités. Pour les variables associées aux parties structurelles, des probabilités conditionnelles sont calculées par

$$P(A\ pertinence|B_1, ..., B_n\ pertinence) \tag{2.12}$$

Où B_i sont les ancêtres de A dans le réseau bayésien.
Le modèle marche en deux temps : apprentissage supervisé et recherche. La phase d'apprentissage est nécessaire pour paramétrer le réseau en fonction du corpus. Elle se fonde sur les jugements de pertinence des utilisateurs pour trouver les paramètres optimaux par rapport au corpus d'apprentissage. La phase de recherche peut avoir lieu grâce aux inférences du réseau bayésien.

2.6 Les langages de requête

De nouveaux langages de requête ont été créés spécifiquement aux documents XML. Beaucoup se fondent sur la norme XPath[November, 1999]. Une schématisation de quelques langages et leurs interactions est donné dans la figure 2.5. Nous présentons deux de ces langages : XQuery et XIRQL.

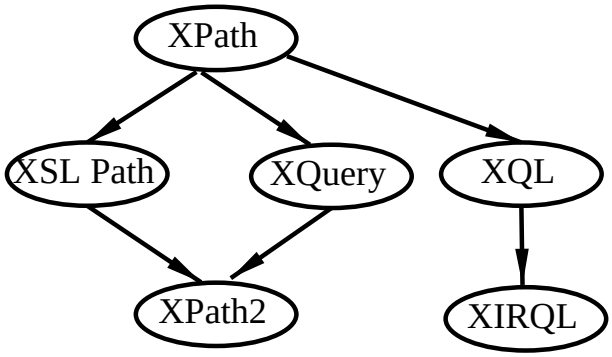

FIG. 2.5 – Les langages de requêtes. Les liens représentent des relations d'inclusion (quelquefois partielles) des différents langages de requête. [Piwowarski, 2003a])

2.6.1 XQuery

XQuery (*XML Query*) est en cours de finalisation auprès du W3C [April, 2005b]. Il permet des expressions de chemin fondées sur XPath, des expressions FLWR (*For, Let, Where, Return*), des expressions conditionnelles (*IF/THEN/ELSE*), des expressions quantifiées (*SOME, EVERY*),

des opérateurs classiques (arithmétiques, de comparaison, booléens) ainsi que l'utilisation de variables et constantes.

L'expression FLWR compose le squelette de l'expression XQuery, elle est similaire à la construction SELECT-FROM-WHERE de SQL. Une expression FLWR est constitué des quatres clauses suivantes :

- FOR : attache une ou plusieurs variables à une séquence de valeurs renvoyées par une autre expression (généralement une expression de chemin), et boucle dans les valeurs.
- LET : attache aussi une ou plusieurs variables à une séquence, mais sans itération.
- WHERE : contient un ou plusieurs prédicats qui filtrent ou limitent le jeu de nœuds générés par les clauses FOR/LET.
- RETURN : génère le résultat de l'expression FLWR. Elle contient généralement un ou plusieurs éléments constructeurs et/ou des références à des variables et est exécutée une fois pour chaque nœud renvoyé par les clauses FOR/LET/WHERE.

Voici un exemple de requête [April, 2005b] en XQuery qui retourne une liste de noms d'auteurs où chaque nom apparaît une fois, suivi par une liste ordonné de ses livres. Il permet donc de créer de nouveaux éléments comme résultat d'une requête.

```
<authlist>
 {
   for $a in ($bib/book/author)
   order by $a
   return
     <author>
        <name> {$a} </name>
        <books>
          {
            for $b in $bib/book[author = $a]
            order by $b/title
            return $b/title
          }
        </books>
     </author>
 }
</authlist>
```

XQuery est donc un langage expressif qui permet d'effectuer des requêtes complexes. Il a cependant une vue orientée données des document et reste donc orienté BD et n'offre pas les fonctionnalités d'un SRI, même s'il intègre un prédicat *contains* pour la recherche par mots clés. Il permet seulement des requêtes de type booléen, les documents retournés ont tous un appariement direct avec la requête. De plus, il suppose une connaissance *a priori* de la structure. Le langage XIRQL est fondé sur XQuery et pallie ces limites.

2.6.2 XIRQL

Le langage XIRQL [Fuhr & Großjohann, 2004] est implémenté dans HyREX[33] et mis à la disposition des participants d'INEX pour les aider à parcourir le corpus et définir les requêtes. Il étend la partie XPath utilisée dans XQuery et est basé sur pDatalog, variante de Datalog (p pour probabilité). Il ajoute les caractéristiques suivantes pour mieux intégrer les fonctions RI :

- Pondération des termes : les termes des documents ainsi que des requêtes sont pondérés ce qui permet un ordonnancement des résultats. La pondération utilisé est BM25 [Robertson & Walker, 1994] ; [S.E. Robertson & Payne, 1995] ; [Robertson & Jones, 1997] normalisée pour avoir une valeur entre 0 et 1.
- Recherche orientée contenu : le système permet de poser une requête en langage naturel et permet de retourner les éléments pertinents. Ils utilisent une technique de propagation des poids des termes. Les nœuds sont considérés comme des unités disjointes. Les poids sont propagés par multiplication par un *facteur d'augmentation* donné.
- Types de données et prédicats vagues : le système permet une approximation de la requête suivant le type de données (une approximation phonétique pour les noms, une approximation géographique pour les lieux, etc.). Ils ont recours à des ressources externes qui spécifient les types de données traités.
- Structure vague : le système ne fait pas de différenciation entre un élément et un attribut, de plus le système permet un traitement vague de la structure en fournissant des opérateurs de similarité entre éléments (l'opérateur ~).

2.7 Evaluation des SRIS

Nous avons vu dans le chapitre 1 que les jugements de pertinence d'un document sont généralement binaire (pertinent ou non pertinent). Dans le cadre de RIS, l'unité d'information n'est plus le document mais l'élément d'information recherché. L'utilisation de pertinence au sein d'un document devient alors problématique : si une section d'un document contient un paragraphe pertinent, quelle valeur de pertinence donner à la section ? Cette section ne peut pas être considéré comme totalement pertinente puisqu'elle contient d'autres paragraphes qui ne sont pas pertinents, elle ne peut pas non plus être considérée comme non pertinente car elle contient quand même de l'information pertinente [Piwowarski & Lalmas, 2004]. Il faudra trouver un moyen pour juger de la pertinence d'une telle section tout en la distinguant de sections qui peuvent être moyennement pertinente dans l'ensemble sans contenir de paragraphes particulièrement pertinents.

2.7.1 Campagne d'évaluation INEX

La campagne d'évaluation INEX [INEX, 2002][INEX, 2003][INEX, 2004] tend à promouvoir l'évaluation de la recherche sur les documents XML en prenant en compte ces différents aspects. Elle fournit une collection de test de documents XML ainsi que des procédures d'évaluation. Les requêtes ainsi que les jugements de pertinence sont fournis par les participants.

[33] http ://ls6-www.informatik.uni-dortmund.de/ir/projects/hyrex/

Les documents constituant la collection de test sont des articles scientifiques provenant de la IEEE Computer Society, balisés en XML. Les articles sont généralement composés d'une entête (<*fm*>), d'un corps (*body*) et d'annexes (<*bm*>). Ces éléments peuvent être composés d'autres éléments, *section* est décomposé de <*sec*>, <*ss1*> <*ss2*>,) etc. par exemple. Les requêtes (*Topics*) doivent comporter (i) un titre qui constitue la requête elle-même, elle peut être en langage naturel ou structurée dans un langage particulier[34], (ii) une partie descriptive qui définit le besoin d'information décrit dans la partie titre, (iii) une partie narrative qui détaille et explicite la requête et spécifie ce qui caractérise une réponse qui pourra être jugée pertinente et (iv) une partie mots clés qui regroupe une liste de mots clés supplémentaire (des termes proches ou synonymes de la requête).
Les requêtes dans INEX peuvent être de deux types : soit sur le contenu uniquement (*CO : Content-only topics*), soit sur le contenu et la structure (*CAS : Content-and-structure topics*).
– Les requêtes *CO* : Correspondent aux requêtes classiques en RI. L'utilisateur ne connaît pas la structure (ou ne veut pas la spécifier dans la requête).
– Les requêtes *CAS* : correspondent à des requêtes contenant des contraintes sur la structure et le contenu. L'utilisateur est supposé avoir une certaine connaissance de la structure et veut des réponses spécifiques.

2.7.2 Les jugements de pertinence

Pour permettre l'évaluation des systèmes de RIS, une échelle à deux dimensions a été proposée lors de la campagne INEX 2002 [Gövert & Kazai, 2002]. Les deux dimensions sont la pertinence et la couverture. La pertinence tient compte de la présence ou l'absence de l'information recherchée. Un élément est considéré comme pertinent même si l'information recherchée se trouve dans une petite partie de cet élément (un seul sous-élément par exemple). On distingue quatre niveaux de pertinence :
– Non pertinent (0) (*irrelevant*) : l'élément ne contient aucune information concernant le sujet de la requête.
– Peu pertinent (1) (*marginally relevant*) : l'élément contient une partie marginale du sujet de la requête.
– Assez pertinent (2) (*Fairly relevant*) : l'élément contient une grande partie du sujet de la requête.
– Très pertinent (3) (*Highly relevant*) : l'élément contient la réponse à la requête.
La couverture s'intéresse au degré avec lequel l'élément traite toute l'information recherchée, elle est de ce fait spécifique à l'évaluation des documents semi-structurés. On distingue également quatre niveaux :
– Pas de couverture (N) (*no coverage*) : L'élément retourné ne contient pas de passages pertinents.
– Trop large (G) (*too large*) : L'élément retourné contient une proportion importante d'information non pertinente.
– Trop petit (P) (*too small*) : L'élément retourné couvre la majorité du sujet de la requête mais il est trop petit pour être considéré comme une unité pertinente.

[34]Les langages de requête sont fixés par les organisateurs, le langage NEXI[O'Keefe & Trotman, 2003] est choisi pour INEX 2004

– Exacte (E) (*exact coverage*) : L'élément retourné constitue exactement la bonne réponse.

Ces jugements ne sont pas tout à fait orthogonaux, lorsqu'un élément n'a pas de couverture il ne peut pas être pertinent (et inversement). Il n'y a donc que 10 valeurs possibles et non 16. Ces valeurs sont représentés par 2 lettres, la première étant la pertinence et la deuxième la couverture. Par exemple, 3E correspond à un élément jugé très pertinent avec une couverture exacte. Dans la campagne INEX 2003, les dimensions de pertinence et de couverture ont été remplacées par les dimensions d'exhaustivité et de spécificité. La terminologie a été modifiée pour éviter les confusions, en effet la pertinence d'un élément est la combinaison de son exhaustivité et spécificité. L'exhaustivité (E) décrit à quel point l'élément discute du sujet de la requête. Une échelle à quatre niveaux est également proposée :

– Non exhaustif (E0) : l'élément ne traite pas du tout le sujet de la requête.
– Faiblement exhaustif (E1) : l'élément traite quelques aspects du sujet de la requête.
– Moyennement exhaustif (E2) : l'élément traite plusieurs aspects du sujet de la requête.
– Totalement exhaustif (E3) : l'élément traite exhaustivement (tout ou la majorité) le sujet de la requête.

La spécificité (S) décrit à quel point l'élément se focalise sur le sujet de la requête. Quatre niveaux sont également distingués :

– Non spécifique (S0) : le sujet de la requête n'est pas un thème de l'élément.
– Faiblement spécifique (S1) : le sujet de la requête est un thème mineur de l'élément (l'élément focalise sur d'autres thèmes non pertinents mais contient quand même des informations pertinentes).
– Moyennement spécifique (S2) : le sujet de la requête est un thème majeur de l'élément (l'élément peut contenir quelques informations non pertinentes).
– Totalement spécifique (S3) : le sujet de la requête est le seul thème de l'élément.

Tout comme les deux dimension de précision et de couverture ces deux dimensions ne sont pas orthogonales et seule la combinaison des 10 valeurs est possible (voir figure 2.6). Un élément jugé par E1S1 contient quelques informations pertinentes (E1) qui constituent un thème mineur de l'élément (S1) (la plupart du contenu est non pertinent par rapport au sujet de la requête).

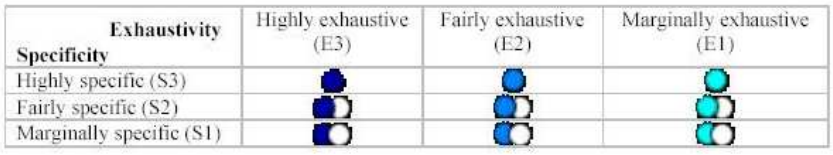

FIG. 2.6 – Les combinaisons possibles de jugements (E0S0 pour les éléments non spécifiques et non exhaustifs n'est pas présentée [Piwowarski & Lalmas, 2004])

2.7.3 Consistance et exhaustivité des jugements

L'aspect structuré des documents XML ne permet pas comme pour la RI classique de juger indépendamment les éléments de structure. En effet, si un élément est pertinent, ses ancêtres sont aussi pertinents (peut être pas au même niveau mais leur pertinence est liée). Le test des consistances des jugements permet d'assurer que les contraintes entre les éléments est respectée.

Des règles d'inférence sur les jugements portés à l'intérieur d'un même document ont donc été définies à partir de INEX 2003[35], ces règles permettent de contrôler la consistance des jugements :

– Si les enfants d'un élément sont non pertinents alors cet élément est non pertinent.

Concernant l'exhaustivité

– L'exhaustivité d'un élément est toujours supérieure ou égale à l'exhaustivité d'un de ses fils. Cette règle est évidente, il est en effet impossible que l'élément englobant soit moins exhaustif que ceux qui le composent.
– Un élément totalement exhaustif doit avoir un enfant totalement exhaustif, ou bien un enfant moyennement exhaustif et un enfant faiblement exhaustif.

Concernant la spécificité

– La spécificité d'un élément est inférieure ou égale au maximum de la spécificité de ses fils. En effet, si un élément est jugé comme hautement spécifique, ses fils ne peuvent pas être que moyennement ou faiblement ou non spécifiques.
– La spécificité d'un élément est supérieure ou égale au minimum des spécificités de ses enfants.

Le nombre important d'éléments (8 millions) rend impossible une évaluation complète de la totalité du corpus INEX. Contrairement aux campagnes TREC, une présélection du corpus à évaluer à partir des résultats des systèmes qui participent à la campagne, n'est pas tout à fait suffisante. On cherche à trouver les éléments totalement spécifiques, il est important alors de juger tous les éléments des documents retournés. Il faut s'assurer que toutes les parties du document ont été jugées. Pour alléger cette phase, une politique de jugement a été proposée, la liste des éléments à juger pour un document est mise à jour en fonction des jugements de l'utilisateur de la manière suivante :

– Si un utilisateur juge un élément non pertinent, aucun élément n'est rajouté à la liste des éléments à évaluer.
– Si un utilisateur juge qu'un élément est totalement spécifique, ses ancêtres sont rajoutés à la liste.
– Si un utilisateur juge qu'un élément n'est pas faiblement ou moyennement spécifique, ses enfants et ses ancêtres sont rajoutés pour forcer l'utilisateur à rechercher un élément hautement spécifique.

L'ajout des éléments à juger alourdit les tâches d'évaluation, un allégement est donc à envisager [Piwowarski & Lalmas, 2004].

[35] Développé dans l'interface d'évaluation Xrai par Piwowarski

2.7.4 Les mesures d'évaluation

La nature des recherches sur les documents XML ne permet pas d'utiliser les mesures d'évaluation classiques comme celles utilisées dans TREC ou CLEF[36]. De nouvelles mesures d'évaluation sont nécessaires. Les mesures proposées dans INEX 2002 [Gövert & Kazai, 2002] sont fondées sur les mesures de rappel/précision.
La mesure $precall$ appliquée aux éléments des documents calcule la probabilité qu'un élément visualisé par l'utilisateur soit pertinent. En supposant que l'utilisateur arrête de visualiser les documents après un certain nombre d'éléments de documents, cette probabilité est calculée par :

$$P(rel|retr) = \frac{x * n}{x * n + esl_{x*n}} \tag{2.13}$$

Où esl_{x*n}(*expected search length*) est le nombre d'éléments non pertinents retournés jusqu'à un point de rappel x, n est le nombre total d'éléments pertinents par rapport à une requête.
Pour pouvoir utiliser les deux dimensions de spécificité et d'exhaustivité, elles sont transformées en une seule valeur de pertinence par une fonction de quantification.

$$\begin{array}{ll} f_{quant} : & \text{exhaustivité} \times \text{spécificité} \rightarrow [0, 1] \\ & (e, s) \rightarrow f_{quant}(e, s) \end{array}$$

Deux différentes fonctions f_{strict} et $f_{generalised}$ ont été adoptées. La fonction f_{strict} permet d'évaluer la capacité des systèmes à retrouver les éléments totalement exhaustifs et spécifiques.

$$F_{strict}(e, s) = \begin{cases} 1 & \text{si (e,s)=E3S3} \\ O & \text{sinon} \end{cases}$$

Cette fonction est très restrictive, seuls les éléments totalement exhaustifs et spécifiques sont pris en compte, alors qu'on peut penser que la notion de pertinence est plus large, est-ce qu'un élément hautement spécifique et moyennement exhaustif est intéressant à retourner ? La fonction $fgen$ ($fgeneralised$) est utilisée pour pouvoir prendre en compte les différents degrés d'exhaustivité et spécificité.

$$F_{gen}(e, s) = \begin{cases} 1 & \text{si (e,s)=E3S3} \\ 0.75 & \text{si (e,s)} \in \text{(E2S3), (E3S2), (E3S1)} \\ 0.5 & \text{si (e,s)} \in \text{(E1S3), (E2S2), (E2S1)} \\ 0.25 & \text{si (e,s)} \in \text{(E1S2), (E1S1)} \\ O & \text{si (e,s)=E0S0} \end{cases}$$

Cette fonction favorise les éléments jugés exhaustifs, elle leur assigne un score élevé même si la spécificité est faible.
D'autres fonctions de quantification ont été proposées, elles combinent aussi les deux dimensions en favorisant soit la spécificité, soit l'exhaustivité. Kazai [2003] propose une fonction généralisée orientée spécificité f_{sog} (*Specificity Oriented Generalised*)

[36]CLEF (*Cross-Language Forum*) est un programme international d'évaluation des techniques et approches de croisement de langues en recherche d'information.

$$F_{sog}(e,s) = \begin{cases} 1 & \text{si (e,s)=E3S3} \\ 0.9 & \text{si (e,s)=E2S3} \\ 0.75 & \text{si (e,s)} \in \text{(E1S3), (E3S2)} \\ 0.5 & \text{si (e,s)=E2S2} \\ 0.25 & \text{si (e,s)} \in \text{(E1S2), (E3S1)} \\ 0.1 & \text{si (e,s)} \in \text{(E2S1), (E1S1)} \\ O & \text{si (e,s)=E0S0} \end{cases}$$

Des fonctions préférant la spécificité ont été proposées pour évaluer les systèmes en fonction de leur capacité à retrouver les éléments les plus spécifiques où l'exhaustivité peut varier de faiblement et moyennement à totalement exhaustif (équation 2.14), ou seulement de moyennement à totalement exhaustif (équation 2.15).

$$Fs3\_e321(e,s) = \begin{cases} 1 & \text{si e} \in \text{E3,E2,E1 et s=S3} \\ 0 & \text{sinon} \end{cases} \tag{2.14}$$

ou

$$Fs3\_e32(e,s) = \begin{cases} 1 & \text{si e} \in \text{E3,E2 et s=S3} \\ 0 & \text{sinon} \end{cases} \tag{2.15}$$

De la même manière les fonctions préférant l'exhaustivité ont été définies. Elles permettent d'évaluer les systèmes en fonction de leur capacité à retrouver les éléments les plus exhaustifs où la spécificité peut varier de faiblement et moyennement à totalement exhaustif (équation 2.16), ou seulement de moyennement à totalement exhaustif (équation 2.17).

$$Fe3\_s321(e,s) = \begin{cases} 1 & \text{si s} \in \text{S3,S2,S1 et e=E3} \\ 0 & \text{sinon} \end{cases} \tag{2.16}$$

$$Fe3\_s32(e,s) = \begin{cases} 1 & \text{si s} \in \text{S3,S2 et e=E3} \\ 0 & \text{sinon} \end{cases} \tag{2.17}$$

En absence d'un modèle d'utilisateur bien défini, toutes ces fonctions sont combinées pour donner une moyenne à la pertinence des systèmes. La principale critique faite à ces mesures est qu'elles favorisent les modèles qui renvoient des éléments emboîtés, une même information pertinente est comptée plusieurs fois. De plus, les règles d'inférence utilisées lors de l'élaboration des jugements propage la pertinence d'un nœud à ces ancêtres, les systèmes qui retournent des éléments imbriqués se trouvent favorisés. Or ce genre d'information peut être considéré comme une perte du temps pour l'utilisateur à consulter des informations redondantes. Les systèmes qui retournent des éléments qui se recouvrent doivent être pénalisés ou au moins non favorisés. D'autres mesures sont en cours pour résoudre ce problème [Piwowarski, 2003b], [Kazai *et al.*, 2005] .

2.7.5 Synthèse

La Recherche d'Information Semi-structurée est un champ de recherche récent. Si nous sommes conscients de l'avantage d'incorporer la structure dans le processus d'indexation et de recherche, beaucoup reste à faire quant à l'étude d'un modèle de l'usager pour arriver à cerner

les attentes des utilisateurs d'un tel système. Les études en cours sur les mesures d'évaluation sont primordiales pour pouvoir juger objectivement de la pertinence des systèmes. D'autres thématiques autour de la RIS sont en train d'être définies. Lors d'INEX 2004, quatres nouvelles tâches se sont ouvertes aux candidats à savoir (i) le TAL pour extraire des requêtes orientées structure et contenu, (ii) la RIS dans des corpus hétérogènes, (iii) la définition de l'interactivité avec l'utilisateur et (iv) le retour sur pertinence dans le cadre de documents XML.
A la section suivante, nous décrivons brièvement ces différentes tâches en présentant succintement les propositions faites lors de ces ateliers.

2.8 Les champs de recherche émergents

2.8.1 Utilisation d'outils de Traitement Automatique de la Langue

Les requêtes orientées structure et contenu (CAS) supposent une connaissance de la structure des documents dans le corpus. De plus l'expression des requêtes dans un langage comme XPath n'est pas aisée, ainsi lors de la campagne INEX 2003 près des deux tiers (63%) des requêtes étaient erronées et ont nécessité pas moins de 12 tours de corrections successives[37]. Le but de la "tâche Traitement Automatique de la Langue"[38] est de regrouper les chercheurs en TAL et ceux de la RIS en vue de tester l'utilisation des outils TAL pour la génération de requêtes orientés structure et contenu à partir de description en langage naturel. L'objectif est de pouvoir extraire des descriptions les informations concernant la structure et celle concernant le contenu. Il faut aussi différencier les éléments à retourner des autres éléments. Par exemple à partir de la description suivante (exemple donné dans [Woodley & Geva, 2004])[39]

```
<description>
Find sections about compression in
articles about information retrieval.
</description>
```

Il faut distinguer les éléments section et article du texte à rechercher (information retrieval et compression) ainsi que retrouver que l'élément à retourner est section. La requête générée en syntaxe de NEXI est la suivante :

```
//article[about](.,information retrieval)]//sec[about(.,compression)]
```

Les participants se sont heurtés à plusieurs obstacles :
– Les descriptions des requêtes CAS ne sont pas toujours fidèles à l'expression de la requête orientée structure et contenu. En théorie la partie descriptive de la requête est censée décrire la requête en langage naturel en donnant éventuellement d'autres précisions.

[37] C'est ce qui a emmené à l'utilisation d'un autre langage NEXI (*Narrow Extended XPath I) pour INEX 2004* [O'Keefe & Trotman, 2003].
[38] NLP Track.
[39] Retrouver les sections qui parlent de compression dans des articles sur la recherche d'Information.

– Les noms d'éléments dans le corpus INEX ne sont pas intuitifs, un appariement exact entre le nom de l'élément et celui en langue naturel n'est pas trivial. Il faut savoir par exemple que l'élément *sec* réfère à section, *p* à paragraphe, etc. Cela suppose de réaliser des outils dédiés au corpus INEX où les règles d'appariement doivent changer si on change de corpus [Tannier *et al.*, 2004].

2.8.2 Corpus hétérogènes

La collection INEX actuelle est fondée sur une seule DTD. En pratique, la réalité de ces hypothèses est autre, la plupart des collections XML proviennent de différentes sources et ont de ce fait des DTDs différentes. Les collections hétérogènes posent de nouveaux défis :
– Pour les requêtes sur le contenu, des méthodes indépendantes des DTDs doivent être développées pour définir les éléments pertinents à retourner à l'utilisateur.
– Pour les requêtes orientées structure et contenu se pose le problème d'appariement d'une DTD à une autre (avec la possibilité d'avoir des collections sans DTD). Les méthodes des bases de données fédérées peuvent être appliquées pour apparier les différentes structures où les différents appariements sont définis manuellement. Il serait intéressant pour de grands corpus hétérogène (où il y a un large nombre de DTDs) de définir des méthodes automatiques (e.g fondés sur des ressources sémantiques).

Le but de cette tâche est d'apporter des réponses à ces questions :
– Pour les requêtes orientées contenu, quelles sont les méthodes qui peuvent être appliquées pour produire des réponses raisonnables ? Est-ce que les méthodes statistiques sont suffisantes, ou des approches fondées sur des ontologies[40] sont plus pertinentes ?
– Quels sont les critères qui peuvent être utilisés pour apparier les structures des différentes DTDs ?
– L'appariement doit-t-il porter seulement sur la structure ou incorporer le texte contenu dans ces éléments ?
– Quelle est la méthode d'évaluation appropriée à des collections hétérogènes ?

Les participants ont tenté d'explorer ces différentes questions ainsi que de pointer les problèmes soulevés. Une collection de corpus hétérogènes a été déterminée et quelques requêtes proposées. Une analyse qualitative des résultats a été réalisée étant donné qu'il est prématuré d'envisager une évaluation quantitative des requêtes, beaucoup de zones d'ombre restent à éclaircir. Les principales contributions reposent sur (i) la définition manuelle d'un dictionnaire d'éléments [Sauvagnat & Boughanem, 2004b] et (ii) l'extraction semi-automatique d'une DTD unifiée à partir des différentes DTDs [Abiteboul *et al.*, 2004].

2.8.3 Interactivité

Le but de la tâche "interactivité" est (i) de cerner le comportement des utilisateurs vis-à-vis des éléments retournés ainsi que (ii) de développer des approches de RIS qui sont en adéquation avec l'environnement des utilisateurs. Les participants se sont focalisés pour la première année

[40] Nous définissons à ce niveau une ontologie comme une ressource sémantique qui représenterait les éléments des différentes DTDs avec leurs relations respectives (des relations hiérarchiques et des relations horizontales). Une définition complète se trouve dans le chapitre suivant.

sur le premier objectif : définir un modèle utilisateur dans le contexte de RIS. Le but étant de valider ou invalider certaines hypothèses sur lesquelles s'appuient des mesures d'évaluation.
Le processus expérimental est défini comme suit [Tombros *et al.*, 2004] :

– La définition des requêtes : un sous ensemble des requêtes CO a été extrait et réparti dans deux catégories : B (*Basic*) qui sont les requêtes qui expriment un besoin de recherche d'information du type "chercher des informations concernant X", C (*Comparative*) qui sont les requêtes du type "chercher les différences entre X et Y"[41]
– La définition d'un SRI unique pour tous les participants : le système HyRex[42] a été fourni par les organisateurs comme système de requêtes. Il retourne le titre, l'auteur et le chemin de l'élément pertinent avec sa valeur de pertinence. L'utilisateur aura à juger les éléments suivant une échelle de 10 (qui correspond aux 10 valeurs possibles des différents jugements (voir section 2.7.2) le terme exhaustivité a cependant été remplacé par *utilité* parce que jugé plus parlant à un utilisateur "naïf".
– La définition d'un questionnaire : un questionnaire a été soumis aux utilisateurs avant et après les expérimentations. Un total de 88 utilisateurs (ayant en moyenne 29 ans) de niveaux universitaire différents [43]ont évalué les réponses du système.

Des résultats préliminaires montrent que les utilisateurs ne sont pas intéressés par les éléments qui se recouvrent (les éléments emboîtés) [Tombros *et al.*, 2004]. Les participants n'ont pas trouvé de corrélation entre la satisfaction des utilisateurs et les différents niveaux de pertinence, ils ont aussi remarqué que la classification des jugements suivant les 10 catégories n'a pas été utile pour évaluer les résultats [Kim & Son, 2004].

2.8.4 Retour sur pertinence pour les documents XML

Les recherches en RI ont montré que même si l'utilisateur peut avoir des difficultés à bien spécifier son besoin avec précision en terme de requête, il peut reconnaître qu'une information est utile quand elle lui est présentée. Partant de ce constat, les outils de retour sur pertinence se sont développés en se fondant sur les jugements de pertinence des utilisateurs (en marquant les documents par pertinent/non pertinent).
Le but de la tâche "retour arrière sur pertinence" (*relevance feedback*) est d'appliquer ce processus dans le cadre des documents XML. Ils nécessitent des techniques plus sophistiquées que celles utilisées en RI (ajout ou retrait) des groupes de mots jugés pertinents (ou non). Les approches doivent aussi prendre la structure en considération. La reformulation de la requête doit alors inférer quels sont les contenus et structures qui sont importantes. Les participants se sont focalisés sur les requêtes sur le contenu. Les premiers résultats obtenus améliorent très peu les précisions [Mass & Mandelbrot, 2004].

2.9 Conclusion

Dans ce chapitre, nous avons présenté dans un premier temps les nouveaux défis posés par les documents XML et les orientations des recherches actuelles. L'utilisation de la structure a mon-

[41] Ce type de requête est plus rare dans INEX, 2 requêtes seulement demandent ce type d'information.
[42] http ://www.is.informatik.uni-duisburg.de/projects/hyrex/
[43] 33% d'étudiants en licence, 49% d'étudiants en maîtrise et 12% de doctorants

tré qu'elle est utile pour améliorer les résultats et avoir une granularité de réponse plus fine que le document. Cependant, la recherche dans les documents XML en est à ses débuts, beaucoup de questions restent posées notamment celles liées aux mesures d'évaluation des systèmes. L'apparition de nouvelles tâches au sein d'INEX montrent que tout le potentiel de l'information structurelle n'est pas encore abordé et que la sémantique d'une telle structuration ainsi que celle des contenus est importante à prendre en compte. On commence à s'orienter vers les ontologies (voir section 2.8.2) se rapprochant ainsi des problématiques du Web Sémantique. Notre système propose une prise en compte de la sémantique tant au niveau de la structure qu'au niveau du contenu. Il repose sur l'utilisation d'ontologie(s). Nous présentons dans le prochain chapitre les ontologies, et leur utilisation dans le cadre de RI et RIS.

Deuxième partie

Texte et Sémantique

Chapitre 3

Ontologies

Sommaire

3.1 Introduction

Dans le dictionnaire Hachette en ligne[44], le terme **ontologie** n'apparaît qu'au singulier avec la définition suivante :

[44] http ://www.dictionnaire.com/hachette/

Partie de la métaphysique qui s'applique à « l'être en tant qu'être » (Aristote), indépendamment de ses déterminations particulières.
En informatique ce terme désigne une ressource sémantique caractérisée surtout par une structure hiérarchique. Dans ce chapitre -comme l'indique le titre- nous parlons d'ontologie**s** au pluriel.
La vision ambitieuse des premiers systèmes d'Intelligence Artificielle [Lenat & Guha, 1990] de construire une ontologie universelle est devenue utopique, il existe autant d'ontologies différentes que de domaines d'application, de tâches envisagées et de points de vue (des experts, utilisateurs, etc.). Les travaux actuels ont des objectifs plus raisonnables et tendent à développer des ontologies spécifiques.
Le but de l'ontologie est de représenter les *connaissances* d'un domaine. Kayser [1997] définit intuitivement la connaissance par rapport à l'information par : « Il n'y a présomption de connaissance qui si la faculté d'utiliser des informations à bon escient est attestée ».

Définition

> Une ontologie, selon [Guarino, 1998], est constituée par un vocabulaire utilisé pour décrire une certaine réalité, avec un ensemble d'hypothèses quant au sens des mots du vocabulaire.

Le triangle de sens [Sowa, 2000], présente un objet qui est une entité du monde, exemple « un chat » ; le symbole qui permet d'identifier cette entité, le terme « chat » ; et le concept qui est le sens porté par le mot « chat » et qui constitue la représentation sémantique entre l'objet et le symbole. [Web, 2003]utilisent ce triangle de sens pour différencier les termes des concepts et montrer la place de l'ontologie dans un tel processus.

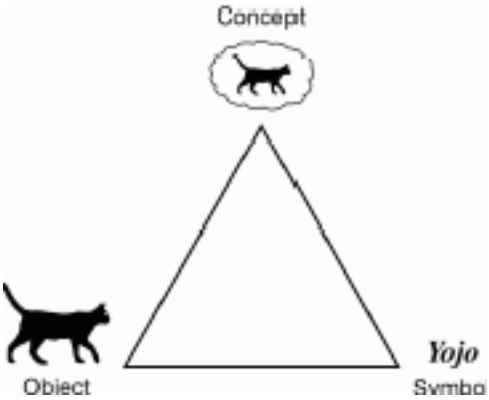

FIG. 3.1 – Le triangle de sens

3.2 Ressources sémantiques

Plusieurs ressources sémantiques existent : Thésaurus, taxonomie, réseau sémantique

3.2.1 Thésaurus

Un thésaurus est une sorte de dictionnaire hiérarchisé, un vocabulaire normalisé sur la base de termes génériques et de termes spécifiques à un domaine. Les termes y sont organisés de manière conceptuelle et reliés entre eux par des relations sémantiques. Un thésaurus fournit accessoirement des définitions. L'accent est mis sur le choix des termes (on parle alors de descripteurs) et des relations entre termes. Les relations communément exprimées dans un thésaurus sont :
(i) les relations taxonomiques (de hiérarchie),
(ii) les relations d'équivalence (synonymie),
(iii) relations d'association (relations de proximité sémantique, proche-de, relié-à, etc.).
Le thésaurus peut être exploré par ordre alphabétique ou en parcourant une liste hiérarchique des termes. Plusieurs standards existent pour le développement de thésaurus Monolingue (NISO, 1998 ; ISO, 1986) et multilingues (ISO, 1985).
Un des meta-thésaurus les plus connus est UMLS (*Unified Medical Language System*) du National Library of Medicine qui définit plus de 40 relations.

3.2.2 Taxonomie (ou taxinomie)

La taxonomie (du grec taxis : rangement et nomos : loi) est la partie de la biologie visant à établir une classification systématique des êtres vivants. Par extension, on appelle taxonomie une organisation de concepts unis par des relations hiérarchiques.

3.2.3 Les réseaux sémantiques

Les réseaux sémantiques [Quillian, 1968]ont été conçus à l'origine comme un modèle de la mémoire humaine. Un réseau sémantique est un graphe orienté et étiqueté (ou, plus précisément, multigraphe). Un arc lie (au moins) un nœud de départ à (au moins) un nœud d'arrivée. Les relations vont des relations de proximité sémantique aux relations partie-de, cause-effet, parent-enfant, etc.
Les concepts sont représentés sous fourme de nœuds et les relations sous forme d'arcs. L'héritage des propriétés par les liens est matérialisé par un arc (sorte-de) entre les nœuds. Les liens de différentes nature peuvent être mélangés ainsi que les concepts et instances. Diverses évolutions ont vu le jour et ont conduit à la définition de langages formels (cf. section 3.4).
Un des réseaux sémantiques les plus utilisés est Wordnet, largement utilisé pour la recherche d'information. Wordnet est un réseau lexical et sémantique qui a été initialement élaboré à partir du corpus Brown. Il regroupe les mots selon leur sens dans des synsets. (cf. annexe A).

3.2.4 Ontologies

Les ontologies permettent, d'une part de décrire les connaissances d'un domaine spécifique et d'autre part de représenter des relations complexes entre les concepts, ainsi que des axiomes et règles qui manquaient aux réseaux sémantiques.
Il y a généralement trois niveaux de représentation :

– Niveau représentation des connaissances : définit les constructeurs qui seront utilisés au niveau de l'ontologie des concepts. Nous présenterons dans ce chapitre les trois familles de langages de représentation de connaissance les plus répandus qui ont précédé les langages de représentation pour le Web Sémantique (cf chap suivant).
– Niveau ontologie de concepts : l'ontologie est définie en utilisant les constructeurs du niveau représentation des connaissances. A ce niveau, on s'intéresse à modéliser les connaissances du domaine.
– Niveau ontologie d'instances : à ce niveau, les constructeurs sont des instances des constructeurs de l'ontologie de concepts.

Dans ce qui suit nous parlerons d'ontologie pour décrire les deux niveaux (concept et instance). Une classification des ontologies selon Guarino [1997] distingue quatre types d'ontologies (cf. figure 3.2) :

– Ontologies génériques (Top-level ontologies) décrivent des concepts génériques indépendamment d'un domaine et d'une application particulière. Elle défini des catégories abstraites de temps, d'espace, d'événement, d'action, etc. (eg. Mikrokosmos)
– Ontologies du domaine décrivent la terminologie d'un domaine spécifique. (eg. Menelas).
– Les ontologies de tâche décrivent des tâches génériques, comme les transactions bancaires par exemple. (eg. ONTOLINGUA)
– Les ontologies d'application sont la combinaison des ontologies du domaine et des ontologies orientés tâche.

FIG. 3.2 – Types d'ontologie

En Ingénierie des connaissances, le terme ontologie désigne implicitement *ontologie formelle*, c'est-à-dire une ontologie dotée d'une syntaxe et d'une sémantique avec des mécanismes d'inférence. La conceptualisation du domaine passe par une explicitation des connaissances du domaine. Uschold [2002] expose les différents types de sémantique :

– Sémantique implicite : le sens est fondé sur une compréhension commune, fruit d'un consensus. Le problème majeure est l'ambiguïté, puisque le sens n'est pas explicitement donné, ceci donne lieu à des interprétations diverses.
– Sémantique explicite et informelle : le sens est explicite mais exprimé d'une manière informelle, comme par exemple un glossaire. Le problème d'ambiguïté est réduit mais reste posé car l'interprétation est laissé au soin du lecteur et peut de ce fait être multiple.
– Sémantique explicite et formelle destinée aux humains : La formalisation du sens explicite réduit nettement l'ambiguïté mais la destination aux humains peut laisser place à des erreurs.

– Sémantique explicite et formelle destinée aux machines : La sémantique est formellement identifiée et peut de ce fait s'intégrer automatiquement dans un processus d'inférence pour générer d'autres connaissances.

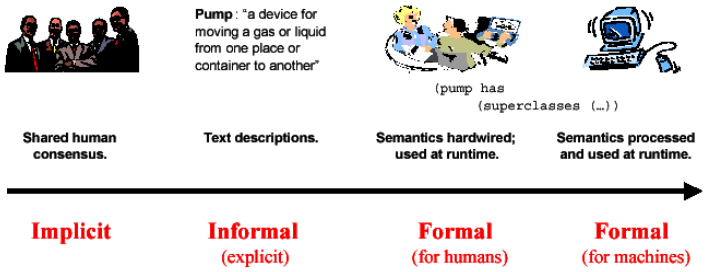

FIG. 3.3 – Continuum sémantique [Uschold, 2002]

3.3 Construction d'ontologies

La construction d'ontologies requiert un effort considérable de conceptualisation du domaine à représenter et fait appel à des domaines de recherche interdisciplinaires qui regroupent des recherches dans les domaines d'ingénierie et représentation des connaissances, de linguistique informatique, d'extraction d'information et d'apprentissage [Maedche & Staab, 2001]. La conceptualisation d'un domaine nécessite le choix des entités à modéliser ainsi que leur organisation en une ontologie. Guarino *et al.* [1994] précise qu'il est important de fixer préalablement à toute construction d'ontologie les engagements ontologiques généraux, les catégories de haut-niveau (appellations et significations) ainsi que la spécialisation de ces catégories.
Une construction automatique d'ontologies est difficilement envisageable, une intervention humaine est toujours nécessaire pour le choix de conceptualisation à faire. Il est plus important d'outiller cette intervention que de chercher à la minimiser [Charlet *et al.*, 2003].
En Ingénierie des Connaissances, la définition des entités de connaissance à représenter, repose souvent sur des échanges avec les experts. Or, cela suppose la disponibilité de ces experts ainsi que leur capacité à expliciter leurs propres connaissance. Dans la pratique cela reste rare, l'utilisation des textes représentatifs de l'expertise s'est donc imposée. Ces ressources, si elles sont bien choisies, peuvent constituer un bon départ pour modéliser les connaissances. Les experts auront pour rôle de valider les connaissances extraites ainsi que leur organisation.

Nous présentons dans ce qui suit des travaux qui proposent une méthodologie de construction d'ontologie ainsi que quelques outils d'apprentissage d'ontologie. Nous nous intéressons particulièrement à la construction d'ontologies à partir de corpus textuels[45]. D'autres méthodes de construction d'ontologie existent : à partir de dictionnaires [Hearst, 1992] ; [Rigau, 1998], à partir de bases de données [Rubin *et al.*, 2002] et à partir de données semi-structurées [Deitel *et al.*, 2001].
Le lecteur intéressé par un état de l'art exhaustif peut se référer à [Gómez-Pérez & Manzano-Macho, 2003].

[45]Nous attirons l'attention du lecteur qu'un corpus textuel dans ce contexte représente un ensemble de textes représentatifs d'un domaine et accepté par les experts du domaine. Ce corpus doit couvrir le maximum de notions utilisées dans ce domaine.

3.3.1 Méthodologie de construction d'ontologies

Les méthodologies de construction d'ontologies que nous présentons proposent d'assister la conceptualisation en repérant les connaissances du domaine à travers l'analyse terminologique. Nous présentons dans ce qui suit Terminae [Aussenac-Gilles *et al.*, 2003] ainsi que la méthode de Bachimont [2000], deux méthodes issues du groupe TIA (*Terminologie et Intelligence Artificielle*)[46]. Ce groupe de travail pluridisciplinaire tend à combiner les méthodes de la terminologie et ceux de l'intelligence artificielle. Le groupe TIA vise à élaborer et tester des méthodes et outils de travail sur corpus pour produire de manière systématique des ressources terminologiques riches en informations sémantiques et exploitables dans diverses applications.

Terminae (Aussenac-Gilles *et al.* [2003])

Terminae propose un cadre pour la construction d'ontologies en s'aidant des outils de Traitement Automatique de la Langue tels que Cordial[47], Lexter[48] [Bourigault, 1994], TreeTagger (cf. section 1.5). Le processus est cyclique et nécessite une validation continue par l'ingénieur des connaissances. La dernière version de Terminae permet de générer des ontologies en OWL à partir de corpus textuels.
Terminae propose une méthodologie générique indépendante du langage utilisé et du domaine d'étude. Il préconise les étapes incrémentales suivantes [Szulman *et al.*, 2002] :

- Mise en place de corpus de référence : le corpus doit couvrir tout le domaine spécifié par l'application et être validé par un expert.
- Analyse linguistique : cette étape consiste à sélectionner les outils linguistiques appropriés et les appliquer aux textes. Lexter est utilisé dans un premier temps pour proposer un ensemble de candidats-termes mais aussi Cordial et SynoTerm[49] [Hamon, 2000]. La principale difficulté est dans le choix de ces outils linguistiques en fonction de la langue d'étude. Les outils gênèrent une liste de candidats-termes, de relations et des groupes de synonymes.
- Normalisation : cette étape se fait en deux temps. Dans un premier temps, les informations lexicales sont exploitées et conduisent à la création de fiches terminologiques à partir des candidats-termes et des relations lexicales (hyperonymie, etc.) retenus dans l'étape précédente. Puis des fiches modélisations sont créées suite à une interprétation sémantique ainsi qu'une structuration de concepts par l'ingénieur.
- Formalisation : cette étape comprend la construction et la validation de l'ontologie. Les concepts et relations sémantiques des fiches de modélisation sont traduits en concepts et relations formels puis insérés dans l'ontologie. Ceci peut amener à restructurer l'ontologie ou créer de nouveaux concepts. La validation de l'ontologie (en vérifiant sa cohérence) peut se faire quand cette dernière atteint un niveau stable. Les langages formels utilisés sont les logiques de description et récemment OWL [Szulman & Biébow, 2004].

Terminae offre la traçabilité des textes vers l'ontologie et vice versa, ce qui est très utile pour comprendre l'ontologie, l'utiliser et la maintenir [Szulman *et al.*, 2002].

[46] http ://www.biomath.jussieu.fr/TIA

[47] Cordial permet un étiquetage morpho-syntaxique des textes en français, une analyse statistique des caractéristiques stylistiques des textes ainsi qu'une aide à l'analyse terminologique et sémantique de corpus.

[48] Lexter est un analyseur syntaxique dédié à l'extraction de syntagmes (nominaux et adjectivaux) pour le français à partir de corpus spécialisés, dans une perspective d'acquisition terminologique.

[49] SynoTerm a pour objectif la détection de relations de synonymie dans une liste de termes extraits d'un corpus.

Nobécourt [1999] présente une approche pour la construction d'ontologies du domaine à partir de texte. La méthode proposée se compose en deux activités :

– Modélisation : cette activité regroupe une phase d'analyse linguistique et une phase d'analyse conceptuelle. L'analyse linguistique extrait des *primitives conceptuelles* du corpus. L'expert sélectionne les primitives pertinentes pour le domaine d'étude, ces primitives sont modélisés en concepts ou propriétés et constituent le squelette de l'ontologie qui sera affinée par un processus itératif.
– Représentation : cette activité consiste à traduire la modélisation dans un langage de représentation de connaissances (cf. section 3.4).

La méthodologie proposée par (Bachimont [2000])

La méthode proposée par Bachimont propose de contraindre l'utilisateur à un *engagement sémantique* Bachimont [2000] en introduisant une normalisation sémantique des termes manipulés dans l'ontologie. Elle prend en considération les outils linguistiques qui proviennent de la *sémantique différentielle*[50] présentée par [F. Rastier & Abeillé, 1994].

La méthode de normalisation suit trois étapes (voir figure3.4) :

– Normalisation sémantique : l'utilisateur doit choisir les termes du domaine et les normaliser en explicitant leurs propriétés et en exprimant les identités et les différences dans leur voisinage proche. La place d'une *notion*[51] dans l'ontologie doit être justifiée par rapport à la communauté et la différence avec le père et la fratrie.
– Formalisation des connaissances : Cette étape consiste à désambiguïser les notions de l'ontologie référentielle obtenue par l'étape précédente et choisir leurs sens pour un domaine spécifique. Cela peut nécessiter la création de nouveaux concepts, l'ajout de propriétés et d'axiomes.
– Opérationnalisation des connaissances : Le système utilise un langage opérationnel de représentation de connaissances (cf. section 3.4) qui possède les caractéristiques nécessaires pour répondre aux besoins exprimés lors de la spécification du système.

[50] Le sens se construit par des relations d'opposition entre les unités du système linguistique. Les sens sont attribués aux termes grâce à des traits sémantiques ou *sèmes*.
[51] Une notion est une unité d'information retenue.

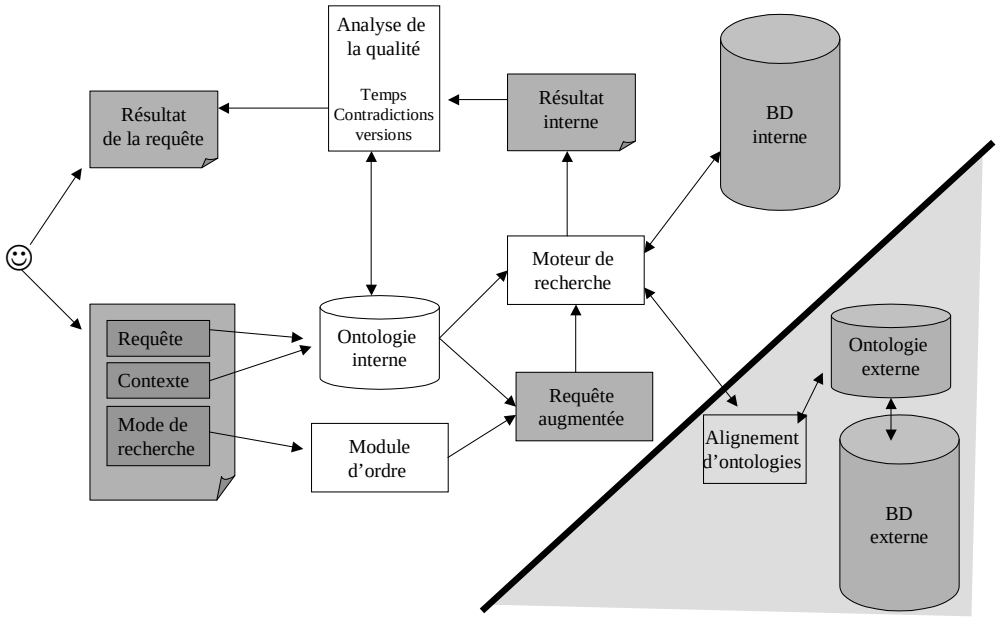

FIG. 3.4 – Les trois étapes de la méthode de construction d'ontologies selon [Bachimont, 2000]

3.3.2 Apprentissage d'ontologie

L'apprentissage d'ontologie appelé aussi *extraction* ou *génération* consiste à construire semi-automatiquement des ontologies à partir du corpus.
L'apprentissage d'ontologies tend à développer des méthodes et outils qui réduisent la tâche manuelle d'ingénierie et de gestion des ontologies. Nous présentons dans ce qui suit deux exemples d'outils.

ASIUM *Acquisition of SemantIc knowledge Using Machine learning methods*([Faure & Nédellec, 1999])

Le but du système ASIUM est d'aider les experts pour l'acquisition sémantique de connaissance à partir de l'analyse syntaxique de corpus de textes techniques. Le système apprend des cadres de sous-catégorisation de verbes et une ontologie pour un domaine spécialisé.
Les cadres de verbes appris sont de la forme :

<verbe><rôle syntaxique|préposition : concept>*.*

Le système fonctionne en deux étapes :
– Factorisation : La première étape consiste à extraire des cadres syntaxiques par SYLEX [Constant, 1995] sous forme *<verbe> <préposition|rôle : nom de tête>*. L'apprentissage se fait à partir des régularités syntaxiques : si deux noms de têtes apparaissent après un même couple verbe+préposition/fonction (ils forment alors une classe de base) dans des contextes

différents alors ils représentent un même concept. Une mesure de similarité calcule le recouvrement entre deux classes et propose le regroupement si la similarité est supérieure à un seuil fixé par l'expert.
– Clusterisation : Les classes de bases sont successivement agrégées afin de former les concepts de l'ontologie niveau par niveau. Les concepts doivent être validés à chaque niveau pour pouvoir être utilisés pour le niveau suivant.

Par exemple, à partir de cadres syntaxiques suivants [Faure & Poibeau, 2000] :

```
<voyager>
    <sujet :[père,voisin,ami]>
    <en : [voiture,train]>
<conduire>
    <sujet :[ami,collègue]>
    <objet : [voiture,moto]>
```

ASIUM apprendra les deux concepts :

```
Humain :
Père ;voisin ;ami ;collègue.
Véhicule motorisé :
Voiture ;train ;moto.
```

Et deux cadres de sous-catégorisation :

```
<voyager>
        <sujet : Humain>
        <en : Véhicule motorisé>
<conduire>
        <sujet : Humain>
        <objet : Véhicule motorisé >
```

Les systèmes de construction semi-automatique posent toujours le problème de validation. L'expert doit intervenir à tous les niveaux et doit aussi juger du niveau de granularité des concepts appris. Un effort particulier est donc fait pour faciliter au plus la tâche de l'expert par des interfaces intuitives et ergonomiques. Ces outils constituent une aide précieuse pour la construction d'ontologies et permettent d'alléger la tâche de l'ingénieur de connaissance ou terminologue.
Le système SVETLAN' que nous présentons dans la section suivante propose une approche similaire mais dans un processus complètement automatique indépendamment du domaine d'étude.

SVETLAN' [de Chalendar & Grau, 2000]

SVETLAN' est un système qui traite des corpus généraux (par opposition à corpus spécialisés) comme les dépêches de l'AFP. Il ne nécessite donc pas la validation d'experts. Le système crée des clusters à partir des mots apparaissant dans les textes. La méthode d'apprentissage est fondée sur une approche distributionnelle : les noms ayant le même rôle syntaxique avec les mêmes verbes sont agrégés dans la même classe.
Le processus d'apprentissage commence par une analyse syntaxique suivie d'une phase d'agrégation et d'un filtrage. L'analyse syntaxique détermine la place du verbe dans la phrase, les

verbes sont supposés permettre la catégorisation des noms. L'étape d'agrégation construit à partir des triplets *(nom-relation-verbe)*, générés par l'étape précédente, des groupes de mots ayant un sens similaire. La phrase de filtrage supprime les mots qui ne sont pas pertinents (en fonction de leurs poids) pour la classe.

Le processus est donc indépendant du domaine d'étude, il trouve une application dans les corpus généraux. Pour les corpus de spécialité le système risque d'être moins performant qu'un système tel que ASIUM [de Chalendar & Grau, 2002].

3.4 Langages de représentation

Les langages de représentation de connaissances reposent sur une **syntaxe** qui précise l'ensemble des expressions admissibles du langage et d'une **sémantique** qui définit le sens de ces formules. Ils offrent des **services inférentiels** tels que l'héritage des propriétés, la classification automatique de descriptions de concepts, etc.

Les travaux en Représentation des Connaissances ont donné naissance à plusieurs formalismes. Nous présentons dans ce qui suit les trois grandes famille de langages à savoir : les logiques de description (section 3.4.2), les graphes conceptuels (section 3.4.3) et les langages de représentation à objet (section 3.4.4).

Outre le fait qu'ils soient les plus utilisés, ces trois langages ont comme ancêtres communs, les réseaux sémantiques (cf. section 3.2.3) et le langage de frames que nous présentons dans la section qui suit.

3.4.1 Le langage de frames

Le langage de frame ou *schéma* a été introduit par Minski [1975]. Un frame représente soit une classe (*class frame*) ou un objet (*instance frame*) et peut être vu comme un réseau de nœuds et relations. Un frame contient des attributs (appelés *slots*) pour décrire les propriétés des objets. Les valeurs des slots peuvent être spécifiées ou calculées [Sowa, 2000].

Les frames sont organisés dans une hiérarchie suivant un lien de spécialisation. Les propriétés sont héritées des *super-frames* aux *sous-frames* dans la hiérarchie suivant une certaine stratégie d'héritage. Il existe trois opérations principales permettant de faire des inférences sur ces structures : (1) la propagation permet de faire hériter en suivant les liens de spécialisation à un frame donné l'ensemble des propriétés de ses pères (2) l'appariement permet de retrouver le frame classe correspondant à un objet et (3) la classification permet d'insérer une classe dans la hiérarchie des frames. Toutefois les frames n'ont pas de sémantique formelle.

Le langage des frames et les réseaux sémantiques ont largement influencé les langages de représentation de connaissance.

3.4.2 Logiques de Description (LD)

Les Logiques de Description (LD) sont nées de la volonté de donner une sémantique solide aux langages de frame et aux réseaux sémantiques. Les logiques de description, connus aussi par le terme de *logiques terminologiques* forment une classe de langages utilisée pour construire et accéder à des bases de connaissances.
Les logiques de description permettent de représenter les connaissances relatives à un domaine de référence à l'aide de "descriptions" qui peuvent être des concepts, des rôles et des individus. Les concepts modélisent des classes d'individus et les rôles des relations entre classes. Une sémantique est associée aux descriptions par l'intermédiaire d'une fonction d'interprétation. La relation de subsomption permet d'organiser les concepts et les rôles en hiérarchie. La classification et l'instanciation sont les opérations qui sont à la base du raisonnement sur les descriptions, ou raisonnement terminologique.
Une logique de description comporte deux composantes : une composante terminologique (T-BOX) et une composante assertionnelle (A-BOX) qui ont leur propre langage et mécanismes d'inférence associés. Le langage terminologique de la T-BOX permet de définir les concepts. Le langage assertionnel de la A-BOX permet de définir des individus et des règles. On trouve dans Napoli [1997] une introduction plus complète aux logiques de description.

Syntaxe

Les objets qui sont définis et manipulés dans une logique de description sont les concepts, les individus et les rôles. Les concepts peuvent être vus comme des prédicats logiques unaires. Les concepts peuvent être définis par extension (par la liste d'instances des concepts) ou intension en terme de descriptions de propriétés que les instances doivent satisfaire pour appartenir aux concepts. Ils peuvent être décrits par des expressions constituées avec des opérateurs syntaxiques comme l'intersection (AND), l'union (OR), des constructeurs introduisant les rôles associés aux concepts et les restrictions attachées à ces rôles. Un concept dénote un ensemble d'individus [Borgida & P.F.Patel-Schneider, 1994]. Trois types de concepts peuvent être dégagés :
– Les concepts définis sont des concepts décrits par des propriétés (liens hiérarchiques et rôles) qui constituent un ensemble de conditions nécessaires et suffisantes pour la reconnaissance d'instances ;
– Les concepts atomiques sont définis uniquement par leurs noms ;
– Les concepts primitifs sont définis avec des conditions nécessaires mais pas suffisantes.
Les individus sont les instances des concepts. Ce sont les objets du domaine. Les rôles sont similaires aux prédicats logiques binaires. Les restrictions des rôles portent généralement sur le co-domaine et la cardinalité.
Par exemple la description du concept suivante décrit tous les enfants qui sont scolarisés. Exemple

$$Enfant \sqcap \exists\, estScolarise \;:\; Ecole$$

Le pouvoir d'expression associé à une logique de description dépend des opérateurs utilisés pour définir les concepts et les rôles. Il existe plusieurs variantes de LD. Chaque variante correspond à un langage de description qui précise clairement les opérateurs utilisables. On présente souvent le langage AL comme le langage de base, il utilise les constructeurs suivants :

$C, D \leftarrow T$// concept primitif
| $\top$ | $\bot$ // concept le plus général | absurde
| $C \sqcap D$ // conjonction de concept
| $C \sqcup D$ // disjonction de concept
| $\neg C$ // négation
| $\forall R : C$ //restriction universelle
| $\exists R : C$ //restriction existentielle
| $\geqslant_n R.C$ //cardinalité minimum
| $\leqslant_n R.C$ //cardinalité maximum

$R \leftarrow P$ // rôle primitif
| $R_1 \sqcap R_2$ //conjonction de rôles
| $R_1 \sqcup R_2$ //disjonction de rôles
| R^{-1} // inverse de rôles

AL = $\sqcap, \forall, \exists, \neg, \top, \bot$. On ajoute à ce langage différents opérateurs pour former les différentes LD [Nebel, 1990]. Le nom de ces langages est donné par référence à AL et par le nom des constructeurs supplémentaires acceptés. ALCNR est considéré comme le langage de référence le plus large. L'étude de la complexité de plusieurs langages (ALC, ALCN, ALCNR, ALUR, etc.) est donné dans F. Donini [1995].

Sémantique

Les LDs sont dotées d'une **sémantique extensionnelle**. Soit le domaine Δ : un ensemble non vide d'individus et I une fonction d'interprétation. Selon cette sémantique, la relation de subsomption extensionnelle est définie comme suit :

Définition

> Un concept C subsume un concept D (respectivement D est subsumé par C), noté $D \sqsubseteq C$ si et seulement si $D^I \subseteq C^I$. Pour toute interprétation I, le concept C est appelé le subsumant et D le subsumant.

L'extension d'un rôle de nom R est un sous-ensemble de $\Delta \times \Delta$ noté R^I. La sémantique des constructeurs de la LD sont présentés dans le tableau 3.1.

Les mécanisme d'inférence dans les logiques de description

L'intérêt des logiques de description est de posséder des mécanismes d'inférence qui permettent de raisonner sur les concepts qui ont été définis. Deux mécanismes d'inférence basés sur le calcul de subsomption.

– La relation de subsomption : elle est la relation qui permet d'organiser les concepts et les rôles en hiérarchies. Par exemple l'ensemble des instances du concept $Enfant$ est inclus dans l'ensemble des instances du concept $Personne$.

Constructeur	Sémantique
$\top \mid \bot$	$\Delta^I \mid \emptyset$
$C \sqcap D$	$C^I \cap D^I$
$C \sqcup D$	$C^I \cup D^I$
$\neg C$	$\Delta^I \; C^I$
$\forall R : C$	$\{x \mid x \in \Delta : \forall y, < x, y > \in R^I \Rightarrow y \in C^I \}$
$\exists R : C$	$\{x \mid x \in \Delta : \exists y, < x, y > \in R^I \wedge y \in C^I \}$
$\geqslant_n R.C$	$\{x \in \Delta : \{y, < x, y > \in R^I\} \geqslant n \}$
$\leqslant_n R.C$	$\{x \in \Delta : \{y, < x, y > \in R^I\} \leqslant n \}$
$R_1 \sqcap R_2$	$\{x, y \mid < x, y > \in \Delta^I \times \Delta^I :< x, y > \in R_1^I \wedge < x, y > \in R_2^I \}$
$R_1 \sqcup R_2$	$\{x, y \mid < x, y > \in \Delta^I \times \Delta^I :< x, y > \in R_1^I \vee < x, y > \in R_2^I \}$
R^{-1}	$\{x, y \mid < x, y > \in \Delta^I \times \Delta^I :< y, x > \in R_1^I \}$

TAB. 3.1 – Syntaxe et sémantique des constructeurs d'une LD

Les deux mécanismes d'inférence sont principalement utilisés dans les logiques de description à savoir [Napoli, 1997] :

– La classification : elle s'applique aux concepts, le cas échéant aux rôles, et permet de déterminer la position d'un concept et d'un rôle dans leurs hiérarchies respectives. La construction et l'évolution de ces hiérarchies est ainsi assistée par le processus de classification. La classification s'effectue au niveau de la T-BOX.
– La reconnaissance des instances : elle permet de retrouver les concepts les plus spécifiques dont un individu est susceptible d'être une instance. L'instanciation s'effectue au niveau de la A-BOX.
– La relation de subsomption : elle est la relation qui permet d'organiser les concepts et les rôles en hiérarchies. Par exemple l'ensemble des instances du concept $Enfant$ est inclus dans l'ensemble des instances du concept $Personne$.

Les logiques de description ont aussi l'avantage -par rapport aux bases de données classiques- de raisonner à partir de connaissances incomplètes [Franconi, 2000]. De nombreuses études théoriques ont permis de donner des résultats précis sur les propriétés des algorithmes (complexité, complétude, correction) en fonction des connecteurs choisis [Nebel, 1990]. Ils ont permis de mettre en évidence les rapports qui existent entre la complexité du raisonnement terminologique et la richesse -en termes de nombre et de types de constructeurs- des langages de descriptions de concepts et de rôles. Deux attitudes ont guidé les concepteurs de logiques de description : avoir un langage de description plutôt pauvre mais une procédure d'inférence correcte, complète et de complexité polynomiale ou au contraire avoir un langage de description riche et une procédure d'inférence de complexité exponentielle. Nous verrons dans le chapitre suivant que cette approche est aussi préconisée dans le cadre du Web sémantique avec la définition de différents niveaux de OWL. Plusieurs systèmes de gestion de bases de connaissances ont vu le jour. Parmi les plus connus, on peut citer CLASSIC [Brachman *et al.*, 1991], LOOM [MacGregor & Bates, 1987],BACK [T. Hoppe & Fischer, 1995] et FaCT (*Fast Classification of Terminologies*)[52] [Horrocks, 1998].

[52]http ://www.cs.man.ac.uk/ horrocks/FaCT/

3.4.3 Les Graphes Conceptuels (GC)

Les GCs ont été introduits par Sowa, comme un langage de représentation de connaissances basé sur la linguistique, la psychologie et la philosophie [Sowa, 1984]. Nous présentons les Graphes Conceptuels Simples (CGS) tels que présentés dans [Chein & Mugnier, 1992] et [Mugnier & Chein, 1996].

Le modèle des GCSs est un modèle déclaratif de représentation de connaissances basé sur des graphes étiquetés, les calculs d'inférence se faisant par des algorithmes de graphe, ayant une sémantique logique consistante et complète. Les GCs ont été répandu grâce à la facilité de représentation graphique. Ainsi, la projection qui est une opération fondamentale est facilement visualisable.

Syntaxe

Un support définit le vocabulaire de base avec lequel on représente des connaissances sur un domaine, sa terminologie. Un support $S = (T_c, T_r, \sigma, M, conf)$, avec :

- T_c : est un treillis des types de concepts. La relation d'ordre est notée $<_{T_c}$. Le plus grand élément du treillis $\top$ est le type absolu. Le type absurde $\bot$ est le plus petit élément.
- T_r : est un treillis des relations d'arité a, ordonné suivant la relation $< T_{r_a}$.
- σ est l'ensemble des signatures des relations. Elle associe à chaque type de relation le type maximal de chacun de ses arguments.
- M un ensemble de marqueurs individuels dénombrable mais pas nécessairement fini. * est un marqueur dit générique.
- $conf$ est une relation de conformité sur l'ensemble des couples (T_c, M) qui à tout marqueur individuel m associe un type de concept t.

Un exemple de support de graphe conceptuel est donnée dans les figures 3.5.
Un graphe conceptuel est un multigraphe biparti, c'est-à-dire ayant deux types de nœuds. Il est basé sur plusieurs éléments :

- Un ensemble de nœuds relation (représentés par des rectangles).
- Un ensemble de nœuds concept qui ne peut être vide (représentés par des ovales).
- Un ensemble d'arêtes reliant un nœud concept à un nœud relation. Les arêtes adjacentes à un nœud relation sont ordonnées, ceci étant représenté par une numérotation des arêtes.

A chaque nœud est associé une étiquette qui diffère si le noeud est une relation conceptuelle ou un concept. L'étiquette associée à un concept est composée d'un référent et du type de ce référent. Le type et le référent doivent être liés par une relation de conformité. Si un référent est conforme à un type alors il est conforme à tous ses super-types définis dans le treillis des concepts. Si le sommet possède un référent générique alors ce concept est dit concept générique $Enfant$ ou $Enfant : *$. Sinon le concept est un sommet individuel $Enfant : Marie$. L'étiquette associée à un sommet relation est seulement composée du type de relation.

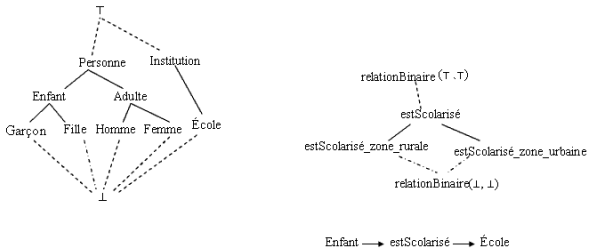

FIG. 3.5 – Un exemple de support de graphe conceptuel

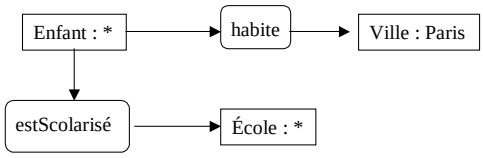

FIG. 3.6 – Un exemple de graphe conceptuel

Il existe plusieurs notations pour les graphes conceptuels : la notation graphique (DF : display form) et la notation linéaire (LF : linear form) qui est une représentation textuelle. Une norme ISO nnn-n spécifie une syntaxe et une sémantique pour les GCs et leur représentation en *Conceptual Graph Interchange Form* (CGIF).
La figure 3.6 montre un graphe conceptuel avec trois concepts [Enfant], [École], [Ville : Paris]. Il a deux relations conceptuelles : (habite) relie [Enfant] à [Ville :Paris] et (estScolarisé) relie [Enfant] à [École].
La forme linéaire s'écrit de la manière suivante.
*[Enfant : *]-*
(Habite) $\mapsto$ *[Ville : Paris]*
(est_scolarisé) $\mapsto$ *[École : *]*

La représentation CGIF s'écrit de la manière suivante.
*[Enfant *x] [Ville : Paris *y] [École *z] (Habite ?x ?y) (est_scolarisé ?x ?z)*

Sémantique

Les GCs sont dotés d'une sémantique obtenue en traduisant les supports et les graphes en logique de premier ordre, par le biais de l'opérateur Φ [Sowa, 1984].

L'interprétation d'un support S en logique de premier ordre est donnée par $\Phi(S)$ définie comme la conjonction des formules suivantes [Mugnier & Chein, 1996] :
- Pour tous t_1 et t_2 de T_c, tels que $t_1 <_{T_c} t_2$[53], on considère la formule : $\forall x, t_1(x) \rightarrow t_2(x)$, qui correspond à l'interprétation des liens *sorte-de* entre types de concepts.
- Pour tous t_{r_1} et t_{r_2} de T_{r_a} tels que $t_{r_1} <_{T_r} t_{r_2}$, on considère la formule : $\forall x_1 \ldots \forall x_n \; t_{r_1}(x_1 \ldots x_n) \rightarrow t_{r_2}(x_1 \ldots x_n)$, qui correspond à l'interprétation des liens *sorte-de* entre types de relation.
- Pour tout t_r de T_r, soit $\sigma(t_r) = (t_1 \ldots t_n), \forall x_1 \ldots \forall x_p \; t_r(x_1 \ldots x_n) \rightarrow t_1(x_1) \wedge \ldots t_n(x_n)$, interprétation des signatures de relations.

On construit la formule $\Phi(G)$ (l'interprétation d'un graphe conceptuel G) de la façon suivante :
- A tout sommet est associé un atome. A un sommet concept c, on associe l'atome $t_c(id_c)$, où t_c est le prédicat associé au type de c et id_c est le terme associé à c.
- A un sommet relation r, on associe l'atome $t_r(id_{c1} \ldots id_{cn})$ où t_r est le prédicat associé au type de r, n est l'arité de ce prédicat et id_{c_i} désigne le terme associé au i^{eme} voisin de r dans G.

$\Phi(G)$ est obtenue en faisant la conjonction de ces atomes, puis en fermant existentiellement la formule. On associe par exemple au graphe de la figure 3.6 les formules suivantes :
$\exists x \exists y \; Enfant(x) \wedge Ville(Paris) \wedge École(y) \wedge Habite(x, Paris)) \wedge est\_scolarisé(x, y)$

Les mécanismes d'inférence dans les graphes conceptuels

Il existe beaucoup d'extensions au modèle de base de Sowa [2000] et aux GCs de Mugnier & Chein [1996]. Ces extensions se sont traduites par l'ajout de nouveaux mécanismes d'inférence. Nous présentons dans ce qui suit les opérations de base :
- Opérations élémentaires de spécialisation : ce sont des opérations internes sur l'ensemble des graphes sur un support donné. Ces opérations peuvent être (i) unaires (simplification, restriction de relation, restriction de concept, jointure interne) et binaires (somme disjointe).
- Opérations élémentaires de généralisation : on peut les définir comme les inverses des règles de spécialisation. Ce sont la duplication, l'augmentation de relation, l'augmentation de concept, l'éclatement et décomposition.
- Projection : la projection est un morphisme de graphes. C'est l'opération fondamentale qui permet le calcul effectif de la relation de spécialisation sur les graphes conceptuels. Une projection d'un graphe G en H est un couple d'applications suivantes : conserver les arcs et leur numérotation, restreindre si possible les étiquettes des sommets.

On trouve dans la communauté plusieurs *outils graphe conceptuel*. Ces outils peuvent être des éditeurs de graphes[54], des *boites à outils* configurables par l'utilisateur [Haemmerlé, 1995], etc.

3.4.4 La Représentation de Connaissances à Objets (RCO)

L'expression systèmes à objet recouvre l'ensemble des systèmes où l'unité de connaissance est la classe (accompagnée de ses instances), que ce soit pour programmer ou pour représenter des connaissances. Les classes sont organisées en hiérarchies conceptuelles et le processus de

[53] t_2 couvre t_1 par un lien direct sans transitivité.
[54] http ://www.cs.uah.edu/~ delugach/CharGer/

classification fait partie intégrante des opérations de raisonnement. Les systèmes de RCO sont des langages hybrides qui sont le fruit de l'évolution des langages de frames sous la double pression des langages de classes et des logiques de description.

Dans [Chouvet *et al.*, 1996], les auteurs essaient d'apporter des éléments de réflexion et d'analyser : ce qu'est un système RCO et ce que pourrait être le système RCO "idéal". Un système de RCO peut se définir comme un environnement de représentation et de programmation qui permet de résoudre des problèmes posés sur un domaine donné. Un objet est un ensemble de couples attributs-valeurs associé à un identifiant. En général cette identification se fait à l'aide d'un nom. La valeur d'un attribut peut être soit un objet, soit une valeur d'un type primitif du langage (entier, chaîne de caractères, etc.). Les objets sont regroupés en classes (ils sont alors dits instances de cette classe). Les classes permettent de regrouper des traits communs à ces objets [Euzenat, 1999]. Une classe C représente un concept du monde réel. Elle possède une identité (son nom) et décrit un ensemble de propriétés correspondant aux caractéristiques du concept représenté (de façon générale en spécifiant leur domaine de valeur, le type des valeurs, la cardinalité, etc.). La classe hérite, éventuellement en les affinant, des propriétés des classes qu'elle spécialise. Les classes sont donc organisées par niveau d'abstraction par l'intermédiaire de relations comme la spécialisation ou la subsomption et elles partagent des propriétés sous le contrôle du mécanisme d'héritage. La RCO privilégie donc ce lien particulier entre les classes. Des problèmes subsistent comme la multi-généralisation quoique bien traités en programmation orientée objet. Contrairement au modèle original des frames, seules les classes associent des facettes aux attributs et seul un ensemble restreint de facettes est autorisé et clairement identifié. Les facettes servent principalement à : (i) préciser les valeurs des attributs admissibles dans une classe (typage) et (ii) déclarer les mécanismes capables de déduire la valeur d'un attribut manquant (inférence) [Euzenat, 1999].

Dans [Carré *et al.*, 1995], les auteurs tentent de démarquer les systèmes de RCO par rapport aux langages de programmation orientés objet. Voici les différences principales qui ont été signalées.

- La RCO porte peu d'intérêt vis à vis de l'encapsulation. En programmation orienté objet, les attributs sont considérés comme la structure d'implémentation de l'objet sur laquelle repose la réalisation de ses méthodes, ils sont de ce fait encapsulés (privés) dans l'objet.
- La RCO confère aux attributs un statut à part entière d'éléments de connaissance, l'objet se définit avant tout par les attributs qui le caractérisent.
- Méthodes ou réflexes : les réflexes constituent un mécanisme de déclenchement de procédure dont la sémantique opérationnelle est complètement spécifiée. Un réflexe n'est jamais appelé de façon impérative contrairement aux méthodes qui le sont.
- La classe en programmation orienté objet sert de moule à partir duquel s'engendrent les instances (ou objets) qui représentent des entités individuelles. En RCO, les objets élémentaires se regroupent en classes sur les bases de propriétés structurelles et comportementales communes. Les propriétés structurelles consistent à décrire l'état de l'objet. Les propriétés comportementales décrivent par des méthodes ou réflexes les opérations que l'objet est capable d'exécuter [Simon & Napoli, 1999].

Syntaxe

Nous présentons dans ce qui suit un exemple de classe $Personne$ avec la sous-classe $Enfant$, nous empruntons la syntaxe de TROEPS [Euzenat, 1999].

```
<Personne
  attributs = {
     <nom type = chaine;>,
     <prénom type = chaine;>,
     <age type = entier;,
     intervalles =[0,150];>,
     <sexe type = {féminin, masculin};>
     ;}
>

<Enfant
   sorte-de = personne;
   attributs =
   {<age intervalles = [3,10];>
   ;}
>
```

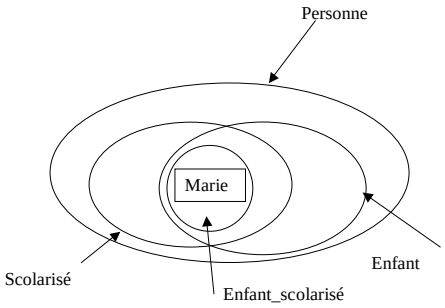

FIG. 3.7 – Représentation ensembliste où Marie est une instance de Enfant_scolarisé

La figure 3.7 présente une interprétation ensembliste de l'objet "Marie" où Marie est une instance de la classe Enfant_scolarisé.
Les constituants d'un système à objet S sont les suivants :

– un ensemble de classes C_S, un ensemble de propriétés P_S et un ensemble d'instances I_S. Les classes sont munies de propriétés structurelles et comportementales -attributs et méthodes- et sont instanciables. Si le système est réflexif, toute entité peut alors être manipulée comme une instance, y compris les relations et les méthodes.
– une hiérarchie H_S : c'est une classification de l'ensemble des classes. Une hiérarchie conceptuelle $H_S = (X, \_, <)$ est un graphe orienté sans circuit, où X est un ensemble de classes, < une relation d'ordre partiel et _ l'élément maximal de X suivant <; _ est appelé racine de

l'hiérarchie. La gestion de l'hiérarchie et le partage de propriétés sont à la charge du mécanisme d'héritage.

Les classes s'organisent aussi autour de relations pour lesquelles il doit être possible de définir un partage de propriétés spécifique.

- chaque classe est décrite en intension Int_S par l'ensemble de ses propriétés.
- chaque classe est décrite en extension Ext_S par un ensemble d'instances qui lui est associé.
- une description de l'état des instances par un ensemble d'expressions. Les expressions de la forme (ipv) associent, à une instance $i \in I_S$ et une propriété $p \in P_S$, une valeur v qui peut être une instance.

A tout système à objets S, est associée une sémantique dénotationnelle[55] qui permet d'interpréter les constituants dans un domaine d'interprétation. Cette sémantique, inhérente à tout système à objets, répond principalement à la préoccupation de fonder logiquement les opérations de raisonnement, rejoignant la démarche des LDs.

Les inférences dans la RCO

De façon générale, le raisonnement hiérarchique ou raisonnement par classification s'appréhende comme une procédure de déduction qui manipule les objets d'un système de RCO. Les opérations principales qui sont à la base du raisonnement hiérarchique sont les suivantes [Simon & Napoli, 1999] :

- la subsomption ;
- la classification : établir la position d'un objet X (classe ou instance) dans une hiérarchie H ;
- la cohérence et instanciation : toute classe possède des instances et pour un objet X on sait déterminer sa classe.

3.4.5 Quel formalisme choisir ?

Les différents formalismes de représentation sont arrivés à maturation et il est difficile de décider quel langage serait le plus adéquat dans l'absolu. Généralement, l'ingénieur de la connaissance utilise le formalisme qu'il maîtrise ou dont il possède un langage implémenté. De plus en plus, ces ingénieurs se sont rendu compte que le choix suivant ces critères est biaisé.

Un souci collectif au sein de la communauté Ingénierie des Connaissances tend à spécifier et à identifier quelle est la procédure à suivre pour choisir tel ou tel formalisme. Dans [Barry *et al.*, 2001], les auteurs tentent de situer les langages (LOOM, DefOnto, OIL, CoGiTo, OCML) suivant leur puissance d'expression, leurs modes d'opérationalisation ainsi que la disponibilité d'environnement d'aide à la conception d'ontologies dans ces langages. La comparaison s'est faite sur la base d'une ontologie « jouet » et a permis de ressortir des disparités mais les auteurs concluent qu'il faudrait porter cette étude sur une ontologie de grande taille ainsi que la généraliser à d'autres langages.

[55] Une sémantique dénotationnelle est donnée par une fonction d'interprétation entre les termes d'un langage et un domaine.

3.5 Utilisation d'ontologies en Recherche d'Information

Les ressources sémantiques (thésaurus, ontologies, etc.) ont un apport considérable pour le traitement des documents textuels ou multimédia. Leur utilisation en Recherche d'Information (RI) peut intervenir lors de la phase de recherche ou lors de la phase d'indexation. La phase de recherche consiste à retrouver les documents les plus pertinents par rapport à une requête donnée (cf. chapitre 1). En général les documents retournés sont ordonnés à l'aide d'une mesure de similarité calculée entre le document et la requête. La phase d'indexation consiste à construire au préalable une structure d'accès aux documents qui facilitera la phase de recherche. Plus la phase d'indexation est sophistiquée, plus la phase de recherche sera facile.

3.5.1 Phase d'indexation

Les documents peuvent être indexés par un groupe de concepts, où on sait qu'un tel document traite des concepts A et B mais on ne connaît pas les relations entre eux dans le texte. Cette méthode permet néanmoins une recherche plus pertinente qu'une recherche par des termes simples, même si les concepts ne sont pas reliés entre eux, les termes référant au même concept le sont.

Une autre méthode attribue à chaque document une description sémantique (ou annotation) où les concepts sont représentés avec leurs relations sémantiques [Alhulou *et al.*, 2003]. Cette représentation confère un grand pouvoir d'expression mais peut par ce fait ralentir les traitements et la construction des descriptions sémantiques associées à chaque document n'est pas une tâche facile.

L'indexation automatique dans les deux cas pose des problèmes notamment celui de l'ambiguïté des termes (homonymie et polysémie) et on a généralement recours à des outils linguistiques de Traitement Automatique des Langues (TAL). Le problème de la désambiguïsation des termes est un vaste champ de recherche encore en expansion. D. Steffen [2003] procèdent à une analyse lexicale : partie du discours, morphologie et analyse de phrases pour l'allemand. Mihalcea & Moldovan [1999] calculent la densité sémantique entre des couples de mots ce qui leur permet de retrouver le sens voulu des termes. Stetina *et al.* [1998] travaillent sur des petits corpus d'apprentissage en considérant les phrases comme unité de contexte. Ils exploitent la probabilité d'apparition des termes dans ces unités ainsi que les liens sémantiques explicites. Mais ces techniques ne résolvent pas totalement le problème et il faut toujours faire le compromis entre la finesse des traitements et la complexité des systèmes.

Gonzalo *et al.* [1998] a proposé une méthode d'indexation de documents s'appuyant sur des concepts d'une base de données sémantique qui a nettement amélioré la recherche (jusqu'à 25% d'amélioration). Indexer les documents par les concepts uniquement peut induire en erreur car les techniques de désambiguïsation ne sont pas complètement fiables et se fonder uniquement dessus risque d'entraîner une perte d'information. [Krovetz, 1997] a montré la nécessité d'indexer par les concepts (i.e. sens des mots) ainsi que par les mots. Nous indexons dans notre système les termes indépendamment du fait qu'ils soient reliés ou non à une ontologie. Les liens sémantiques constituent ainsi un plus mais un terme qui n'est pas relié à l'ontologie peut aussi être retrouvé. Nous montrerons aussi que le problème d'ambiguïté est pris en charge en intégrant la notion de contexte dans nos calculs de similarité.

3.5.2 Phase de recherche

L'intérêt d'utiliser des ressources sémantiques en recherche d'information est de pouvoir retourner, lors d'une recherche par similarité, les documents qui partagent avec la requête le maximum de concepts plutôt que le maximum de mots-clés. Les ontologies ont montré leur efficacité en RI [Andreasen *et al.*, 2003], leur utilité s'est vu confirmé par le web sémantique. Une ontologie permet d'affiner les résultats en réduisant le silence et le bruit [Hernandez & Aussenac-Gilles, 2004].

Expansion de requêtes

Les réseaux sémantiques ont montré leur apport en expansion de requêtes [Lu & Keefer, 1994] [Moldovan & Mihalcea, 2000] [Baziz *et al.*, 2003]. Le but de l'expansion de requête est soit d'élargir l'ensemble de documents retournés ou d'augmenter la précision. Dans le premier cas, la requête peut être étendue en ajoutant des termes similaires (généralement des synonymes) à ceux de la requête. Dans le deuxième cas, les termes peuvent être complètement changés pour reformuler la requête, une technique utilisée dans les retours arrière sur pertinence [Buckley *et al.*, 1994]. La désambiguïsation des termes dans les requêtes influence la qualité du système. La plupart des systèmes utilisent WordNet comme ressource sémantique, ceci est dû au fait que WordNet est parmi les rares ressources accessibles en ligne. [Guarino *et al.*, 1999] ont montré l'apport des ontologies dans leur système OntoSeek en sélectionnant manuellement les synsets de WordNet qu'ils ont rajouté à leurs catégories sur les catalogues de produits et les pages jaunes.

Baziz *et al.* [2003] commencent la phase d'expansion de la requête en lemmatisant les termes de la requête par un étiqueteur. WordNet est ensuite interrogé pour récupérer des termes reliés à la requête par des relations de synonymie, généralisation et spécialisation. Une pseudo-désambiguïsation est entamé pour retrouver le synset adéquat en comptant le nombre de termes de la requête communs avec chacun des synsets. Si plusieurs synsets présentent le même résultat, ceux qui possèdent le plus de mots différents priment. Une seconde phase consiste à détecter des concepts à partir de groupes de mots pour favoriser les concepts « longs » qui servent à plus cerner le sens. Les évaluations sur un corpus issu du programme CLEF2001 (avec 50 requêtes) de la méthode d'expansion intégré dans le système Mercure [Boughanem, 1992] montre que le choix de concepts longs est à favoriser sur les concepts isolés ceci s'explique par la réduction considérable du risque d'erreur. Un poids est ajouté aux termes ajoutés à la requête, ce poids ne doit pas dépasser 0.5. Le niveau des relations rajoutées ne doit pas dépasser 1, la prise en compte du synonyme du synonyme par exemple n'est pas pertinente. Les limites de WordNet ont été soulignés, en effet un terme peut avoir une multitude de synsets rattachés alors qu'un autre terme technique n'en aura aucun.

Nous présentons dans ce qui suit deux exemples de systèmes de recherche d'information : DSIR (*Distributional Semantics based Information Retrieval*), déjà présenté dans la section 1.8.4 du chapitre 1 et XXL (*Flexible XML Search Language*) un langage de requêtes pour recherche d'information structuré. Les deux systèmes utilisent WordNet pour prendre en compte la sémantique dans leurs traitements.

3.6 DSIR

Rappelons que le modèle DSIR (*Distributional Semantics based Information Retrieval*) propose d'étendre le modèle vectoriel en introduisant une représentation des documents qui intègre une **information sémantique**, en utilisant une représentation distributionnelle, fréquentielle et co-fréquentielle de la sémantique d'un terme [Rajman *et al.*, 2000].
Pour améliorer la qualité des représentations des textes, Besançon [2002] propose un espace de représentation défini à partir des concepts. Le but est de regrouper les termes synonymes dans la même entrée de l'index. Cela nécessite un modèle permettant d'associer les termes aux concepts et donc de gérer l'ambiguïté sémantique.
L'intégration des concepts passe par le calcul des co-occurrences entre concepts contrairement au modèle DSIR de base où les co-occurrences sont calculées entre termes. La désambiguïsation se fait également sur la base de ces co-occurrences. L'hypothèse de travail est que les termes se désambiguïsent entre eux [Rajman *et al.*, 2000]. La séquence des concepts à associer à une séquence de termes est celle qui maximise la probabilité d'affecter des concepts aux nœuds du graphe de co-occurrence dérivé de la séquence des termes.
Pour calculer ces probabilités, l'auteur fait les hypothèses suivantes :
– H1 : Un terme est choisi comme instanciation d'un concept, indépendamment du choix des autres termes de la phrase.
– H2 : Un terme est conditionné seulement par son contexte.
Les probabilités calculées par le modèle probabiliste de Champ de Markov caché. La fonction d'appariement reste la même que celle présentée dans la section 1.8.4. Les résultats présentent une amélioration très significative des performances pour un faible rappel.

3.7 XXL

Le langage de requêtes XXL (*Flexible XML Search Language*) reprend un noyau des langages de requêtes tels que XML-QL, XQuery et rajoute l'opérateur ~ : un opérateur de similarité sémantique qui peut s'appliquer aux éléments ou aux termes du corpus. L'évaluation de la requête est fondé sur le calcul de similarité dans une ontologie ainsi que des techniques de pondération des termes.
Le moteur de recherche XXL dispose de trois structures d'index [Schenkel *et al.*, 2005b] :
– *L'index contenu d'élément* : Cet index utilise un fichier inversé et un arbre B+ pour les noms d'éléments. A partir d'un texte, le système est alors capable de retourner les éléments dans lesquels il apparaît. Le système actuel utilise *Oracle Text*[56], les fréquences des termes sont calculés par le $tf - idf$[57].
– *L'index chemin d'élément* : Cet index permet l'accès aux nœuds parents, enfants, descendants et ancêtre d'un nœud donné. Il calcule aussi la distance entre deux nœuds connectés.
– *L'index ontologie* : Le but de cet index est de retrouver des mots reliés sémantiquement à un mot donné. Il calcule pour cela une similarité qui peut être restreinte à un certain type de liens. A partir de cette valeur une mesure de similarité peut être calculée entre deux concepts.

[56] http ://otn.oracle.com/products/text/.
[57] L'unité de base étant l'élément, le *tf-idf* revient alors à calculer le $tf - ief$

Pour cela il faut d'abord désambiguïser les termes. Nous présentons la mesure de similarité ainsi qu'une méthode de désambiguïsation ci-dessous.

L'ontologie est construite à partir de WordNet, elle est défini comme un graphe $O = (V_O, E_O)$, où V_O représente la liste des concepts et E_O la liste des arcs qui représentent les liens sémantiques entre deux concepts. Les concepts et les relations sont extraits de WordNet. Les arcs typés par la relation (hyperonymie par exemple) et sont pondérés par des fréquences fondés sur la corrélation entre concepts à partir de pages retournés par un *crawler* sur le web.

Calcul de similarité

La mesure de similarité sim_p entre deux concepts C_1 (n_0), C_2 (n_k) suivant un chemin $p = (n_0 \dots n_k)$ est calculée de la manière suivante :

$$sim_p(C_1, C_2) = \prod_{i=1}^{|p|-1} poids(n_i, n_{i+1})$$

Où $|p|$ est la longueur du chemin p, $poids(n_i, n_{i+1})$ dénote le poids de l'arc $e = (n, n+1)$. On peut restreindre les types d'arcs autorisés dans un chemin (ne considérer que les hyponymes par exemple).

Le calcul de la similarité entre deux concepts est calculé par :

$$sim(C_1, C_2) = max(sim_p(C_1, C_2))$$

Désambiguïsation sémantique

Un terme peut être rattachés à plusieurs concept (ou *synsets* dans WordNet). Il faut alors décider du sens à fixer au terme.

La désambiguïsation dans XXL est effectuée en calculant la corrélation d'un contexte d'un terme donné avec le contexte du concept qui peut être rattaché à ce terme. Le contexte d'un mot est l'ensemble de termes co-occurents dans un document ou dans la requête. Le contexte d'un concept est la liste des termes reliés aux concepts proches dans l'ontologie (comme les hypéronymes et les homonymes). La corrélation est calculé par le $cosinus$ du vecteur des termes du contexte du terme à désambiguïser et le vecteur du concept potentiel.

Voici un exemple de requête qui cherche à retrouver les titres des articles scientifiques concernant l'échange d'informations et XML :

```
SELECT $T
FROM INDEX
WHERE ~ article AS $A
AND $A/~ title AS $T
AND $A/#/~ section ~ "Information exchange & XML"
```

L'évaluation de l'opérateur de similarité est ~ retourne la liste des termes les proches dans l'ontologie, c'est-à-dire ceux qui ont la plus grande similarité avec le terme recherché. Pour l'élément $article$ par exemple $paper$ peut être retourné par une similarité de 0.9. Au niveau des éléments, la pertinence des éléments retournés sont multipliés par la similarités sémantique, $pertinence(article) = 0.9 * pertinence(paper)$ par exemple.
Les termes de la requête sont remplacés par la disjonction entre eux et les termes reliés (ayant une forte similarité), ainsi *"Information exchange & XML"* devient *("Information exchange"* | *"data exchange"* | *"heterogeneous data")& ("XML"* | *"semistructed data").*
Dans [Schenkel *et al.*, 2005b], les auteurs précisent qu'ils construisent leur ontologie à partir de WordNet. Malheureusement ils ne précisent pas quelle pondération est affectée aux concepts complexes (avec des termes composés) ni comment cette pondération est prise en compte pour le calcul du score. De plus, les termes synonymes présentés dans l'exemple ne figurent pas dans WordNet. Les résultats sont encourageants et montrent une amélioration avec la prise en compte de la sémantique.

3.8 conclusion

Dans ce chapitre nous avons défini les ontologies dans le cadre de l'ingénierie des connaissances. Les ontologies en RI connaissent un cadre moins formel mais présentent un intérêt certain pour pallier les problèmes de bruit et silence dus à la richesse de la langue. Les ressources utilisées comme WordNet ne peuvent pas être qualifiées de formelles mais présentent par leur accessibilité un intérêt certain pour la RI. La principale lacune d'une telle ressource est sa généralité qui empêche son utilisation pour un domaine de spécialité. La difficulté majeure pour l'utilisation d'une ontologie dans le cadre d'une RI est la désambiguïsation sémantique qui a une influence directe sur la qualité des résultats obtenus.
L'apparition du Web Sémantique et d'un cadre unificateur pour proposer de nouveaux standards a vivifié l'intérêt de l'utilisation des ontologies.
Nous présentons dans le chapitre qui suit le Web Sémantique, ses standards et les nouvelles possibilités d'une recherche d'information sémantique dans ce cadre.

Chapitre 4

Web Sémantique

Sommaire

4.1 Introduction

Selon la vision de Berners-Lee [1999] Berners-Lee *et al.* [2001] le Web Sémantique est le web du futur dans lequel les informations possèdent un sens explicite facilitant ainsi aux machines leur traitement et leur intégration. Le Web sémantique offre une infrastructure qui permet une utilisation de connaissances formalisées en plus du contenu informel du web actuel. Le W3C

présente une architecture en couches du Web sémantique (voir figure 4.1) qui s'appuie sur une pyramide de langages dont les couches de base sont les plus stabilisées. L'architecture repose sur la notion d'URI (*Uniform Resource Identifier*) qui permet de localiser une ressource sur le Web (ou dans d'autres domaines) en lui attribuant un identifiant unique. XML et XML-Schema représentent la couche syntaxique à laquelle vient s'ajouter les langages qui permettent d'annoter les ressources (RDF) et de définir le vocabulaire des annotations (RDF-S).
La pyramide du W3C inclut aussi la possibilité de faire des déductions on pourra s'appuyer sur la standardisation d'un langage de règles, comme RuleML (*Rule Markup Language*) encore en cours de maturation. A plus long terme la capacité de produire des preuves des déductions faites pourra augmenter le niveau de confiance des utilisateurs dans ces déductions.

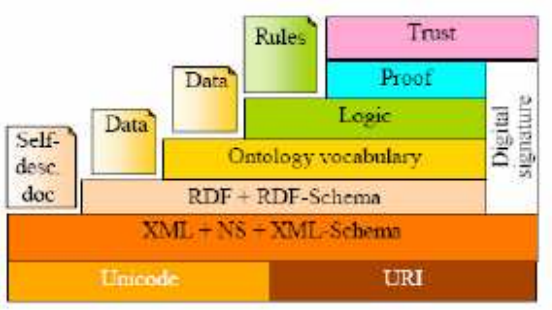

FIG. 4.1 – Les couches du Web sémantique selon le W3C

Même si l'infrastructure du Web sémantique ne se réduit pas à ses couches, cette structuration permet de présenter les efforts faits notamment en standardisation pour permettre une utilisation uniforme des outils. Le Web sémantique promet de transformer le Web en un espace d'information sémantique qui rend l'information accessible par son sens (d'une manière sémantique) aux machines ou agents. La thématique du Web sémantique regroupe autour d'elle des communautés de recherche différentes (Recherche d'Information, Représentation et Ingénierie de Connaissances, Bases de Données, Apprentissage Automatique, etc.), comme en témoigne l'Action Spécifique du département STIC du CNRS qui a débuté en Novembre 2001[58] ainsi que la profusion des listes de diffusion[59]. Suivant la structure en couches présentée dans la figure, plusieurs standards ont vu le jour. Nous présentons ces standards qui constituent le fondement du Web sémantique.
RDF est un modèle de données dédié à la description des ressources et des relations entre ces ressouces, en fournissant une sémantique simple. RDF peut être sérialisé en XML, ce qui facilite l'échange de données.
RDF-S (*RDF Schema*) est un langage de spécification des schémas associés à RDF.
OWL est un langage de représentation d'ontologies sur le web. Il possède plus de primitives que RDF(S).

[58] http ://www.lalic.paris4.sorbonne.fr/stic/as5.html
[59] Liste de diffusion internationale : semanticweb@yahoogroups.com, liste de diffusion française :websemantique@xmlfr.org

Nous nous intéressons dans ce chapitre à l'aspect données dans le Web Sémantique et leur utilisation dans le cadre d'une recherche d'information guidée par la sémantique. Ainsi nous ne présentons pas les technologies nécessaires pour la mise en place des services web sémantiques (UDDI, WSDL, etc.).

Nous discutons des problématiques que pose le Web sémantique par son aspect dynamique, distribué et hétérogène. Le web sémantique est fondé sur la notion de méta-donnée et d'annotation. Nous commençons par donner une définition de ces deux termes.

4.2 Méta-données et annotations

Une méta-donnée est littéralement une donnée sur une donnée. Plus précisément, c'est un ensemble structuré (ou pas) de données décrivant une ressource quelconque. Une méta-donnée peut être utilisée à des fins diverses (la description et la recherche de ressources, la gestion de collections de ressources, la préservation des ressources, etc.). Ces informations peuvent être évidentes (tels que l'auteur, la date de publication, l'éditeur d'un livre, etc), ou plus complexes et moins aisément définies ainsi que la gestion des avis d'un collectif de lecture d'un article nécessitent une structure de méta-données évoluée capable d'annoter des portions de cet article, et cela, de façon multiple. Les méta-données sont particulièrement importantes pour les ressources (images ou vidéos) qui, sans elles, peuvent demeurer pratiquement inexploitables et difficiles à retrouver.

Plusieurs standards concernant les méta-données existent (MARC : *MAchine-Readable Cataloging* - pour la description des ouvrages, LOM : *Learning Object Metada* - pour la description des ressources liées à l'éducation). Ces standards sont orientés méta-données « métier ». Le *Dublin Core Metadata* Initiative (DCMI) définit un ensemble de méta-données communes à diverses communautés. Il comporte un ensemble minimal de méta-données qui ont trait :

– au contenu : Title, Description, Subject, Source, Coverage, Type, Relation.
– à la propriété intellectuelle : Creator, Contributor, Publisher, Rights.
– à la version : Date, Format, Identifier, Language.

Une annotation est une information graphique ou textuelle attachée à un document [Desmontils & Jacquin, 20
C'est une note critique ou explicative accompagnant un document. Desmontils & Jacquin présentent les différentes dimensions d'une annotation suivant son utilisation, le formalisme[60] dans lequel elle est décrite et son rôle, etc. Dans le cadre du Web sémantique on parle d'annotations sémantiques (dans le sens formel). Dans ce cadre on essaie de représenter le contenu du document par une description formelle. Annoter des documents c'est les décrire par des méta-données. Les termes annotation et méta-donnée sont souvent utilisés indifféremment. Une discussion sur la possible différenciation par rapport au caractère plus situé[61] de la ressource est proposée dans [Charlet *et al.*, 2003].

[60] ou l'absence de formalisme pour les annotations informelles
[61] Le caractère situé à l'intérieur ou à l'extérieur des documents.

4.3 Description des ressources

Les DTDs et les schémas XML (cf. chapitre 2) peuvent être utilisés pour l'échange de données (entre des communautés qui ont d'abord convenu d'un vocabulaire). Cependant, l'apport de XML reste essentiellement au niveau syntaxique, son manque de sémantique ne permet pas de créer de nouveaux vocabulaires XML. Le même terme peut être utilisé avec différents sens dans différents contextes (polysémie) et des termes différents peuvent être utilisés pour des documents ayant le même sens. RDF et RDF-S ont essayé de résoudre ce problème en permettant d'associer des sémantiques simples aux ressources. Avec RDF-S, une personne peut définir des classes hiérarchisées ainsi que des propriétés pouvant avoir des sous-propriétés, associées à des domaines ou des intervalles. Nous présentons dans ce qui suit RDF et RDF-S ainsi que TopicMaps (une alternative à RDF).

4.3.1 RDF : *Resource Description Framework*

RDF (*Ressource Description Framework*) a été développé par le W3C. La première ébauche de RDF a été publiée en octobre 1997, et les spécifications sont devenues une recommandation du W3C en février 1999. La dernière mise à jour comprenant des changements dans la terminologie, la syntaxe et les concepts manipulés, date de Février 2004 [February, 2004]. RDF est un langage de représentation des ressources du web, et en particulier les méta-données (telles que le titre, l'auteur, la date de modification d'une page, etc.). Un document RDF est un ensemble de triplets de la forme <sujet,prédicat, objet>. Où le sujet est la ressource à représenter, le prédicat est la propriété et l'objet est la valeur de cette propriété. Les éléments de ces triplets peuvent être des URIs, des littéraux ou des variables. Cet ensemble de triplets peut être représenté de façon naturelle par un multigraphe orienté étiqueté où les éléments apparaissant comme sujet ou objet sont des sommets, et chaque triplet est représenté par un arc dont l'origine est son sujet et la destination est son objet (voir figure 4.2). Ce document peut être sérialisé en machine par un document RDF/XML (pour faciliter l'échange de données) ou N3, mais est souvent représenté sous une forme graphique.
Voici un exemple des différentes représentations[62]. Nous voulons décrire le fait que *Emile Ajard est l'auteur d'un livre qui coûte 12Eur*.

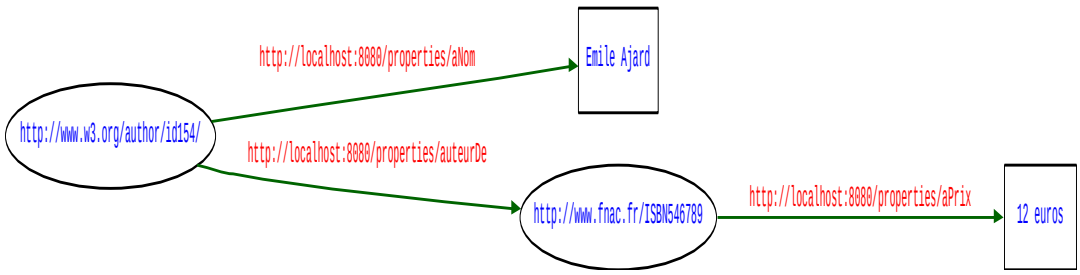

FIG. 4.2 – Exemple de graphe RDF

Exemple de la notation RDF/XML

[62] généré par RDF Validator http ://www.w3.org/RDF/Validator/ à partir de la notation RDF/XML

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:s="http://localhost:8080/properties/">
  <rdf:Description rdf:about="http://www.w3.org/author/id154/">
    <s:aNom>Emile Ajard</s:aNom>
    <s:auteurDe>
      <rdf:Description rdf:about="http://www.fnac.fr/ISBN546789">
         <s:aPrix> 12 euros </s:aPrix>
      </rdf:Description>
    </s:auteurDe>
  </rdf:Description>
</rdf:RDF>
```

Exemple de notation N3

```
@prefix s: http://localhost:8080/properties //déclaration des
espaces de noms en N3

<http://www.w3.org/author/id154> <s:aNom> <Emile Ajard>
<http://www.w3.org/author/id154> <s:auteurDe>
<http://www.fnac.fr/ISBN546789>
<http://www.fnac.fr/ISBN546789> <s:aPrix> <12Eur>
```

Une assertion peut aussi être considérée comme une ressource. Cette réification rend possible de faire des assertions sur des assertions. Les déclarations réifiées sont des instances de la classe *rdf :statement* et sont décrit par les propriétés suivantes : *rdf :subject*, *rdf :predicate* et *rdf :object* pour réifier une assertion à une ressource. Ainsi on peut exprimer que le livre coûte 12Eur dans le cadre du salon du livre. En plus de ces primitives, RDF définit trois types de conteneurs pour représenter des collections de ressources ou de littéraux :
– Bags : listes non ordonnées de ressources,
– Sequences : listes ordonnées de ressources,
– Alternatives : un choix parmi une liste de ressources.
Même si RDF est une amélioration (« au niveau sémantique »), il reste insuffisant comme langage de représentation de connaissances. Il manque de primitives et possède peu de mécanismes pour gérer l'évolution des schémas.

4.3.2 RDF-S : RDF Schema

RDF Schema est un langage de spécifications de schéma associé à RDF. RDF-S offre une représentation du vocabulaire défini sur RDF pour permettre la modélisation des objets du modèle. Les primitives RDF-Schema les plus pertinentes sont : *rdf :ressource* qui est la classe la plus générale. Elle possède deux sous classes *rdfs :class* et *rdfs :property*. En spécifiant un schéma spécifique à un domaine en RDF-S, les classes et propriétés vont devenir des instances de ces deux classes. *rdfs :class* dénote l'ensemble des classes (dans un sens orienté objet), idem pour *rdfs :property* où chaque propriété sera une instance de cette classe. *rdfs :subClassOf* définit la relation de sous-classe entre classes. Cette relation est transitive donc si A est sous-classe de B et B est une sous-classe de C Alors A est une sous-classe de C. RDF-S interdit la réflexivité de

cette relation ainsi que l'héritage cyclique, une classe ne peut pas être sous-classe d'elle-même ni d'une classe fille. *rdfs :subPropertyOf* définit à l'instar de *rdfs :subClassOf* une hiérarchie entre les propriétés. RDF-S permet aussi de définir le domaine et le type des propriétés par *rdfs :domain*, *rdfs :type*, *rdfs :range*.
RDF-S est doté d'une sémantique en théorie des modèles, il ressemble en cela aux graphes conceptuels. S'il permet de représenter des ontologies simples, il reste néanmoins pas assez expressif. Une mutualisation des efforts au sein du W3C a conduit à la définition de OWL que nous présentons plus loin.

4.3.3 TopicMaps

Les TopicMaps (cartes Topiques) [Biezunski & Newcomb, 2001] sont un standard ISO. C'est une proposition concurrente à RDF(S). Elles comprennent quatre primitives :
– Les topics qui peuvent être n'importe quel sujet ou entité.
– Les noms donnés aux topics
– Les occurrences qui sont des ressources, disposant d'une URI, qui peuvent être liées à des topics.
– Les portées permettent de spécifier le contexte dans lequel une relation est valide.
Une syntaxe XML existe depuis 2001 sous le nom XML Topic Maps (XTM) [Pepper & Moore, 2001].
Les Topic Maps ne disposent pas d'une sémantique claire ce qui peut procurer une liberté d'utilisation mais aussi limiter ses applications. Une des particularités des Topic Maps, par rapport à RDF(S) du W3C, est la notion de « scope » qui permet de définir des contextes différents dans lesquels les éléments nommés identiquement peuvent avoir des significations différentes. Par contre, la notion de hiérarchies de classes n'existe pas dans les Topic Maps. Les Topic Maps permettent aussi bien l'expression de méta-données que l'exploitation des relations entre éléments pour différentes tâches, par exemple l'aide à la navigation sur le Web sémantique.

4.4 Ontologies et Web sémantique

Les ontologies facilitent le partage d'information entre les communautés des humains et des agents logiciels. Elles jouent, de ce fait, un rôle clé dans la mise en place du web sémantique.
Les langages suivants constituent aussi des langages de description de ressources mais avec un niveau logique. C'est pourquoi nous les classifions comme langages de définition d'ontologies. Nous présentons OWL (*Ontology Web Language*), un langage de représentation d'ontologies sur le Web. Nous présentons d'abord les deux langages qui l'ont précédé et qui ont permis sa mise en place.

4.4.1 OIL et DAML

Le projet on-To-knowledge est un projet européen qui a été à l'origine de la proposition du langage OIL (*Onntology Inference Layer*) qui combine les primitives des langages de frame avec les inférences des logiques de description. OIL dispose d'une syntaxe XML et définit ses primitives en RDF-S.

En parallèle la DARPA a lancé le programme de recherche DAML (*DARPA Agent Markup*) pour développer des langages qui facilitent aux machines l'accès au contenu des documents. La fusion de DAML et OIL (DAML+OIL) [Patel-Schneider *et al.*, 2002] a constitué le point de départ au W3C pour la définition de OWL.

4.4.2 OWL : *Web Ontology Language*

Le langage OWL (*Web Ontology Language*) [2004, 2004] a été conçu pour ajouter plus de sémantique à RDF et RDF-S ainsi que des mécanismes d'inférence. C'est un nouveau standard du Web sémantique. OWL possède des sous-langages avec des expressivités différentes et des pouvoirs d'inférence en conséquence (voir figure 4.3) : OWL LITE, OWL DL et OWL FULL.

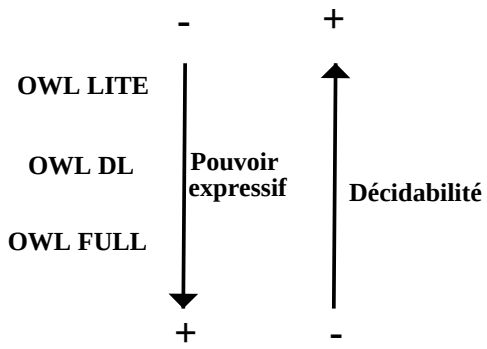

FIG. 4.3 – Pouvoir expressif vs. décidabilité pour OWL

– OWL LITE : est utile aux applications qui ont besoin des hiérarchies de classifications et des caractéristiques de contraintes simples.
– OWL DL : est utile aux applications qui demandent un maximum d'expressivité tout en maintenant la complétude et la décidabilité. OWL DL contient tous les constructeurs du langage OWL mais avec des restrictions (par exemple, l'héritage multiple est interdit).
– OWL FULL : destiné aux applications qui demandent un maximum d'expressivité avec la liberté syntaxique de RDF sans aucune garantie de calculs. Il relaxe les contraintes de OWL DL et de ce fait viole certains raisonnements des logiques de description.

Pour ces trois sous langages, seulement les deux premiers maintiennent les tâches d'inférence principales : satisfiabilité/classification. Chaque sous-langage est construit sur celui qui le précède et de ce fait :
– Une ontologie OWL LITE est une ontologie OWL DL.
– Une ontologie OWL DL est une ontologie OWL FULL.
– Un raisonnement dans OWL LITE est aussi valide dans OWL DL.
– Un raisonnement dans OWL DL est aussi valide dans OWL FULL.

L'inverse n'est, bien sûr, pas vrai.
OWL FULL peut être vu comme une extension de RDF tandis que OWL LITE et OWL DL peuvent être vues comme une restriction de RDF. Tout document OWL est donc un document RDF mais seulement un sous-ensemble de documents RDF est un document OWL DL ou LITE.

OWL LITE

OWL LITE assure fondamentalement la classification ainsi que quelques contraintes simples. Par exemple, il prend en compte les cardinalités mais les restreint à 0 ou 1. *owl :minCardinality*, *owl :maxCardinality* spécifient les cardinalités minimales et maximales et *owl :cardinality* exprime une seule cardinalité si la valeur min et max coïncident. OWL reprend tous les constructeurs RDF ainsi que le vocabulaire de RDF-S à savoir *rdfs :class*, *rdfs :subClassOf*, *rdfs :property*, *rdfs :subPropertyOf*, *rdfs :domain* et *rdfs :range*. Les classes *owl :ObjectProperty* et *owl :DatatypeProperty* sont des sous-classes de la classe *rdfs :property*. OWL LITE permet d'exprimer des caractéristiques sur les propriétés.

– *owl :inverseOf* permet d'exprimer qu'une propriété est l'inverse d'une autre.
– *owl :TransitiveProperty* permet de spécifier qu'une propriété est transitive.
– *owl :SymetricProperty* exprime la symétrie d'une propriété.
– *owl :FunctionalProperty* permet d'exprimer qu'une propriété a au plus une valeur unique et *owl :InverseFunctionalProperty* exprime que l'inverse de cette propriété.

Pour exprimer des contraintes sur les propriétés OWL fournit *owl :allValuesFrom* qui exprime la quantification universelle et *owl :someValuesFrom* la quantification existentielle. OWL LITE permet d'exprimer des relations d'égalité ou de différence entre

– Individus, par *owl :sameAs*, *owl :differentFrom* et *owl :allDifferent*.
– Classes, par *owl :equivalentClass*.
– Propriétés, par *owl :equivalentProperty*.

OWL permet aussi d'exprimer l'intersection par *owl :intersectionOf* mais en limite l'usage.

OWL DL

OWL DL inclut tous les constructeurs du langage OWL mais avec certaines contraintes. Une séparation de type est nécessaire, une classe par exemple ne peut pas être aussi un individu ou une propriété. Il lève certaines contraintes de OWL LITE comme la restriction sur les cardinalités. Voici les constructeurs de OWL DL (mais aussi de OWL FULL) :

– *owl :oneOf* permet de décrire une classe par son extension (la liste de ses individus).
– *owl :disjointWith* permet d'affirmer que deux classes sont disjointes (n'ont aucune valeur commune).
– *owl :unionOf* permet d'exprimer qu'une classe est l'union de deux autres.
– *owl :ComplementOf* permet d'exprimer qu'une classe est le complément d'une autre.
– *owl :hasValue* que la valeur d'une propriété est un individu.

OWL FULL

OWL FULL reprend tous les constructeurs d'OWL DL mais il les interprète de manière plus large. Il relâche toutes les contraintes, une classe par exemple peut être vue comme un individu (intension) ou une collection d'individus (extension). Il permet aussi de rajouter du sens au vocabulaire prédéfini de RDF ou OWL.

Ce gain en expressivité détériore la complexité. En effet OWL Full n'est pas décidable.

4.5 Recherche d'Information dans le Web Sémantique

La recherche d'information sur le Web partage les mêmes objectifs qu'un recherche d'information classique[63]. Le but est de retrouver des résultats pertinents en réponse à une requête posée par un utilisateur. La nature même du Web (hétérogène, dynamique, distribué) impose de définir des critères supplémentaires :

– Le contexte : déterminé par le profil de l'utilisateur, son mode de recherche (navigation, fouille, etc.), son domaine d'intérêt, etc.
– La qualité de l'information retrouvée qui pose la question du temps de réponse, de la validité des réponses, leur "fraîcheur" et impose la gestion des versions des différents documents.

Nous avons discuté dans le chapitre précédent du rôle que peuvent jouer les ontologies dans le processus de RI. Dans le cadre du Web Sémantique et grâce à leur aspect *formel*, les ontologies promettent d'améliorer les techniques d'accès à l'information. En effet, on aura accès au *contenu sémantique* des documents grâce aux annotations et on pourra retourner des informations qui n'étaient pas directement accessibles grâce aux inférences.
La figure 4.4 présente une architecture possible pour les applications de RI "sémantique" [Léger *et al.*, 2002].

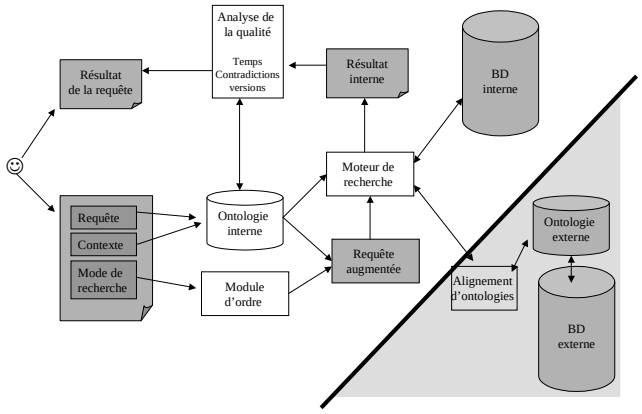

FIG. 4.4 – Architecture possible pour les applications de RI sémantique

Nous présentons dans ce qui suit deux moteurs de recherche qui se fondent sur une ontologie formelle : Corese et OWLIR (*OWL Information Retrieval*).

4.5.1 Corese

Corese [Corby *et al.*, 2004] est un moteur de recherche pour le Web sémantique fondé sur une ontologie. Il permet de retrouver des ressources web annotées en RDF(S).
L'ontologie est représentée en graphe conceptuel au dessus de RDFS. Le moteur de recherche prend en considération la subsomption des concepts et des relations pour calculer un appariement entre la requête et les annotations. La requête est traduite en graphe conceptuel et

[63] Les documents sont généralement fixes, au contraire des pages web qui sont en perpétuel changement.

l'appariement correspond à la projection de ce graphe sur les graphes conceptuels relatifs aux documents.
Le système propose des règles RDF intégrées dans le moteur d'inférence. Les règles sont appliquées aux annotations avant le traitement de la requête, les annotations sont donc enrichies par les conclusions des règles.
Corese implémente une méthode de recherche approchée qui permet de retourner des réponses approximatives quand par exemple l'utilisateur pose une requête avec des termes qui n'ont pas servi à annoter les ressources. Ceci permet d'éviter de retourner un résultat vide. L'évaluation des réponses est faite en calculant une distance sémantique dans l'ontologie.
La distance entre une ressource web et une requête est la somme des distances entre les types de concepts de la requête et les types de concepts cibles. La distance entre deux concepts type est la somme des distances entre chaque concept et son super type commun. La longueur d'un arc entre un fils et un père de profondeur d est de $\frac{1}{2^d}$.
Le langage de requête est basé sur le modèle de triplets de RDF et traduit un graphe conceptuel. Par exemple la recherche ci-dessous recherche les personnes (1) ayant un nom (2) qui sont les auteurs (3) d'une thèse (4) qui a un titre (5) qui contient le mot "web" (6).
(1) ?p rdf :type c :person
(2) ?p c :name ?n
(3) ?p c :author ?doc
(4) ?doc rdf :type c :Thesis
(5) ?doc c :Title ?title
(6) ?title ~ "web"

Le système permet aussi d'utiliser les variables pour désigner des propriétés dans une requête ainsi que de rechercher des chemins de longueur supérieure à 1 reliant deux ressources. Le système de recherche raisonne essentiellement à partir des annotations. On n'interroge pas les documents mais leurs annotations. L'efficacité du système est alors dépendante de la qualité de ces annotations. Le principal inconvénient est de perdre l'information textuelle des documents.

4.5.2 OWLIR

OWLIR [Mayfield & Finin, 2003]est un système de recherche d'information pour le web sémantique. Le système veut combiner les techniques de recherche d'information efficaces et les mécanismes d'inférence du Web Sémantique. Les requêtes peuvent alors être soit classiques dans le sens RI (avec possibilité d'intégrer des balises), soit des faits à prouver. L'application actuelle est dans le domaine des annonces d'événements dans une université. OWLIR est composé d'un ensemble d'ontologies et de mécanismes de recherche d'information hybrides. OWLIR définit une ontologie représenté en DAML+OIL ou OWL qui permet de décrire les intérêts des utilisateurs pour différents événements à l'université. Ces ontologies sont utilisées pour annoter les documents d'annonce d'événements. La plateforme comprend : (1) un outil d'extraction de texte, (2) un outil d'annotation et (3) des mécanismes d'inférence.
L'extraction d'information est effectuée par AeroText^{TM} qui extrait des phrases clés. Ces phrases sont ensuite traduites automatiquement en triplets RDF qui utilise une syntaxe OWL. Ces annotations sont augmentées par le système d'inférence. Ainsi si un document est annoté par le triplet suivant [Shah *et al.*, 2003] :

_ :j00255 owlir :moviegenre "action"
Le système rajoute, par des règles spécifiques au domaines, l'annotation suivante :
_ :j00255 owlir :movie ;dc :title "spiderman"
Un moteur de recherche indexe ces annotations ainsi que le texte des documents. Les annotations sont alors indexés de la même manière que le texte.
Les résultats des expérimentations préliminaires semblent prometteurs [Mayfield & Finin, 2003], les auteurs envisagent l'utilisation d'une base de connaissances scalable.
L'intérêt de ce système est de combiner l'information textuelle avec les annotations sémantiques du document ; contrairement à Corese, ce système profite du contenu textuel des documents, même si on regrette que les expérimentations ont été effectuées sur des documents très courts. Dans le cas de documents plus longs, nous pensons que le système risque d'être moins efficace. En effet, l'indexation des annotations dans le moteur de recherche est effectuée comme pour l'indexation des textes, dans le cas de textes longs, les annotations risquent de se disperser dans le vocabulaire d'indexation et il serait moins évident de retrouver les associations entre texte et annotation.

4.6 Problématiques autour du Web sémantique

Le développement du Web sémantique comme un méta-Web fondé sur le World Wide Web va nécessiter un grand nombre d'ontologies du domaine. De nouvelles méthodes d'ingénierie sont nécessaires parce que (i) les ontologies seront développées non seulement par des ingénieurs de la connaissance mais aussi par des experts de domaine, (ii) les ontologies développées pour le Web sémantique utilisées par les machines et les humains nécessitent de nouveaux paradigmes. Les recherches en construction d'ontologies (cf. chapitre 3) ont atteint une certaine maturation mais n'ont pas été réalisées dans une optique de Web sémantique. Des questions restent posées : Comment générer, analyser, modifier, et maintenir les ontologies dans le temps.

4.6.1 Construction d'ontologie

Plusieurs éditeurs d'ontologies existent : Protégé2000, OntoEdit, OILEd. Ce sont des éditeurs graphiques qui permettent à l'utilisateur de construire des ontologies. Dans le cadre ouvert du Web où tout le monde peut ajouter du contenu, les utilisateurs « naïfs » ont besoin d'outils semi-automatiques et d'une approche simple pour définir leur propre ontologie ou en ré-utiliser d'autres. D'autres types d'ontologies s'imposent comme par exemple les ontologies coopératives, les ontologies distribuées, etc.
Le processus de développement des ontologies a toujours été considéré comme un processus coopératif qui implique plus d'un sujet humain. L'acceptation d'un vocabulaire consensuel partagé est un aspect important. L'ingénierie des ontologies coopératives fondée sur une architecture client/serveur est une tâche complexe et a de ce fait été peu abordée. VerticalNet [Das *et al.*, 2001] est un outil commercial qui permet de supporter des ingénieries d'ontologie multiples. La collaboration a aussi besoin de contrôle de verrouillage et une gestion des versions et de systèmes sécurisés.
Des vues spécifiques doivent aussi être mises en place. Pour représenter ces vues on a besoin de la mise en place d'ontologies distribuées où les connaissances explicites sont échangées

par un appariement avec l'ontologie. L'approche des ontologies fédérées ressemble à celle des multi-bases en bases de données. La question de l'interopérabilité sémantique reste posée. A l'état actuel des recherches il est impossible d'envisager un apprentissage complètement automatique des ontologies. Une coopération entre les êtres humains (ingénieur de la connaissance, utilisateur) et les machines est nécessaire, pour un apprentissage semi-automatique.
L'apprentissage d'ontologies tend à développer des méthodes et outils qui réduisent la tâche manuelle d'ingénierie et gestion des ontologies. Les ontologies devront être générées et maintenus par des experts du domaine et non plus par des ingénieurs de la connaissance [Maedche & Staab, 2001]. Des outils doivent être fournis pour les aider dans cette tâche.
Il faut aussi permettre l'accès aux ontologies développées pour le Web sémantique. Si les ontologies ne sont pas accessibles (via une recherche sur le web par exemple), les utilisateurs risquent d'être découragés d'annoter leurs pages ou seront obligés de créer de nouvelles ontologies.

4.6.2 Passage à l'échelle

L'application des ontologies au web nécessite un passage à l'échelle. Il est important de le prendre en compte en terme de taille d'ontologies comme du nombre d'utilisateurs connectés simultanément. De plus les applications peuvent avoir besoin de différents niveaux d'accès à des portions d'ontologies. Il est donc important d'avoir un environnement qui n'expose que des portions fondées sur un modèle de sécurité. Le mode de sécurité devra supporter l'accès en lecture et écriture.

4.6.3 Annoter les documents

Quelques outils d'annotation qui génèrent des méta-données sémantiques existent déjà. OntoAnnotate [Staab & Maedche, 2001] inclut des outils pour une annotation manuelle ou semi-automatique des pages. Annotea [J. Kahan & Swick, 2001] offre un balisage RDF mais ne permet pas d'intégrer des outils d'extraction d'information ou une possibilité de rattachement à une ontologie. SHOE [Heflin & Hendler, 2000] fût l'un des premiers systèmes à proposer une annotation sémantique des pages Web. Il permet aux utilisateurs d'annoter les pages via des ontologies accessibles par des URIs ou localement ainsi que faire des raisonnements sur ces annotations par SHOE Search.
Les travaux de [Lerman *et al.*, 2001] [N. Kushmerick & Doorenbos., 1997] tentent d'annoter les pages automatiquement par des algorithmes d'apprentissage mais restent tributaires d'une longue phase d'apprentissage. [Li *et al.*,] proposent une annotation automatique fondée sur une ontologie du domaine. Dill *et al.* [2003] proposent *SemTag and Seeker*, un outil d'annotation automatique du Web en se fondant sur une base de connaissance ATP (avec une taxonomie lexicalisée des entités tels que le sport, la santé, etc.) ainsi qu'un nouvel algorithme de désambiguïsation. Ils affichent d'excellents résultats en annotant 264 millions de pages Web, 434 millions annotations ont été générés. Il reste à tester la validité d'une telle méthode pour des textes spécialisés dans un Web sémantique par domaine (ou communautaire). Les annotations doivent-elle porter sur les documents ou dépendre des usages ? Sont-elles crées par les concepteurs des ressources ou par les utilisateurs ? Le Web Sémantique repose sur la possibilité d'annoter sémantiquement les documents. Cette tâche ne peut pas être manuelle. De plus, que faire

des documents qui existent actuellement. Il est nécessaire de s'aider d'outils d'extraction d'information pour définir des annotations pertinentes. D'autres méthodes s'intéressent à rattacher les ontologies aux schémas ou DTDs [G. Giraldo, 2002] [Alhulou *et al.*, 2003].

4.6.4 Retrouver une ontologie adéquate

La multiplication des ontologies en ligne donnera un large choix pour annoter les ressources. Cette (possible) profusion pose aussi le problème de l'adéquation de l'ontologie. Face à deux (ou plusieurs ontologies) d'un même domaine, il s'agit de distinguer l'ontologie la plus adéquate. Des outils de validation de l'utilité d'une ontologie plutôt qu'une autre sont nécessaires. [Hernandez & Aussenac-Gilles, 2004] proposent des mesures numériques pour évaluer l'adéquation d'une ontologie par rapport à un corpus.

4.6.5 Fusionner plusieurs ontologies

Annoter les documents grâce à des ontologies suppose qu'elles elles existent dans la durée et donc qu'elles soient maintenues et développées. De plus, il peut être nécessaire de fusionner plusieurs ontologies ou portions d'ontologies car un système est amené à être interconnecté avec divers systèmes. Il devient important de gérer cette évolution en fusionnant les concepts, ou les décomposant, etc. L'intégration et la fusion d'ontologies devient cruciale.
Des mesures de similarité sémantique peuvent être utiles dans ce contexte pour évaluer la similarités entre différentes parties des ontologies [Despres & Szulman, 2005]. Nous projetons de tester notre mesure de similarité et la comparer à d'autres existantes comme celle de [Euzenat & Valtchev, 2004].

4.6.6 Gérer les versions

Dans le cadre du Web sémantique, les ontologies ont des durées de vie plus longue et doivent évoluer en fonction des ressources et des utilisations qui en sont faites. La gestion de versions des ontologies est difficile, il serait intéressant d'envisager des périodes où l'ontologie peut être dans un état incomplet ou incohérent et pouvoir porter ces incohérences à l'utilisateur. [Heflin & Hendler, 2000] mettent le point sur les difficultés à gérer des ontologies dans des environnements dynamiques, distribués et hétérogènes.
Plusieurs ontologies pourront être accessibles sur le net et on peut être amené à en utiliser plusieurs simultanément pour une même application. Il faudrait s'assurer de la cohérence et la consistance de l'ontologie obtenue. Des choix s'imposent quand au degré de granularité, le choix des concepts à garder ou à éliminer.

4.7 Conclusion

L'émergence du Web Sémantique a accru l'intérêt pour les ontologies. La construction d'une ontologie générale est désormais reconnue utopique et les travaux semblent s'orienter de plus en plus vers un web sémantique par domaine, comme l'a confirmé la tenue du premier Workshop

du Web Sémantique Médical [WSM, 2003], où l'accent a été mis sur la nécessité des ontologie du domaine.
Le Web Sémantique fédère des recherches de diverses disciplines. Sa propagation tient à l'acceptation des standards et la facilitation de leur mise en œuvre. Les ontologies formelles dans le cadre du Web Sémantique promettent de nouvelles possibilités d'accès à l'information. La construction de telles ontologies ainsi que l'annotation des documents sont deux tâches importantes dont dépend le bon fonctionnement du système.
Nous proposons un système de Recherche d'Information qui intègre l'ontologie dont tout le processus de RI. Il n'en reste pas pour autant tributaire, l'indexation ne repose pas seulement sur les annotations mais aussi sur les documents. Les termes qui ne sont pas reliés à l'ontologie ne sont pas perdus. Nous présentons notre système dans les chapitres suivants.

Troisième partie

Texte, structure et Sémantique

Chapitre 5

Similarité sémantique

Sommaire

5.1 Introduction

Nous avons mis en évidence dans le chapitre 3, les problèmes des modèles classiques de RI dus à la correspondance exacte entre les termes de la requête et les termes d'indexation. Les auteurs des documents et les utilisateurs de SRI utilisent une grande variété de mots pour exprimer le même concept. De plus, ils peuvent aussi utiliser les mêmes termes pour exprimer des concepts différents. Le but d'utiliser une ontologie dans un processus de RI est d'apporter une réponse à ces problèmes et permettre de retourner des réponses fondées sur la sémantique des concepts à la place des termes. L'ontologie joue donc un rôle important et de sa qualité dépend celle de l'indexation.
La difficulté de l'indexation des documents par une ontologie est qu'elle doit couvrir la totalité des concepts dans le corpus. Cela est possible si l'ontologie est construite à partir des corpus, mais elle doit néanmoins être évolutive pour prendre en compte l'émergence de nouveaux

concepts (si d'autres documents sont rajoutés). On doit souvent faire un choix entre le degré de finesse des descriptions extraites de l'ontologie et la complexité des traitements (une représentation des contenus des documents par des réseaux sémantiques est fastidieuse et demande une validation). Le choix des concepts pour décrire les documents est crucial, tout comme le choix des termes d'indexation en RI. Quels sont les concepts qui représentent le document ? Quels sont les concepts qui intéresseraient l'utilisateur ? Ce même problème se pose pour une recherche d'information sur le Web Sémantique (cf. chapitre 4) où le choix des annotations pour un document sont determinants pour pouvoir le retrouver. Le pari du Web Sémantique porte sur la profusion sur le Web d'ontologies par domaine et par point de vue, ce qui permet d'avoir un large éventail de choix de l'ontologie pour annoter les documents.

Pour interroger les documents indexés *sémantiquement*, l'utilisateur doit poser sa requête par les concepts de l'ontologie. Pour cela, il faut lui permettre de naviguer dans l'ontologie et de sélectionner ces concepts. Les langages actuels pour une interrogation sémantique des documents (e.g. RDQL, OWLIR cf. chapitre 4) sont complexes et supposent une connaissance du langage de représentation. Ils restent inaccessibles pour des utilisateur non expérimentés. De plus la recherche est booléenne, les documents répondent ou non à la requête et sont tous présentés au même niveau.

Pour résoudre ces problèmes, nous proposons de :

– Mettre à profit toutes les descriptions de l'ontologie sans pour autant rester tributaire de son degré de couverture. Nous défendons la thèse que l'ontologie n'est plus au cœur de la recherche mais vient la renforcer quand c'est possible. Nous indexons aussi les termes qui ne sont pas couverts par l'ontologie.
– Utiliser l'ontologie d'une manière transparente à l'utilisateur, sans qu'il n'ait besoin de comprendre les fondements sous-jacents. Nous voulons que l'utilisateur continue de poser ses questions en langage naturel sans se soucier de l'ontologie qu'il y a derrière. C'est au système de s'adapter aux questions de l'utilisateur, d'en extraire le maximum de sens et de chercher les correspondances dans l'index.
– Utiliser la richesse de la structure de l'ontologie pour permettre de retourner à l'utilisateur des documents qui comportent les synonymes des termes recherchés mais aussi les termes plus spécifiques, plus généraux et plus proches.

Afin de réaliser ces propositions, nous avons utilisé la notion de similarité sémantique pour permettre d'indexer les termes par leur voisinage sémantique (le sens des termes apparaissant dans un même contexte). Les termes d'indexation restent les mêmes que pour une indexation classique, ils sont enrichis s'ils ont des concepts reliés à l'ontologie.[64]

Pour cela nous avons besoin de quantifier la notion de similarité. L'ontologie est aussi utilisée dans la phase de recherche afin de trouver des documents qui contiennent des termes apparentés à ceux de la requête. Elle est particulièrement utile dans le cas où les termes recherchés n'apparaissent pas dans le corpus mais dans l'ontologie, nous retournons dans ces cas les documents apparentés sémantiquement même s'ils ne contiennent aucun terme de la requête.

La **similarité sémantique** a intéressé diverses communautés de recherche en intelligence artificielle, psychologie et sciences cognitives. Elle a comme principaux champs d'application la recherche d'information et le traitement automatique de la langue.

[64]Nous présenterons notre système d'indexation complet ainsi que le langage de requêtes dans les chapitres suivants.

Il est important de noter que dans la littérature, on parle aussi de **proximité sémantique** (*semantic relatedness*) qui est une notion plus large que la **similarité sémantique**. En effet la proximité sémantique prend en considération tout type de relation entre les concepts. Ainsi deux concepts peuvent être proches sémantiquement par leur similarité (e.g. voiture et automobile), mais aussi par d'autres relations comme *partie-de* (voiture-roue) ou *contraire* (guerre-paix), etc.

Dans ce travail, nous traitons la similarité sémantique qui même si elle est plus restrictive nous semble plus adéquate pour une recherche d'information. Un utilisateur est (généralement) plus intéressé par des documents qui traitent de moyens de transports quand il recherche voiture que par roue, essuie glaces, etc.

Deux approches principales sont utilisées pour la mesure de similarité entre concepts dans une ontologie : (i) en utilisant la structure arborescente ou (ii) en utilisant le contenu informatif des différents concepts en intégrant des mesures statistiques. D'autres approches proposent de combiner les deux. Notre approche fait partie de la première catégorie de méthodes et adapte la mesure de Wu & Palmer que nous présentons dans la section suivante. La plupart de ces propositions portent sur WordNet[65] [66], elles peuvent cependant être généralisées à une ontologie puisqu'elles exploitent la structure taxonomique.

WordNet possède l'avantage d'être une ressource assez fournie, le grand nombre de synsets [67] ainsi que sa facilité d'accès[68] en ont fait une ressource très utilisée en RI et en TAL. Néanmoins des lacunes existent ; nous mettrons en évidence certaines incohérence de WordNet qui peuvent fausser les résultats des mesures de similarité (section 5.8.1). Le besoin de s'assurer de la consistance de la ressource sémantique utilisée nous mène à l'utilisation d'une ontologie formelle, ce qui nous permet aussi d'utiliser les inférences. Nous présentons notre mesure de similarité appliquée à WordNet -à l'instar des mesures que nous présentons dans les sections 5.3, 5.4 et 5.5- qui utilise les liens hiérarchiques entre concepts. Nous présentons également l'application de cette mesure dans le cadre d'une ontologie formelle (en OWL), qui, couplée à l'utilisation d'un algorithme de calcul du concept le plus spécifique qui subsume deux concepts donnés **(PPG : Plus Petit Généralisant)**, permet de prendre en compte tous les différents types de relation entre deux concepts.

Nous commençons par présenter les différentes terminologies et notations qui seront utilisées dans ce chapitre. Nous présentons notre mesure de similarité précédé par une présentation succincte des mesures de similarités proposées dans la littérature. Nous exposons ensuite le processus d'évaluation de telles mesures pour WordNet. Nous mettons en évidence les limites de WordNet en tant qu'ontologie et présentons, enfin, un algorithme de calcul du **Plus Petit Généralisant : PPG** et son apport pour notre calcul de la similarité.

[65] Plus spécifiquement sur les noms, cette partie étant la plus fournie de WordNet.

[66] Il existe aussi dans la littérature des mesures de similarité sémantique qui portent sur d'autres thésaurus (e.g. le Roget [Roget, 1995]), elles utilisent la structure spécifique au thésaurus [Morris & Hirst, 1991] ou proposent une utilisation des liens hiérarchiques [Jarmasz & Szpakowicz, 2003] et sont semblables aux mesures qu'on présente dans la section .

[67] 115424 synsets toute catégorie grammaticale confondue, dans WordNet 2.0.

[68] WordNet est accessible en ligne et en téléchargement avec des APIs pour avoir accès à sa structure interne.

5.2 Terminologie et notation

Les approches présentées ci-dessous manipulent des **concepts** et des **mots**[69]. Un concept réfère à un sens particulier d'un mot donné. Quand on dit que deux mots sont similaires, c'est dans le sens que les concepts qui leur sont reliés sont similaires. La similarité est la fonction inverse de la distance, plus deux mots sont similaires, moins ils sont distants. Une distance $dist$ est une **mesure** qui respecte les trois propriétés suivantes :
– nullité de la distance d'un concept avec lui même : $dist(a, a) = 0$
– symétrie : $dist(a, b) = dist(b, a)$
– inégalité triangulaire : $dist(a, b) \leq dist(a, c) + dist(c, b)$
Nous partageons les mêmes intuitions que Lin [1998] concernant les caractéristiques que devrait respecter la notion de similarité. La similarité entre deux entités A et B est
– fonction des caractéristiques communes. Plus les entités ont des caractéristiques en commun, plus elles sont similaires.
– fonction de leurs différences. La similarité décroît inversement aux caractéristiques différentes.
– maximale quand A est identique à B.

Définition

Une mesure de similarité est une fonction sim : $S^2 \longrightarrow [0, 1]$ avec S l'ensemble de concepts.

Les opinions divergent quant aux propriétés que doit respecter une mesure de similarité. Tversky [1977] a montré que les mesures de similarités conformes à la perception humaine ne satisfont pas toujours les propriétés d'une mesure, comme par exemple la symétrie. Rada *et al.* [1989] stipulent que cette asymétrie ne provient pas de la similarité mais des relations asymétriques entre deux concepts.
Il est donc généralement admis qu'une mesure de similarité doit être réflexive et symétrique :
– $sim(x, x) = 1$: réflexivité
– $sim(x, y) = sim(y, x)$: symétrie
Nous évaluons la similarité entre les concepts d'une ontologie. Nous commençons par en définir les composantes qui serviront pour nos calculs.

Définition

Une ontologie $O =< S, R, type, \top, \bot >$
– S : l'ensemble des concepts de l'ontologie.
– R : l'ensemble des relations entre les concepts.
– $type$: une fonction qui associe un type à une relation (e.g. $is-a(R)$ est la relation hiérarchique de spécialisation).
– $\top$: est la racine de l'ontologie.

[69] Nous utiliserons aussi "mot" dans notre cas pour désigner les termes sélectionnés pour l'indexation et sur lesquels portera la mesure de similarité

– $\perp$: est l'anti-racine de l'ontologie.

WordNet peut donc être représenté comme une ontologie [70], où S est l'ensemble des synsets, R est l'ensemble de relations (hyperonymie, hyponymie, antonymie, etc.). $\top$ est un concept que nous ajoutons au dessus des 11 entrées de WordNet, pour s'assurer d'avoir toujours un chemin entre deux concepts. La plupart des mesures reposent sur le lien taxonomique $is-a$ (hyperonymie, hyponymie) pour le calcul de similarité.
Nous utiliserons les notations suivantes[71] :
– $chemin(C_1, C_2)$ est le chemin le plus court en nombre d'arcs qui mène d'un concept C_1 à un concept C_2.
– $profondeur(C)$ est la longueur du plus court chemin entre le concept C et la racine, on a $profondeur(C) = chemin(C, \top)$
– $ppac(C_1, C_2)$ est le plus petit ancêtre commun des concepts $C_1 et C_2$, c'est le concept le plus spécifique qui les subsume.
– Dans les formules que nous présentons, la similarité sémantique entre deux mots, w_1 et w_2, est calculée par :

$$Sim(w_1, w_2) = \max_{c_1 \in S(w_1), c_2 \in S(w_2)} sim(c_1, c_2)$$

où $S(w_i)$ est l'ensemble des concepts qui sont des sens de w_i. La similarité entre deux mots est donc égale à la similarité maximale entre les différents sens des mots.

5.3 Calcul de similarité par le nombre d'arcs

Rada *et al.* [1989] ont été, à notre connaissance, les premiers à suggérer que la similarité dans un réseau sémantique[72] peut être calculée en se fondant sur les liens taxonomiques « is-a ». Plus généralement, le calcul de la similarité entre concepts peut être fondé sur les liens hiérarchiques de spécialisation/généralisation. Un moyen des plus évidents pour évaluer la similarité sémantique dans une taxonomie est alors de calculer la distance entre les concepts par le chemin le plus court. Les auteurs soulignent que cette proposition est valable pour tous les liens de type hiérarchique (est-un (*is-a*), sorte-de (*kind-of*), partie-de (*part-of*) mais doit être adaptée pour d'autres types de liens (cause, etc.).[73]
Hirst & St-Onge [1998] calculent la proximité sémantique qui est une notion plus large que la similarité sémantique. Toutes les relations dans WordNet sont prises en compte. Les liens sont classés comme *haut* (hyperonymie et meronymie), *bas* (hyponymie et holonymie) et *horizontal*

[70] Cette appellation est un choix de généralité, notre mesure pouvant s'appliquer à n'importe quel réseau sémantique. Nous préciserons à la fin de chapitre, pourquoi WordNet ne peut pas, en définitive, être considérée comme une ontologie.
[71] Pour avoir une présentation uniforme, nous avons pris la liberté de changer les notations originales des formules.
[72] Les auteurs utilisent Mesh (*Medical Subject Heading*), un réseau sémantique de termes utilisés pour indexer les articles dans le système de recherche bibliographique Medline.
[73] Le succès d'une telle approche est probablement dû à la spécificité du domaine (médecine) qui assure une homogénéisation de la hiérarchie.

(antonymie). En plus du classement des types de relation, les auteurs définissent un degré de relation[74] :

– *Très fort : (extra-strong)* fondé sur la forme de surface des mots.
– *Fort : (strong)* selon trois cas : (i) quand les mots font partie du même synset, (ii) quand les mots ont une relation horizontale entre leurs synsets, ou (iii) quand un mot fait partie d'un mot-composé.
– *Moyen-fort : medium-strong* quand il y a un chemin acceptable entre deux mots.
– *Faible :* il s'agit de tous les autres mots.

Les mots qui ont une relation extra-forte ou forte ont une similarité constante de $3 * T$ et $2 * T$ respectivement (T étant une constante). Les mots qui ont une relation faible ont une similarité nulle. Pour les relations moyennes, la proximité est calculée entre les mots par le poids du chemin le plus court qui mène du synset du mot à un autre. Elle est calculée en fonction des classements des types de relations qui indiquent les changements de direction.

$$Prox(C_1, C_2) = T - chemin(C_1, C_2) - k * d \tag{5.1}$$

Tels que T et K sont des constantes, dans la pratique les auteurs fixent $T = 8$ et $K = 1$ et d le nombre de changements de direction. L'idée est que deux mots sont proches sémantiquement si leurs synsets sont connectés par un chemin qui n'est pas très long et qui ne change pas souvent de direction. S'il n'y a pas de chemin, ce calcul est égal à zéro. Cette mesure s'éloigne de la similarité proprement dite car elle traite tout type de relations, comparée aux autres mesures de similarité. Elle ne donne pas, de ce fait, de bons résultats[75].

La limite d'utiliser le chemin le plus court est qu'on ne prend pas en considération la position des concepts dans l'ontologie. Intuitivement, deux concepts classés en bas de l'ontologie sont très spécifiques et sont donc à un degré de granularité plus fin que deux concepts classés en haut de l'ontologie. Ainsi les synsets *plant*[76] et *animal* ont la même distance entre eux que *egyptian-cat*[77] et *siamese-cat*[78], alors que intuitivement les deux derniers sont plus proches. La mesure de [Wu & Palmer, 1994] apporte une réponse à ce problème en comptant la position des concepts par rapport à la racine de l'ontologie.

Wu & Palmer [1994] ont défini une mesure de similarité entre concepts pour la traduction automatique entre l'anglais et le mandarin chinois. Pour éviter les problèmes d'ambiguïté, leur mesure s'applique à un domaine conceptuel qui correspond à un point de vue donné pour lequel un mot a un seul sens. La similarité est définie par rapport à la distance qui sépare deux concepts par rapport au concept le plus spécifique qui subsume les deux concepts dans l'ontologie, ainsi que la racine de la hiérarchie. La similarité entre C_1 et C_2 (voir figure 5.1) est :

$$sim_{WP}(C_1, C_2) = \frac{2 * N3}{N1 + N2 + (2 * N3)} \tag{5.2}$$

[74] Extra strong, Strong, Strong-medium, Weak.
[75] Cela est peut être dû aussi au type d'évaluation où les évaluateurs devaient juger de la similarité (synonymie) entre mots cf. section 5.8
[76] Plante.
[77] Chat égyptien.
[78] Chat siamois.

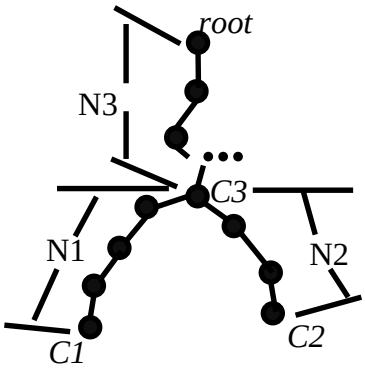

FIG. 5.1 – Les relations conceptuelles[Wu & Palmer, 1994]

Plus formellement cette mesure devient :

$$sim_{WP}(C_1, C_2) = \frac{2 * profondeur(C_3)}{profondeur_C(C_1) + profondeur_C(C_2)} \tag{5.3}$$

Où C_3 est le $ppac(C_1, C_2)$, $profondeur(C_3)$ est le nombre d'arcs minimal qui séparent C de la racine et $profondeur_C(Ci)$ le nombre d'arcs minimal qui séparent C_i de la racine en passant par C. La profondeur de C est une profondeur globale qui permet de normaliser le calcul par rapport à la position des concepts dans la hiérarchie. Deux concepts identiques ont une similarité maximale de 1, plus les concepts sont éloignés, plus la mesure décroît, elle atteint 0 pour deux concepts qui ont $\top$ comme $ppac$.
Cette mesure a l'avantage d'être rapide à calculer, en restant aussi expressive que les autres méthodes (elle a d'aussi bonnes performances que les autres mesures de similarité) [Lin, 1998][79].

5.4 Calcul de similarité par le contenu informatif

Les mesures de [Resnik, 1995] et [Jiang & Conrath, 1997] sont fondées sur la notion de contenu informatif. La notion de **contenu informatif (CI)** a été introduite pour la première fois par Resnik. Elle utilise conjointement l'ontologie et un corpus. Le contenu informatif d'un concept traduit la pertinence d'un concept dans le corpus en tenant compte de la fréquence de l'apparition des mots auxquels il se réfère ainsi que de la fréquence d'apparition des concepts qu'il généralise. Plus précisément le contenu informatif se calcule par la formule suivante :

$$CI(C) = -log(p(C)) \tag{5.4}$$

Où $p(C)$ est la probabilité de retrouver qu'un mot du corpus soit une instance du concept C (un des mots référés par le concept C ou par un de ses descendants). Elle est monotone quand on remonte dans la hiérarchie $(p(A) \leq p(B))$ si A est plus grand B (e.g. dans la figure 5.2, $p(hill) < p(geological - formation)$[80]). Dans les expérimentations de [Resnik, 1995], ces

[79] En comparaison avec la mesure de Resnik et Lin. Les expérimentations ont porté sur la corrélation avec les jugements humains sur les paires de mots définis par Miller & Charles (cf. section 5.8)
[80] colline, information géologique.

probabilités sont calculées par : $p(C) = \frac{frequence(C)}{N}$ où
N est le nombre total de concepts et $frequence(C) = \sum_{w \in instance(C)} count(w)$.

Plus un concept est général, plus son contenu informatif est faible, ainsi $CI(p(\top)) = 0$ parce qu'il a un contenu informatif nul. A l'inverse, plus le concept est spécifique plus son contenu informatif est important (e.g. le contenu informatif de *hill* est plus important que celui de *geological-formation*). L'intuition de la notion de contenu informatif est que la similarité entre deux concepts est la portion d'information qu'ils ont en commun qui, dans le cadre d'une ontologie, peut être déterminée par le concept le plus spécifique qui les subsume (*ppac*). Cette intuition est indirectement appliquée par les mesures présentées dans la section précédente qui calculent la similarité avec le nombre d'arcs qui séparent deux concepts.

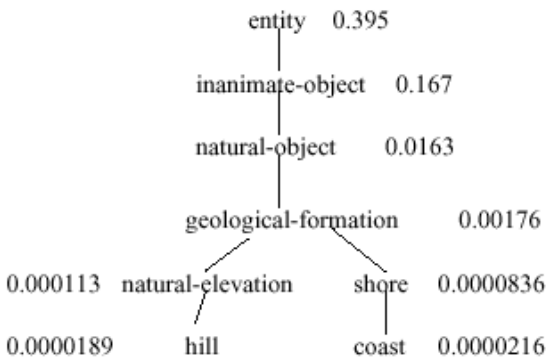

FIG. 5.2 – Extrait de WordNet présenté dans [Lin, 1998] avec les probabilités correspondantes aux différents concepts[82].

Resnik définit la similarité sémantique entre deux concepts par la quantité d'information qu'ils partagent. Cette information partagée est évaluée numériquement par le contenu informatif du plus petit ancêtre commun ($ppac$).

$$sim_{Res}(C_1, C_2) = CI(ppac(C_1, C_2)) \tag{5.5}$$

Ainsi, si deux termes sont très éloignés et ont comme $ppac$ la racine, leur similarité est égale à 0. L'approche de Resnik essaie d'éviter le problème de granularité, cité au dessus, en diminuant le rôle des arcs dans le calcul de similarité. En effet, les arcs ne sont utilisés que pour retrouver le $ppac$, elle est de ce fait un peu sommaire car nous pouvons avoir $ppac(hill, shore^{83}) = ppac(shore, natural - elevation)$ même si *shore* et *natural-elevation* sont plus proches de leur $ppac$ (*geological-formation*) que *shore* et *hill*.

Les mesures qui font un calcul hybride, combinent le calcul d'arcs avec le contenu informatif.

[83] rivage

5.5 Calcul de similarité hybride (contenu informatif et liens taxonomiques)

La mesure de [Jiang & Conrath, 1997] pallie les limites de la mesure de Resnik en combinant le contenu informatif du $ppac$ à ceux des concepts. Elle prend en considération aussi le nombre d'arcs en calculant le contenu informatif de chaque concept.
Ainsi une distance est définie :

$$distance(C_1, C_2) = CI(C_1) + CI(C_2) - (2.CI(ppac(C_1, C_2))) \tag{5.6}$$

La mesure de similarité revient donc à calculer :

$$sim_{JC}(C_1, C_2) = \frac{1}{distance(C_1, C_2)}^{84} \tag{5.7}$$

[Lin, 1998] propose une mesure de similarité qui calcule la proportion d'information commune entre deux concepts par rapport à leur description.

$$sim_{Lin}(C_1, C_2) = \frac{2 * CI(ppac(C_1, C_2))}{CI(C_1) + CI(C_2)} \tag{5.8}$$

Notons que si la probabilité $p(C|C')$ est la même pour toutes les paires de concepts, cette mesure coïncide avec celle de Wu & Palmer.

5.6 Discussion

Le calcul de la similarité par nombre d'arcs avec une restriction sur le lien « is-a » pose deux problèmes :
- La granularité : Les concepts placés en bas de l'ontologie, donc plus spécifiques sont plus similaires que les concepts en haut de l'ontologie (cf. exemple donné plus haut).
- La densité : Pour un concept qui a plusieurs fils, il paraît logique de supposer que sa similarité avec ses fils est plus grande qu'avec les autres concepts. Ainsi dans les parties plus denses de l'ontologie, la similarité doit augmenter.

Ces problèmes peuvent être résolus en associant des poids aux liens. L'affectation de ces poids peut être basée sur : les types de liens présents, la profondeur du lien dans la taxonomie et la densité du concept par ses voisins immédiats [Sussna, 1997; Agirre & Rigau, 1996].
La performance des mesures hybrides ainsi que celles fondées sur le contenu informatif dépend de la qualité du corpus sur lequel sont calculés les contenus informatifs. Il faut disposer d'un corpus qui contient des occurrences des mots qui vont servir pour le calcul de similarité avec une répartition qui donnerait une idée de la relation entre les concepts. C'est la raison pour laquelle nous avons préféré l'utilisation d'une mesure qui prend en compte le nombre d'arcs, plus spécifiquement celle de Wu & Palmer.

[84] On pourrait aussi calculer la similarité par $1 - distance$, l'essentiel est que la similarité soit une fonction inverse de la distance.

5.7 Adaptation de la mesure de Wu & Palmer

Les mesures qui utilisent la notion de contenu informatif sont tributaires des corpus d'apprentissage qui servent de référence pour le calcul des probabilités des concepts. De plus, ces mesures ne peuvent pas être appliquées dans notre cas car elle serait redondante puisque nous combinons la mesure de similarité à la mesure distributionnelle des termes (voir chapitre 6). Nous avons donc retenu la mesure de Wu & Palmer parce qu'il a été prouvé qu'elle est aussi efficace que les autres mesures [Lin, 1998]. Cette mesure a été aussi utilisée dans le système *THESUS* pour la classification (*clustering*) de pages web [Halkidi *et al.*, 2003] ainsi que pour évaluer la proximité sémantique de deux concepts d'une page HTML relativement à un thésaurus dans le cadre d'une indexation d'un site web par des ontologies [Desmontils & Jacquin, 2001; Simon *et al.*, 2003].
Rappelons que la mesure de Wu & Palmer calcule la similarité de la manière suivante.

$$sim_{WP}(C_1, C_2) = \frac{2 * profondeur(C)}{profondeur_C(C_1) + profondeur_C(C_2)}$$

Cette mesure de similarité est intéressante parce qu'elle prend en considération le plus petit ancêtre commun (elle permet, donc, de prendre en considération la granularité). Plus cet ancêtre est général, moins les concepts sont similaires (et inversement). Néanmoins, cette mesure présente une limite car elle vise essentiellement à détecter la similarité uniquement par rapport à ce *ppac*. Dans le cas de recherche d'information, il est à notre sens plus intuitif de retourner en priorité les concepts qui sont subsumés par les concepts de la requête avant les concepts voisins. Si l'utilisateur recherche les documents qui contiennent *chat*, il est plus intéressant de lui retourner les différentes *races de chat* avant de lui retourner des documents qui contiennent les différents *types de félin*. La similarité doit de ce fait prendre en considération les liens père/fils et la densité des concepts. Nous voulons obtenir une mesure de similarité qui classe les concepts de la manière suivante :

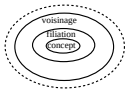

FIG. 5.3 – Relations entre concepts

Ceci n'est pas assuré par la mesure de Wu & Palmer. Si nous calculons la similarité (voir figure 5.4 entre *feline*[85] et *blue-point-siamese-cat*[86] (qui est un des descendants de feline) et entre *feline* et *canine*[87] (qui est un frère) nous obtenons les valeurs rapportées dans le tableau 5.1 :

Or un *félin* est intuitivement plus proche d'un *chat siamois bleu point*, parce qu'ils partagent toutes les caractéristiques de félin, que d'un *canin* où ils partagent toutes les caractéristiques de carnivore qui est plus général que félin.

[85] Félin.
[86] Chat siamois bleu point.
[87] Canin.

sim(mot#catégorie#numsens, mot#catégorie#numsens)	Trace d'exécution
$sim_{wp}(feline\#n\#1, canine\#n\#2) = 0.9$	$ppac : carnivore\#n\#1$ $profondeur = 9$ $profondeur(feline\#n\#1) = 10$ $profondeur(canine\#n\#2) = 10$
$sim_{wp}(feline\#n\#1, blue\_point\_siamese\_cat\#n\#1) = 0.833$	$ppac : feline\#n\#1$ $profondeur = 10$ $profondeur(feline\#n\#1) = 10$ $profondeur(blue\_point\_siamese\_cat\#n\#1) = 14$

TAB. 5.1 – Résultat de sim_{WP} avec trace d'exécution

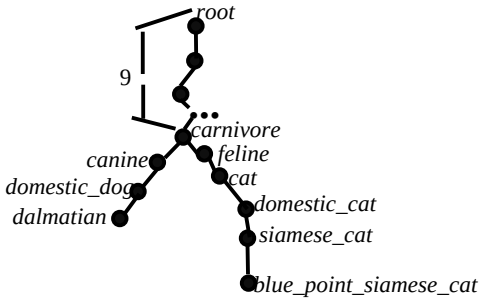

FIG. 5.4 – Exemple

Nous voulons éviter qu'un concept soit plus proche de son voisinage que de ses descendants. Pour cela nous pénalisons les concepts qui ne sont pas de la même lignée. Nous rajoutons une mesure $spec$ qui prend en considération le **degré de spécialisation** du concept par rapport au nombre d'arcs qui le séparent de $\perp$ (l'anti-racine). Rappelons que le concept $\perp$ n'existe pas dans l'ontologie, c'est un concept virtuel qui symbolise le concept le plus bas de l'ontologie, il est subsumé par tous les concepts de l'ontologie, nous avons donc $distance(\top, \perp)$ est égal à la profondeur de l'ontologie plus 1.
Le degré de spécialisation nous informe donc, d'une certaine manière, sur la densité d'un concept. Si un concept a de nombreux descendants, il est plus spécialisé qu'un concept qui en a moins.
Notre mesure prend donc en compte la **spécificité** d'un concept, en calculant sa position par rapport à $\top$, ainsi que sa **spécialisation**, en calculant sa position par rapport à $\perp$.

$$spec(C_1, C_2) = profondeur_{\perp}(C) * distance(C, C_1) * distance(C, C_2) \tag{5.9}$$

Avec $profondeur_{\perp}(C)$, le nombre d'arc qui sépare C (le $ppac(C_1, C_2)$) de $\perp$ et $distance(C, C_i)$ la distance en nombre d'arc entre C et C_i. $profondeur_{\perp}(C)$ calcule la spécialisation du plus petit ancêtre commun. Notons que $spec$ compte seulement pour les concepts qui sont voisins (ou frères), quand $C \neq C_1$ et C_2. Dans le cas où $C = C_1$ ou C_2, $spec$ devient nulle (e.g. $spec(feline\#n\#1, blue\_point\_siamese\_cat\#n\#1) = 0$).
Ainsi la mesure de similarité devient :

$$sim(C_1, C_2) = \frac{2 * profondeur(C)}{profondeur_C(C_1) + profondeur_C(C_2) + spec(C_1, C_2)} \tag{5.10}$$

Dans le cas où $spec$ est nulle, pour les concepts de même lignée, notre similarité revient à calculer sim_{WP}. Si nous recalculons les similarités sur le même exemple précédent, nous retrouvons les valeurs suivantes (voir tableau 5.2) qui sont cohérentes avec nos intuitions.

sim(mot#catégorie#numsens, mot#catégorie#numsens)	Trace d'exécution)
$sim(feline\#n\#1, canine\#n\#2) = 0.620$	$ppac : carnivore\#n\#1$ $profondeur = 9$ $profondeur_{\perp} = 9$ $profondeur(feline\#n\#1) = 10$ $profondeur(canine\#n\#2) = 10$
$sim(feline\#n\#1, blue\_point\_siamese\_cat\#n\#1) = 0.833$	$ppac : feline\#n\#1$ $profondeur = 10$ $profondeur_{\perp} = 6$ $profondeur(feline\#n\#1) = 10$ $profondeur(blue\_point\_siamese\_cat\#n\#1) = 14$

TAB. 5.2 – Résultat de notre similarité avec trace d'exécution

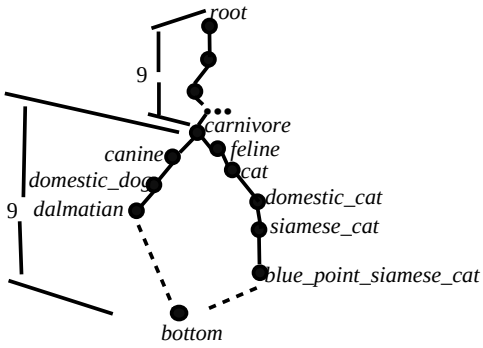

FIG. 5.5 – Exemple extrait de la taxonomie de WordNet avec représentation de $bottom$

Notre mesure vérifie bien les caractéristiques que nous nous sommes fixées :
- La similarité augmente en fonction des caractéristiques communes. En effet, deux concepts frères situés en bas de l'ontologie, sont plus similaires car ils ont plus de propriétés en communs que deux concepts frères situés en haut de l'ontologie.
- La similarité diminue en fonction des caractéristiques différentes. En effet, plus deux concepts sont éloignés, plus ils ont des propriétés différentes.
- La similarité est maximale (égale à 1) quand les deux concepts sont identiques.

5.8 Evaluation des mesures de similarité

L'évaluation des mesures de similarité peut se faire en étudiant la corrélation avec les jugements humains[88]. Rubenstein & Goudenough [1965] furent les premiers à proposer de conduire une expérience quantitative avec 51 sujets humains qui ont été amenés à évaluer 65 paires de mots en leur donnant une note allant de 0 à 4. Les paires de mots varient de très similaires (les synonymes) à des couples non reliés. Le but de cette expérimentation était d'expliquer la perception humaine de la synonymie. Miller & Charles [1991] ont reproduit la même expérimentation en étudiant 30 paires de mots extraites de celles proposées par Rubenstein & Goudenough (10 paires ayant un score entre 3 et 4, 10 paires ayant un score entre 1 et 3 et 10 autres ayant un score entre 1 et 0). L'expérimentation a porté sur 38 sujets. La corrélation entre les jugements

[88]Les jugements humains sont supposés corrects par définition, une bonne mesure doit bien approximer les jugements humains.

humains de Rubenstein & Goudenough et Miller & Charles est de 0.95. Les expérimentations ont été effectuées à 25 ans d'intervalle, ce qui peut expliquer la variation de perception de similarités, qui peut aussi être plus simplement due au fait que les sujets ne sont pas les mêmes (voir tableau 5.3). Ces paires de mots ont été reprises par d'autres expérimentations ([Resnik, 1995; Lin, 1998; Hirst & St-Onge, 2004; Patwardham, 2003]), le but étant de détecter les corrélations avec les résultats des différentes mesures de similarités. De telles expérimentations ont des limites, nous en dégageons quelques unes dans ce qui suit.

Paires de mots	M&C		R&G		Paires de mots	M&C		R&G	
	score	rang	score	rang		score	rang	score	rang
car-automobile	3.92	1	3.92	2	crane-implement	1,66	15	2,37	14
gem-jewel	3.84	2	3.94	1	journey-car	1.16	16	1.55	15
journey-voyage	3.84	2	3.58	6	monk-oracle	1.10	17	0.91	21
boy-lad	3.76	3	3.82	3	cemetery-woodland	0.95	18	1.18	17
coast-shore	3.70	4	3.60	5	food-rooster	0.89	19	1.09	21
asylum-madhouse	3.61	5	3.04	9	coast-hill	0.87	20	1.26	16
magician-wizard	3.5	6	3.21	7	forest-graveyard	0.84	21	1.00	19
midday-noon	3.42	7	3.94	1	shore-woodland	0.63	22	0.90	22
furnace-stove	3.11	8	3.11	8	monk-slave	0.55	23	0.57	24
food-fruit	3.08	9	2.69	11	coast-forest	0.42	24	0.85	23
bird-cock	3.05	10	2.63	12	lad-wizard	0.42	24	0.99	20
bird-crane	2.97	11	2.63	12	chord-smile	0.13	25	0.02	27
tool-implement	2.95	12	3.66	4	glass-magician	0.11	26	0.44	25
brother-monk	2.82	13	2.74	10	rooster-voyage	0.08	27	0.04	26
lad-brother	1.68	14	2.41	13	noon-string	0.08	27	0.04	26

TAB. 5.3 – Les paires de mots dans les expérimentations avec les jugements humains (extrait de [Miller & Charles, 1991].

5.8.1 Discussion

La comparaison avec le jugement des sujets humains constitue le cadre idéal pour valider une mesure de similarité. Seulement, en pratique, il est difficile de mener une expérimentation d'envergure, avec un grand nombre de paires de mots ainsi qu'un grand nombre de sujets défini et fiable. Finkelstein & E. Gabrilovich [2002] ont publié les similarités entre 353 paires de mots[89], ce qui constitue un début, le nombre de sujets restant limité[90] [91]. Ainsi pour évaluer les jugements des sujets humains, il est nécessaire de définir des variables (telles que le sexe, l'age, la catégorie socio-culturelle, etc.) et mesurer leur influence sur les jugements.
De plus, il est implicite dans les expérimentations de [Rubenstein & Goudenough, 1965] et [Miller & Charles, 1991] que les sujets humains utiliseraient le sens dominant des mots, or cela n'est pas évident. Le sens des mots est défini par le contexte dans lequel ils apparaissent et rien ne peut présupposer du sens qui serait attribué par l'utilisateur. Il pourrait s'avérer intéressant de laisser le sujet associer un ensemble de mots à un mot donné, ce qui donnerait une idée du sens assigné par le sujet.[92]

[89] Avec les 30 paires de mots de Miller & Charles

[90] 16 sujets ont participé à l'évaluation

[91] Il est intéressant de noter que la mesure de Resnik qui avait 79% de corrélation avec les jugements humains a donné 39% sur les 350 paires de mots.

[92] Il nous paraît aussi plus pertinent de présenter des couples de mots avec des mots redondants. Si on demande à des sujets de donner une valeur à la similarité entre *chat* et *félin* ils ne donneraient pas la même estimation que si on leur présente *(chat, félin)* et *(chat, chat égyptien)*.

Une évaluation des mesures dans le cadre d'une application peut s'avérer plus pertinente (ou du moins plus facile à mener). Des tests sur la détection des incongruités (*malproprism*) sont menées par Budanitsky & Hirst [2001]. Une variation de l'orthographe est faite aléatoirement sur un corpus et il s'agit de détecter et corriger ses mots en calculant la similarité avec d'autres mots apparaissant dans la même fenêtre. La fenêtre peut être le paragraphe dans lequel le mot apparaît (scope =1), ce paragraphe avec un ou deux paragraphes adjacents de chaque côté (scope=3 ou 5) ou tout le document (scope=Max). Les expérimentations ont porté sur cinq mesures de similarité [Hirst & St-Onge, 1998; Leacock & Chadorow, 1998; Resnik, 1995; Jiang & Conrath, 1997; Lin, 1998]. Celle de Jiang & Conrath a montré les meilleurs résultats en rappel/précision[93]. Les expériences menées ont montré aussi que la meilleure fenêtre est le paragraphe (scope=1).
Il est important de souligner que les limites des expérimentations avec des sujets humains ne se limitent pas seulement au fait qu'elles soient menées hors du contexte applicatif. Les incohérences même de WordNet font qu'avec les mesures les plus pertinentes, nous ne pouvons pas obtenir la même corrélation que les sujets humains, nous présentons ici deux exemples étonnants de la structuration de WordNet[94].

Incohérence de WordNet

La relation de subsomption "is-a" définit un lien de specialisation/généralisation, elle est utilisée pour structurer les ontologies. Cette relation permet formellement l'héritage de propriétés des concepts. Si un concept C_1 hérite les propriétés de C et C', tels que C subsume C', il sera classé directement sous C'. C'est sur cette base que se fondent les mesures de similarité sémantique qui prennent en considération les liens "is-a". Or dans WordNet, ceci n'est pas toujours respecté. Nous avons voulu calculer la similarité entre *(européen, asiatique)*, *(européen, africain)* et *(européen, américain)*, étant au même niveau de granularité (habitants d'un continent), nous nous attendions à avoir la même similarité. Les résultats de la mesure de similarités sur WordNet sont les suivants :

sim(mot#catégorie#numsens, mot#catégorie#numsens)	Trace d'exécution)
$sim(european\#n\#1, asian\#n\#1) = sim(european\#n\#1, american\#n\#1) = 0.454$	$ppac : inhabitant\#n\#1$ $profondeur = 5$ $profondeur_{\perp} = 10$ $profondeur(european\#n\#1) = 6$ $profondeur(asian\#n\#1) = 6$
$sim(european\#n\#1, african\#n\#1) = 0.242$	$ppac : person\#n\#1$ $profondeur = 4$ $profondeur_{\perp} = 11$ $profondeur(european\#n\#1)) = 6$ $profondeur(african\#n\#1) = 5$

TAB. 5.4 – Similarité entre *european*, *asian* et *african*

Une vérification de la place de ces synsets dans WordNet a montré que *européen*, *asiatique* et *américain* sont classés en dessous d'*habitant* par contre *africain* est classé en dessous de

[93] Il reste cependant inexpliqué la différence des résultats (inférieurs) donnés par la mesure de Lin qui consiste juste à une combinaison arithmétique différente de celle de Jiang & Conrath

[94] Nous relevons ici les incohérences qui peuvent influencer les résultats des mesures de similarité. WordNet ne fait pas l'unanimité parmi les chercheurs, [Slodzian, 1999] [Slodzian, 2000] développe des arguments épistémologiques et linguistiques à ce sujet.

personne (voir figure 5.6). Rien dans la définition (*gloss*)[95] de *africain* n'explique sa position en dessous de personne au lieu d'habitant.

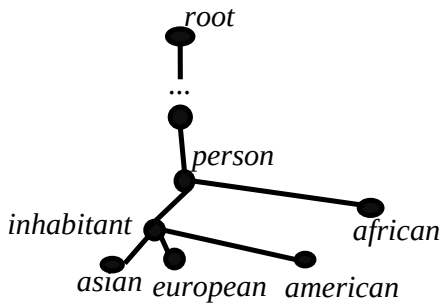

FIG. 5.6 – Extrait de Wordnet

La différence entre le résultat des mesures de similarité et les jugements humains (même en contexte) ne peuvent pas corréler si la structure même de WordNet est pour le moins inattendue. Le même problème existe entre *alsacien* et *breton* où l'un est classé en dessous d'*inhabitant* et l'autre sous *french person*.
D'ailleurs un autre problème existe avec ce dernier synset. WordNet différencie *french person* (une personne de nationalité française) de *french people*[96] (une personne vivant en France) (voir figure 5.7 et 5.8 pour les synsets et leurs hypéronymes).
Cette différenciation, devrait aussi s'appliquer à toutes les personnes, on devrait avoir par exemple *german* et *german person* ce qui n'est pas le cas. Nous avons donc, les similarities suivantes :

sim(mot#catégorie#numsens, mot#catégorie#numsens)	Trace d'exécution)
$sim(french\_person\#n\#1, german\#n\#1) = 0.55$	$ppac : european\#n\#1$ $profondeur = 5$ $profondeur_{\perp} = 6$ $profondeur(french\_person\#n\#1) = 6$ $profondeur(german\#n\#1) = 6$
$sim(french\#n\#2, german\#n\#1) = 0$	$ppac : noppac$

TAB. 5.5 – Similarité entre *french*, *french person* et *german*

S: (n) French#2, French people#1 (the people of France)
◦ ***direct hypernym*** / *inherited hypernym* / *sister term*
• S: (n) nation#2, land#8, country#3 (the people who live in a nation or country)
• S: (n) people#1 ((plural) any group of human beings (men or women or children) collectively)
• S: (n) group#1, grouping#1 (any number of entities (members) considered as a unit)
• S: (n) abstraction#6 (a general concept formed by extracting common features from specific examples)
• S: (n) abstract entity#1 (an entity that exists only abstractly)
• S: (n) entity#1 (that which is perceived or known or inferred to have its own distinct existence (living or nonliving))

FIG. 5.7 – Synset de *french* et ses hypéronymes

[95] African is a native or **inhabitant** of Africa.
[96] Synonyme de french.

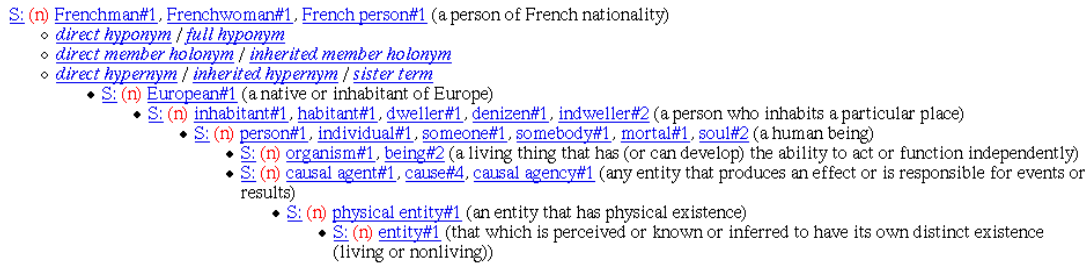

FIG. 5.8 – Synset de *french person* ainsi que son hypéronyme

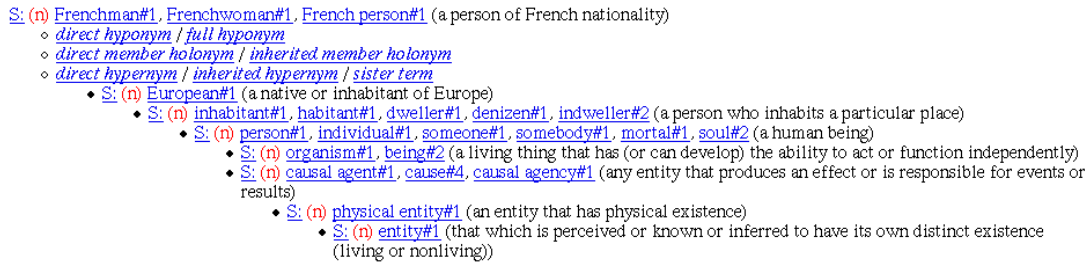

FIG. 5.9 – Synset de *german* et ses hypéronymes

Les quelques problèmes pointés ci-dessus auraient pu être détectés si WordNet était représenté formellement. La formalisation permet de bénéficier des inférences, mais aussi de s'assurer de la cohérence de l'ontologie. Elle permet l'explicitation des caractéristiques des concepts. Or, le fait qu'un synset subsume un autre synset signifie qu'ils partagent des caractéristiques mais ces caractéristiques ne sont pas explicitées dans WordNet. La formalisation de WordNet n'est pas sans poser des problèmes. Gangemi *et al.* [2001] s'est essayé à la formalisation des hauts niveaux de WordNet et a pointé plusieurs difficultés de conceptualisation. Charlet *et al.* [2003] affirment même que "WordNet n'est pas une ontologie, cela n'en prend pas le chemin et essayer de l'utiliser tel quel ou avec un minimum de modification dans un système formel est voué à l'échec". WordNet peut néanmoins servir de ressource pour aider à la construction d'ontologie. Le Web sémantique apporte une réponse à ces problèmes, en proposant d'intégrer des ontologies formelles, il facilite l'interprétation des connaissances aux agents mais assure aussi la cohérence des catégorisations et les choix de structurations sont ainsi explicités. Nous proposons d'appliquer notre mesure de similarité à des ontologies formelles. Nous utilisons un algorithme de calcul du Plus Petit Généralisant (notion équivalente au *ppac*).

5.9 Calcul de similarité dans une ontologie formelle

La formalisation de l'ontologie nous permet d'utiliser des ontologies cohérentes, les calculs peuvent ainsi être faits avec plus de confiance. Nous proposons de calculer notre mesure de similarité sur des ontologies en OWL. La formule de la mesure reste la même, la nouveauté réside dans la construction du Plus Petit Généralisant (PPG). Pour cela, nous adaptons les travaux de

[Ventos, 1997] sur C-CLASSIC [Borgida & Patel-Schneider, 1994][97](une variante de la logique de description CLASSIC) à OWL-Lite en restreignant les constructeurs à ceux de C-CLASSIC.

Le problème de granularité et de densité reste inhérent à n'importe quelle structure taxonomique même formelle. Nous montrons dans ce qui suit comment le PPG nous permet de réduire ce problème.

Les calculs de similarité sont largement influencés par la cohérence des ressources utilisés, nous avons montré dans la section précédente que les meilleures mesures de similarité peuvent échouer si le réseau est mal structuré. Ainsi si nous reprenons l'exemple dans la figure 5.6, voici les définitions de WordNets (les gloss) donnés aux synsets *inhabitant*, *african* et *european* :
Inhabitant is a **person** who inhabits a particular **place**
African is a native or **inhabitant** of **Africa**
European is a native or **inhabitant** of **Europe**

Nous pouvons les décrire semi-formellement par :
Inhabitant Is-a Person (AND inhabits :Place)
African Is-a Inhabitant (AND inhabits :Africa)
European Is-a Inhabitant (AND inhabits :Europe)

Il devient évident que *African* est à classer au dessous de Inhabitant (et ceci même s'il est décrit par *African Is-a Person (AND inhabits :Africa)*).

Dans un précédent travail [Zargayouna, 2001] ; [Zargayouna *et al.*, 2001] dans le cadre d'une Recherche d'Information multimédia, nous avons calculé les similarités entre des descriptions formalisées dans une logique de description : C-CLASSIC [Borgida & Patel-Schneider, 1994]. La similarité entre deux concepts est évaluée en calculant le **Plus Petit Généralisant : PPG** (*Least Commun Subsumer*) les subsumant. Nous utilisons les travaux de Ventos [Ventos, 1997] pour calculer une description symbolique de ce PPG, ce dernier n'est pas forcément nommé il peut de ce fait ne pas exister dans l'ontologie. Ceci est intéressant car nous arrivons ainsi à déterminer des similarités à une granularité plus fine que celle de l'ontologie. De plus nous utilisons, dans le calcul du PPG, toutes les relations (rôles) qui existent entre les concepts.

Une des limites de ce travail est le manque de relation d'ordre total entre les similarités. Nous savons que PPG(A, B) = C et PPG (D,E) = F mais ceci ne nous permet pas de les comparer entre eux, car les seules comparaisons possibles correspondent au cas où un PPG subsume un autre (C subsume F, par exemple). Ce problème peut être résolu par les mesures numériques de calcul de la similarité conceptuelle. Si sim(A, B)=0.6 et sim(D,E)=0.75, nous pouvons dire que D est plus proche de E que A de B.

Nous proposons de calculer le PPG entre deux concepts et de calculer la similarité par rapport à ce PPG à la place du $ppac$, le calcul de similarité reste le même, c'est la définition du plus petit ancêtre commun qui change. Ainsi nous pallions la limite du calcul en nombre d'arc, en prenant en considération les rôles.

Nous commençons par présenter C-CLASSIC ainsi que les fondements du calcul du PPG.

[97]Cette logique de description a le mérite de présenter un bon compromis expressivité/complexité et complétude.

5.9.1 Le langage C-CLASSIC

La syntaxe de C-CLASSIC est définie par induction à partir de :
- Un ensemble R de rôles atomiques ;
- Un ensemble P de concepts atomiques ;
- Des constantes $\top$ et $\bot$ qui correspondent respectivement au concept le plus général et au concept le plus spécifique ;
- Un ensemble I d'individus ;
- De la règle syntaxique suivante : (C et D sont des concepts, P est un concept atomique, R est un rôle atomique, n un nombre réel, m est un nombre entier et I_i des individus).

C,D ← T
| $\top$ // concept le plus général
| $\bot$ // concept le plus spécifique
| P // concept atomique
| ONE-OF $\{I1, .., I_n\}$ // concept en extension
| MIN n // ensemble de réels > n
| MAX n // ensemble de réels $\leq$ n
| C $\sqcap$ D // conjonction de concept
| $\forall$ R :C //précise le co-domaine de R
| R FILLS $\{I_1..I_n\}$ //restriction de valeurs pour R
| R AT-LEAST m //restriction de nombre pour R (minimum)
| R AT-MOST m //restriction de nombre pour R (maximum)

Le constructeur $\sqcap$ dénote la conjonction de concepts. Le constructeur ONE- OF permet de définir un domaine de valeurs en donnant la liste des individus. Il est à noter que ces individus ne possèdent aucune propriété et aucun fait de la A-BOX[98]. La quantification universelle ($\forall r : C$) précise le co-domaine du rôle r. Le constructeur $FILLS$ associe une ou plusieurs valeurs à un rôle, *AT-LEAST* (respectivement *AT-MOST*) sont des restrictions de cardinalité qui fixent le nombre minimal (respectivement maximal) de valeurs que peut prendre le rôle.

5.9.2 La forme normale

Du point de vue algorithmique, les descriptions de concepts sont difficilement manipulables. En effet, pour pouvoir déterminer une relation de subsomption entre deux concepts à partir de leurs descriptions, il faut pouvoir les comparer. Afin de rendre les termes de C-Classic facilement utilisables, une forme normale des descriptions a été définie appelée forme normale descriptive[99].
La forme normale d'un concept décrit par un terme T est de la forme :
(dom, min, max, P, r)
où :

[98] Ces individus sont appelés "individus classic" et ont donc un statut particulier, se référer à [Borgida & Patel-Schneider, 1994] pour plus de détails

[99] Dans [Ventos, 1997] C-CLassic a été doté d'une **sémantique intentionnelle** où les concepts sont dénotés par leur forme normale de l'ensemble de leurs propriétés.

- dom est un ensemble d'individus si la définition de T contient une propriété $ONE-OF$ où le symbole $UNIV$ qui représente l'absence de propriété $ONE-OF$ dans le terme T ;
- min (respectivement max est un réel si T contient une propriété MIN (respectivement MAX), où le symbole ***MIN-R*** (respectivement ***MAX-R***) sinon ;
- P est l'ensemble des concepts atomiques contenus dans T ;
- r est un ensemble d'éléments, chaque élément étant défini comme suit : $< R, fillers, least, most, c >$ où :
  - R est un nom de rôle ;
  - $fillers$ est un ensemble d'individus si T contient une propriété R FILLS dans sa définition, $\emptyset$ sinon ;
  - $least$ (respectivement $most$) est un entier représentant une propriété AT-LEAST (respectivement AT-MOST) ou 0 (respectivement NOLIMIT) sinon ;
  - c est la forme normale de C si T contient $(\forall r : C)$ dans sa définition ou t sinon (t est une constante représentant la forme normale du concept $\top$).

Principe de calcul de la forme normale

Pour obtenir une forme normale, on peut appliquer différentes stratégies de normalisation. La stratégie de normalisation choisie par Ventos est de calculer **la saturation**, la forme normale obtenue est une forme normale saturée. Ce qui consiste à ajouter les informations implicites, par exemple R FILLS $\{I_1..I_n\}$ devient R FILLS $\{I_1..I_n\} \sqcap R\ AT-LEASTn$. Ceci permet de simplifier considérablement la définition des opérations du PPG.
On définit, par exemple, un citoyen français (*CitoyenFrancais*) qui est un citoyen qui a pour pays la France :
CitoyenFrançais = Citoyen $\cap \forall$ a-pour-pays :France.
La forme normale du concept est :
(UNIV, MIN-R, MAX-R, Citoyen,<a-pour-pays, $\emptyset$,0, NOLIMIT, France>)
Nous supposons dans cet exemple que Citoyen et France sont des concepts primitifs, nous détaillerons plus loin cet exemple pour illustrer le calcul du PPG.
Le calcul d'une forme normale se fait en temps polynomial, la comparaison de deux formes normales se fait en temps polynomial et la taille d'une forme normale correspondant à un terme est polynomiale en fonction de la taille du terme. Les preuves de ces théorèmes se trouvent dans [Ventos, 1997].

5.9.3 L'algorithme du PPG

L'algorithme du PPG calcule le plus petit généralisant entre deux concepts représentés par leurs formes normales. Les opérations de base du calcul du PPG sont l'union(dom), l'intersection($P, fillers$) et la recherche du minimum et maximum de deux entiers ou de deux réels $(least, most, min, max)$. On cherche le Plus Petit Généralisant de deux concepts A1 et A2.
Soient a1 et a2 les formes normales de A1 et A2 avec :

$a1 = (a1.dom, a1.min, a1.max, a1.P, a1.r)$ et

$a2 = (a2.dom, a2.min, a2.max, a2.P, a2.r)$.

Soit $appg$ le résultat du PPG avec $appg = (appg.dom, appg.min, appg.max, appg.P, appg.r)$.

Algorithme PPG (a1, a2)

DEBUT
appg.dom ← a1.dom $\bigcup$ a2.dom
appg.min ← min (a1.min , a2.min)
appg.max ← max (a1.max , a2.max)
appg.P ← a1.P $\bigcap$ a2.P
appg.r ← $\emptyset$
Pour chaque $< R, fillers1, least1, most1, c1 >$ de a1.r Faire
Si il existe $< R, fillers2, least2, most2, c2 >$ appartenant à a2.r Alors
Début c ← PPG (c1, c2)
fillers ← fillers1 $\bigcap$ fillers2
least ← min (least1 , least2)
most ← max (most1 , most2)
appg.r ← appg.r $\bigcup$ $< R, fillers, least, most, c >$
FinSi
FinFaire
Retourner(appg)
FIN

Nous présentons dans ce qui suit un exemple qui montre le calcul du PPG ainsi que son intérêt pour notre calcul de similarité.
*C*itoyen ≡ Personne
CitoyenFrançais ≡ Citoyen ⊓ ∀ a-pour-pays :France
CitoyenAllemand ≡ Citoyen ⊓ ∀ a-pour-pays :Allemagne
CitoyenTurc ≡ Citoyen ⊓ ∀ a-pour-pays :Turquie
France ≡ Pays ⊓ localisé-dans Fills{Europe} ⊓ a-langue Fills{Français}
Allemagne ≡ Pays ⊓ localisé-dans Fills{Europe} ⊓ a-langue Fills{Allemand}
Turquie ≡ Pays ⊓ localisé-dans Fills{Asie} ⊓ a-langue Fills{Turc}

Les formes normales des concepts *CitoyenFrançais*, *CitoyenAllemand*, *CitoyenTurc* sont les suivants :

FN(CitoyenFrançais) = (UNIV, MIN-R, MAX-R, Citoyen,<a-pour-pays, ∅, NOLIMIT, (UNIV, MIN-R, MAX-R, Pays, <localisé-dans, Europe,1, NOLIMIT,t><a-langue, Français,1, NOLIMIT,t>)>)

FN(CitoyenAllemand) = (UNIV, MIN-R, MAX-R, Citoyen,<a-pour-pays, ∅, NOLIMIT, (UNIV, MIN-R, MAX-R, Pays, <localisé-dans, Europe,1, NOLIMIT,t><a-langue, Allemand,1, NOLIMIT,t>)>)

FN(CitoyenTurc) = (UNIV, MIN-R, MAX-R, Citoyen,<a-pour-pays, ∅, NOLIMIT, (UNIV, MIN-R, MAX-R, Pays, <localisé-dans, Asie,1, NOLIMIT,t><a-langue, Turc,1, NOLIMIT,t>)>)

Le PPG entre les formes normales de *CitoyenFrançais* et *CitoyenAllemand* :

PPG(FN(CitoyenFrançais), FN(CitoyenAllemand)) = (UNIV, MIN-R, MAX-R, Citoyen,<a-pour-pays, (,0, NOLIMIT, (UNIV, MIN-R, MAX-R, Pays, <localisé-dans, Europe,1, NOLIMIT,t><a-langue, ,1, NOLIMIT,t>)>)

Ce qui peut correspondre à *CitoyenEuropéen* (voir figure 5.10) :
CitoyenEuropéen $equiv$ Citoyen ⊓ ∀ a-pour-pays : (Pays ⊓ localisé-dans Fills{Europe} ⊓ a-langue AT LEAST 1)

Par contre le PPG entre les formes normales de *CitoyenFrançais* et *CitoyenTurc* :

PPG(FN(CitoyenFrançais), FN(CitoyenTurc))=
(UNIV, MIN-R, MAX-R, Citoyen,<a-pour-pays, (,0, NOLIMIT, (UNIV, MIN-R, MAX-R, Pays, <localisé-dans, ,1, NOLIMIT,t><a-langue, ,1, NOLIMIT,t>)>)
Ce qui est équivalent à *Citoyen* (voir figure 5.10) :

Citoyen $equiv$ Citoyen ⊓ ∀ a-pour-pays :(Pays ⊓ localisé-dans AT-LEAST 1 ⊓ a-langue AT-LEAST 1)

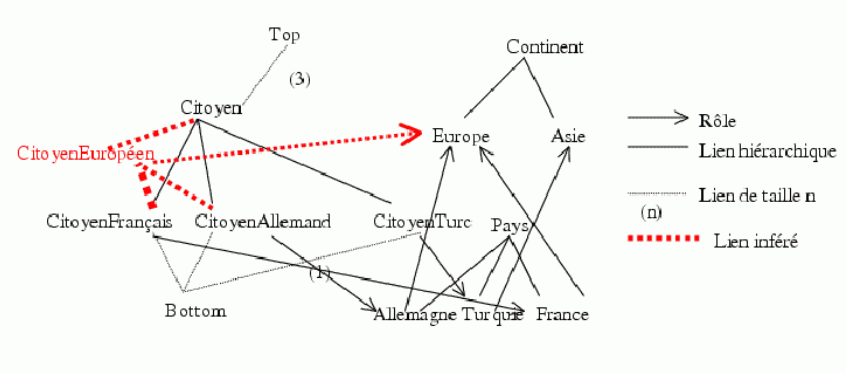

FIG. 5.10 – Exemple de PPG

Si nous calculons la similarité en prenant en compte seulement les liens taxonomiques, nous obtenons :

$sim(CitoyenFrancais, CitoyenAllemand) = \frac{2*3}{4+4+2} = 0,6$

$sim(CitoyenFrancais, CitoyenTurc) = \frac{2*3}{4+4+2} = 0,6$

En prenant en compte le PPG, les similarités changent par la création du concept virtuel *CitoyenEuropéen* subsumé par Citoyen et subsumant *CitoyenFrançais* et *CitoyenAllemand* :
$sim_{ppg}(CitoyenFranais, CitoyenAllemand) = \frac{2*4}{5+5+2} = 0,66$

$sim_{ppg}(CitoyenFranais, CitoyenTurc) = \frac{2*3}{4+4+2} = 0,6$
Nous nous limitons dans ce travail à la T-Box ainsi qu'aux opérateurs définis dans C-CLASSIC. Nous présentons dans le tableau 5.9.3 les constructeurs de OWL Lite qui sont pris en considération par rapport à ceux de C-CLASSIC.

Opérateurs C-CLASSIC	Syntaxe abstraite OWL-Lite
$\top$	owl :Thing
$\bot$	owl :Nothing
C $\sqcap$ D	intersectionOf (C, D)
Min m	restriction (base = xsd :float xsd :minExclusive value = "m")
Max m	restriction (base = xsd :float xsd :maxInclusive value = "m")
ONE-OF $\{I_1..I_n\}$	// Le ONE-OF $\{I_1..I_n\}$ de C-CLASSIC n'est pas le même que le oneOf$\{I_1..I_n\}$ de OWL (qui n'existe pas dans OWL-Lite mais dans OWL-DL et Full). Comme noté plus haut, les individus du ONE-OF et de FILLS ne sont pas relié à la A-BOX.
$\forall$ R :C	restriction(R allValuesFrom (C))
R FILLS $\{I_1..I_n\}$	restriction(R hasValue $\{I_1..I_n\}$)
R AT-LEAST m	restriction (R minCardinality (m)[100])
R AT-MOST m	restriction (R maxCardinality (m))

Nous n'utilisons pas tous les constructeurs de OWL-Lite, mais il s'agit dans un premier temps de restreindre les constructeurs pour maîtriser la complexité.
L'algorithme du PPG nous permet de prendre en considération tous les types de relations entre concepts et nous permet d'avoir une mesure de similarité qui prend en considération le nombre d'arcs ainsi que la richesse des descriptions des concepts.
Nous avons implémenté cet algorithme pour C-CLASSIC [Zargayouna, 2001], son adaptation pour prendre en considération les opérateurs de OWL est envisagé.

5.10 Conclusion

Nous avons présenté une mesure de similarité sémantique qui nous permet d'évaluer la similarité entre deux concepts de l'ontologie. Cette mesure prend en considération la spécificité et la spécialisation des concepts et permet de faire les calculs dans n'importe quelle structure

taxonomique. Nous avons proposé, dans le cadre d'une ontologie formelle, l'utilisation d'un algorithme de calcul du Plus Petit Généralisant (PPG) qui permet de prendre en considération la richesse des descriptions des concepts.
La mesure de similarité sémantique a comme champ d'application la recherche d'information et le TAL. Elle a connu récemment un renouveau d'intérêt avec le Web sémantique. Euzenat & Valtchev [2004] proposent une mesure de similarité pour comparer des entités de deux ontologies fondées sur une comparaison qui prend en considération les types des entités ainsi que leurs descriptions. Cette mesure sert pour l'alignement de deux ontologies décrites en OWL-Lite. Bach *et al.* [2004] proposent un algorithme d'appariement d'ontologies qui peut servir pour l'intégration ou la comparaison d'ontologies. Cet algorithme utilise une mesure de similarité qui exploite toutes les informations disponibles (e.g. les noms, les étiquettes, les descriptions, etc.). Dans la même lignée [Giunchiglia & Shvaiko, 2004] proposent un algorithme d'appariement sémantique qui recherche des correspondances entre les concepts en prenant en considération leur place dans l'ontologie ainsi que les relations de généralisation/spécialisation.
Dans les chapitres suivants nous présentons notre modèle d'indexation SemIndex (Semantic Indexing) ainsi que notre langage de requête SemIR (Semantic Information Retrieval). La mesure de similarité sémantique sert à enrichir les termes d'indexation ainsi que ceux de la requête.

Chapitre 6

SemIndex : Indexation sémantique de documents XML

Sommaire

6.1 Introduction

La recherche d'information semi-structurée est un domaine de recherche récent en plein essor, nous avons présenté dans le chapitre 2, les différents travaux liés à cette problématique.
Le modèle vectoriel a largement démontré son efficacité dans les différentes campagnes d'évaluation TREC dans le cadre d'une recherche d'information classique (chapitre 1) ainsi que dans la campagne d'évaluation INEX dans le cadre d'une recherche d'information semi-structurée (chapitre 2). Il s'avère donc être un bon candidat pour la recherche d'information semi-structurée et conforte en cela nos intuitions.
Nous avons exposé dans le chapitre 2, les nouveaux défis posés par la recherche d'information semi-structurée :

– Comment prendre en considération la dimension structurelle ? Ceci nécessite la définition des éléments de structure à indexer.
– Quelle politique choisir pour le choix des termes d'indexation ? Doivent-ils être indexé par rapport à l'élément où ils apparaissent ? comment éviter la redondance dans l'indexation des termes ? ce qui pose la question de la nature des relations entre les éléments de structures, doivent-ils être considérés comme disjoints ou imbriqués ?
– La définition des relations entre les éléments de structure influence le choix de la pondération des termes d'indexation. Cela nécessite la définition des poids des termes en fonction de la politique choisie pour les éléments de structure.
– La définition d'une mesure de concordance entre les requêtes et les unités indexées.
– Quelle est l'unité d'information pertinente à retourner ? ce qui amène à définir une fonction d'agrégation de la mesure de concordance au niveau de la structure des documents.

Les trois premières problématiques concernent l'indexation, les deux dernières concernent l'interrogation et seront traitées dans le chapitre suivant.

Nous prenons aussi en considération la dimension sémantique, tant au niveau des termes que de la structure. Cependant très peu de travaux s'intéressent à combiner les aspects structurels des documents avec l'aspect sémantique de leur contenu. A notre connaissance seul le système XXL (cf chapitre 3) intègre une ontologie dans le processus d'indexation. Nous avons présenté, dans le chapitre précédent, une mesure de similarité sémantique qui évalue le degré de similarité entre deux concepts. Nous exposerons dans la section 6.5 comment nous incorporons cette mesure dans notre index, afin de garantir une gestion transparente à l'utilisateur de l'ontologie au niveau de la recherche. Les deux questions fondamentales au niveau de l'indexation sont donc :

– Comment prendre en considération la **structure** ainsi que les **termes**, notre but étant de concevoir un système générique qui prend en considération n'importe quelle structure et traite ainsi des corpus hétérogènes ?
– Comment prendre en considération la **sémantique** des documents tout en gardant une souplesse d'interrogation par les mots clés ?

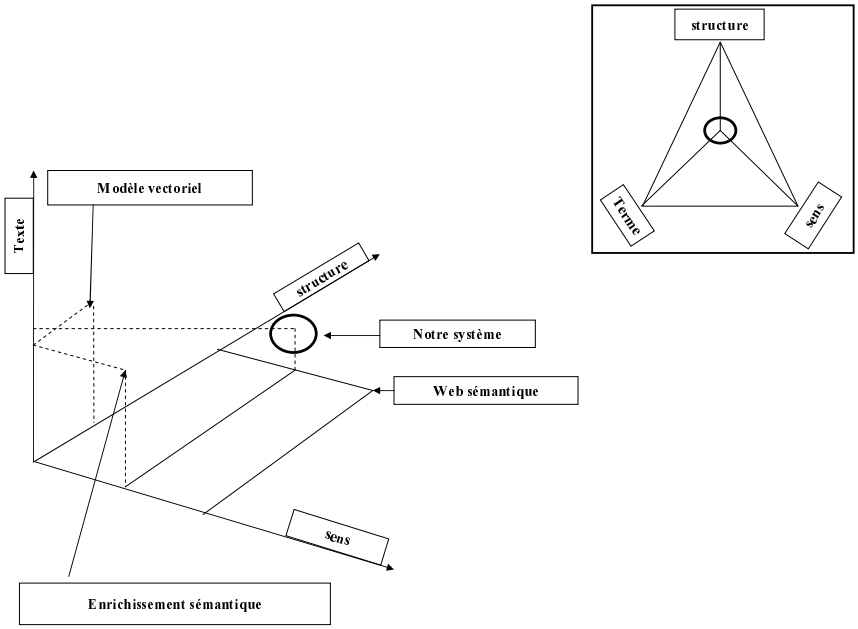

FIG. 6.1 – Objectif de notre système : structure, texte et sens

Le chapitre est organisé de manière à répondre graduellement à ces questions. Ainsi, nous exposons les trois entrées de notre index : par le contenu textuel, par le contenu structurel et par le contenu sémantique (voir figure 6.1). Nous exposons également la manière dont ces trois éléments sont combinés de manière à garder leur indépendance en cas de l'ignorance de l'un d'eux. Par une telle approche, notre modèle logique d'index supporte chacun ou des combinaisons de ces éléments lors de la recherche.
Nous présentons dans la section 6.2 la représentation du contenu textuel des documents. Nous présentons ensuite dans la section 6.3, la représentation du contenu structurel. L'adaptation du modèle vectoriel, par rapport à ces deux dimensions, est présentée dans la section 6.4. Nous présentons, enfin, la représentation du contenu sémantique dans la section 6.5 ainsi que son intégration dans l'index dans la section 6.6.

6.2 Contenu textuel des documents

Les corpus documentaires visés essentiellement par notre approche sont des documents XML présentant un contenu **structurel** et un contenu **textuel**. Afin que notre système supporte des requêtes par le contenu (mots clés) , nous proposons, dans un premier temps, une entrée par le texte en ignorant la structure conformément aux méthodes d'indexation classiques.
Nous procédons à l'extraction des termes en appliquant les outils de TAL traditionnellement utilisés en RI, ce qui correspond à :

- ignorer les mots qui appartiennent à des anti-dictionnaires (*stop lists*) ; c'est à dire les listes pré-définies qui contiennent les mots qui peuvent être ignorés (comme les mots communs) ;
- prendre la forme canonique des mots (*lemme*), nous utilisons pour cela le TreeTagger ;
- ne prendre en considération que les mots pleins. Les unités linguistiques seront restreintes aux catégories suivantes : noms et adjectifs. C'est dans un but de simplification que nous

procédons à cette épuration, il est évident que les verbes et autres structures grammaticales sont porteurs d'information.

Les techniques de TAL sont beaucoup plus riches que les outils présentés. L'apport des études morphologiques, syntaxiques et sémantiques (avec plus ou moins de robustesse) est évident pour la RI, notamment pour retrouver des unités d'index plus pertinentes (cf. chapitre 1). Nous préférons, dans un premier temps, mettre en oeuvre des outils robustes qui ne nécessitent pas une validation. Notre système reste néanmoins générique et peut permettre une phase d'extraction plus fine, notamment pour extraire des termes composés.

Les lemmes sélectionnés constituent l'entrée textuelle de notre index. Il est important, cependant, de noter que contrairement aux pratiques classiques, nous ne procédons pas à une épuration par rapport à la fréquence des termes. Même si les termes ne caractérisent pas le contenu d'un document (ou d'un fragment de document), ils nous sont utiles pour guider le calcul des similarités sémantiques entre les termes. Ils jouent un rôle important pour la determination du contexte sémantique, nous reviendrons sur ce point plus loin.

Dans ce qui suit, nous notons par Ψ la liste des étiquettes (noms) de balises[101] et Ω l'ensemble des termes (lemmes) issus de la phase d'extraction qui représentent le contenu textuel des balises.

6.3 Contenu structurel des documents

6.3.1 Arbre réduit

Les documents XML offrent la possibilité de délimiter leur contenu en un ensemble d'unités sémantiques. Toutefois, ce découpage reste arbitraire et une même unité sémantique peut être représentée selon des structures différentes. Ainsi, par exemple pour représenter une adresse nous pouvons avoir les structures suivantes :

```
<adresse>                    <adresse>
45 avenue de Choisy           <rue>45 avenue de Choisy</rue>
75013                         <codePostal> 75013 </codePostal>
Paris                         <ville> Paris </ville>
</adresse>                    </adresse>
```

Néanmoins, la structure offre un apport sémantique non négligeable. Dans [Schlieder & Meuss, 2002] il est même affirmé que : "Ignorer la structure du document revient à ignorer sa sémantique". La structuration hiérarchique du document peut être associée à la notion de **contexte documentaire** (cf. chapitre 1). L'objectif d'un contexte documentaire est de mieux prendre en compte la structure des documents [Habert *et al.*, 1997]. Un tel contexte documentaire est défini comme une unité textuelle à l'intérieur d'un document, il peut représenter la structure logique d'un document (paragraphe, section, chapitre) mais aussi une structure thématique. Ces unités textuelles peuvent avoir des relations entre elles, de plus en plus de travaux visent à trouver ces

[101] Un élément est composé d'une balise ouvrante et d'une balise fermante, dans la suite nous parlerons indifféremment de balise ou d'élément

relations et à les représenter par des annotations [Hernandez, 2004]. La structuration d'un document devient intéressante quand elle dépasse la représentation en paragraphe, section, etc. et prend en compte le contenu "sémantique" du texte. Les travaux sur la structuration thématique de texte peuvent s'avérer précieux pour avoir des documents annotés guidés par le contenu (cf. chapitre 4) . Nous faisons le pari d'une telle structuration, notamment pour les nouveaux documents dans le Web Sémantique où les balises seraient plus pertinentes que . Nous modélisons la structure d'un document XML comme un arbre étiqueté où chaque élément (et attribut) correspond à un nœud. Nous ne faisons aucune distinction entre les éléments et les attributs (voir figure 6.2).

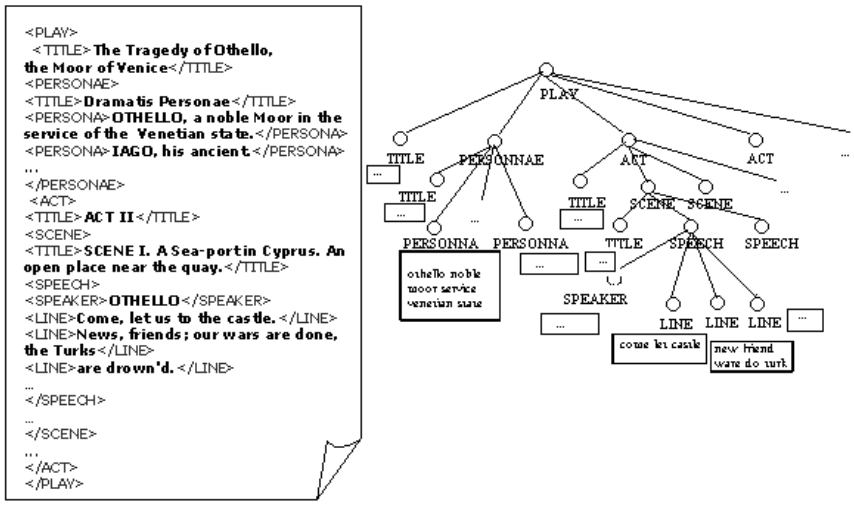

FIG. 6.2 – Exemple de document XML avec son arbre étiqueté

Définition

Un arbre étiqueté d'un document d, noté AE_d, est un n-uplet $< n_0, N, A, label, W >$
Où :

- N est un ensemble fini de nœuds
- n_0 est le nœud racine,
- $A \subset N \times N$ est un ensemble fini d'arcs avec
  - pour $n_i, n_j \in N$ on a $(n_i, n_j) \in A$ ssi n_i est le père direct du nœud n_j
  - on note par $suiv(n) = \{n_i | (n, n_i) \in A\}$ et par $pred = suiv^{-1}$, $\forall n \neq n_0$ on a $|pred(n)| = 1$. Un nœud possède un seul père.
- $label : N \rightarrow \Psi$ est une fonction qui associe à un nœud une étiquette.
- $W : N \rightarrow 2^{\Omega}$ avec $W(n) = \{t \in \Omega | t$ est un terme apparaissant dans $n\}$

Un chemin est un ensemble de nœuds $n_1, .., n_n$ tel que $\forall n_i, n_{i+1} \exists (n_i, n_{i+1}) \in A$ (i :1..n).
Notre modèle d'indexation n'oblige pas à ce que les documents aient une structure commune par rapport à une DTD ou un schéma XML. Nous extrayons les structures des documents à partir de leur représentation en arbre. Nous faisons l'hypothèse que deux nœuds ayant le même label et le même chemin à partir de la racine, représentent le même contexte. Nous approximons la grammaire de chaque document en regroupant ensemble ces nœuds. Ainsi nous effectuons une abstraction pour définir un modèle de documents qui se rapproche de sa grammaire. Cette démarche a pour but de simplifier la représentation du document et de ne garder que les éléments de la sémantique structurelle qui le définissent. L'abstraction de la représentation en arbre du document est appelée **arbre réduit** et est définie comme suit :

Définition

Nous définissons un arbre réduit d'un document d par un arbre étiqueté $AR_d =< n_0, N', A', label, W' >$ où :

- $N' \subset 2^N$ où $\forall n' \in N'$, on a $n' = \{n \in N | \forall n_j, n_i \in n'$ on a $label(n_i) = label(n_j)$ et $chemin(n_0, n_i) = chemin(n_0, n_j)\}$. On appelle **instance** de n' l'ensemble des $n \in n'$, noté $inst(n')$. $label(n)\#i$ est le ième élément de cet ensemble.
- $A' \subset N' \times N'$ où $\forall (n, n') \in A'$ on a $\forall n_j \in n' \; \exists n_i \in n$ avec $(n_i, n_j) \in A$
- $W'(n') = \bigcup_{n_i \in n'} W(n_i)$. W' associe à un nœud n' de l'arbre minimal, l'ensemble des termes apparaissant dans n.

L'arbre réduit revient à essayer de déduire une grammaire possible pour la structure d'un document. La génération de l'arbre réduit de tous les documents nous permet d'approximer les DTDs possibles.

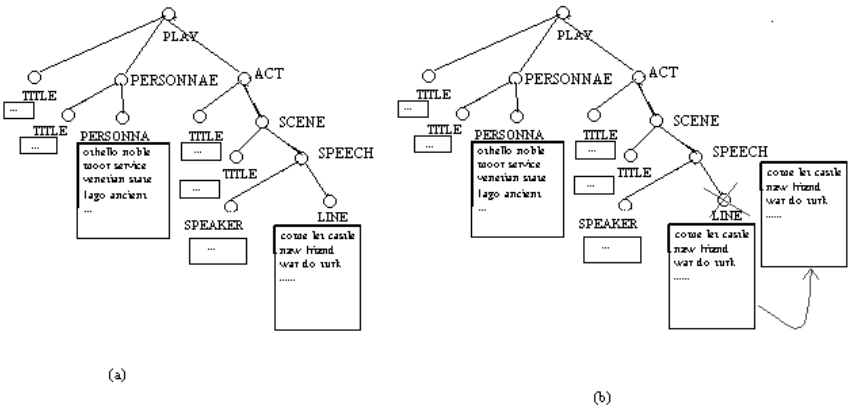

FIG. 6.3 – Exemple d'arbre réduit du document présenté dans la figure 6.2

Dans la figure 6.3 (a), nous présentons un exemple d'arbre réduit du document de la figure 6.2. L'algorithme de construction de l'arbre réduit est polynomial en fonction de la profondeur de l'arbre. Son principe peut se résumer comme suit :
– Nous commençons par $N' = \{\{n_0\}\}$ et $A' = \emptyset$
– Nous dépilons n' :

1. Nous calculons $suiv(n') = \bigcup_i suiv(n_i)$ pour $n_i \in n'$
2. Pour chaque sous-ensemble de nœuds de $suiv(n')$ partageant la même étiquette, on crée un nouveau nœud n'' portant le même label et nous l'empilons $(N' = N' \cup \{n''\})$
3. Nous rajoutons à $A' = A' \cup (n', n'')$

– L'algorithme se termine quand la pile est vide.

6.3.2 Indexation de la structure

La structure d'un document est un élément non négligeable de sa sémantique. Il est évident que pour un contenu informationnel donné, il existe plusieurs découpages ou structurations possibles. Mais quelque soit le degré d'arbitraire dans ce découpage, il existe une sémantique appliquée aux éléments informationnels qu'il est important de mettre en évidence dans le processus d'indexation et par conséquent dans le processus de recherche. L'index que nous proposons possède une entrée par la structure où chaque document est indexé par ses unités structurelles. Deux points doivent dès lors être clarifiés :
– Nous devons d'abord définir quelle sera l'unité de base pour l'indexation structurelle i.e. la plus petite unité sémantique de structuration.
– Quelles seront les indicateurs d'indexation (comment les quantifier ?) compte tenu du choix des unités d'index

Nous définissons la notion de contexte structurel qui est fondé sur l'hypothèse que le contenu textuel d'un nœud de l'arbre réduit constitue son vocabulaire et définit de ce fait un contexte.

Contexte structurel

Dans un document XML, on peut distinguer trois types d'éléments :
– **Élément de structuration :** ce sont les éléments qui ne portent pas de contenu textuel. Ils jouent essentiellement un rôle de structuration.
– **Élément feuille :** ce sont généralement les éléments porteurs de texte.
– **Élément mixte :** ce sont des éléments internes (non feuilles) qui comportent aussi bien un contenu textuel qu'un contenu structurel.

Dans notre index, une de nos préoccupations est de capter l'information de structuration afin d'affiner la compréhension du texte. Dans cette perspective, nous nous intéressons aux deux derniers types d'éléments, c'est à dire ceux qui comportent du texte que nous appelons *contextes*. Nous verrons par la suite que ce choix réduit considérablement la taille de l'index, sans pour autant perdre de l'information moyennant le processus de propagation. Dans ce qui suit, les représentations de la structure du document sont des arbres réduits.

Définition

Soit un arbre réduit AR_d=$< n_0, N', A', label, W' >$ on définit un **modèle d'élément** noté E_e^σ avec $e = label(n')$ pour $n' \in N'$ et $\sigma = label(n_0), .., label(n_i), .., label(n_n - 1)$ tel que $n_0, ..n_i, ..n_{n-1}, n$ est un chemin de AR_d . On note par E_d^{AR} l'ensemble des modèles d'élément de l'arbre.

L'ensemble des modèles d'éléments pour l'exemple de la figure 6.3 est $E_d^{AR} = \{E_{play}^{\emptyset}, E_{title}^{play}, E_{personnae}^{play}, E_{title}^{play,personnae}, E_{speaker}^{play,act,scene,speech}, ... \}$.
Il faut noter ici que deux nœuds portant le même label peuvent donner lieu à deux éléments différents si les chemins qui les relient à la racine sont différents.
Notons aussi que l'**instance** d'un modèle d'élément $inst(E_e^\sigma)$ correspond à $inst(n')$
Pour chaque modèle une seule occurrence existe dans un document réduit (e.g. E_{title}^{play}, $E_{title}^{play,personnae}$).
Les instances du modèle E_{act}^{play} sont $E_{act\#1}^{play}$, $E_{act\#2}^{play}$, etc.
Les modèles d'éléments peuvent être validées par l'utilisateur en fonction des éléments qui sont pertinents pour l'interrogation, ainsi par exemple si on n'a pas besoin d'une structuration qui va jusqu'à $< Line >$, on peut ignorer le modèle $E_{line}^{play,act,scene,speech}$, son contenu textuel est alors affecté au modèle d'élément englobant $E_{speech}^{play,act,scene}$ (figure 6.3 (b)).
Comme indiqué plus haut, nous considérons que l'essentiel d'une structuration dans notre modèle se rapporte donc au *modèle d'élément* porteur de texte et nous appelons cette classe de modèle d'élément ***contexte structurel***.

Définition

Soit l'arbre réduit $AR_d =< n_0, N', A', label, W' >$ On appelle ***contexte structurel*** noté tout modèle d'élément $C_e^\sigma \in E_d^{AR}$, tel que $W'(n') \neq \emptyset$. On note par C_d^{AR} l'ensembles de contextes qui définissent le document d. On dit que un document d *réalise un contexte* c ($c \in C_d^{AR}$) s'il existe au moins une instance de ce contexte dans le document. On appelle **instances** d'un contexte c dans le document d l'ensemble des apparitions de ce contexte dans le document.

L'ensemble des contextes structurels pour l'exemple de la figure 6.3 est $C_d^{AR} = \{C_{title}^{play}, C_{title}^{play,personnae}, C_{speaker}^{play,act,scene,speech}, ...\}$
L'ensemble de contextes constitue l'entrée structurelle de l'index. Avant d'exposer le schéma de l'index structurel, nous introduisons la relation de **spécialisation** et de **généralisation** entre deux contextes.

Définition

[102]

Soit deux contextes $C_{e_1}^{\sigma_1}$ et $C_{e_2}^{\sigma_2}$ on dit que $C_{e_2}^{\sigma_2}$ une **spécialisation** de $C_{e_1}^{\sigma_1}$ et $C_{e_1}^{\sigma_1}$ est une **généralisation** de $C_{e_2}^{\sigma_2}$ ssi $\sigma_2, e_2 = \sigma_1, e_1, \sigma'$ (σ' peut être égal à $\emptyset$ c'est à dire, qu'un contexte est une spécialisation ou généralisation de lui même).

[102] Cette définition peut être généralisée aux éléments.

On note par $dist(C_{e_1}^{\sigma_1}, C_{e_2}^{\sigma_2}) = |\sigma'|$ la distance en nombre d'arcs entre deux contextes qui représente le **degré** de spécialisation.

Notons que pour chaque instance n_2 du contexte $C_{e2}^{\sigma 2}$ ($n_2 \in C_{e2}^{\sigma 2}$), $\exists n_1, n_1 \in inst(C_{e1}^{\sigma 1})$ tel que $n_1, \ldots, n_2$ est un chemin de l'arbre étiqueté.
Nous présentons dans la figure 6.4, un exemple d'arbre réduit avec les contextes et leurs relations.

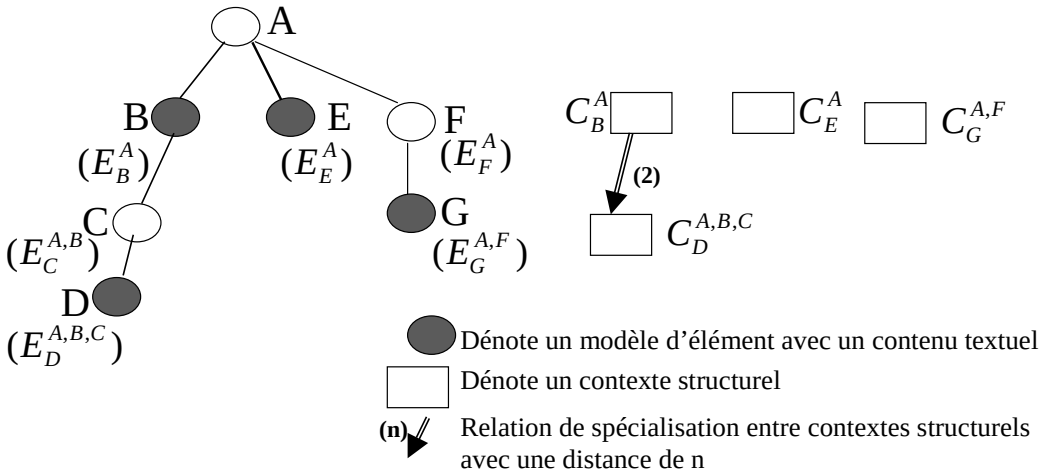

FIG. 6.4 – Exemple d'arbre réduit et des contextes associés avec leurs relations

Les contextes constituent la composante structurelle de notre index. Ils servent avec le contenu textuel à représenter les termes en fonction de leur apparition dans ces contextes.

Définition

Soit D un corpus de document XML, Ψ l'ensemble des labels des balises et Ω l'ensemble des termes du corpus.
Un index structurel est un tuple

$$IS =< C, D, \Psi, \Omega, f, g >$$

- C : ensemble des contextes avec $\forall c \in C$, $\exists d \in D$, avec $c \in C_d^{AR}$ (AR étant la représentation en arbre réduit du document d).
- $g : C \rightarrow 2^D$ avec $g(c) = \{d \in D | c \in C_d^{AR}\}$. g associe à un contexte l'ensemble des documents dont il est la réalisation (où une instance du contexte apparaît au moins une fois).

– $f : C \rightarrow 2^{\Omega}$ avec $f(c) = \bigcup_{d \in g(c)} W'_d(c)$. f associe à un contexte son vocabulaire par rapport au corpus.

L'index structurel maintient une liste de contextes. Il associe à chaque contexte la liste des documents où le contexte apparaît au moins une fois ainsi que l'ensemble des termes qui apparaissent au moins une fois dans ce contexte structurel. La figure 6.5 schématise la structure de notre index.

6.4 Indexation de structure et texte : adaptation du modèle vectoriel

La structure de l'index telle que nous l'avons présentée dans la section précédente nous offre des informations sur les dimensions structurelle et textuelle. Nous utilisons ces informations pour pondérer la structure ainsi que les termes.

Les contextes sont considérés comme des unités atomiques (à la place des documents). Chaque contexte est représenté par un vecteur des termes qui constituent son vocabulaire. Néanmoins, nous ne considérons pas les contextes comme des unités indépendantes, comme c'est le cas des documents dans le modèle vectoriel. Les contextes peuvent être liés par des relations de spécialisation/généralisation. Nous gardons donc la dimension de document car l'imbrication des contextes à l'intérieur d'un même document (ou d'une manière générale dans le corpus) est porteuse de sens qu'on exploite dans les pondérations des termes.

Nous illustrons nos calculs sur l'exemple présenté dans la figure 6.5. Quatre documents sont représentés par leurs contextes structurels. Les trois documents (d_1, d_2, d_3) partagent les mêmes contextes structurels. Nous pouvons remarquer que les contextes $C_Z^{A,B,C,X}$, $C_K^{A,E}$, $C_J^{A,E,L}$, $C_K^{A,F,G,I}$ sont optionnels parce qu'ils n'apparaissent pas dans la totalité des trois documents. Le document d_4 présente une structure différente.

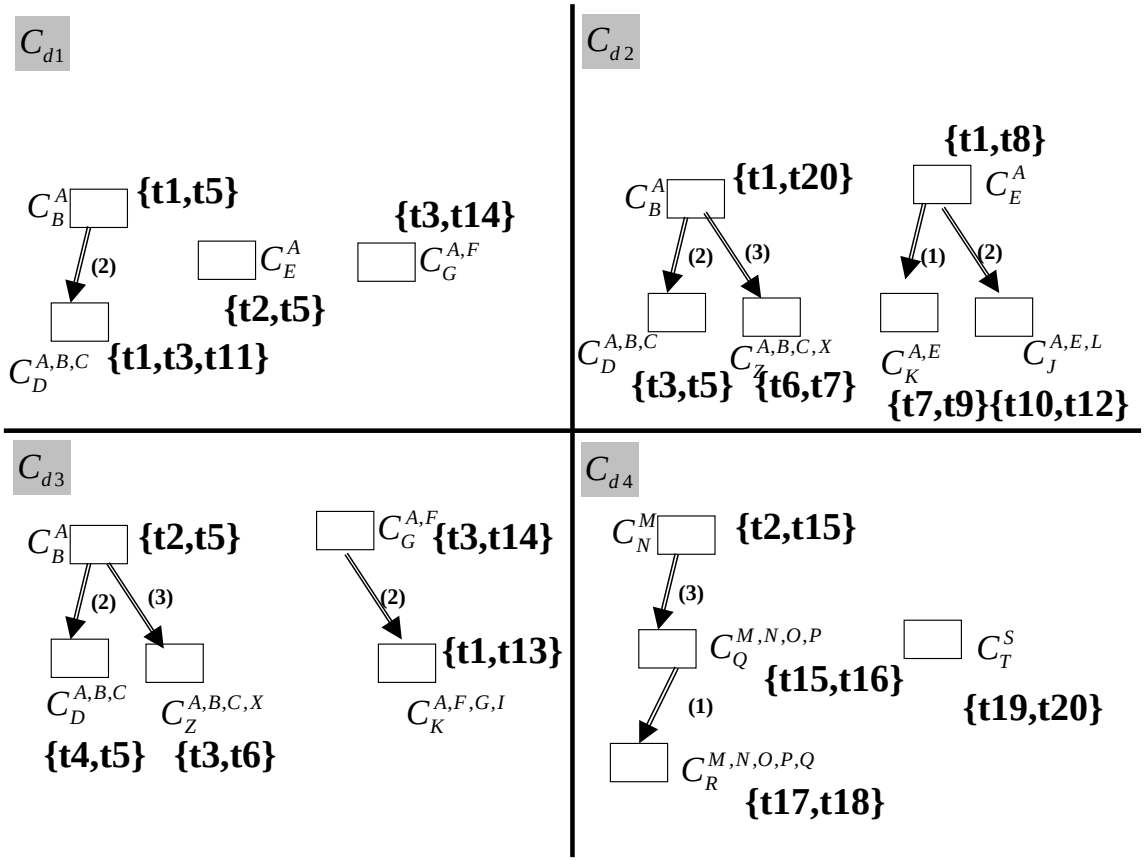

FIG. 6.5 – Exemple de 4 documents avec 12 contextes structurels

6.4.1 Indicateurs sur la structure

Les documents tels que nous les avons présentés dans la section précédente sont composés d'un ensemble de contextes qui peuvent être imbriqués (avec une relation de généralisation/spécialisation). Chaque contexte représente un modèle réalisé par une partie plus au moins importante du document. L'entrée structurelle de l'index comporte des contextes qui ont été réalisés par au moins un document du corpus. Nous maintenons, pour un contexte c deux indicateurs sur sa force informationnelle (appelée CF pour *Context Force*) :

– Un indicateur **local** qui représente la proportion informationnelle du contexte **dans le document**. Plus un contexte est spécialisé (c.a.d la taille de son sous arbre est importante), plus il comporte un contenu informationnel riche, puisque il véhicule son propre contenu ainsi que le contenu de son sous-arbre.
– Un indicateur **global** qui représente la force informationnelle du contexte par rapport **au corpus**.

Pour une même classe de documents (qui répondent à un même schéma) certains contextes peuvent avoir un contenu informationnel plus conséquent que d'autres, il est donc important de capter ce contenu informationnel en exprimant la force d'un contexte par rapport à l'ensemble des contextes d'un document.

Soit donc l'index structurel $IS =< C, D, \Psi, \Omega, f, g >$ et un document d avec $AR_d =< n_0, N, label, W' >$ sa représentation réduite. Pour simplifier la notation, nous notons C_d pour C_d^{AR}, la totalité des contextes dans d. On dénote par $SP_c = \{c_i \in C | c_i$ est une spécialisation de $c\}$, l'ensemble des spécialisations de c et par $SP_n = \{n_i \in inst(c_i) | c_i \in SP_c, n_i \blacktriangleleft n$, n_i est une spécialisation de $n\}$, l'ensemble des spécialisations de l'instance contexte n.
Nous calculons la proportion informationelle (*CF*) d'un contexte c pour un document d par la formule suivante :

$$CF(c,d) = log(\frac{|SP_c \cap C_d|}{|C_d|} + 1)$$

$|C_d|$ est le nombre de contextes qui définissent d.
$|SP_c \cap C_d|$ est le nombre de contextes spécialisations de c et qui appartiennent à C_d
Nous calculons la proportion informationelle (*CF*) d'un contexte c pour un corpus D par la formule suivante :

$$CF(c) = log(\frac{|SP_c|}{|C|} + 1)$$

$|C|$ est le nombre de contextes dans le corpus.
$|SP_c|$ est le nombre de contextes spécialisations de c dans le corpus.
L'intuition derrière cette mesure est que plus un contexte est haut dans l'arbre réduit (c'est à dire plus général) et plus ce contexte est spécialisé c'est à dire la taille de son sous arbre est importante, plus il participe à la part d'information du document. Dans la figure 6.5, le contexte C_B^A a une plus grande force informationnelle que C_E^A dans le document d1, puisque C_B^A contient 3 spécialisations. Au niveau du corpus $CF(C_E^A) = CF(C_B^A)$ puisqu'ils contiennent tous les deux le même nombre de contextes.
Pour calculer la pondération des termes, nous instancions les valeurs de CF en prenant en considération le terme. Ainsi, SP_c, C_d et C, deviennent SP_c^t, C_d^t et C^t qui dénotent respectivement, le nombre de spécialisations du contexte c qui contiennent t, le nombre de contextes dans le document d qui contiennent t et le nombre de contextes dans le corpus qui contiennent t.

6.4.2 Pondération des termes

La structure de notre index comporte deux fonctions de répartition des termes. Ces fonctions expriment des distributions de niveau de généricité différents : (ii) La fonction W_d associe les termes aux instances contextes qui composent le document d(e.g $W(C_B^A) = (t1, t5)$ pour le document $d1$) (iii) la fonction f associe les termes aux contextes à l'échelle du corpus (e.g $f(C_B^A) = (t1, t2, t5, t7, t20)$ pour l'ensemble des documents).
Comme pour les contextes, la distribution des termes est calculée à deux niveaux :
– Niveau local : calcule la répartition des termes par rapport à leur **instance de contexte structurel** d'apparition au niveau du **document**.
– Niveau global : calcule la répartition des termes par rapport à leur **contexte structurel** d'apparition au niveau du **corpus**.
Même si le document ne constitue plus l'unité de base pour l'index, il est important de garder cette dimension dans le calcul de la pondération des termes. L'apparition des contextes à l'intérieur d'un même document est importante et il est utile de garder cette information. Nous calculons la pondération d'un terme en fonction de sa fréquence ainsi que sa représentativité et son pouvoir discriminatoire, comme suit.

Fréquence d'un terme

– Au niveau du document : $TF(t, d, c, n)$[103] calcule la fréquence du terme t dans l'instance n du contexte c dans le cadre du document d comme suit :

$$TF(t, d, c, n) = \sum^{c_i \in SP_c \cap C_d} (\frac{1}{1 + dist(c, c_i)} * \sum^{n_j \in inst(c_i) \bigcap SP_n} count(t, n)) \quad (6.1)$$

$count(t)$ compte le nombre d'occurrences d'un terme dans l'instance de contexte n ainsi que ses descendants. La fréquence d'un terme dans une instance de contexte n est en fonction de son occurrence dans son sous arbre et de sa distance. Dans le cas où le terme n'apparaît que dans cette instance de contexte ou que le contexte n'a pas de spécialisation $TF(t, d, c, n)$ revient à calculer le nombre d'occurrences dans ce seul contexte $count(t)$ (e.g $TF(t1, d1, C^A_{B\#1}) = 1 + \frac{1}{1+2} = 1.33; TF(t1, d2, C^A_B, C^A_{B\#1}) = 1$).

– Au niveau du corpus : $TF(t, c)$ calcule le nombre d'occurrences du terme t dans le vocabulaire du contexte c ($f(c_i)$) comme suit :

$$TF(t, c) = \sum^{c_i \in SP_c} \frac{1}{1 + dist(c, c_i)} * count(t, c_i) \quad (6.2)$$

$count(t, c_i)$ compte le nombre d'occurrences d'un terme dans $f(c_i)$, la fréquence d'un terme dans un contexte est en fonction de son occurrence dans son sous arbre dans le corpus (SP_c). Dans le cas où le terme n'apparaît que dans ce contexte ou que le contexte n'a pas de spécialisation $TF(t, c)$ revient à calculer le nombre d'occurrences dans ce seul contexte $count_{t \in f(c)}(t)$ (e.g $TF(t1, C^A_B) = 2 + (1 * \frac{1}{1+2}) = 2.33; TF(t3, C^{A,B,C}_D) = 2$).

Représentativité d'un terme

– Au niveau du document : $CF(t, c, d)$[104] exprime la représentativité d'un terme t pour un contexte c dans le cadre du document.

$$CF(t, c, d) = log(\frac{|SP^t_c \cap C_d|}{|C^t_d|} + 1) \quad (6.3)$$

avec :

$|SP^t_c \cap C_d|$ le nombre de spécialisations de c dans d contenant au moins une fois t et $|C^t_d|$ le nombre total de contextes de d comportant au moins une fois le terme t. La fonction $CF(t, c, d)$ est monotone croissante, elle appartient à l'intervalle $]0, log(2)]$. Si un terme n'apparaît que dans un contexte (ou avec ses spécialisations) dans un document il représente fortement le contenu de ce contexte (e.g $CF(t1, C^A_B, d1) = log(\frac{2}{2} + 1) = 0.3$ qui est la valeur maximal pour CF, en effet $t1$ représente fortement le contenu de C^A_B puisqu'il apparaît exclusivement dans ce contexte et ses spécialisations dans $d1$).

[103] Term Frequency
[104] CF : Context Force.

– Au niveau du corpus $CF(t,c)$ exprime la représentativité d'un terme t pour un contexte c par rapport aux différents contextes du corpus. Cet indicateur est calculé comme suit :

$$CF(t,c) = log(\frac{|SP_c^t|}{|C^t|} + 1) \tag{6.4}$$

avec :

$|SP_c^t|$ est le nombre de spécialisations de c dans le corpus contenant au moins une fois t et $|C^t|$ le nombre total de contextes comportant au moins une fois le terme t. La fonction $CF(t,c)$ est également monotone croissante, elle appartient à l'intervalle $]0, log(2)]$. Si un terme n'apparaît que dans un contexte (ou avec ses spécialisations) dans un corpus il représente fortement le contenu de ce contexte (e.g $CF(t1, C_B^A) = log(\frac{2}{4}+1) = 0.17$, au niveau du corpus t1 apparaît dans d'autres contextes $C_E^A, C_K^{A,F,G,I}$).

Pouvoir discriminatoire d'un terme

– $IDF(t,c)$[105] exprime la force discriminatoire du terme t qui permet de pondérer les contextes c par rapport à l'ensemble de leurs apparitions dans les différents documents (g(c)) qui contiennent le terme t :

$$IDF(t,c) = log(\frac{|D|_c}{|D|_{SP_c}^t} + 1) \tag{6.5}$$

avec :

$|D|_{SP_c}^t$ le nombre de documents qui comportent c (ou une de ses spécialisations) avec au moins une occurrence de t et $|D|_c = |g(c)|$ le nombre de documents qui contiennent c. La fonction $IDF(t,c)$ est monotone décroissante, elle appartient à l'intervalle $[log(2), log(|D|_c)]$. Notons qu'elle s'applique à une classe de documents qui partagent la même structure (e.g $IDF(t1, C_B^A) = log(\frac{3}{2} + 1) = 0.39$).

En utilisant l'ensemble de ces indicateurs, nous mettons en place deux mesures de pondération : au niveau du corpus (TF-CF : term frequency-context force) et au niveau du document (TF-ICDF : term frequency-inverse context and document frequency).
Au niveau du corpus, la pondération d'un terme porte sur le contexte et son vocabulaire sans prendre en considération les documents qui réalisent ce contexte. L'idée est de pouvoir repérer pour un terme les contextes qu'il représente le mieux, cela nous servira dans la phase d'interrogation pour pré-sélectionner les contextes.
Au niveau du document, la pondération d'un terme porte sur la réalisation du contexte dans ce document ainsi que la force discriminatoire de cette réalisation.
La pondération d'un terme t dans un contexte c

$$TF - CF(t,c) = TF(t,c) * CF(t,c) \tag{6.6}$$

Cette fonction nous renseigne sur la représentativité du terme par rapport à un contexte (e.g $TF - CF(t1, C_B^A) = 2.33 * 0.17 = 0.396$), elle nous servira lors de la phase d'interrogation à faire une pré-selection des contextes à retourner.

[105] IDF : Inverse Document Frequency

La pondération d'un terme t dans un contexte c réalisant un document d est inspirée du $TF - IDF$ en faisant intervenir la notion de contexte.

$$TF - ICDF(t, d, c, n) = TF(t, d, c, n) * CF(t, c, d) * IDF(t, c, d) \tag{6.7}$$

Cette fonction nous renseigne sur la représentativité et la force discriminatoire d'un terme pour un contexte donné réalisant un document donné (e.g $TF - ICDF(t1, d1, C_B^A, C_{B\#1}^A) = 1.33 * 0.3 * 0.39 = 0.155$). C'est cette valeur qui sera représentée dans les vecteurs de contexte pour un document. Dans le cas de documents non structurés, on peut considérés qu'ils possèdent une seule structure qui est le document lui-même, le $TF - CF = TF - IDF * Cste$[106] dans ce cas.

6.5 Contenu sémantique des documents

L'ensemble des pondérations présentées dans la section précédente offre une information riche sur le poids du terme par rapport à son contexte d'apparition au niveau du document et du corpus. Néanmoins, aucune considération de la sémantique du terme dans le calcul de son poids n'est prise en compte. Par exemple, le poids est calculé indépendamment de l'apparition des termes synonymes dans le même contexte. C'est un des principaux reproches faits au modèle vectoriel (ainsi qu'aux autres modèles qui supposent pour simplifier l'indépendance des termes). Nous avons présenté dans le chapitre précédent une mesure de similarité sémantique entre deux concepts dans une ontologie, nous enrichissons les termes d'un même contexte par leur voisinage sémantique. Nous enrichissons les termes par leurs synonymes mais aussi par leurs hypéronymes, hyponymes ou n'importe quelle autre relation si les termes sont similaires au dessus d'un certain seuil. Nous devons donc dans un premier temps rattacher les termes à leurs concepts et décider du sens que prend un terme à l'intérieur d'un contexte. Nous présentons dans ce qui suit l'intégration de l'ontologie dans le processus d'indexation et la prise en compte de la sémantique des termes. Nous proposons un algorithme de désambiguïsation sémantique des termes qui tient compte de la fréquence des termes et leurs similarités sémantiques au sein d'un même contexte.

6.5.1 Utilisation de l'ontologie

Nous proposons d'utiliser des ontologies *lexicalisées* (les concepts possèdent des entrées par les lemmes) lors de la phase d'indexation pour enrichir le poids d'un terme par l'espace de sens qui l'entoure. Les concepts peuvent aussi être reliés aux contextes et serviront à créer des classes d'équivalence lors de la phase d'interrogation. Ainsi si un contexte C1 est similaire à un contexte C2, ils peuvent se substituer l'un à l'autre (eg. Auteur et Author). Cependant, contrairement aux méthodes présentées dans l'état de l'art, nous ne nous restreignons pas à l'utilisation des concepts. En effet, les termes sont enrichis s'ils sont reliés aux concepts mais il est important de noter que lors de la recherche, nous pouvons aussi retrouver les termes qui ne sont pas reliés à l'ontologie (cas où l'ontologie ne couvre pas tout le corpus). Nous calculons la similarité entre les termes indexés lors de la phase d'indexation pour alléger le temps de

[106] $cste = log(\frac{1}{1+1}) = 0.3$

traitement des requêtes. Lors de la phase de recherche, seule la similarité entre des termes qui n'existent pas dans le vocabulaire du contexte est calculée (cf. chapitre suivant).

Problématique : Projection d'une ontologie sur un corpus

Il s'agit dans un premier temps de trouver un appariement entre chaque terme indexé et le (ou les) concept(s) de l'ontologie.

Définition

Une ontologie lexicalisée est un tuple

$$OL =< L, S, \varphi >$$

- L : l'ensemble des lemmes qui sont les entrées des concepts de l'ontologie.
- S : l'ensemble des concepts.
- $\varphi : L \rightarrow 2^S$ affecte à chaque lemme l'ensemble des sens ou concepts possibles.

Nous utilisons dans un premier temps WordNet, en ne considérant que sa structure taxonomique. WordNet est une ressource largement utilisée car disponible en ligne et assez bien fournie. Les liens entre les termes d'indexation (sous forme de lemme) sont faciles à trouver grâce à la structure interne de WordNet. Chaque entrée de WordNet est un groupe de mots lemmatisés (appelés *indexedWord*). Ainsi, si le terme existe dans WordNet (en fonction de sa catégorie grammaticale généré par le TreeTagger) nous lui associons un ensemble de synsets. Cet ensemble dénote l'ensemble de sens possibles du terme. Par exemple le terme *message* possède deux sens dans Wordnet en tant que nom et trois sens en tant que verbe. Comme beaucoup de travaux, nous nous sommes intéressés exclusivement aux noms dans WordNet. Cela est dû à la structuration de WordNet. En effet, les mots sont regroupés par leurs catégories grammaticales et il n'est pas possible de calculer la similarité intra-catégories (par liens taxonomiques). Le problème de désambiguïsation des termes n'est pas posé à ce niveau. Il sera résolu lors de l'intégration du calcul de la similarité, nous en reparlerons plus loin.
WordNet possède une structure taxonomique mais n'est pas pour autant une ontologie. Le but de notre système étant à terme de profiter de la plate-forme du Web sémantique, nous avons fait le choix d'utiliser des ontologies formelles représentées dans le formalisme OWL. Ce langage devenu un standard, le pari du Web Sémantique est qu'il existera, dans un futur proche, de plus en plus d'ontologies en ligne en OWL.
Le lien entre les termes d'indexation et une ontologie en OWL se fait sur la supposition que l'ontologie est lexicalisée. Sans entrée lexicale il est impossible de trouver une concordance entre le terme du corpus et le concept qui lui est relatif. Une analyse lexicale consiste à comparer la liste des termes dans le texte à ceux qui existent dans l'ontologie. Il est à noter que la problématique de projection d'une ontologie sur un texte n'est pas aisée. Même si un terme et un concept partagent le même lemme, cela ne prouve en rien que le terme est relatif au concept. Une validation par un expert est toujours nécessaire, nous avons montré dans le chapitre 4 que la définition même de l'expert n'est pas évidente. Celui qui indexe les documents n'est pas forcément celui qui en est l'auteur qui n'est pas à son tour forcément un expert du domaine (au sens Ingénierie des Connaissances).

6.5.2 Indexation du contenu sémantique

L'indexation sémantique d'un corpus consiste dans un premier temps à faire correspondre à chaque terme de Ω les lemmes des concepts de l'ontologie OL.

Définition

L'index sémantique est un tuple

$$ISem =< C, D, \Omega, f, g, OL, h >$$

- D : l'ensemble de documents XML.
- C : l'ensemble des contextes dans D.
- OL : l'ontologie lexicalisée $< L, S, \varphi >$
- $g : C \rightarrow 2^D$ avec $g(c) = \{d \in D | c \in C_d\}$. g associe à un contexte l'ensemble de documents ou il apparaît.
- $f : C \rightarrow 2^\Omega$ avec $f(c) = \bigcup_{d \in g(c)} W_d(c)$. f associe à un contexte son vocabulaire.
- $h : \Omega \rightarrow L \cup \emptyset$: associe à chaque terme le lemme correspondant à un (ou plusieurs) concept(s) de l'ontologie il faut noter que $\varphi(t \in \Omega) = \varphi(l \in L)$ avec $l = h(t)$. h est une fonction de correspondance entre le terme et le lemme de l'ontologie, dans notre cas cette fonction se base sur l'égalité des lemmes mais elle pourrait aussi prendre en compte des traductions (e.g. *cat=h(chat)*)

Chaque terme du corpus se trouve alors relié à un ensemble de concepts qui correspondent à l'ensemble des sens possibles du terme. Nous avons besoin de désambiguïser les termes avant de procéder à leur enrichissement sémantique.

Désambiguïsation sémantique

Chaque auteur d'un document, essaye, quelque soit le niveau d'arbitraire dans sa structuration des documents, de définir une logique particulière de décomposition du contenu textuel des documents. La structure donc, dans certains cas, joue un rôle de limitation sémantique où la co-occurrence des termes a une signification relié au critère de structuration. Dans ce cas chaque terme apporte une information particulière sur les termes voisins. Il est donc possible d'utiliser aussi bien la limite du contexte ainsi que la co-occurrence afin de désambiguïser les termes dans le cadre d'un contexte. Rappelons que nous voulons attribuer un seul sens au terme (au plus) par contexte structurel. On considère dans ce qui suit l'ontologie enrichie par une fonction de similarité entre les concepts présentée dans le chapitre précédent.

Définition

Soit $OS =< L, S, \varphi, sim >$ une ontologie enrichie par la mesure de similarité $Sim : S \times S \rightarrow [0..1]$ et l'index sémantique est un tuple
$ISem =< C, D, \Omega, f, g, OS, h, Sim >$
Nous définissons pour chaque contexte c le graphe bipartite $G_c =< \omega_c, S_c, \rightarrow_1, \rightarrow_2 >$ avec :
- ω_c : désigne le vocabulaire du contexte c (i.e $f(c)$)

- $S_c = \bigcup_{t \in \omega_c} \varphi(t)$ l'ensemble de concepts possibles des termes du contexte c.
- $\rightarrow_1$: $S_c \times S_c \rightarrow [0..1]$: désigne les relations entre les concepts S_c étiquetées par la valeur de la similarité ($s_i \longrightarrow_1^{Sim(s_i,s_j)} s_j$)
- $\rightarrow_2$: $\omega_c \rightarrow S_c$ désigne les correspondances entre chaque terme t du contexte à ses sens possibles, nous avons $(t, s) \in \rightarrow_2$ ssi $s \in \varphi(t)$

La désambiguïsation sémantique d'un contexte consiste à choisir pour chaque terme de son vocabulaire le sens correspondant conformément au sens choisi pour les autres termes. Le processus de désambiguïsation consiste à choisir pour chaque terme le sens qui maximise la similarité sémantique avec les autres sens.

L'algorithme de désambiguïsation se réduit donc au problème d'affectation dans un graphe bipartite. La désambiguïsation d'un contexte consiste donc à trouver l'affectation maximale des sens aux termes. La construction de la solution d'affectation est guidée par trois critères à maximiser :

La fréquence du terme dans le contexte : ce critère permet de favoriser l'attribution d'un sens aux termes qui ont une grande fréquence, rappelons qu'à ce niveau aucune épuration sur la fréquence des termes n'est faite.

La similarité sémantique : ce critère permet de sélectionner l'ensemble des concepts les plus proches sémantiquement.

La densité du graphe de sens : ce critère nous permet de sélectionner les concepts les plus reliés (le graphe le plus complet).

Il est à noter que cette opération est faite une seule fois. A l'issue de l'étape de désambiguïsation, la fonction φ sera adaptée à chaque contexte (notée φ_c). La désambiguïsation d'un contexte consiste à attribuer un seul sens aux termes qui y apparaissent. L'idée derrière cette désambiguïsation d'un contexte c est de trouver la fonction φ_c (qui est une sous-relation de φ) tel que $\forall t \in \omega_c$, on a :

- $\underset{t_i \in \omega_c}{|\varphi_c(t)|} = \{0, 1\}$. Un terme a au plus un sens attribué.
- $\sum_{t_i \in \omega_c} \sum_{t_j \in \omega_c} sim(\varphi_c(t_i), \varphi_c(t_j))$[107] est optimale
- $|\varphi_c|$ est optimale (regroupe le maximum de termes).

Un terme t peut se voir attribuer des sens différents suivant son contexte d'apparition. Soit par exemple les contextes présentés dans la figure 6.6.

[107] $t_i < t_j$, notre mesure de similarité étant symétrique, cela nous évite de calculer la similarité deux fois pour le même couple.

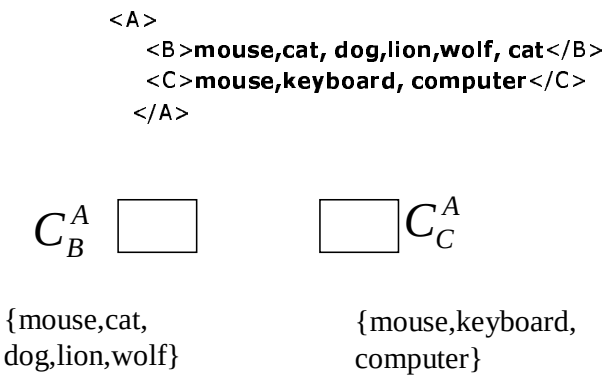

FIG. 6.6 – Exemple de contextes avec un terme polysémique

Le terme *mouse* (souris) comporte deux sens[108] dans WordNet. L'algorithme de désambiguïsation va selon le contexte (le sens des termes co-occurrents dans les deux contextes C_B^A et C_C^A) sélectionner le sens approprié.

Les figures 6.7 et 6.8 comportent respectivement la représentation graphique du graphe sémantique de chaque contexte B et C. Les arcs en bleu correspondent à $\rightarrow_2$. Les arcs en noir représentent $\rightarrow_1$, la similarité sémantique entre les concepts du contexte. Les synsets en rouge sont ceux retenus pour chaque terme dans le cadre de chaque contexte. Dans le contexte C_B^A le sens retenu pour le terme $mouse$ est celui d'un rongeur alors que pour C_C^A c'est celui d'une unité électronique périphérique.

[108] sens1 = any of numerous small rodents typically resembling diminutive rats having pointed snouts and small ears on elongated bodies with slender usually hairless tails
sens2 = a hand-operated electronic device that controls the coordinates of a cursor on your computer screen as you move it around on a pad ; on the bottom of the device is a ball that rolls on the surface of the pad

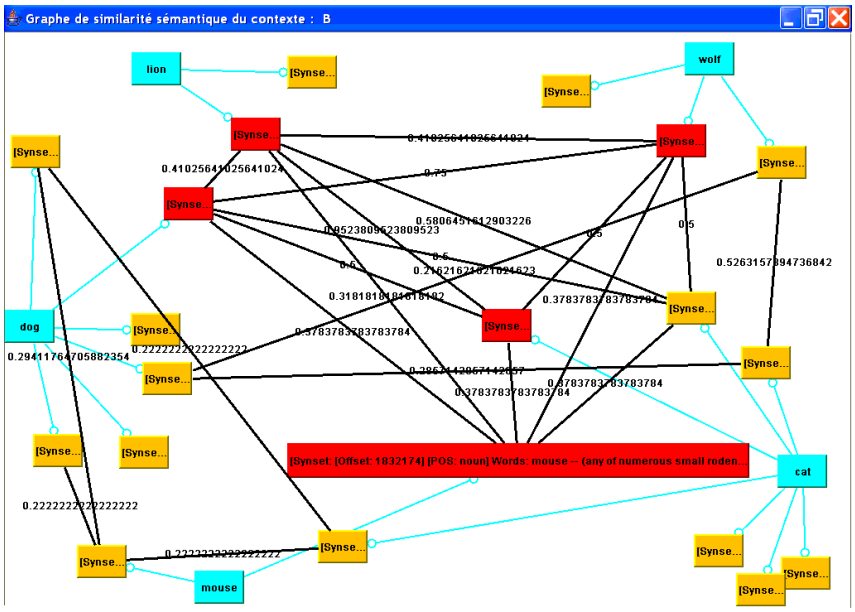

FIG. 6.7 – Résultat de la désambiguïsation du contexte B(φ_B) (copie d'écran de *SemIndex*)

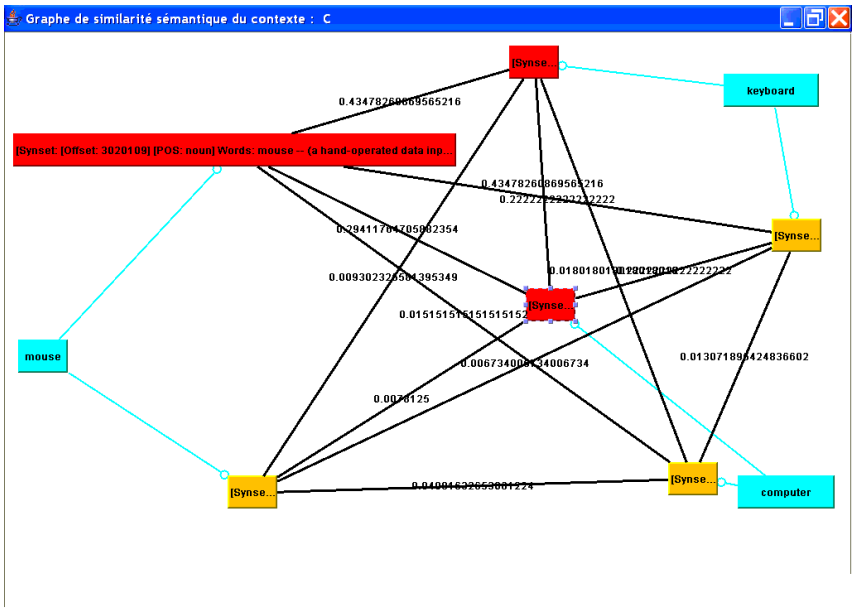

FIG. 6.8 – Résultat de la désambiguïsation du contexte C(φ_C) (copie d'écran de *SemIndex*)

6.6 Indexation de structure, texte et sémantique : enrichissement du modèle vectoriel

Une fois que la désambiguïsation des termes dans un contexte est réalisée, il est possible maintenant d'enrichir la pondération d'un terme par celle des termes similaires dans un même contexte. Nous étendons alors les mesures de pondération des termes définies dans la section 6.4.

Fréquence sémantique d'un terme

- Au niveau du document : $SemTF(t,d,c,n)$[109] calcule la fréquence du terme t dans l'instance n du contexte c dans le cadre du document d enrichie par la similarité des termes co-occurrents (au niveau du contexte) pondérée par leurs fréquence comme suit :

$$SemTF(t,d,c,n) = TF(t,d,c,n) + \sum^{t_i \in \omega_c} TF(t_i,d,c,n) * Sim(\varphi_c(t), \varphi_c(t_i)) \quad (6.8)$$

  Avec $Sim(\varphi_c(t), \varphi_c(t_i)) > seuil$, $seuil$ est une valeur à déterminer empiriquement. Si pour l'exemple de la figure 6.5, t1 et t2 sont des synonymes (leur similarité sémantique est égale à 1), $SemTF(t1, d1, C_B^A, C_{B\#1}^A) = 1.33 + (1 * 1) = 2.33$, la fréquence de t_5 dans l'instance de contexte $C_{B\#1}^A$ se trouve identiquement enrichie. Supposons que t_2 est similaire à t_1 avec une valeur de 0.6, t_2 qui avait une fréquence nulle dans $d1$ est enrichie par la fréquence de t_1 : $SemTF(t2, d1, C_B^A, C_{B\#1}^A) = 0 + (0.6 * 1.33) = 0.798$
- Au niveau du corpus : $semTF(c,t)$ calcule le nombre d'occurrences du terme t dans le vocabulaire du contexte c ($f(c_i)$) enrichie par la similarité des termes co-occurrents pondérée par leurs fréquence dans le corpus comme suit :

$$SemTF(t,c) = TF(t,c) + \sum^{t_i \in \omega_c} TF(t_i,c) * Sim(\varphi_c(t), \varphi_c(t_i)) \quad (6.9)$$

  Ainsi, $SemTF(t1, C_B^A) = 2.33 + (1 * 2.66^{110}) + (1 * 1^{111}) = 5.99$

Représentativité sémantique d'un terme

- Au niveau du document : $SemCF(t,c,d)$[112] exprime la représentativité sémantique d'un terme t pour un contexte c dans le cadre du document. En d'autres termes, nous nous intéressons à l'apparition du sens de t ($\varphi_c(t)$) à la place de t[113].

$$SemCF(t,c,d) = log(\frac{|SP_c^{\varphi_c(t)} \cap C_d|}{|C_d^{\varphi_c(t)}|} + 1) \quad (6.10)$$

  avec :

[109] SemTF : Semantic Term Frequency
[110] $TF(t_5, C_B^A, d1) = 2 + (\frac{1}{1+2} * 2) = 2.66$
[111] $TF(t_2, C_B^A, d1) = 1$
[112] SemCF : Semantic Context Force.
[113] Ce qui revient à prendre en compte les synonymes.

$|SP_c^{\varphi_c(t)} \cap C_d|$ le nombre de spécialisations de c dans d contenant au moins une fois $\varphi_c(t)$ et $|C_d^{\varphi_c(t)}|$ le nombre total de contextes de d comportant au moins une fois le sens $\varphi_c(t)$. Par exemple, si le terme t_5 a un sens différent dans C_E^A, $SemCF(t5, C_B^A, d1) = log(\frac{2}{2}+1) = 0.3$ alors que si c'était le même sens $SemCF(t5, C_B^A, d1) = log(\frac{2}{3}+1) = 0.22$.

– Au niveau du corpus $SemCF(t, c)$ exprime la représentativité d'un sens $\varphi_c(t)$ pour un contexte c par rapport aux différents contextes du corpus. Cet indicateur est calculé comme suit :

$$SemCF(t, c) = log(\frac{|SP_c^{\varphi_c(t)}|}{|C^{\varphi_c(t)}|} + 1) \tag{6.11}$$

avec :

$|SP_c^{\varphi_c(t)}|$ est le nombre de spécialisations de c dans le corpus contenant au moins une fois $\varphi_c(t)$ et $|C^{\varphi_c(t)}|$ le nombre total de contextes comportant au moins une fois le terme t. $semCF(t1, C_B^A) = log(\frac{2}{2} + 1) = 0.3$, si au niveau du corpus le sens de t1 dans les contextes $C_E^A, C_K^{A,F,G,I}$) est diffèrent de $\varphi_{C_B^A}$.

Pouvoir discriminatoire de la sémantique d'un terme

– $SemIDF(t, c)$[114] exprime la force discriminatoire du sens $\varphi_c(t)$ qui permet de désigner les contextes c par rapport à l'ensemble des extensions du contexte dans les différents documents (g(c)) qui contiennent le sens $\varphi_c(t)$ (réalisation du contexte) :

$$SemIDF(t, c) = \frac{|D|_c}{|D|_c^{\varphi_c(t)}} \tag{6.12}$$

avec :

$|D|_c^{\varphi_c(t)}$ le nombre de documents qui comportent c avec au moins une occurrence de $\varphi_c(t)$ et $|D|_c = |g(c)|$ le nombre de documents qui contiennent c. Ainsi $IDF(t1, C_B^A) = log(\frac{3}{3} + 1) = 0.30$).

Les fonctions de $TF - CF$ et $TF - ICDF$ sont les mêmes que celles présentées dans la section 6.4 mais appliquées au $SemTF, SemCF$ et $SemIDF$.
La pondération d'un terme t dans un contexte c enrichie par son voisinage sémantique.

$$semTF - CF(t, c) = semTF(t, c) * semCF(t, c) \tag{6.13}$$

$$semTF - ICDF(t, c, d) = semTF(t, d, c, n) * semCF(t, c, d) * semIDF(t, c, d) \tag{6.14}$$

6.7 Architecture générale de l'index

Le modèle d'indexation *SemIndex* met en concert trois éléments d'indexation : le contenu textuel, le contenu structurel et le contenu sémantique. Contrairement à d'autres modèles d'indexation structurée, notre entrée structurelle est fondée sur la notion de contexte ce qui nous permet

[114] SemIDF : Semantic Inverse Document Frequency

d'éviter l'indexation de chaque chemin de l'arbre du document. Nous faisons une abstraction, par la notion de contexte, des chemins qui répondent à une même grammaire structurelle. Les termes (entrées textuelles de l'index) sont distribués selon leur apparition dans les contextes du corpus nous permettant ainsi de définir le contexte comme unité de recherche et de mettre en place la notion de vocabulaire d'un contexte. Le vocabulaire d'un contexte nous offre la possibilité de décider du sens d'un terme (s'il est ambigu selon son contexte).
La figure 6.9 expose une schématisation de la structure de l'index et souligne également la possibilité d'accès au corpus indexé aussi bien par la structure, par le contenu, par le sens ou par une combinaison des trois critères. Dans le chapitre suivant, nous exposerons notre langage de requête *SemIR* ainsi que la manière d'utiliser les informations stockées par *SemIndex* pour sélectionner les documents (ou fragments) qui répondent au mieux au besoin de l'utilisateur.

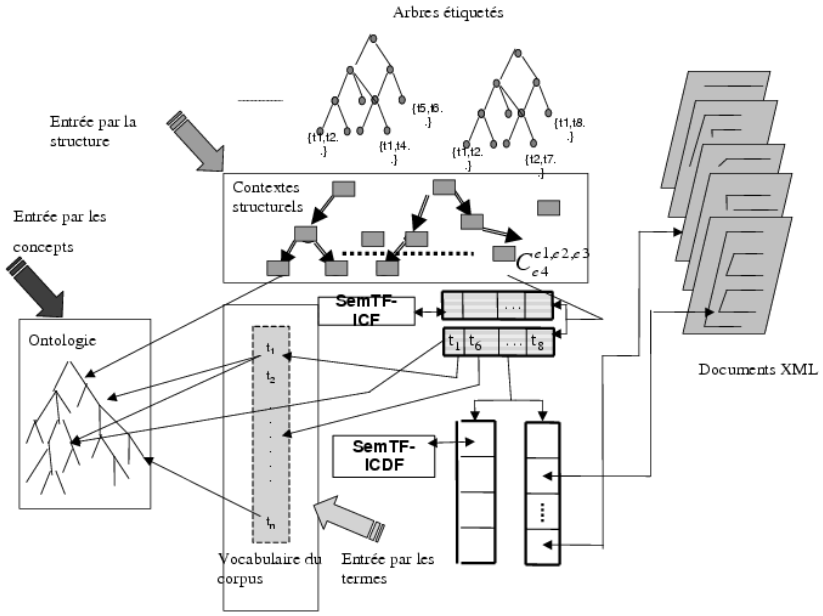

FIG. 6.9 – Structure du modèle SemIndex

6.8 Conclusion

Nous avons présenté, dans ce chapitre, la structure de notre index. Il possède une indexation structurelle fondé sur la notion de contexte qui défini un modèle de structure comportant une information textuelle. Dans chaque contexte les termes ainsi que les documents associés sont indexés selon deux mesures de pondération : le $TF - CF$ qui défini la distribution des termes dans les contextes définissant ainsi la notion de vocabulaire du contexte et le $TF - ICDF$ qui exprime la distribution au sein d'un même contexte des termes dans les documents qui instancient le modèle de structure du contexte. Notre index procède également à une indexation sémantique des termes fondé sur une ontologie lexicalisée. La polysémie des termes est traitée par une procédure de désambiguïsation sémantique qui se base sur la correspondance entre le contexte structurel et la notion de contexte sémantique. Nous attribuons un sens à chaque terme

en fonction de son contexte d'apparition et des sens retenus pour les termes co-occurrents dans le même contexte. L'attribution d'un sens à chaque terme pour un contexte nous offre un moyen d'estimer le poids sémantique du terme et de désambiguïser les termes de la requête lors de la phase d'interrogation. Les indicateurs de notre index sont étendus en tenant compte de la similarité sémantique entre les termes.

Chapitre 7

L'index en action : le langage de requêtes SemIR

Sommaire

7.1 Introduction

Les langages proposés dans la littérature pour l'interrogation de documents XML sont généralement issus de la communauté Bases de Données. Ils sont très expressifs et permettent un

traitement efficace de la structure mais nécessitent néanmoins une connaissance poussée de la structure des documents pour préciser les conditions sur la structure et spécifier l'élément à retourner. De plus, ces langages possèdent une syntaxe qui est difficile d'accès pour beaucoup d'utilisateurs potentiels. Ainsi, prés des deux tiers (63%) des requêtes exprimées en XPath étaient erronées[115], lors de la campagne INEX 2003 et ont nécessité pas moins de 12 tours de corrections successives [O'Keefe & Trotman, 2003]. Or il est probable que le succès d'un langage dépende certes de son expressivité mais aussi de sa facilité d'utilisation et de sa facilité d'accès à de nombreux utilisateurs potentiels.

Nous proposons un langage de requête (**SemIR : Semantic Information Retrieval**) qui s'inspire des langages de requête par l'exemple (*QBE*[116] [Zloof, 1977]). Il possède une syntaxe simple et permet à l'utilisateur de poser sa requête sous forme de document XML.

Notre langage permet les requêtes suivantes :

- Formulation de requêtes à base de simples mots-clés, appelées aussi requêtes orientées contenu : l'utilisateur ne précise pas la structure. Ce type de requêtes est utile quand l'utilisateur n'a aucune connaissance de la structure ou que cette dernière lui importe peu. Le système doit décider de l'unité à retourner.
- Formulation de requêtes orientées contenu avec spécification de l'élément à retourner : l'utilisateur ne précise pas la structure pour les mots-clés recherchés mais a une idée de ce qu'il veut avoir comme résultat.
- Formulation de requêtes orientées structure : ces requêtes ne contiennent pas de mots-clés, les conditions de la requête portent seulement sur la structure. Notre système permet de traiter des corpus hétérogènes et il peut s'avérer utile, dans un but d'exploration, de parcourir les documents qui satisfont une certaine structure.
- Formulation de requêtes orientées structure et contenu : ces requêtes peuvent ou non préciser l'élément à retourner. Les conditions peuvent aussi bien porter sur les mots-clés que la structure. Les contraintes sur les chemins peuvent être vagues en ne donnant pas de chemin absolu.

Contrairement à XXL (cf chapitre 3), nous n'avons pas de syntaxe particulière pour différencier les termes des concepts. Le système procède à un enrichissement sémantique des termes et leur fixe un sens quand c'est possible.

Nous présentons dans la section 7.2 la syntaxe de notre langage de requêtes. Nous décrivons ensuite la méthode de recherche que nous préconisons, en détaillant l'évaluation des requêtes sur le contenu (section 7.4) et celles sur le contenu et la structure (section 7.5) ainsi que le traitement des opérateurs booléens (section 7.7).

7.2 Syntaxe de SemIR

Notre langage de requête suit une grammaire XML, nous présentons le schéma XML de notre langage dans l'annexe B. La grammaire offre une structure qui supporte des éléments exten-

[115] Les requêtes sont syntaxiquement correctes mais ne traduisent pas le besoin d'information, décrit dans la partie descriptive.

[116] QBE : Query By Example. Pour interroger une table avec ce système, il suffit de proposer une image de la réponse (un "squelette" de table) et, par un mécanisme d'analogie, QBE va rechercher les occurrences dans les tables qui correspondent aux différents critères établis.

sibles qui permettent à l'utilisateur de définir les éléments de structure recherchés. L'utilisateur pose sa requête en esquissant la structure recherchée ainsi que son contenu.
Nous introduisons deux catégories de balises prédéfinies (i) la balise **ANY** qui désigne une balise générique (elle peut se substituer à n'importe quelle structure du document) et (ii) les balises qui expriment des opérateurs booléens : **OR**, **NOT** et **AND**. Le tableau 7.1 contient une BNF simplifiée de notre language de requête (le schema XML complet se trouve dans l'annexe B).

terme : := string
requête : :=any | or | not | and | struct
struct : :="<"terme">" terme* requête* | " ?" "</"terme">"
any : :=<ANY> terme* requête*</ANY>
and : :=<AND> requête requête </AND>
or : :=<OR> requête requête </OR>
not : :=<NOT> requête </NOT>

TAB. 7.1 – Grammaire simplifiée du language de requête SemIR

La requête peut comporter de simples mots clés délimités par la balise ANY pour exprimer le fait que l'utilisateur recherche les termes n'importe où dans le document. Cette même balise ANY, si elle est imbriquée à des balises de structure, traduit l'expression d'un chemin vague. Le signe **?** à l'intérieur d'une balise signifie que c'est l'élément à retourner. Nous présentons dans ce qui suit des exemples de requêtes exprimées dans notre langage en explicitant leur signification.

7.2.1 Requêtes orientées contenu

Dans notre langage, les requêtes à base de mots-clés sont un cas particulier de requêtes structurées où aucune restriction sur la structure, n'est imposée. La requête est un document XML, ce qui permet à l'utilisateur de préciser, s'il le veut, l'élément à retourner. Les requêtes à base de mots clés doivent figurer uniquement dans la balise ANY. Voici quelques exemples.

<ANY> recherche information semi-structurée </ANY>

TAB. 7.2 – Requête orientée contenu : rechercher l'unité d'information minimale qui parle de *recherche information semi-structurée*

<article> <titre> **?** </titre> </article> <ANY> recherche d'information semi-structurée </ANY>

TAB. 7.3 – Requête orientée contenu : rechercher les **titres** des articles qui parlent de *recherche information semi-structurée*

Dans la requête du tableau 7.2, l'utilisateur ne spécifie pas l'élément à retourner, le système doit décider du degré de granularité de l'élément pertinent à retourner. Les mots-clés peuvent

éventuellement être reliés par des opérateurs booléens exprimés par les balises prédéfinies. L'exemple du tableau 7.4 signifie que l'utilisateur souhaite retrouver les documents parlants de recherche d'information mais pas du modèle vectoriel.

```
<AND>
   <ANY> recherche  d'information </ANY>
   <NOT>
      <ANY> modèle vectoriel  </ANY>
   </NOT>
</AND>
```

TAB. 7.4 – Expression de requête booléenne à base de mots-clés

7.2.2 Requêtes orientées structure et contenu

Une requête orientée structure et contenu est un document XML qui comporte les éléments définis par l'utilisateur qui traduisent l'idée qu'a utilisateur de la structure dans le document qu'il recherche. Elle peut comporter des constructions avec les balises prédéfinies ANY, OR, AND et NOT pour exprimer des requêtes booléennes.
Les éléments définis par l'utilisateur peuvent être imbriqués, l'expression du chemin peut être à des degrés de précision variés grâce à la balise ANY. Ils peuvent contenir des conditions sur leur contenu ou attributs. L'élément à retourner peut être spécifié dans la requête ou déterminé par le système.
Nous présentons dans ce qui suit quelques exemples de requête.

```
<chapitre>
<section> web sémantique </section>
</chapitre>
```

TAB. 7.5 – Retrouver les *chapitres* contenant des *sections* qui parlent de *web sémantique*
[a]

[a] Les chapitres sont l'unité maximale, le système peut retourner des éléments plus spécifiques.

```
<chapitre>
<titre> ? </titre>
</chapitre>
<chapitre>
<section> web sémantique </section>
</chapitre>
```

TAB. 7.6 – Retrouver les **titres** des *chapitres* contenant des *sections* qui parlent de web sémantique

L'expression de chemin vague : le constructeur ANY

Ce constructeur est utilisé dans le cas où l'utilisateur ne connaît pas, par exemple, la position de la balise $titre$[117] et veut retrouver des chapitres (ou parties de chapitre) qui contiennent "web sémantique" sous $titre$ (titre peut être, par exemple, en dessous de $chapitre$ ou $section$).

[117] Ou que sa position lui importe peu

```
<chapitre>
<ANY>
<titre> web sémantique </section>
</ANY>
</chapitre>
```

TAB. 7.7 – Retrouver les documents qui contiennent l'élément *titre* n'importe où sous l'élément *chapitre* qui traite du *web sémantique*

Il est à noter que les deux requêtes des tableaux 7.5 et 7.7 sont équivalentes. L'utilisateur peut ne pas spécifier un chemin exact qui part de la racine des documents.

```
<ANY>
<section>
<titre> recherche information structurée </titre>
</section>
</ANY>
```

TAB. 7.8 – Retrouver les documents qui parlent de *recherche information structurée* dans *titre* en dessous de *section* mais n'importe où dans l'arbre du document

```
<section>
<titre> recherche information structuré </titre>
</section>
```

TAB. 7.9 – Requête équivalente à la requête dans le tableau 7.8

La disjonction : le constructeur OR :

L'utilisateur recherche un document qui contient "recherche information structurée" dans le titre du chapitre ou dans le titre d'une section.
Nous exposerons dans la section suivante la transformation des requêtes pour traiter les constructeurs booléens. Les constructeurs sont remontés à la racine. Ainsi, les deux requêtes présentées dans les tableaux 7.10 et 7.11 sont équivalentes et expriment exactement le même besoin.

```
<chapitre>
<OR>
<titre> recherche information structuré </titre>
<section>
<titre> recherche information structuré </titre>
</section>
</OR>
</chapitre>
```

TAB. 7.10 – Exemple requête OR (1)

```
<OR>
<chapitre>
<titre> recherche information structuré </titre>
</chapitre>
<chapitre>
<section>
<titre> recherche information structuré </titre>
</section>
</chapitre>
</OR>
```

TAB. 7.11 – Exemple requête OR (2)

La conjonction : le constructeur AND

La sémantique de l'opérateur AND est différente selon qu'il se trouve à la racine de la requête ou comme un nœud interne de l'arbre de la requête.
La requête dans le tableau 7.12 exprime le fait qu'on recherche les documents qui contiennent aussi bien les chapitres dont le titre contient "recherche information structuré" que les chapitres dont une des sections porte sur "XML". Rien dans la requête n'exige le fait que la section soit une section du chapitre recherché. En effet, les documents qui contiennent un chapitre "recherche information structuré" et qui contiennent un chapitre différent de ce dernier dont une section porte sur "XML" représentent une réponse valide.
La requête du tableau 7.13 exige que la section soit une section du chapitre recherché et est équivalente à la requête du tableau 7.14.

```
<AND>
<chapitre>
<titre> recherche information structuré </titre>
</chapitre>
<chapitre>
<section>
<titre> XML </titre>
</section>
</chapitre>
</AND>
```

TAB. 7.12 – Rechercher les documents qui ont comme *titre* de *chapitre recherche information structuré* et comme *titre* de *section XML*

```
<chapitre>
<AND>
<titre> recherche information structuré </titre>
<section>
<titre> XML </titre>
</section>
</AND>
</chapitre>
```

TAB. 7.13 – Requête AND (1)

```
<chapitre>
<titre> recherche information structuré </titre>
<section>
<titre> XML </titre>
</section>
</chapitre>
```

TAB. 7.14 – Requête AND (2)

La négation : le constructeur NOT

Le constructeur NOT possède deux sémantiques différentes selon que son argument est une requête porteuse de contenu ou pas. Dans le cas où l'argument ne porte pas de contenu textuel, la négation porte sur la présence de la structure de la requête argument. Dans l'exemple du tableau 7.15, l'utilisateur recherche les livres qui n'ont pas d'épilogue.

```
<livre>
<NOT>
<épilogue/>
</NOT>
</livre>
```

TAB. 7.15 – Rechercher les livres qui n'ont pas d'épilogue

Dans le cas où l'argument porte un contenu textuel, la négation porte sur la présence de la structure de la requête argument avec son contenu. Si l'utilisateur recherche les chapitres dont les sections ne portent pas sur la recherche d'information sémantique la requête est exprimée comme dans le tableau 7.16.

```
<chapitre>
<NOT>
<section>
<titre> recherche information sémantique</titre>
</section>
</NOT>
</chapitre>
```

TAB. 7.16 – Rechercher les chapitres qui n'ont pas de section qui ont pour titre "recherche information sémantique"

7.3 Décomposition de la requête

Notre langage de requêtes permet d'exprimer des requêtes en spécifiant les éléments de structure ainsi que le contenu souhaité. Cette construction est enrichie par un ensemble de balises prédéfinies.
L'avantage d'avoir un langage de requête en XML est d'en faciliter l'usage à l'utilisateur. De plus, la requête peut être représentée sous forme d'arbre où sont combinées les balises de la structure et les balises booléennes. Afin de faciliter le traitement des requêtes nous les décomposons en **requêtes élémentaires** qui à leur tour sont décomposées en **sous-requêtes élémentaires**. La décomposition nous permet de décomposer la requête en un ensemble de requêtes élémentaires reliées par des opérateurs ensemblistes qui correspondent aux opérateurs binaires (voir figure 7.3. Nous procédons, cependant, avant la décomposition, à transformation de l'arbre de la requête de manière à retrouver les balises qui portent sur la structure au niveau des feuilles de l'arbre.

Définition

Une requête valide est un arbre $Q =< n_0, N, H, label, T, \Psi_q, \omega_q >$
- n_0 : le nœud racine
- Ψ_q : l'ensemble des termes désignant les noms de balises de structure[118]
- ω_q désigne les termes contenus dans les balises

[118]Les balise de structure sont définis par l'utilisateur et peuvent comporter la balise ANY.

- $T \subseteq N \times 2^{\omega_q}$: la répartition du termes dans les balises.
- N : l'ensemble des nœuds tel que $N = N_u \cup N_b$ avec
  - N_u : l'ensemble des nœud tel que $label(n) \in \Psi_q$.
  - N_b : l'ensemble des nœuds désignant les opérateurs $label(n) \in AND, OR, NOT$.
- $H : N \rightarrow 2^N$ désigne l'ensemble des arcs de l'arbre :
  - $\forall n$, si $label(n) \in \{OR, AND\}$ on a $|H(n)| = 2$ et si $label(n) = NOT$ on a $|H(n)| = 1$.
  - $\forall n \in N$ on a $|H^{-1}| = 1$

La transformation de l'arbre de la requête consiste à obtenir un arbre où les balises de type and et or sont remontées au plus haut de l'arbre suivies par les balises not et le reste des balises. La transformation est fondée sur les trois règles suivantes :
Soit $a1, a2, a3$ l'ensemble des balises de structure ($\in N_u$)

- *règle 1* :
  - $not(and(a2, a3)) \equiv or((not(a2)), (not(a3)))$
  - $not(or(a2, a3)) \equiv and((not(a2)), (not(a3)))$
- *règle 2* :
  $a1(and(a2, a3)) \equiv a1(a2, a3)$
- *règle 3* :
  $a1(or(a2, a3)) \equiv or((a1(a2)), (a1(a3)))$

La figure 7.1 schématise ces règles.

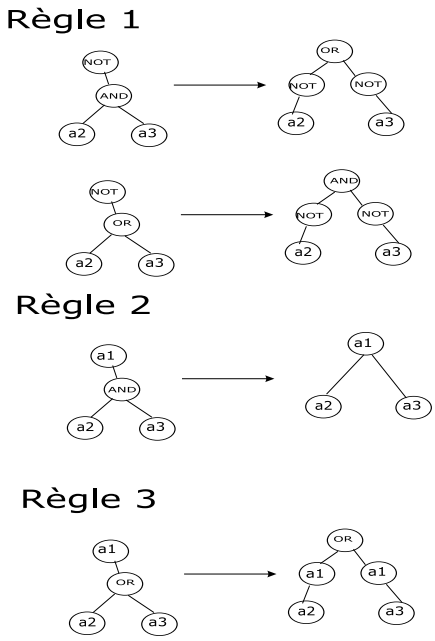

FIG. 7.1 – Les règles de transformation de l'arbre de la requête

Le principe de l'algorithme de transformation consiste à effectuer un parcours en largeur en partant de la racine en appliquant les règle présentées plus haut. La requête de la figure 7.2 b) est la transformation de la requête de la figure 7.2 a)(en remontant les opérateurs AND et OR).

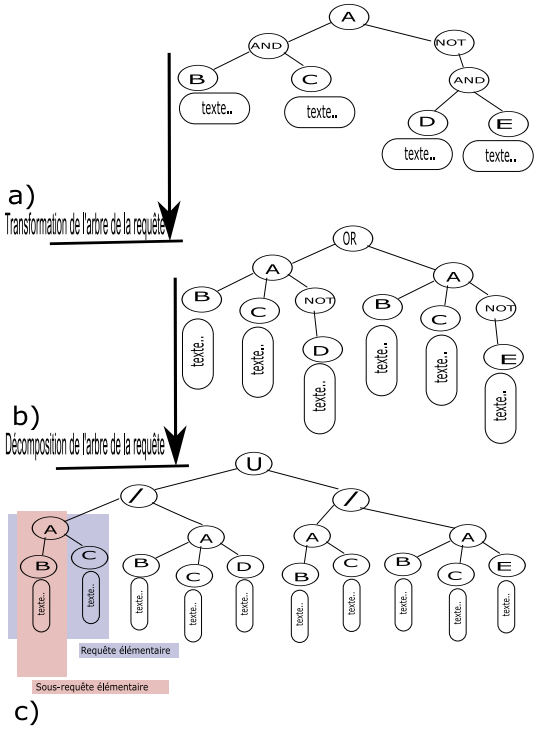

FIG. 7.2 – Exemple de transformation et décomposition de requête

Nous procédons, après la transformation, à la décomposition de la requête en requêtes élémentaires qui à leur tour sont décomposés en sous-requêtes élémentaires (voir figure 7.2). Une sous-requête élémentaire correspond à une chemin. Notez qu'une requête par mots-clés sera directement décomposée en sous-requêtes élémentaires.
Les résultats issus de chaque sous-requête seront agrégés selon des opérations ensemblistes en fonction de l'opérateur booléen.
L'évaluation des requêtes dépend de leur nature, le traitement des requêtes orientées contenu est cependant repris pour les requêtes orientées structure et contenu, en prenant en compte la dimension structurelle. La prise en compte des opérateurs booléens est la même pour les deux types de requêtes.
Nous présentons, dans ce qui suit, la manière dont on utilise la richesse des informations de notre index afin de répondre le mieux aux requêtes de l'utilisateur.

7.4 Évaluation des requêtes orientées contenu

Les requêtes orientées contenu sont formulées par l'utilisateur avec une ignorance ou une indifférence de la structure des réponses. Les requêtes sont décomposées en sous-requêtes élémentaires.
Le but du traitement des requêtes orientées contenu est de retrouver l'élément minimal qui comporte les mots-clés recherchés. Nous faisons donc l'hypothèse que les mots-clés qui figurent dans un même contexte sont à favoriser. Ainsi si l'utilisateur recherche les termes Web et $ontologie$, nous recherchons les réalisations des contextes qui regroupent ces deux termes.

Le traitement des requêtes orientées contenu se compose de trois étapes :

1. Identification des contextes structurels de l'index qui comportent les termes de la requête : nous maintenons pour chaque contexte structurel un vecteur de termes qui constituent son vocabulaire ainsi que leurs représentativité (par le *semTF-CF* au nivau du corpus). Le vocabulaire des contextes nous oriente vers les contextes pertinents. Cette phase d'identification permet de réduire considérablement le nombre de contextes à traiter. En effet, nous n'avons pas besoin de parcourir toutes les réalisations des contextes au niveau des documents. Nous sélectionnons les contextes qui ont le meilleur score de concordance avec les sous-requêtes élémentaires.
2. Calcul des éléments à retourner : pour chaque contexte retenu par l'étape précédente nous calculons la concordance de sa réalisation dans les documents avec la sous-requête élémentaire et nous recherchons le plus petit sous-arbre du document à retourner.
3. Classement des réponses : le système présente les éléments retrouvés à l'étape précédente par ordre décroissant de leur score.

7.4.1 Identification des contextes pertinents

Chaque sous requête élémentaire est représentée par un vecteur de termes avec leurs poids :

$$R_{s-elem} = ((t_1, w_1^q), (t_2, w_2^q), \ldots, (t_m, w_m^q)) \text{ tels que } t_i \in \omega_q, i : 1..m$$

Chaque contexte structurel est représenté par un vecteur de termes qui constituent son vocabulaire avec leurs $semTF - CF$ qui expriment le pouvoir de représentativité des termes dans ce contexte au niveau du corpus 6.6 :

$$C_e^\sigma = ((t_1, w_1^{c_e^\sigma}), (t_2, w_2^{c_e^\sigma}), \ldots, (t_n, w_n^{c_e^\sigma})) \text{ tels que } t_i \in f(c), i : 1..n$$

Nous calculons la concordance par le $cosinus$ des deux vecteurs :

$$concordance_{s-elem-context} = cos(R_{s-elem}, C_e^\sigma)$$

Avant de calculer la concordance entre les contextes et les sous-requêtes élémentaires, nous devons fixer les sens des termes reliés à l'ontologie. En effet, nous avons souligné dans le chapitre précédent qu'un même terme peut être relié à des concepts différents en fonction du contexte où il apparaît. Par exemple le terme *avocat* peut avoir le sens de *fruit* dans un contexte *A* et celui d'un *homme de loi* dans un contexte *B*. La détermination du concept relatif au terme de la requête est donc en fonction du contexte dans lequel on recherche ce terme ; deux cas sont possibles :

1. le terme t_i^q existe dans le vocabulaire du contexte. Dans ce cas le concept (sens) attribué est celui qui a été retenu pour le contexte structurel ($\varphi(t_i^q)$). Rappelons que φ_c n'est pas totalement définie sur le vocabulaire de C_e^σ, (i.e. des termes peuvent ne pas être rattachés à un sens).

2. le terme t_i^q n'apparaît pas dans le vocabulaire, ce problème de **"non-correspondance"** est connu en RI classique par *term mismatch*. Nous définissons le sens du terme en fonction de ceux retenus par les autres termes de la requête. Dans ce cas, si on trouve un sens au terme, nous calculons aussi son $semTF-CF$ dans le contexte de l'index et on le rajoute au vecteur du contexte pour le calcul de la concordance.

On associe à chaque terme de la requête t_i^q, $\varphi_q^c(t)$ qui fixe un sens parmi $S_{t_i}^q$ (l'ensemble des sens possibles du terme selon les fonction S et h de l'ontologie de l'index).

$$\varphi_q^c(t) = \left| \begin{array}{l} \varphi_c(t) \text{ si } t \in f(C_e^\sigma) \\ s \in S_{t_i} \text{qui maximise } desambig \quad \text{sinon} \end{array} \right.$$

Le processus de désambiguïsation est similaire à celui qu'on a appliqué aux termes de l'index. Le sens est fixé selon les poids des termes, les densités du graphe de similarité et la valeur de similarité :

$$desambig = \underset{s \in S_{t_i}^q}{Max} \sum^{t_j^q \in \omega_c} sim(s, \varphi_c(t_j^q)) * \frac{|\{t_j^q \in \omega_c | sim(s, \varphi_c(t_j^q)) > 0\}| * w_{t_j}^q}{|\omega_q|} \tag{7.1}$$

Notons que la fonction $desambig$ est utilisée seulement dans le cas où le terme t_i^q est ambigu, le problème d'attribution de sens ne se pose pas si le terme n'a qu'un seul sens possible.

Cette fonction profite de la désambiguïsation des termes lors de l'indexation et n'alourdit donc pas les traitements au moment de l'interrogation. En effet, les termes qui existent dans le vocabulaire du contexte étant déjà désambiguïsés, la fonction $desambig$ ne concerne que les termes de la requête qui n'existent pas dans l'index et qui sont ambigus.

Si un sens a été attribué, nous calculons le poids du terme qui n'existait pas dans le contexte par :

$$w_i^{c_e^\sigma} = \sum^{t_j \in f(c)} TF-CF(c, t_j) * sim(\varphi^c(t_j), \varphi_q^c(t_i)) \tag{7.2}$$

Si par exemple nous avons une requête R_{s-elem} avec les mots clés $mouse$ et $wolf$ avec un poids de 1. Le terme $mouse$ existe dans le corpus, son sens varie selon qu'il se trouve dans les contextes C_B^A ou C_C^A. Le terme $wolf$ n'existe pas dans les contextes du corpus, la désambiguïsation des termes de la requête fixe le sens de $wolf$ en fonction du sens attribué à $mouse$[119].

Les nouveaux poids du terme $wolf$ dans les contextes est présenté dans le tableau 7.17, par exemple $semTF-CF(wolf, C_B^A) = (1.99 * 0.75) + (0.66 * 0.4) + (1.5 * 0.28) = 2.176$.

[119] Le sens est donc indirectement les sens attribués aux autres termes du contexte.

contexte & terme		$TF-CF$	$SemTF-CF$
$C_A^{\emptyset}$	dog	1.65	1.669
	animal	0.6	0.652
	Wolf	0	1.261
C_B^A	dog	1.99	2.647
	cat	0.66	1.796
	mouse	1.5	2.242
	Wolf	0	2.176
C_C^A	mouse	2.33	2.98
	keyboard	1.25	2.46
	Wolf	0	0

TAB. 7.17 – Exemple de contextes avec les termes, $TF-CF$ et $semTF-CF$

Sim	dog	cat	mouse#1	mouse#2	animal	wolf	Keyboard
dog	1	0.36	0.28	0	0.032	0.75	0
cat	0.36	1	0.28	0	0.036	0.4	0
mouse#1	0.28	0.28	1	0	0.048	0.28	0
mouse#2	0	0	0	1	0	0	0.52
animal	0.032	0.036	0.048	0	1	0.04	0
wolf	0.75	0.4	0.28	0	0.04	1	0
keyboard	0	0	0	0.52	0	0	1

TAB. 7.18 – Similarité entre concepts [120]

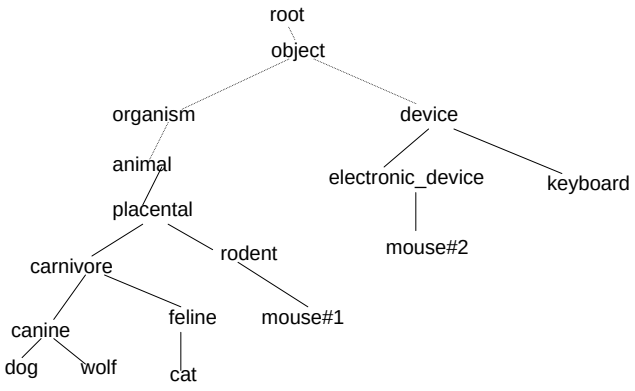

FIG. 7.3 – Extrait de WordNet

7.4.2 Calcul des éléments à retourner

L'étape précédente nous renvoie une liste de contextes du corpus avec un score de concordance avec la sous-requête. Nous sélectionnons les contextes dont le score est supérieur à un seuil

donné à définir empiriquement[121]. Chaque contexte nous fournit un ensemble de documents potentiellement intéressants pour la requête dans lesquels il apparaît. Pour chaque instance n du contexte C retenu dans le document d on calcule, le vecteur de poids des terme de la requête noté $V^{c,d,n}$ comme suit :
$V^{c,d,n} = \{(t_1, w_1^{c,d,n}), ..., (t_n, w_n^{c,d,n})\}$:

Les poids, $w_i^{c,d,n}$ des termes sont les $semTF - ICDF$ des termes t_i dans l'instance n du contexte c dans le document d ($semTF - ICDF(t_i, c, d, n)$), dans le cas où nous avons retenu un sens pour un terme t_i qui n'existait pas dans le vocabulaire du contexte C, son poids dans le vecteur est calculé par :

$$w_i^{c,d,n} = \sum^{t_j \in w_d(n)} TF - ICDF(t_j, d_k, C, n) * sim(\varphi^c(t_j), \varphi_q^c(t_i)) \tag{7.3}$$

Rappelons que $w_d(n)$ est l'ensemble des termes qui apparaissent dans l'instance n dans le document d.
Nous calculons $PR(V^{c,d,n}, R_{s-elem})$, la concordance entre l'instance n du context C du docuement d et la sous-requête élémentaire par le cosinus de leurs vecteurs :

$$PR(C^{c,d,n}, R_{s-elem}) = cos(V^{c,d,n}, R_{s-elem})$$

Un même document d peut être retourné par plusieurs instances de contexte. Nous devons définir celles à retourner à à l'utilisateur. Nous nous fondons sur le calcul de concordance pour décider du contexte à retourner. Les contextes à retourner sont ceux qui *se suffisent à eux-même*, c'est-à-dire qui ont une valeur de concordance supérieure aux instances des contextes plus haut dans l'arbre.
Pour cela nous calculons les concordances pour les nœuds père d'au mmoins deux instances de contexte. Pour avoir la listes des nœuds qui vont servir à ce calcul, nous calculons pour l'ensemble des instances retenues pour chaque document, le $ppeg$ (**le plus petit élément généralisant**), l'élément le plus bas qui généralise leurs abstractions (les contextes qu'ils instancient). Afin de réduire le nombre de nœuds à parcourir, nous calculons le $ppeg$ au niveau de l'arbre réduit.

Définition

> Soit AR l'arbre réduit d'un document, $E_{e'}^{\sigma'} = ppeg(C_{e_1}^{\sigma_1}, C_{e_2}^{\sigma_2}, .., C_{e_n}^{\sigma_n})$ ($E_{e'}^{\sigma'} \in E^{AR}$) est le plus petit élément qui généralise $C_{e_i}^{\sigma_i}$ si $\exists n' \in inst(E_{e'}^{\sigma'})$ tel que $\forall i, j$ avec $n_i \in inst(C_{e_i}^{\sigma_i})$ on a $n_i \in SP_{n'} \wedge \nexists E_{e''}^{\sigma''} \neq E_{e'}^{\sigma'}$ avec $n'' \in inst(E_{e''}^{\sigma'})$ où $\forall i, j$ avec $n_i \in inst(C_{e_i}^{\sigma_i})$ on a $n_i \in SP_{n''}$.

Nous construisons une projection sur l'arbre réduit du document d_k en fonction des contextes retournés par l'étape précédente et leurs $ppeg$ par un arbre où les nœuds feuilles sont les contextes retournés du document d_k, la racine est l'élement $ppeg$ (n') de tous les contextes et les nœuds internes sont les $ppeg$ entre au moins deux contextes.

[121] On peut aussi prendre les n premiers contextes retournés.

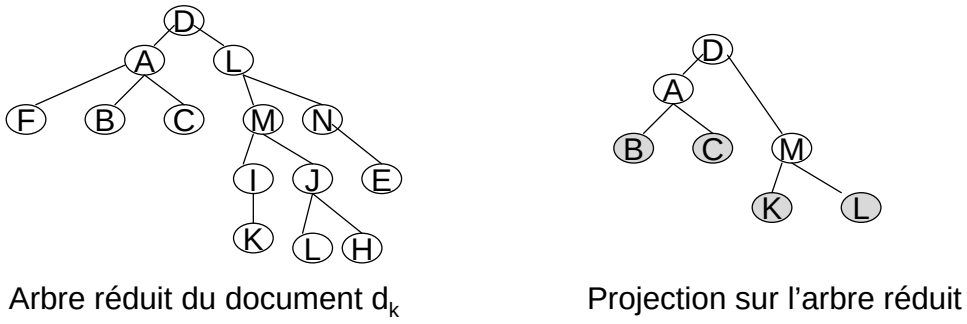

FIG. 7.4 – Un arbre réduit et sa projection en fonction de $C_B^{DA}, C_C^{DA}, C_K^{DLMI}, C_L^{DLMJ}$ et leurs $ppeg$
122

Dans la figure 7.4, $E_D^{\emptyset}$ est le $ppeg$ de $C_B^{DA}, C_C^{DA}, C_K^{DLMI}, C_L^{DLMJ}$, les nœuds internes E_D^A et E_M^{DL} sont respectivement les $ppeg$ de C_B^{DA}, C_C^{DA} et C_K^{DLMI}, C_L^{DLMJ}.
Pour chaque nœud interne n de l'arbre construit, nous construisons son vecteur de poids en agrégeant l'ensemble des nœuds instances qu'il généralise comme suit :
soit n un nœud interne et soit $n_1...n_i..n_m$ les nœuds instances tel que $n_i \in SP_n$ alors on a $V^{d,n} = ((t_1, w_1^{d,n}, ..., (t_j, w_j^{d,n}$ où :

$$w_j^{d,n} = \sum^{n_i \in SP_n} \frac{1}{1+dist(Inst^{-1}(n), Inst^{-1}(n_i))} * w^{dc,d,n_i}$$

Notez que $Inst^{-1}(n)$ est le modèle d'élement de n : $E^{\sigma}_{label(n)}$, et qu'il peut être un contexte et comporter à son tour du texte, si t_i apparaît déjà dans le contexte, son poids n'est pas recalculé et sera égal $semTF - ICDF(t_i, d, C^{\sigma}_{label(n)}, n)$.
Nous calculons pour chaque nœud interne n sa pertinence $PR(n, R_{s-elem})$ par le cosinus entre le vecteur du nœud ($V^{d,n}$) et la sous-requête élémentaire. Nous définissons les règles suivantes, pour chaque $E^{\sigma}_{label(n)}$ de l'arbre construit :

1. Si $PR(n, R_{s-elem}) > Max(PR(n_i, R_{s-elem}))$ alors l'élément n est retenu pour le document d à la place des n_i qui ont servi au calcul.
2. Sinon les instances n_i sont considérées comme indépendantes et seront retournées séparément.

L'idée derrière une telle approche est de retourner à l'utilisateur une portion cohérente du document qui regroupe l'information recherchée. Si nous considérons deux contextes où chacun contient une partie des termes de la requête, pris indépendamment les deux parties du document donnent des éléments de réponse partielle par rapport à la requête. Le fait de les regrouper ensemble donnera une information exhaustive. Or le processus de regroupement des contextes n'a de sens que si les contextes sont proches, il est donc inversement proportionnel à la profondeur de leur $ppeg$. Plus le $ppeg$ est haut, plus le regroupement n'a pas de sens car les termes apparaissent dans des contextes éloignés.

7.4.3 Classement des réponses

La réponse à la requête R_{s-elem} est l'ensemble des éléments retournés par l'étape précédente triés par leur ordre de pertinence. Il est possible d'avoir deux sous-arbres du même document à des emplacements différents dans le classement. Si la requête comporte des opérateurs booléens, les réponses sont agrégées en fonction de ces requêtes (voir section 7.7).

7.5 Évaluation des requêtes orientées structure et contenu

Nous avons exposé dans la section 7.3, la décomposition des requêtes en requêtes élémentaires qui à leur tour sont décomposées en sous-requêtes élémentaires. Une requête élémentaire R_{elem} est donc un ensemble de sous-requêtes élémentaires R_{s-elem}. Nous présentons dans ce qui suit le traitement des requêtes sous-élémentaires ainsi que les requêtes élémentaires.

7.5.1 Traitement des sous-requêtes élémentaires

Les sous-requêtes élémentaires comportent des informations concernant le contenu et la structure. $R_{s-elem} = (C^q_{\sigma_i e_i}, R_{contenu_j})$ où $C^q_{\sigma_i e_i}$ est le contexte de la sous-requête élémentaire et $R_{contenu_j} = (t_1, w^q_1), (t_2, w^q_2), \ldots, (t_m, w^q_m))$ (t_i est un terme de la requête avec son poids w^q_m). Nous supposons que l'utilisateur n'a pas une idée précise de la structure dans laquelle se trouve l'information recherchée. Le système doit quand même retourner des résultats (sur la structure la plus proche de l'esquisse structurelle proposée). Nous considérons donc que le contenu constitue une valeur plus sûre que la structure. Dans un premier lieu nous traitons le contenu textuel comme une requête par le contenu. Ainsi l'exemple de requête de la figure 7.5 est décomposé en $(C^A_B, R_{contenu1})$, $(C^{A,B}_C, R_{contenu2})$, $(C^{A,B}_D, R_{contenu3})$ et $(C^{A,C}_E, R_{contenu4})$.

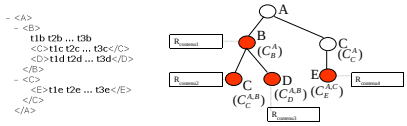

FIG. 7.5 – Exemple de requête et sa représentation en arbre

Chaque $R_{contenu_j}$ est soumis au système comme une requête par le contenu. La réponse à la requête est un ensemble de triplets triés $Rep_{R_{contenu_j}} = \{(d_1, n_1, p_1), \ldots, (d_n, n_m, p_m)\}$ où p_k est la concordance entre $R_{contenu_j}$ et n_i l'instance d'un contexte E_i dans le document d_k : $PR(n^{d_k}_i, R_{contenu_j})$.

Nous voulons retrouver parmi les éléments retournés ceux qui se rapprochent le plus de la structure. Nous proposons une mesure de similarité structurelle qui calcule la similarité entre la structure précisée dans la requête et l'abstraction[123] de celle retournée par la requête sur le contenu ($inst^{-1}(n_i)$), tant au niveau de l'enchaînement des éléments que de leur sémantique. Nous exposons cette mesure de similarité $SSim$ dans la section suivante.
Pour chaque élément retourné, nous recalculons sa pertinence en fonction de sa similarité avec $C^q_{\sigma_i e_i}$. p'_k est la concordance entre $C^q_{\sigma_i e_i}$ et l'instance du contexte $n_i(E^{d_j}_m)$ dans le document d_j :

$$p'_k = p_k * SSim(C^q_{\sigma_i e_i}, E^{d_j}_m)$$

Ainsi par exemple si le contexte $C_C^{AB\#1D\#1}\#1$ est retourné en réponse à la requête $R_{contenu_2}$ avec une pertinence de 0.6, la concordance entre la sous-requête élémentaire $(C_C^{AB}, R_{contenu_2})$ et $C_C^{AB\#1D\#1}\#1$ est calculé par $0.6 * SSim(C_C^{AB}, C_C^{ABD})$.
A l'issu de cette étape nous disposons donc pour chaque sous requête élémentaire d'une liste de triplets $Rep_{R_{s-elem_i}} = \{(d_1, n_1, p'_1)..., (d_n, n_m, p'_m)\}$ triés par ordre de pertinence décroissant.

7.5.2 Traitement des requêtes élémentaires

Il s'agit maintenant de verifier la validité structurelle de la requête élémentaire, pour chacun des documents retournés. Pour ce faire, nous choisissons, pour chaque document, l'élément qui a la pertinence maximale parmi les éléments retournés à l'étape précédente.
Nous disposons, pour chaque document apparaissant au moins une fois dans les réponses, de l'ensemble de ses parties qui répondent au mieux à chacune des sous-requêtes élémentaires $\{(n_1, R_{s-elem_1}, p'_1), \ldots, n_i, R_{s-elem_i}, p'_i)\}$.
Les requêtes R_{s-elem_i} répondent à une organisation entre eux, dans la figure 7.5, le contexte C_C^{AB} et C_D^{AB} sont des spécialisations du contexte C_B^A.
Il s'agit dans cette étape de verifier que les éléments retenus pour chaque document, reprennent au mieux l'organisation exigée par la requête.
Nous calculons pour cela P_{d_k} la projection des abstractions des éléments retournés ($inst^{-1}(n_i)$) sur l'arbre réduit du document d_k. Nous présentons dans la figure 7.6, un exemple d'arbre réduit d'un document et sa projection.

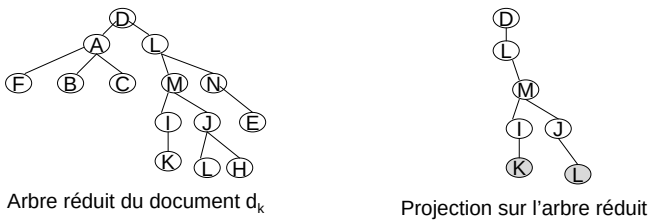

FIG. 7.6 – Un arbre réduit et sa projection en fonction de C_K^{DLMI} et C_L^{DLMJ}

[123]Nous avons besoin de cette abstraction pour comparer deux vues équivalentes de la structure. Les expressions de chemins des requêtes sont équivalents aux modèles de balises dans l'arbre réduit des documents.

Nous calculons la similarité structurelle $SSim$ entre R_{elem} et P_{d_k}. Cette similarité nous renseigne sur la relation qu'entretiennent les éléments de la réponse entre elles par rapport à la structure globale de la requête élémentaire, en gardant le même appariement R (cf. section suivante pour la définition d'un appariement).
Nous calculons la concordance de P_{d_k} pour chaque document d_k avec la requête élémentaire R_{elem} en fonction de la pertinence de ses éléments retenus de l'étape précédente et la mesure de similarité structurelle entre R_{elem} et P_{d_k}, comme suit :

$$p''_k = concordance(P_{d_k}, R_{elem}) = \frac{SSim(R_{elem}, P_{d_k}) * \sum^i p'_i}{|R_{s-elem}|}$$

L'élément retourné pour le document d_k est l'instance du $ppeg$ entre les $inst^{-1}(n_i)$ retenus qui aura comme score p''_k.

7.6 Similarité structurelle

Une structure est un arbre tels que les nœuds sont étiquetés par les balises et les arcs représentent l'imbrication entre ces balises.

Définition

Une structure B est un couple $(S_B, \rightarrow_B)$ avec :

- $S_B \subseteq \Psi$: l'ensemble des balises de la structure.
- $\rightarrow_B$: une relation de $S_B \rightarrow S_B$ qui traduit la relation d'imbrication dans l'arbre :
  - $\forall s \rightarrow_B s' \wedge s'' \rightarrow_B s'$ on a $s = s''$ (chaque nœud a un père unique).

Nous définissons la relation $\Rightarrow_B$: $\mathbb{N} \times S_B \rightarrow 2^{S_B}$ avec
$\forall s \in S_B \wedge n \in \mathbb{N}$ on a $s \overset{n}{\Longrightarrow}_B s'$ ssi $s \rightarrow_B s_1 \rightarrow_B ...s_{n-1} \rightarrow_B s'$ (notez que $\overset{1}{\Longrightarrow}_B = \rightarrow_B$).
$\Longrightarrow_B$ est la relation entre les différents nœuds de la structure avec en exposant la distance entre eux, elle est obtenue par la fermeture transitive de la relation père/fils.

Trois critères régissent la mesure de la similarité entre deux structures B_1 et B_2 :

1. Le coût de modification : représente la correspondance entre la présence ou l'absence des éléments entre les deux structures.
2. Le coût du bruit : concerne la sur-specification des liens entre les éléments appariés de B_1 par rapport à B_2.
3. Le coût du silence de structure : concerne les sous-specifications des liens dans les éléments appariés de B_1 par rapport à B_2.

Nous avons mentionné dans le chapitre précédent que les balises peuvent à leur tour être reliées à une ontologie, ainsi si E^A_B et E^G_F font référence au même concept elles sont considérées comme équivalentes et ont une similarité sémantique de 1. Pour le calcul de la similarité structurelle, nous commençons par la définition d'une fonction d'appariement entre deux structures qui repose sur la fonction de similarité sémantique présentée dans le chapitre 5.

7.6.1 Relation d'appariement entre structures

Soient deux structures B_1 et B_2 définies respectivement par $B_1 = (S_{B1}, \rightarrow_{B1})$ et $B_2 = (S_{B2}, \rightarrow_{B2}$). Soit la fonction $[\ ]_f : \Psi \rightarrow \Psi \times [0..1]$ qui désigne une relation d'équivalence entre les balises du corpus avec $[s]_f = \{(s', f(s,s'))|f(s,s') > \beta$ où $f : \Psi \times \Psi \rightarrow [0..1]$ et β un seuil$\}$. Dans notre travail, la fonction f correspond à la fonction de similarité sémantique $sim()$ entre concepts. Deux balises sont donc équivalentes si leur similarité sémantique est au dessus d'un certain seuil.

Définition

Un appariement valide entre deux structures B_1 et B_2 est une relation $R \subseteq S_{B1} \times S_{B2}$ tels que
- $\forall (s_1, s_2) \in R$ on a $s_1 \in [s_2]_f$
- $\forall i \in \mathbb{N} \wedge (s_1, s_2) \in R \wedge (s'_1, s'_2) \in R$ tel que $s_1 \overset{i}{\Longrightarrow}_{B1} s'_1\ \exists j > 0$ tel que $s_2 \overset{j}{\Longrightarrow}_{B2} s'_2$

On note par $|R| = \sum_{(s_i,s_j)\in R} f(s_i, s_j)$ qui désigne la similarité sémantique d'un appariement.
On note $\Re_f^{B_1,B_2}$ l'ensemble des appariements possibles entre les structures B_1 et B_2.

7.6.2 Évaluation d'un appariement

Soient deux structures Q et D définies respectivement par $(S_Q, \rightarrow_Q)$ et $(S_D, \rightarrow_D)$. La similarité structurelle $SSim_R(D,Q)$ entre D et Q par rapport à l'appariement R est définie selon les critères exposés plus haut :

coût de la modification le coût de la modification peut se mesurer par le taux de participation des balises de D appariés selon R avec les balises de Q :

$$Cr1_R(D,Q) = \frac{|R|}{|S_Q|}$$

Cette mesure défini le rapport des balises de D pondérés par leur correspondance lors de l'appariement, sa valeur vaut 1 quand l'ensemble des nœuds de Q sont appariés avec une similarité sémantique de 1.

le bruit de structure le bruit est calculé par un indicateur qui transcrit le rapport entre le nombre de liens entre les nœuds appariés dans D (noté $D_{>_R}$) par rapport au nombre de liens dans Q :

$$Cr2_R(D,Q) = \frac{|\Longrightarrow_Q \cap \Longrightarrow_{D_{>_R}}|}{|\Longrightarrow_{D_{>_R}}|}$$

le silence de structure le silence est calculé par un indicateur qui pénalise un appariement qui ne couvre pas l'ensemble des relations de structuration entre les balises.

$$Cr3_R(D,Q) = \frac{|\Longrightarrow_Q \cap \Longrightarrow_{D_{>_R}}|}{|\Longrightarrow_Q|}$$

Définition

Nous définissons $V_Q^D = (x_1 = Cr1_R(D,Q), x_2 = Cr2_R(D,Q), x_3 = Cr3_R(D,Q))$

$$SSim(D,Q) = \begin{cases} \overset{R \in \Re_f^{Q,D}}{Max} \frac{\sum^{x_i \in V_Q^D} x_i}{\sqrt{3} * \sqrt{\sum x_i^2}} & \text{si } \Re_f^{Q,D} \neq \emptyset \\ O & \text{sinon} \end{cases}$$

7.7 Traitement des opérateurs booléens

Á l'issue du traitement des requêtes élémentaires (ou sous-élémentaires pour les requêtes orientées contenu), nous disposons d'un ensemble de réponses ordonnées selon leur pertinence. Chaque réponse comporte un triplet document-élément-pertinence. Nous détaillons la prise en compte des opérateurs booléens entre les requêtes élémentaires.

7.7.1 Traitement de l'opérateur AND

Soit $Q = Q_1\ AND\ Q_2$ avec Rep_{Q_1} et Rep_{Q_2} les réponses des requêtes élémentaires. La conjonction exprime que l'utilisateur recherche le (ou la) parties du document qui contiennent les deux éléments de réponse. Toutefois plus les deux éléments recherchés sont éloignés plus l'information retournée est moins précise. Pour chaque document figurant comme réponse a au moins une des requêtes, on procède au calcul de l'éléments à retourner et sa pertinence. Pour agréger la pertinence pour un document $d1$ tels que $(d1, n1, P1) \in Rep_{Q1}$ et $(d1, n2, P2) \in Rep_{Q2}$

1. Nous calculons l'élément à retourner qui est $ppeg(E1, E2)$, tels que $Ei = inst^{-1}(ni)$
2. La pertinence du $ppeg(E1, E2)$ est calculée par la somme des pertinences de $n1$ et $n2$:

$$P1 + P2$$

7.7.2 Traitement de l'opérateur OR

Soient deux requêtes $Q = Q_1\ OR\ Q_2$ et par Rep_{Q_1} et Rep_{Q_2} La réponse à la requête est :

$$Rep_Q = Rep_{Q_1} \cup Rep_{Q_2}$$

La pertinence retenue pour un élément n_i est le maximum de sa pertinence dans Rep_{Q_1} et Rep_{Q_2}.

7.7.3 Traitement de l'opérateur NOT

Le NOT exprime une exclusion d'une partie ou du contenu d'une partie de la requête. Nous calculons la difference entre les résultats issues des requêtes élémentaires. Soient $Q = Q_1\ NOT\ Q_2$, nous avons exposé dans la section 7.3, la transformation des requêtes en fonction des opérateurs booléens. Ainsi si $Q = Q_1\ NOTQ_2$, Q_1 est la requête recherchée et Q_2 est la requête Q_1 augmentée par la partie à exclure. On note par Rep_{Q_1} et Rep_{Q_2} les résultats des deux requêtes élémentaires. Le résultat de Q est défini comme suit :
Pour chaque document d_k apparaissant comme résultat de la requête Q_1 :

- si $(n, d_k, P) \in Rep_{Q_1}$ et $(n', d_k, P') \in Rep_{Q_2}$ et $n' \blacktriangleleft n$ alors on a $(n, d_k, P) \in Rep_Q$.
- si $(n, d_k, P) \in Rep_{Q_1}$ et $(n', d_k, P') \in Rep_{Q_2}$ et $n'\neg \blacktriangleleft n$[124] alors si $P > P'$ on a $(n, d_k, P) \in Rep_Q$ sinon $(n, d_k, P'') \in Rep_Q$, tel que $P'' = P - |P' - P|$.

7.8 Conclusion

Dans ce chapitre, nous avons présenté SemIR, un langage de requête pour la recherche d'information dans les documents semi-structurés. Ce langage permet d'exprimer des requêtes par simples mots-clés, des requêtes orientées structure ainsi que des requêtes orientées structure et contenu. Notre système permet de retrouver l'unité d'information minimale à retourner à l'utilisateur. Nous avons pour cela introduit la notion du **plus petit élément généralisant** qui permet de retourner les unités d'information les plus spécifiques et exhaustives.
Le système permet d'interroger des bases documentaires hétérogènes et permet une recherche sémantique au niveau des termes de la requête ainsi qu'au niveau de la structure. L'utilisateur n'a pas besoin d'avoir une connaissance des concepts de l'ontologie associée aux documents, le système profite de l'indexation sémantique des documents pour retrouver les termes synonymes ou dans le même voisinage sémantique que les termes recherchés. L'indexation sémantique permet aussi au système de traiter le problème de "non-correspondance" des termes et retourne des documents même si le terme ne figure pas dans la base documentaire.

[124] n' n'est pas une spécialisation de n.

Chapitre 8

Expérimentations et Evaluations

Sommaire

8.1 Introduction

Nous avons présenté dans la dernière partie de ce manuscrit, nos contributions pour un système d'indexation et de recherche fondé sur la sémantique et la structure. Nous présentons dans ce chapitre des expérimentations préliminaires notamment concernant la mesure de similarité sémantique proposée.
Nos propositions trouvent leur application sur des textes de spécialité structurés "sémantiquement" qui disposent de ressources sémantiques adéquates. Nous présentons brièvement les corpus à notre disposition et montrons les limites des expérimentations dans ce cadre (section 8.3). Nous présentons les résultats de l'évaluation de la mesure de similarité que nous proposons par rapport au corpus de paires de mots de [Finkelstein & E. Gabrilovich, 2002]. Nous présentons également les expérimentations sur le corpus CACM, un corpus de résumés de 3204 articles. Nous avons *XMLisé* ce corpus en fonction des champs déjà définis (.W –> <Abstract>, A. –> <Author>, ect.). Les documents ainsi *XMLisés* sont donnés en entrée à l'indexation. Les requêtes sont traduites en notre langage simplement en les délimitant pas les balises <*ANY*>.
Un des objectifs de cette thèse est d'avoir à notre disposition un prototype de recherche pour pouvoir tester la validité de nos propositions. Nous mettons en cours une plateforme expérimentale sur des guides de bonne pratique médicale, cette plateforme avec la présence des experts

nous permettra de valider et d'améliorer nos propositions. Le corpus INEX, même s'il n'est pas sémantiquement structuré présente une ressource précieuse, nous pouvons tester l'apport de nos propositions concernant la prise en compte de la structure tels que sur la définition de l'élément le plus pertinent à retourner, la similarité structurelle. La tâche corpus hétérogènes présentée dans la section 2.8.2, nous permettra de tester l'apport de la fonction d'appariement sémantique R. Nous devons pour mener à bien ces expérimentations avoir recours à des bases de donnés pour le stockage des index. En effet le système actuel est fondé sur la sérialisation des objets [125], ce qui ne peut pas être envisageable pour traiter de gros corpus.
Nous commençons par présenter le prototype implémenté **SemIndex** et **SemIR** ainsi que les différentes APIs utilisées.

8.2 Le prototype

8.2.1 Modèle de classes

Le modèle de classes présenté à la figure 8.1 représente les différentes classes implémentées par SemIndex et SemIR.

[125] La sérialisation permet de rendre un objet persistant ; c'est-à-dire le "sauvegarder" sous une certaine forme (fichier) qu'on peut récupérer par la suite sous sa forme d'objet initial.

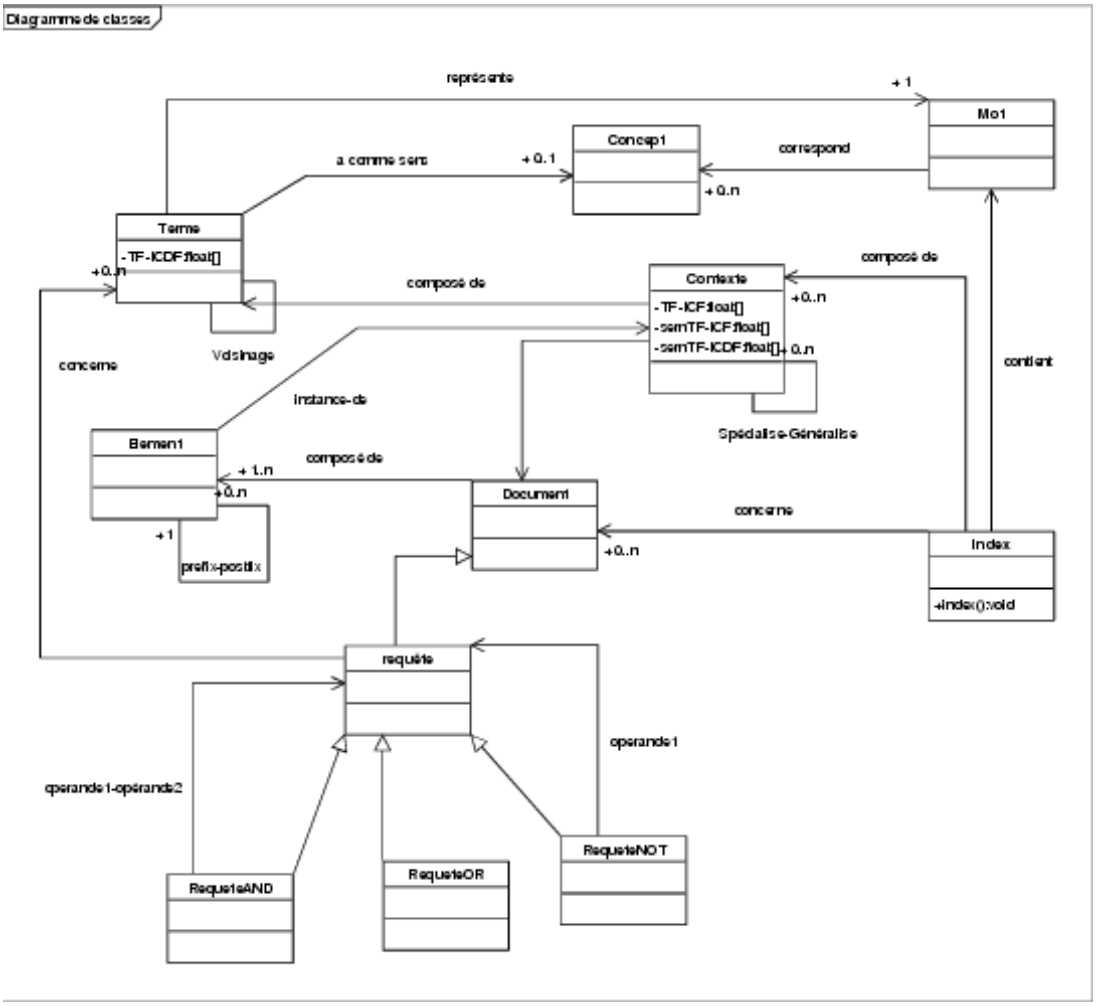

FIG. 8.1 – Le diagramme des classes

Nous décrivons les classes modélisées dans le tableau 8.2.1 :

Classe	Descriptif
Index	C'est la classe principale de l'index. Elle est composée d'une liste de *documents*, une liste de *mots* et une liste de *contextes*.
Contexte	Cette classe implémente la notion de contexte présentée dans le chapitre 6. Chaque contexte est défini par l'ensemble d'*éléments* qui le réalisent ainsi qu'un lien vers toutes ses instances dans les *documents*. Il maintient également la liste des *termes* (le vocabulaire du contexte). Pour chaque *terme* dans un contexte, nous stockons les indicateurs $TF - ICF$ et $semTF - ICF$.
Mot	Cette classe est la représentation des *termes* en tant que chaîne de caractères. Chaque *mot* est ainsi relié aux *concepts* dont il peut être la representation lexicale.
Terme	Elle représente un *mot* dans un contexte donné et maintient donc un lien vers celui-ci. Contrairement au *mot* qui peut être rattaché à plusieurs *concepts*, un *Terme* est rattaché à un seul concept. Ce rattachement est, bien entendu, fixé lors de la phase de désambiguïsation. Pour chaque terme nous maintenons une liste d'indicateurs : $TF - ICDF$ et $semTF - ICDF$.
Document	Chaque document du corpus est parsé et un arbre réduit d'*élément* est crée. Chaque document est défini donc par une liste d'*éléments* organisés sous forme d'arbre.
Element	Les éléments sont structurés en arbre. Les éléments instances d'un *contexte* possèdent un lien vers celui-ci.
Requête	C'est une sous classe de *Document*. Une requête étant un document XML qui répond à un schéma particulier. Cette classe est à son tour spécialisée en trois sous-classes : $ANDRequest$, $ORRequest$, $NotRequest$. Chacune de ses classes spécialise la classe requête en rajoutant les opérandes qui sont des objets de type requête (un ou deux selon l'opérateur).

TAB. 8.1 – Descriptif des classes proposées

8.2.2 Environnement technologique

Le prototype qui implémente notre modèle a été développé entièrement en Java (1.5). Le choix de Java a été effectué en raison du nombre d'APIs existant qui ont servi à la réalisation de notre outil, en voici la liste.

Les APIs XML Afin de parser les documents XML nous avons utilisé le parseur SAX. Le choix de SAX a été effectué afin d'avoir le plus de liberté dans la construction de l'arbre du document. En effet lors du parsing nous générons en même temps aussi bien l'arbre étiqueté du

document que l'arbre réduit. Cela aurait nécessité un traitement en deux temps si nous avions choisi DOM par exemple.

Les APIs WordNet Nous avons utilisé deux APIs (*JwordNet 2.0*)[126] qui permettent d'accéder à la base de données de Wordnet. Chaque API est relative à une version de WordNet (version 1.7 et 2.0). Bien que la version 2.0 est plus riche en terme de synsets, certains problèmes liés à la génération des arbres des hyperonymes et des hyponymes existent. Ces problèmes n'existent pas avec la version C de l'API. Nous nous sommes donc contenté de la version 1.7 de WordNet. Ce choix est de plus motivé par le fait que les autres mesures de similarités ont été testé sur cette version.

Les API OWL Bien que le travail effectué utilise essentiellement Wordnet comme ressource sémantique. Notre outil peut également supporter des ontologies de type OWL. Nous avons utilisé la version 1.4.2 de $owlAPI$[127]. Cette version nécessite, bien évidemment, un parseur RDF et une archive de validation qu'on a également intégré à nos librairies.

Les APIs JGraph L'API JGraph[128] nous offre la possibilité de manipuler des graphes. Cette API est utilisée parce que : (i) elle nous offre un outil pour visualiser les graphes et (ii) elle nous permet de modéliser le problème de désambiguïsation en un problème d'affectation dans un graphe bipartite avec les nœuds mots et concepts (ou sens).

8.3 Point sur les corpus

Les corpus sous format XML ne sont pas nombreux, le seul à notre connaissance qui ait mis à disposition des jugements de pertinence est le corpus d'INEX.
Les autres corpus que nous avons en notre possession ne ce sont pas avérés utiles pour des expérimentations d'envergure. Nous décrivons ces corpus en présentant quelques extraits.

– Shakespeare regroupe les pièces de shakespeare, un exemple de document a été présenté dans le chapitre 6.
– Mondial regroupe des informations sur les continents, pays villes, ... du monde. Voici un extrait :

```
<country car_code="TN" area="163610" capital="cty-Tunisia-Tunis">
  <name>Tunisia</name>
  <population>9019687</population>
  <government>republic</government>
  <encompassed continent="africa" percentage="100"/>
  <ethnicgroups percentage="1">Jewish</ethnicgroups>
  <ethnicgroups percentage="1">European</ethnicgroups>
  <ethnicgroups percentage="98">Arab-Berber</ethnicgroups>
  <religions percentage="98">Muslim</religions>
```

[126] Accessible à http ://prdownloads.sourceforge.net/jwn/JWordNet.zip
[127] Accessible à http ://owl.man.ac.uk/api.shtml
[128] Accessible à http ://www.jgraph.com/

```
    <religions percentage="1">Jewish</religions>
    <religions percentage="1">Christian</religions>
    <city id="cty-Tunisia-Tunis" is_country_cap="yes" country="TN">
        <name>Tunis</name>
        ..
    </city>
  </country>
```

– SIGMod regroupe les références des publications du SIGMod. Voici un extrait :

```
  <article>
    <title articleCode="172089">
    Partition semantics for incomplete information in
    relational databases
    </title>
    <authors>
        <author authorPosition="01">D. Laurent</author>
        <author authorPosition="02">N. Spyratis</author>
    </authors>
   <initPage>66</initPage>
   <endPage>73</endPage>
   ..
    </article>
```

– DBLP regroupe des références bibliographiques de publications (thèses, articles de journaux, etc.).

– Menelas est un corpus dans le domaine médical qui a été construit durant le projet Menelas [Zweigenbaum, 1995]. Il contient un chapitre de livre sur la coronarographie et des comptes rendus d'hospitalisation. Ce corpus a fait l'objet d'un encodage suivant le *TEI XML Corpus Encoding Standard* [Habert *et al.*, 2001]. La structuration concerne essentiellement les méta-données (e.g. <docAuthor>, <address>, etc.). Le contenu textuel est structuré en paragraphe suivant les sauts de lignes du texte original. Voici un extrait d'un compte rendu d'hospitalisation. Voici un extrait :

```
<text complete="n"> <body>
    <p> Malade de 65 ans, ancien tabagique, ayant fait
    un IDM inférieuren 1984 compliqué d'OAP et qui a
    bénéficié en Juillet 1984 d'un triple pontage.
    </p>
    <p> En Juillet 1985, il est à nouveau hospitalisé
    dans le Service pour reprise de l'angor et un Thallium
    positif.
    </p>
    ..
    <p> On augmente la posologie de bétabloquants et de
    TILDIEM. ..
    </p>
```

```
    </body>
  </text>
```

– Inex est le corpus de la campagne d'évaluation INEX, présentée dans le chapitre 2.

L'avènement de XML est assez récent et les corpus constitués sont encore rares. La majorité des corpus présentés (Mondial, DBLP, SIGMod) sont centrés données et ne sont pas accompagnés de requêtes. Le corpus des pièces de Shakespeare est plus riche en texte mais l'ensemble des requêtes fournies est plus orienté pour des systèmes de Question/Réponse. Voici un exemple de requête :

```
<query no="26">Who does Antony marry?</query>
```

Seul le corpus INEX, à notre connaissance, a été réalisé à des fins de recherche d'information. La majeure partie textuelle reste néanmoins confinée dans des balises de structuration logique (section, paragraphe, etc.). Nous avons bâti notre modèle sur l'hypothèse que la structuration permet de limiter le champs sémantique du vocabulaire. Cela n'est réalisé par aucun des corpus que nous avons étudié. Il est difficile de proposer une structuration "sémantique" automatique de corpus, les travaux en segmentation thématique [de Chalendar & Grau, 2000] [Hernandez, 2004] peuvent constituer des pistes intéressantes. Les corpus spécialisés (ou de spécialité) comme dans le domaine juridique ou médical sont probablement plus facile à structurer que les textes généraux.

Le corpus Menelas a été construit pour la mise en place d'une ontologie dans le domaine médical. La version qui a servi à ce travail n'est pas sous forme XML. Le travail de structuration, qui s'est fait en aval, s'est focalisé sur les méta-données, nous pouvons néanmoins imaginer une structuration sémantique pour les comptes rendus d'hospitalisation. Ainsi l'exemple présenté ci-dessus peut devenir :

```
<text complete="n"> <body>
  <age> Malade de 65 ans <age>
  <antecedents>
    <antecedent> ancien tabagique </antecedent>
    <antecedent> ayant fait un IDM inférieuren 1984
    compliqué d'OAP </antecedent>
  </antecedents>
  <interventions>
    <intervention> qui a bénéficié en Juillet 1984
    d'un triple pontage. </intervention>
    <intervention> En Juillet 1985, il est à nouveau hospitalisé
    dans le Service pour reprise de l'angor et un Thallium
    positif.
    </intervention>
  </interventions>
  ..
  <traitement> On augmente la posologie de bétabloquants et de
  TILDIEM. ..
  <traitement>
  </body>
```

```
</text>
```

V. Brunie & Morizet-Mahoudeaux [1999] proposent à ce titre une modélisation des connaissances structurelles documentaires pour les comptes rendus d'hospitalisation. Ils présentent la DTD suivante (extraits) :

```
< ! -- ===== Un compte rendu d'hospitalisation ==== -->
 <!ELEMENT crh - - (En-Tête, Informations, Corps, En-Pied)>

<! --  ===== Informations sur le patient/le document ==== -->
 <!ELEMENT Informations -  - (Titre, Nom, Prenom, Date-Entree,
 Date-Sortie, Date-Naissance, Auteur-CRH, Numero-Patient,
 Destinataire+)>
 <!ELEMENT Nom - - Libelle-Info, %TextField ;)>

<! -- ===== Sections de niveau 1 et 2 ==== -->
 <!ELEMENT Corps - -(Motif-admission, Provenance, Mode-vie-Act,
 Histoire-Maladie, Entree, Sejour, Sortie, Conclusion) >
 <!ELEMENT Entree - - (Titre-section, Parag+, Examen-Clinique,
 Examens-Complémentaires, Autres-Examens?, Section*)>
```

V. Brunie & Morizet-Mahoudeaux prennent le pari que l'existence d'une structure bien définie encouragerait les usagers à produire des textes structurés. Nous pensons également que l'apport de la structure aux systèmes de navigation et d'accès à l'information tend à encourager les producteurs de textes à structurer leurs productions. Nous faisons le pari que dans un futur proche il existera beaucoup plus de documents structurés "sémantiquement".
Notre proposition de désambiguisation des termes indexés est différente des propositions classiques. En effet, nous fixons le sens d'un terme donné dans un contexte donné mais par rapport à **tout le corpus**. Ce qui requiert de désambiguiser un terme par rapport à tout le **vocabulaire du contexte**. La complexité de l'algorithme dépend de la taille de ce vocabulaire et pour peu que les contextes soient larges, le calcul devient très coûteux. Nous proposons une restriction du contexte pour les termes qui s'inspirent de la notion de fenêtre (ou scope). Nous sélectionnons les n termes contiguës pour chaque terme à désambiguiser (voir figure 8.2). n doit être fixé empyriquement, un n trop petit ou trop grand peut nuire au résultat en restreignant ou en élargissant trop l'espace de sens.

8.4 Interfaces graphiques

Notre prototype propose un ensemble d'interfaces graphiques qui permettent de :
- Visualiser l'arbre minimal des corpus, cette visualisation permet de naviguer dans les documents.
- Visualiser le vocabulaire des contextes avec la liste des concepts assignés aux termes ainsi que les différentes mesures de pondération.
- Visualiser les similarités entre les concepts par une matrice de similarité ou une cartographie des documents pas contexte.

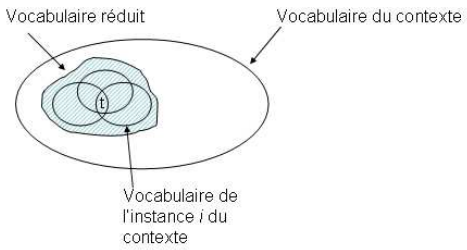

FIG. 8.2 – Réduction des contextes des termes

Accès à l'index La figure 8.3 présente la hiérarchie d'un corpus qui contient des documents de structures différentes. Nous présentons les contextes ainsi que les relations de spacialisation/-généralisation. L'utilisateur peut naviguer dans cette structure et visualiser la liste des termes indexés avec leurs pondérations. L'utilisateur peut ainsi visualiser les concepts qui ont été rattaché aux termes avec la possibilité de supprimer ou d'ajouter des concepts. L'utilisateur peut ainsi valider ou modifier l'indexation.

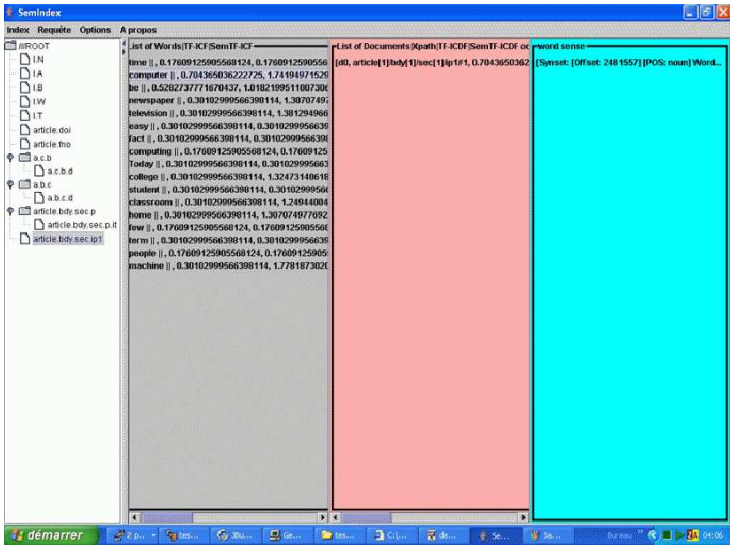

FIG. 8.3 – Visualisation du contenu de l'index

$SemIndex$ permet également de visualiser la cartographie d'un contexte en présentant l'ensemble des concepts retenus et leurs similarités. Cette interface permet de visualiser les relations entre les différents concepts. Cela peut s'avérer intéressant si on veut vérifier la cohérence de la structuration des documents. Les documents faiblement structurés ou structurés non sémantiquement présentent des îlots de concepts éloignés. Dans la figure 8.4 reprend l'exemple des contextes ambigus présenté dans la section 6.5.2. Nous avons combiné les deux contextes sous une même structure, la figure montre clairement qu'il existe de groupes de sens distincts. Le calcul de la similarité sémantique nous permet de détecter les variations sémantiques et donne une idée sur une possible restructuration.

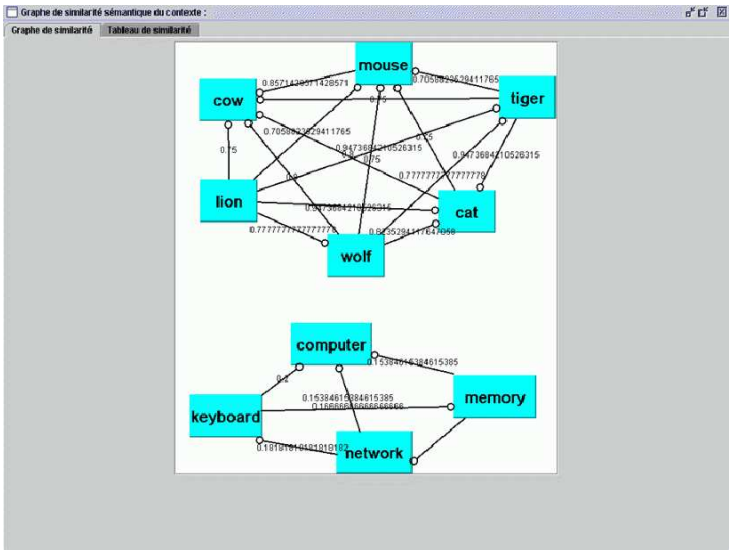

FIG. 8.4 – Visualisation de la cartographie d'un contexte

Recherche d'information $SemIR$ permet à l'utilisateur de poser ses requêtes orientées structure et contenu ou par simples mots clés. Notre outil permet une visualisation de la structure du corpus afin d'aider l'utilisateur à poser ses conditions sur la structure. Nous donnons, dans la figure 8.5, un exemple de requête par mots clés avec les résultats retournés.

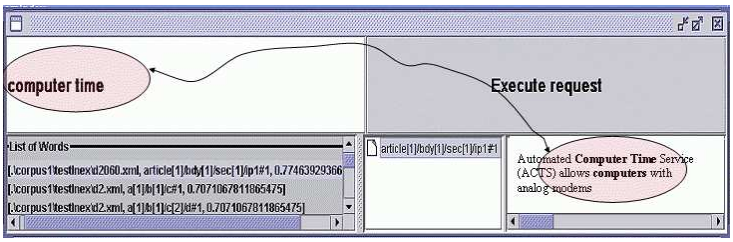

FIG. 8.5 – Visualisation du résultat de l'exécution d'une requête

Le système retourne une liste d'éléments triés pas ordre de pertinence décroissant. L'utilisateur peut visualiser les sous-arbres retournés.

8.5 Evaluation de la similarité sémantique

Nous avons présenté dans le chapitre 5 les différentes expérimentations pour tester la validité des mesures de similarité. Nous reprenons dans cette section ces expérimentations en nous fondants sur des expérimentations plus récentes Finkelstein & E. Gabrilovich [2002] que celles de Miller & Charles [1991] ou Rubenstein & Goudenough [1965]. Ces expérimentations portent

sur 353 paires de mots évaluées par 16 sujets[129]. Nous exposons dans le tableau 8.5 les corrélations des différentes mesures de similarité présentées dans le chapitre 5. Nous utilisons les notations suivantes :
– F pour les jugements humains de Finkelstein & E. Gabrilovich ;
– R pour la mesure de Resnik ;
– HS pour la mesure de Hirst & St-Onge ;
– JC pour la mesure de Jiang & Conrath ;
– L pour la mesure de Lin ;
– WP pour la mesure de Wu & Palmer ;
– HZ pour la mesure que nous proposons [Zargayouna & Salotti, 2004a].

	F	R	HS	JC	L	WP	HZ
Corrélation	1	0.375	0.344	0.357	0.368	0.344	0.371

Les valeurs de R, HS, JC et L présentées dans le tableau 8.5 sont fournies par [Jarmasz, 2003]. Il est aussi possible de les calculer en utilisant par exemple le package $WordNet :: Similarity$[130] développé par Pedersen *et al.* [2004] en Perl, permettant de calculer entre autres mesures celles que nous venons de citer. Nous avons enlevé les paires de mots qui n'existent pas dans WordNet 1.7.1 (Tels que *Arafat*, *Maradona*), ce qui a porté le nombre des paires de mots à 345.
Nous avons implémenté notre mesure similarité et celle de Wu & Palmer, l'étude de corrélation montre que notre mesure est aussi performante que celle de Resnik qui possède la plus forte corrélation. Cela est encourageant sachant que nous ne nous fondons sur aucune analyse distributionnelle de corpus. Les valeurs de corrélation peuvent paraître faibles comparées à celles présentées dans le chapitre 5. Ceci peut être dû à plusieurs facteurs :
– Les jugements de pertinence portent sur la notion de *proximité sémantique*, exposée dans le chapitre 5, qui est plus large que la notion de *similarité sémantique*. Nous donnons l'exemple de *psychology, Freud* jugés 8.210/10 similaires par les utilisateurs. Nous avons par exemple les couples de mots *(telephone, communication)* qui ont eu une moyenne de 7.5/10 par les utilisateurs et 0 par notre mesure. [131]
– L'existence de verbes qui ne sont pas pris en compte pour le calcul de similarité, comme par exemple *(eat, drink)*.
– L'existence d'adjectifs qui ne sont pas pris en compte par notre système, comme par exemple *(smart, student)*. Le sens qui a été pris pour *smart* dans ce cas est celui du nom[132] ce qui éloigne du sens voulu et compris par les utilisateur qui estiment que *student* et *smart* peuvent être similaires à 4.62/10.
Tous ces facteurs diminuent la corrélation, en effet en étudiant de prés les couples de mots qui ont une valeur nulle (voir les figures 8.6, 8.7, 8.8) nous remarquons que les valeurs nulles

[129]Les jugements humains sont accessibles sur http ://www.cs.technion.ac.il/ gabr/resources/data/wordsim353/wordsim353.html
[130]Il est aussi possible de faire ces calculs en ligne à l'adresse : http ://marimba.d.umn.edu/cgi-bin/similarity/similarity.cgi
[131]En revanche dans WordNet, *telephone* est relié à *telecommunication*, *(telephone, telecommunication)* ont une similarité de 0.93
[132]smart, smarting – (a kind of pain such that caused by a wound or a burn or a sore).

correspondent à des valeurs assez élevées dans les jugements des utilisateurs. Les valeurs de similarité entre 0.8 et 1 corrèlent avec celles des jugements des utilisateurs et correspondent généralement aux mots synonymes ou frères proches.

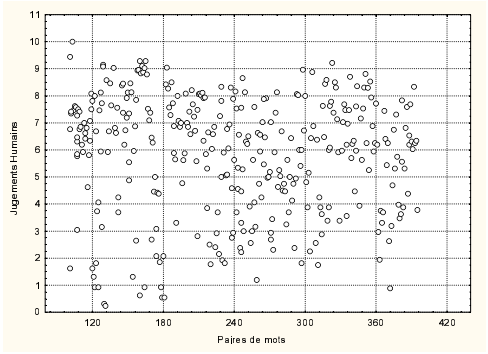

FIG. 8.6 – Répartition des jugements humains par paires de mots

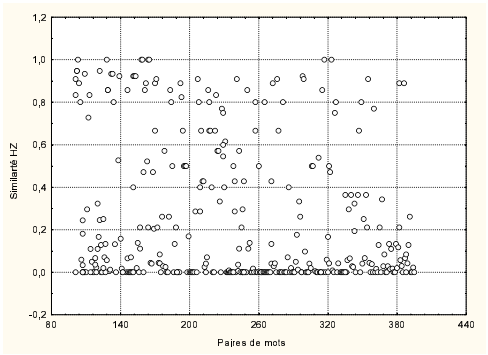

FIG. 8.7 – Répartition de la Similarité HZ par paires de mots

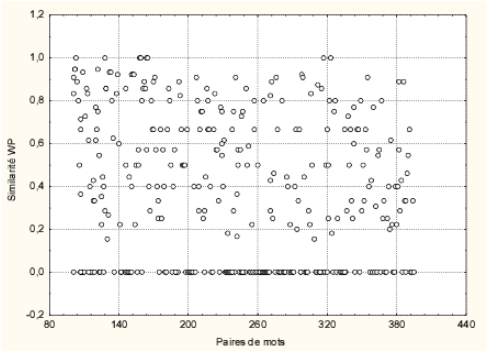

FIG. 8.8 – Répartition de la similarité WP par paires de mots

8.6 Evaluation de la prise en compte de la sémantique

Notre prototype $SemIndex$ stocke les différents index en les sérialisant. Ce choix nous a limité dans le choix des corpus à évaluer. En effet, les corpus volumineux requiérent plus d'espace mémoire. Nous améliorons notre prototype pour pouvoir stocker les index dans des bases de données ce qui nous permettra de créer de gros index et faire des expérimentations sur la structure sur le corpus INEX.
Nous avons choisi le corpus de test CACM qui est accompagné de 64 requêtes avec des jugements de pertinence portant sur 52 d'entre elles. Le corpus présente une liste de documents commençant par ".I". Les documents peuvent contenir les champs : titre indiqué par ".T" ; mots-clés indiqué par ".K" ; résumé indiqué par ".W" ; auteur indiqué par ".A", références croisées ".X", etc. Nous avons structuré les documents en fonction de ces champs : .T –> <Title>, .W –> <Abstract>, etc. Voici un exemple de document *XMLisé* :

```
<I id="1410">
 <Title>Interarrival Statistics for Time Sharing Systems</Title>
 <Abstract>The optimization of time-shared system performance
 requires the description of the stochastic processes
 governing the user inputs and the program activity.
 This paper provides a statistical description of the
 user input process in the SDC-ARPA general-purpose
 Time-Sharing System (TSS).
 The input process is assumed to be stationary, and to be
 defined by the interarrival time distribution. The data
 obtained appear to justify satisfactorily the common
 assumption that the interarrival times are serially independent.
 The data do not appear to justify, except as a very rough
 approximation, the usual assumption off an exponential distribution
 for interarrival time.
 A much more satisfactory approximation to the data can
 be obtained with a biphase
 or triphase hyperexponential distribution.</Abtract>
 <Author>Coffman, E. G. Wood, R. C.</Author>
..
 </I>
```

Le corpus CACM nous a intéressé essentiellement pour sa petite taille (3204 documents), nous voulons tester l'apport de la sémantique dans un premier temps sur des contextes avec un vocabulaire limité.
Nous avons lancé trois tests :

- *Test1* : Cas d'une indexation sans sémantique. Nous calculons seulement le $TF - CF$ et le $TF - ICDF$.
- *Test2* : Cas d'une indexation avec désambiguïsation sémantique. Les termes des requêtes qui n'existent pas dans le corpus sont aussi désambiguïsés.
- *Test3* : Cas d'une indexation avec désambiguïsation sémantique. Les termes des requêtes ne sont pas désambiguïsés.

Nous avons indexé les contextes *Author*, *Abtract*, *KeyWord* et *Title*. Nous présentons dans le tableau 8.6, le nombre de termes indexés (NI) par contexte ainsi que le nombre de termes ayant un sens dans WordNet (NS), le nombre de termes ambigus (NA) et le nombre de termes désambiguïsés (ND)[133].

Contexte	NI	NS	NA	ND
Abstract	5964	3220 56%	2293 40.27%	3027 53.16%
KeyWord	2334	1410 60.4%	1088 46.61%	1263 54.11%
Title	3202	1841 57.49%	1352 42.22%	1374 42.91%

TAB. 8.2 – Statistiques sur le corpus CACM

Les requêtes sont aussi constitué de champs (.I début de la requête, .W contenu de la requête et .N l'auteur). Voici un extrait de la requête 3 :

```
.I 3
.W
 Intermediate languages used in construction of multi-targeted
 compilers; TCOLL
.N
 3. Donna Bergmark, Comp Serv, Uris Hall (intermed lang)
```

Nous extrayons le contenu du champ .W et le délimitons par la balise <*ANY*>.

Les résultats présentés dans la figure 8.9 ne nous permettent pas de juger de manière catégorique de la prédominance d'une méthode sur une autre. Les valeurs de précision sont assez basses par rapport à ce qui a été rapporté dans les travaux classiques sur ce même corpus (indexation par le TF-IDF par exemple). Cela s'explique par la structuration des documents que nous avons effectuée, les contextes *Abstract*, *Title*, etc sont considérés comme indépendants et peuvent être retournés séparément. Si un document est retourné plusieurs fois pour une même requête, nous retenons la valeur maximale. Les jugements de pertinence portent sur le document dans sa globalité et ont tendance à défavoriser les fragments de documents.

L'indexation sémantique avec désambiguïsation des termes de la requête montrent de légères améliorations pour de valeurs de rappel faibles et des résultats comparables aux autres méthodes pour des valeurs de rappel fort (à partir de 0.6). L'analyse des valeurs de précision dans les zones médianes de rappel montrent que la désambiguïsation des termes de la requête, en d'autres termes la prise en compte des termes qui n'existent pas dans le vocabulaire donne de meilleurs résultats que les tests sans désambiguïsation (Test2).

[133] Rappelons que les termes peuvent ne pas se voir attribuer un sens même s'il en existe dans l'ontologie

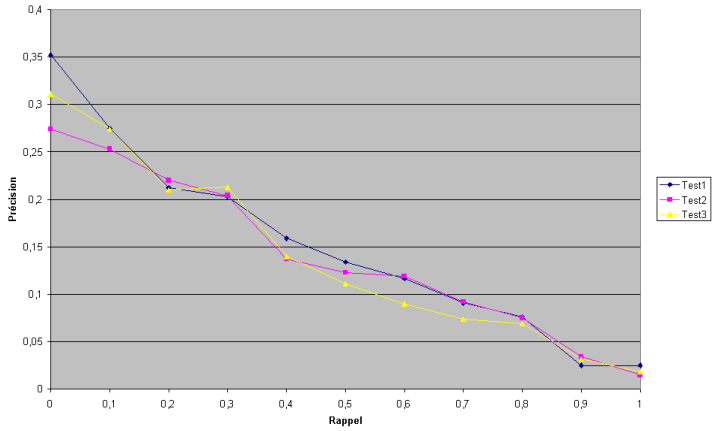

FIG. 8.9 – Courbes de Rappel/Précision pour les trois tests

Ces résultats appellent plusieurs commentaires :

– Nous avons remarqué que les jugements de pertinence portent essentiellement sur les documents qui contiennent exactement les mots-clés recherchés. Les documents ramenés par notre système qui contiennent des terms similaires sont jugés non pertinents alors qu'ils nous ont paru intéressants.
– Nous n'avons pas fixé un seuil pour l'enrichissement sémantique, ce qui peut amener à retourner des documents qui comportent des termes assez éloignés de ceux recherchés.

Beaucoup d'éléments restent à évaluer, notamment l'efficacité de notre algorithme de désambiguïsation. En effet, si les sens fixés aux termes sont erronés, les résultats des requêtes ne correspondent pas adéquat au sens recherché par les termes de la requête.

Le fait d'enrichir les termes de la requête par des termes qui sont éloignés sémantiquement (sans fixer le seuil) augmente le bruit. Ceci nous permet de penser que les résultats actuels sont néanmoins encourageants car ils sont légèrement meilleurs ou équivalents aux résultats des tests sans sémantique. Ils peuvent être améliorés et être plus probants si nous déterminons la valeur optimale du seuil de similarité sémantique.

La prise en compte de concepts complexes peut s'avérer aussi une piste intéressante comme prouvé par Baziz *et al.* [2005], les concepts longs sont souvent moins ambigus. Par exemple le concept *operating system* possède un seul sens dans WordNet, alors que *operating (adj)* possède deux sens et system en possède neuf.

Les valeurs précision du système en fonction des valeurs de rappel doivent être estimés en fonction du but du système de recherche d'information. En effet, un système qui a une bonne précision pour un rappel élevé est préféré dans le cadre d'un domaine de spécialité (le domaine médical ou juridique par exemple). Pour un domaine général comme c'est le cas pour le Web (où ce qui compte est d'avoir les documents pertinents parmi les premiers retournés), une bonne précision pour une faible valeur de rappel est en général à favoriser.

8.7 Conclusion

Dans ce chapitre nous avons présenté le prototype que nous avons implémenté et qui nous a servi pour les expérimentations. Nous avons discuter de la limite des corpus disponibles sous format XML et l'absence de ressources sémantiques accessibles. Nous avons utilisé WordNet dans le cadre de nos expérimentations. WordNet possède l'avantage d'être une ressource assez fournie, et de disposer d'APIs qui facilitent la navigation entre les synsets. Notre système reste néanmoins générique et peut prendre en compte n'importe quelle ontologie décrite en OWL.
Nos expérimentations ont porté sur la mesure de similarité sémantique proposée dans ce mémoire. Notre mesure a montré des résultats équivalents à la meilleure mesure sans nécessiter une analyse distributionnelle de corpus.
Les expérimentations portant sur l'apport de la sémantique présentent des résultats encourageants. Ils peuvent être améliorés, notamment dans le cadre d'expérimentations sur des corpus de spécialité.

Chapitre 9

Conclusion et Perspectives

Sommaire

9.1 Synthèse

Les travaux présentés dans ce mémoire se situent dans le contexte général de la recherche d'information et de l'ingénierie des connaissances. Nous nous positionnons plus particulièrement dans le cadre d'une *recherche d'information structurée* profitant du cadre du *web sémantique*.
De nombreux systèmes d'accès à l'information pourraient être améliorés s'ils avaient à disposition des ressources sémantiques adaptées adaptées au contenu textuel des documents. C'est le pari du Web Sémantique, nous avons décrit dans les chapitres 3 et 4, les fondements de cette vision ainsi que les standards émergeant. Le Web sémantique repose sur la capacité de XML à définir des balises "personnalisables". Il représente un ensemble d'efforts qui vise une transposition des outils de l'ingénierie des connaissances au problème de grosses masses de données hétérogènes et distribués.
Les documents XML, par la définition explicite de leur structure, offrent la possibilité de répondre de la manière la plus spécifique et exhaustive possible au besoin en information de l'utilisateur. Il est désormais possible de traiter les documents à un degré de granularité plus fin, le document ne constitue plus une unité atomique mais un ensemble de fragments cohérents qu'il est possible de retourner à l'utilisateur.
Les ontologies jouent un rôle central pour l'amélioration de la "compréhension des textes" par les machines. Nos propositions permettent l'intégration de l'ontologie dans le processus d'indexation et d'interrogation des documents. Nous défendons la thèse que le système ne doit pas rester tributaire de cette ressource. En effet, beaucoup de problèmes doivent d'abord être résolus :

– D'où vient l'ontologie ? Cette question renvoie au fait à beaucoup d'autres problèmes : qui sont les experts ? partagent-ils le même point de vue que les utilisateurs ?

– Quelle est le degré de couverture de l'ontologie ? Ce qui pose la question de l'adéquation de l'ontologie à un corpus. Comment choisir entre plusieurs ontologies ?

L'ontologie nous sert à enrichir la pondération des termes par ses similarités avec les termes qui apparaissent dans un même contexte. Les termes qui ne sont pas reliés à l'ontologie peuvent être retrouvés grâce à leur pondération.

Nous avons proposé une mesure de similarité sémantique qui nous permet d'évaluer la similarité entre deux concepts de l'ontologie. Cette mesure prend en considération la spécificité et la spécialisation des concepts et permet de faire les calculs dans n'importe quelle structure taxonomique. Cette mesure a montré de bons résultats dans le cadre d'une évaluation par rapport à des jugements humains. Nous avons proposé, dans le cadre d'une ontologie formelle, l'utilisation d'un algorithme de calcul du Plus Petit Généralisant (PPG) qui permet de prendre en considération la richesse des descriptions des concepts.

Nous avons proposé un algorithme de désambiguïsation qui fixe un sens aux termes indexés en fonction de leurs similarités avec les termes apparaissant dans le même contexte. La définition même de ce contexte est importante pour fixer le champs sémantique des termes.

Nous nous sommes intéressés dans ce mémoire à proposer des méthodes qui mettent à profit la structure et le contenu sémantique des documents. Le modèle que nous proposons repose sur :

– un modèle générique qui permet d'indexer des documents qui possèdent des structures hétérogènes et qui permet de retrouver et d'apparier ces structures.
– un langage de requête qui à la différence des langages de requêtes qui existent est plus intuitif et repose sur une syntaxe XML. Notre langage permet de poser des requêtes vagues sur la structure et sur le contenu ainsi que des requêtes booléennes. Il permet aussi de poser des requêtes par simple mots-clés, le système décide alors de l'élément à retourner. L'utilisateur n'est pas tenu d'avoir une connaissance précise de la structure. La prise en compte de la sémantique est totalement transparente à l'utilisateur.

9.2 Perspectives

Les perspectives de notre travail peuvent être classés en perspectives interne (concernant le système proposé et réalisables à court terme) et des perspectives externes (concernant les débouchés de notre travail et les travaux futurs).

Les perspectives internes concernent essentiellement l'évaluation de notre système d'indexation. Les expérimentations menées sur un petit corpus ne sont pas concluantes et sont limités du fait de la nature même du corpus testé. Nous participons à la mise en place d'une plateforme sur les guides de bonne pratique qui nous permettra de tester notre approche sur un corpus de spécialité ainsi que d'avoir accès aux avis des experts. Nos propositions sont multiples concernant la sémantique et chaque proposition doit être éprouvée :

– L'apport de la mesure de similarité dans le cadre de recherche d'information. Le seuil de la prise en compte de la similarité doit être fixé empiriquement, un seuil trop faible risque de trop élargir la recherche et un seuil trop élevé risque d'ignorer des documents proches.
– L'apport de la méthode de désambiguïsation qui a une influence directe sur l'enrichissement sémantique des termes. Les corpus de spécialité offrent l'avantage de contenir moins de termes ambigus que les corpus généraux.

Pour évaluer nos propositions concernant la prise en compte de la structure, nous avons commencé l'intégration des bases de données pour stocker nos index. Les bases de données nous permettront d'évaluer notre système dans le cadre d'INEX afin de :
– fixer le seuil des contextes à retourner lors de la phase de préselection.
– tester l'apport de nos propositions de pondération.
– tester l'efficacité de notre mesure de similarité structurelle.
– tester la pertinence de notre stratégie d'agrégation des résultats.
La participation à la tâche *Corpus hétérogènes* nous permettra aussi de tester l'apport de la sémantique de la structure.

Notre système peut être appliqué au Raisonnement à Partir de Textes Textuel (RàPCT) où les cas peuvent être semi-structurés [Zargayouna & Salotti, 2004b]. Bellia [2004] applique dans son travail de DEA nos propositions (notre mesure de similarité ainsi que les calculs de poids des termes) à un corpus de demandes d'intervention dans le domaine des ambiances intérieures. Les demandes sont de la forme suivante :

```
<Effets id="StageFinal_Instance_1">
   <Antecedent>notre enfant de 4
ans est asthmatique </Antecedent>
   <Gene>nous avons un problème
d'odeur d'une chaudière a gaz </Gene> </Effets>
<Environnement id="StageFinal_Instance_2">
<AirExterieur>lors de
l'aération il n'y a aucune odeur </AirExterieur>
  <Appareil>la chaudière a gaz, cheminée installées dans la cour,
  ventouse de chaudière a gaz installée au norme </Appareil>
  <Habitat> Notre appartement haussmannien a 3 fenêtres sur une
  cour </Habitat>
  <Voisinage>cour de environs 20 m2</Voisinage>
</Environnement>
<Personne id="StageFinal_Instance_3">
<Locataire>Nous sommes un couple, nous avons un enfant de 3 ans
</Locataire></Personne>
```

La similarité entre documents permet d'extraire dans une base de cas des cas similaires à différents degrés au cas cible dans une base de cas source.

Nous nous intéressons également à la structuration sémantique de documents. Notre système permet de dresser une cartographie d'un ou de plusieurs documents. Ces cartographies peuvent servir à détecter les variations thématiques au sein d'un document et aider à la structuration des documents.

Un minimum de standardisation est certes nécessaires pour que le Web sémantique puisse voir le jour. L'emergence de travaux qui mettent à profit ces standards et démontrent l'intérêt de les adopter jouera sans doute un rôle important à la mise en place et l'expansion du Web Sémantique. Les utilisateurs du monde professionnel ou même les particuliers doivent être convaincus que la structuration "sémantique" de leurs documents et leur rattachement à une (ou des ontologies) peuvent leur apporter une plus value certaine pour investir de leurs temps pour aider à la création d'un web de "sens" à la place du web "de liens" actuel.

C'est dans l'optique de proposer un système de recherche d'information sur le chemin entre le web d'aujourd'hui et celui de demain qu'a été centré nos propositions. Nous n'avons pas valider une par une toutes nos propositions. Les corpus XML restent rares et les ressources sémantiques associées encore plus. Un des objectifs de cette thèse est d'avoir à notre disposition un outils de recherche d'information générique et modulaire qui nous facilitera dans le futur l'intégration de nouvelles idées (analyse linguistique plus fine des corpus, intégration de termes composés, de concepts complexes, etc.).

Annexe A

WordNet

WordNet est un dictionnaire électronique de l'anglo-américain, développé depuis 1985 et initialement conçu pour tester les déficits lexicaux dans des expériences de psychologie cognitive. Il a été développé à Princeton et continue à être mis à jour. Sa structure est celle d'un thésaurus, il est organisé autour de la structure des synsets, c'est-à-dire des ensembles de synonymes et de pointeurs décrivant des relations vers d'autres synsets. Chaque mot peut appartenir à un ou plusieurs synsets, et à une ou plusieurs catégorie du discours suivantes : nom, verbe, adjectif, adverbe. Nous ne nous intéressons dans ce rapport qu'aux noms de WordNet, pour des raisons de simplicité. Notons que la base de connaissance de WordNet est composée de plusieurs graphes de concepts : en effet, il y plusieurs racines, ou concepts les plus généraux, et donc plusieurs graphes, censés se couper le moins possible. Les racines sont :

– entity ;
– location ;
– abstraction ;
– event ;
– group ;
– phenomenon ;
– psychological_feature ;
– shape ;
– state ;
– act ;
– possession.

Chacun des graphes dont nous venons de préciser la racine a un ensemble de concepts différent : de 43950 concepts pour entity a 688 pour shape.

Annexe B

Schéma XML du langage de requêtes SemIR

```
 <?xml version="1.0" encoding="UTF-8" ?> - <xs:schema
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:ad="urn:ourLanguage" targetNamespace="urn:ourLanguage"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
elementFormDefault="qualified">
  <xs:element name="query" type="ad:tquery" minOccurs="1" />
- <xs:complexType name="tquery"> - <xs:complexContent> -
<xs:extension> - <xs:choice minOccurs="1" maxOccurs="unbounded">
  <xs:element name="OR" type="ad:tOR" minOccurs="0" maxOccurs="unbounded
  <xs:element name="AND" type="ad:tAND" minOccurs="0" maxOccurs="unbound
  <xs:element name="NOT" type="ad:tquery" minOccurs="0" maxOccurs="unbou
  <xs:element name="ANY" type="ad:tANY" minOccurs="0" maxOccurs="unbound
  <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##other" proces
  </xs:choice>
  </xs:extension>
  </xs:complexContent>
  </xs:complexType>
- <xs:complexType name="tOR"> - <xs:complexContent> -
<xs:extension> - <xs:sequence> - <xs:choice minOccurs="1"
maxOccurs="1">
  <xs:element name="OR" type="ad:tOR" minOccurs="1" maxOccurs="1" />
  <xs:element name="AND" type="ad:tAND" minOccurs="1" maxOccurs="1" />
  <xs:element name="NOT" type="ad:tNOT" minOccurs="1" maxOccurs="1" />
  <xs:element name="ANY" type="ad:tANY" minOccurs="1" maxOccurs="1" />
  <xs:any minOccurs="1" maxOccurs="1" namespace="##other" processContent
  </xs:choice>
- <xs:choice minOccurs="1" maxOccurs="1">
  <xs:element name="OR" type="ad:tOR" minOccurs="1" maxOccurs="1" />
  <xs:element name="AND" type="ad:tAND" minOccurs="1" maxOccurs="1" />
  <xs:element name="NOT" type="ad:tNOT" minOccurs="1" maxOccurs="1" />
```

```
  <xs:element name="ANY" type="ad:tANY" minOccurs="1" maxOccurs="1" />
  <xs:any minOccurs="1" maxOccurs="1" namespace="##other" processContents="
  </xs:choice>
  </xs:sequence>
  </xs:extension>
  </xs:complexContent>
  </xs:complexType>
- <xs:complexType name="tAND"> - <xs:complexContent> -
<xs:extension> - <xs:sequence> - <xs:choice minOccurs="1"
maxOccurs="1">
  <xs:element name="OR" type="ad:tOR" minOccurs="1"
  maxOccurs="1" />
  <xs:element name="AND" type="ad:tAND" minOccurs="1"
  maxOccurs="1" />
  <xs:element name="NOT" type="ad:tNOT" minOccurs="1"
  maxOccurs="1" />
  <xs:element name="ANY" type="ad:tANY" minOccurs="1"
  maxOccurs="1" />
  <xs:any minOccurs="1" maxOccurs="1" namespace="##other"
  processContents="lax" />
  </xs:choice>
- <xs:choice minOccurs="1" maxOccurs="1">
  <xs:element name="OR" type="ad:tOR" minOccurs="1"
  maxOccurs="1" />
  <xs:element name="AND" type="ad:tAND" minOccurs="1"
  maxOccurs="1" />
  <xs:element name="NOT" type="ad:tNOT" minOccurs="1"
  maxOccurs="1" />
  <xs:element name="ANY" type="ad:tANY" minOccurs="1"
  maxOccurs="1" />
  <xs:any minOccurs="1" maxOccurs="1" namespace="##other"
  processContents="lax" />
  </xs:choice>
  </xs:sequence>
  </xs:extension>
  </xs:complexContent>
  </xs:complexType>
- <xs:complexType name="tNOT"> - <xs:complexContent> -
<xs:extension> - <xs:choice minOccurs="1" maxOccurs="1">
  <xs:element name="OR" type="ad:tOR" minOccurs="1"
  maxOccurs="1" />
  <xs:element name="AND" type="ad:tAND" minOccurs="1"
  maxOccurs="1" />
  <xs:element name="NOT" type="ad:tNOT" minOccurs="1"
  maxOccurs="1" />
```

```
  <xs:element name="ANY" type="ad:tANY" minOccurs="1"
  maxOccurs="1" />
  <xs:any minOccurs="1" maxOccurs="1" namespace="##other"
  processContents="lax" />
  </xs:choice>
  </xs:extension>
  </xs:complexContent>
  </xs:complexType>
- <xs:complexType name="tANY"> - <xs:complexContent> -
<xs:extension> - <xs:choice minOccurs="1" maxOccurs="1">
  <xs:element name="OR" type="ad:tOR" minOccurs="1"
  maxOccurs="1" />
  <xs:element name="AND" type="ad:tAND" minOccurs="1"
  maxOccurs="1" />
  <xs:element name="NOT" type="ad:tNOT" minOccurs="1"
  maxOccurs="1" />
  <xs:element name="ANY" type="ad:tANY" minOccurs="1"
  maxOccurs="1" />
  <xs:any minOccurs="1" maxOccurs="1" namespace="##other"
  processContents="lax" />
  </xs:choice>
  </xs:extension>
  </xs:complexContent>
  </xs:complexType>
  </xs:schema>
```

Bibliographie

[2004, 2004] W3C Recommendation February 2004. Owl web ontology language. W3C Recommendation, February 2004. 93

[Abiteboul *et al.*, 2004] S. Abiteboul, I. Manolescu, B. Nguyen, et N. Preda. A test platform for the inex hetregeneous track. In INEX INEX [2004]. 58

[Agirre & Rigau, 1996] E. Agirre et G. Rigau. Word sense disambiguation using conceptual density. In *Proceedings of COLING'96*, pages 16–22, 1996. 111

[Ait-El-Mekki, 2004] T. Ait-El-Mekki. *Construction semi-automatique d'index de fin de livre*. PhD thesis, LIPN, Université Paris 13, France, december 2004. 31

[Alhulou *et al.*, 2003] R. Alhulou, A. Napoli, et E.Nauer. Une mesure de similarité pour raisonner sur des documents. In *Journées Nationales sur les Modèles de Raisonnement*, pages 27–28, Paris, novembre 2003. 82, 99

[Andreasen *et al.*, 2003] T. Andreasen, H. Bulskov, et R. Knappe. On ontology-based querying. In *18th International Joint Conference on Artificial Intelligence, Ontologies and Distributed Systems, IJCAI*, 2003. 83

[April, 2005a] W3C Working Draft 04 April. Xml path language (xpath) 2.0, 2005. 38

[April, 2005b] W3C Working Draft 04 April. Xquery 1.0 : An xml query language, 2005. 49, 50

[Aussenac-Gilles *et al.*, 2003] N. Aussenac-Gilles, B. Biébow, et S. Szulman. D'une méthode à un guide pratique de modélisation des connaissances à partir des textes. In *5ième Journées Terminologie et Intelligence Artificielle*, pages 41–53, 2003. 68

[Azevedo *et al.*, 2004] M.I.M. Azevedo, L.P. Amorim, et N. Zivian. A universal model for xml information retrieval. In INEX INEX [2004]. 43

[Bach *et al.*, 2004] T.L Bach, Rose Dieng-Kuntz, et Fabien Gandon. On ontology matching problems (for building a corporate semantic web in a multi-communities organization). In *Proceedings of ICEIS'04*, 2004. 125

[Bachimont, 2000] B. Bachimont. *Engagement sémantique et engagement ontologique : conception et réalisation d'ontologies en ingénierie des connaissances*. Eyrolles, 2000. xv, 68, 69, 70

[Baeza-Yates & Ribeiro-Neto, 1999] R. Baeza-Yates et B. Ribeiro-Neto. *Modern information retrieval.* ACM press, 1999. 10

[Barry *et al.*, 2001] C. Barry, C. Cormier, G. Kassel, et J. Nobécourt. Évaluation de langages opérationnels de représentation d'ontologies. In Jean Charlet, editor, *journées ingénierie des connaissances*, pages 309–325, Grenoble, 2001. 81

[Baziz *et al.*, 2003] M. Baziz, N. Aussenac-Gilles, et M. Boughanem. Exploitation des liens sémantiques pour l'expansion de requêtes dans un système de recherche d'information. In *INFORSID*, 2003. 83

[Baziz *et al.*, 2005] M. Baziz, M. Boughanem, N. Aussenac-Gilles, et C. Chrisment. Semantic cores for representing documents in ir. In ACM Press, editor, *SAC'2005- 20th ACM Symposium on Applied Computing*, pages 1020–1026, 2005. 185

[Beigbeder & Mercier, 2003] M. Beigbeder et A. Mercier. Étude des distributions de tf et de idf sur une collection de 5 millions de pages html. In *actes de l'atelier « Recherche d'information ; un nouveau passage à l'échelle », associé à Inforsid*, 2003. 18

[Bellia, 2004] Z. Bellia. Modélisation d'un système informatique pour la gestion des demandes d'intervention dans le domaine des ambiances intérieures : Une approche basée sur le raisonnement à partir des cas - ràpc. Master's thesis, Université Paris VI, 2004. 189

[Berners-Lee *et al.*, 2001] T. Berners-Lee, J. Hendler, et O. Lassila. The semantic web : A new form of web content that is meaningful to computers will unleash a revolution of new possibilities. *Scientific American*, pages 35–43, 2001. 2, 87

[Berners-Lee, 1999] T. Berners-Lee. *Weaving the Web*. Harper San Francisco, 1999. 5, 87

[Besançon *et al.*, 2003] R. Besançon, M. Rajman, et J-C. Chappelier. The form of information in science. analysis of an immunology sublanguage. *Assistance intelligente à la recherche d'informations*, 2003. 25, 26

[Besançon, 2002] R. Besançon. *Intégration de connaissances syntaxiques et sémantiques dans les représentations vectorielles de textes. Application au calcul de similarités sémantiques dans le cadre du modèle DSIR.* PhD thesis, Ecole polytechnique Fédérale de Lausanne, 2002. 25, 26, 84

[Biezunski & Newcomb, 2001] M. Biezunski et S.R. Newcomb. Making semantics addressable – topic maps unclothed. In *Semantic Web Working Symposium*, 2001. 92

[Borgida & Patel-Schneider, 1994] A. Borgida et P.F. Patel-Schneider. A semantic and complete algorithm for subsumption in the classic description logic. *Journal of Artificial Intelligence Research*, 1 :278–308, 1994. 119, 120

[Borgida & P.F.Patel-Schneider, 1994] A. Borgida et P.F.Patel-Schneider. A semantics and complete algorithm for subsumption in the classic description logic. *Artificial Intelligence Research*, 1 :278–308, 1994. 73

[Boughanem, 1992] M. Boughanem. *Recherche d'information : d'un modèle classique à un modèle connexionniste.* PhD thesis, Thèse de doctorat de l'Université Paul Sabatier de Toulouse, 1992. 83

[Bourigault, 1994] D. Bourigault. *LEXTER, un Logiciel d'EXtraction de TERminologie. Application à l'acquisition des connaissances à partir de textes*. PhD thesis, Ecole des Hautes Etudes en Sciences Sociales, Paris, 1994. 68

[Brachman *et al.*, 1991] R. J. Brachman, D. L. MCGuiness, P. F. Patel-Schneider, L. Aperien-Resnick, et A. Borgida. *Principles of Semantic Networks*, chapter Living with CLASSIC : When and how to use a KL-ONE like langage, pages 401–456. Number 14. Morgan Kaufmann, 1991. 75

[Brill, 1992] E. Brill. A simple rule-based part of speech tagger. In *Proceedings of the Third Conference on Applied Natural Language Processing*, 1992. 16

[Buckley *et al.*, 1994] C. Buckley, G. Salton, J. Allan, et A. Singhal. Automatic query expansion using smart. In *Overview of the Third Text REtrieval Conference*, 1994. 83

[Budanitsky & Hirst, 2001] A. Budanitsky et G. Hirst. Semantic distance in wordnet : An experimental, application-oriented evaluation of five measures. In *NAACL Workshop on WordNet and Other Lexical Resources*, 2001. 116

[Carmel *et al.*, 2002] D. Carmel, D. Efraty, G. Landau, Y. Maarek, et Y. Mass. An extension of the vector space model for querying xml documents via xml fragments in xml and information. In *Retrieval workshop of SIGIR*, 2002. 41, 42

[Carré *et al.*, 1995] B. Carré, R. Ducournau, J. Euzenat, A. Napoli, et F. Rechenmann. Classification et objets : programmation ou représentation ? In *cinquièmes journées nationales PRC GDR Intelligence Artificielle*, pages 213–237, Nancy, France, 1995. Teknea, Toulouse. 79

[Charlet *et al.*, 2003] J. Charlet, P. Laublet, et C. Reynaud. Ontologies pour le web sémantique. Chap. 3. Rapport de l'action spécifique « Web sémantique ». CNRS., 2003. 67, 89, 118

[Chein & Mugnier, 1992] M. Chein et M. L Mugnier. Conceptual graphs : Fundamental notions. *Revue d'Intelligence Artificielle*, 6-4 :365–406, 1992. 76

[Chouvet *et al.*, 1996] M. P. Chouvet, F. Le Ber, J. Lieber, L. Mangelinck, A. Napoli, et A. Simon. Analyse des besoins en représentation et raisonnement dans une représentation à objets l'exemple de y3. In Y. Dennebouy, editor, *Langages et Modèles à Objets*, pages 150–169, Leysin, Suisse, 1996. 79

[Church & Hanks, 1989] K. W. Church et P. Hanks. Word association norms, mutual information, and lexicography. In *Proceedings of the 27th. Annual Meeting of the Association for Computational Linguistics*, pages 76–83, Vancouver, B.C., 1989. Association for Computational Linguistics. 25

[Consortium, 1999] World Wide Web Consortium. Namespaces in xml, 1999. 36

[Constant, 1995] P. Constant. L'analyseur linguistique sylex. 5ième école d'été du CNET, 1995. 70

[Corby *et al.*, 2004] O. Corby, R. Dieng, et C. Faron-Zucker. Querying the semantic web with corese search engine. *PAIS, ECAI 2004*, 2004. 95

[Crouch *et al.*, 2002] C.J. Crouch, S. Apte, et H. Bapat. Using the extended vector model for xml retrieval. In INEX INEX [2002]. 44

[Crouch *et al.*, 2003] C.J. Crouch, S. Apte, et H. Bapat. An approach to structured retrieval based on the extended vector model. In INEX INEX [2003]. 44, 45

[Crouch *et al.*, 2004] C.J. Crouch, A. Mahajan, et A. Bellamkonda. Flexible xml retrieval based on the extended vector model. In INEX INEX [2004]. 44, 45

[D. Steffen, 2003] B. Sacaleanuand P. Buitelaar. D. Steffen. Domain specific sense disambiguation with unsupervised methods. In *GermaNet-Workshops des GLDV-AK Lexikografie*, 2003. 82

[Das *et al.*, 2001] A. Das, W. Wu, et D.L. McGuinness. Industrial strength ontology management. In *Proceedings of the International Semantic Web Working Symposium*, 2001. 97

[de Chalendar & Grau, 2000] G. de Chalendar et B. Grau. Svetlan' ou comment classer des noms en fonction de leur contexte. In *TALN*, 2000. 71, 177

[de Chalendar & Grau, 2002] G. de Chalendar et B. Grau. Structuration de domaines sémantiques. In *13ème journées francophones d'Ingéniérie des connaissances, IC'2002*, 2002. 72

[Deerwester *et al.*, 1990] S. Deerwester, S.T. Dumais, G.W. Furnas, T. K. Landauer, et R. Hrashman. Indexing by latent semantic analysis. *Journal of th american society for information science*, 41(6) :391–407, 1990. 22, 24

[Deitel *et al.*, 2001] A. Deitel, C. Faron, et R. Dieng. Learning ontologies from rdf annotations. In *IJCAI Workshop in Ontology Learning*, Seattle, 2001. 67

[Desmontils & Jacquin, 2001] E. Desmontils et C. Jacquin. Des ontologies pour indexer un site web. In *Actes des journées francophones d'Ingénierie des Connaissances (IC'2001)*, 2001. 112

[Desmontils & Jacquin, 2002] E. Desmontils et C. Jacquin. Annotation sur le web : notes de lecture. Journées de l'AS-CNRS Web sémantique, 2002. 89

[Despres & Szulman, 2005] S. Despres et S. Szulman. Construction d'ontologies à partir de textes, alignement et fusion d'ontologies, droit communautaire. In *IC 2005*, 2005. 99

[Dill *et al.*, 2003] S. Dill, N. Eiron, D. Gibson, D. Gruhl, R. Guha, A. Jhingran, T. Kanungo, S. Rajagopalan, A. Tomkins, J. A. Tomlin, et Jason Y. Zien. Semtag and seeker : Bootstrapping the semantic web via automated semantic annotation. In *proceedings of WWW2003*, 2003. 98

[Euzenat & Valtchev, 2004] J. Euzenat et P. Valtchev. Similarity-based ontology alignment in owl-lite. In *Proceedings 16th european conference on artificial intelligence (ECAI)*, 2004. 99, 125

[Euzenat, 1999] J. Euzenat. *Représentations de connaissance : de l'approximation à la confrontation*. Habilitation à diriger des recherches, Université Joseph Fourier, Grenoble, France, 1999. 79, 80

[F. Donini, 1995] D. Nardi et W. Nutt F. Donini, M. Lenzirini. The complexity of concept languages. research report RR-95-07, Deutsches Forschungszentrum fur kunstlische Intelligenz, Kaiserlautern, 1995. 74

[F. Rastier & Abeillé, 1994] M. Cavazza F. Rastier et A. Abeillé. *Sémantique pour l'analyse*. Masson, Paris, France, 1994. 69

[Faure & Nédellec, 1999] D. Faure et C. Nédellec. Knowledge acquisition of predicate argument structures from technical texts using Machine Learning : the system ASIUM. In Dieter Fensel Rudi Studer, editor, *11th European Workshop EKAW'99*, pages 329–334, Dagstuhl Castle Allemagne, Mai 1999. Springer-Verlag. 70

[Faure & Poibeau, 2000] D. Faure et T. Poibeau. First experiments of using semantic knowledge learned by asium for information extraction task using intex. In S. Staab, A. Maedche, C. Nedellec, et Wiemer-Hastings P., editors, *Ontology Learning ECAI-2000 Workshop*, pages 7–12, Berlin Allemagne, 22 Août 2000. 71

[February, 2004] W3C Recommendation 10 February. Rdf/xml syntax specification (revised), 2004. 90

[Ferret *et al.*, 2001] O. Ferret, B. Grau, M. Hurault-Plantet, G. Illouz, et C. Jacquemin. Document selection refinement based on linguistic features for qalc, a question answering system. In *RANLP*, pages 5–7, 2001. 30

[Finkelstein & E. Gabrilovich, 2002] L. Finkelstein et E. Rivlin Z. Solan G. Wolfman E. Ruppin E. Gabrilovich, Y. Matias. Placing search in context : the concept revisited. *Source ACM Transactions on Information Systems (TOIS)*, 20(1) :116 – 131, 2002. 115, 171, 180, 181

[Fox, 1983] E.A. Fox. *Extending the Boolean and Vector Space Models of information retrieval with p-norm queries and multiple concept types*. PhD thesis, Department of Computer Science, Cornell University, 1983. 40, 44

[Franconi, 2000] E. Franconi. Description logics and databases, 2000. 75

[Fuhr & Großjohann, 2004] N. Fuhr et K. Großjohann. XIRQL : An XML query language based on information retrieval concepts. *ACM Transactions on Information Systems*, 22 :313–356, 2004. 51

[G. Giraldo, 2002] C. Reynaud G. Giraldo. Construction semi-automatique d'ontologies à partir de dtds relatives à un meme domaine. *13èmes journées francophones d'Ingénierie des Connaissances*, 2002. 99

[Gangemi *et al.*, 2001] A. Gangemi, N. Guarino, et A. Oltramari. Conceptual analysis of lexical taxonomies : The case of wordnet top-level. In *Proceedings de FOIS'2001*, 2001. 118

[Gaussier *et al.*, 1997] E. Gaussier, G. Grefenstette, et M. Schulze. Traitement du langage naturel et recherche d'informations : quelques expériences sur le français. In *Premières Journées Scientifiques et Techniques du Réseau Francophone de l'Ingénierie de la Langue de l'AUPELF-UREF*, 1997. 15

[Gaussier *et al.*, 2000] E. Gaussier, G. Grefenstette, D. Hull, et C. Roux. Recherche d'information en français et traitement automatique des langues. *revue Traitement Automatique des Langues (TAL)*, 41(2) :473–494, 2000. 15

[Giunchiglia & Shvaiko, 2004] F. Giunchiglia et P. Shvaiko. Semantic matching. *The Knowledge Engineering Review Journal*, 18(3) :265–280, 2004. 125

[Gómez-Pérez & Manzano-Macho, 2003] A. Gómez-Pérez et D. Manzano-Macho. Ontoweb deliverable 1.5 a survey of ontology learning methods and techniques. deliverable1.5, 2003. 67

[Golub & Loan, 1996] G. Golub et C. Van Loan. *Matrix computations*. The Johns Hopkins University Press, Baltimore, 1996. 22

[Gonzalo *et al.*, 1998] J. Gonzalo, F. Verdejo, I. Chugur, et J. Cigarrán. Indexing with wordnet synsets can improve text retrieval. In *Proceedings of the COLING/ACL 1998 Workshop on Usage of WordNet for Natural Language Processing, Montreal*, 1998. 82

[Géry, 1997] M. Géry. Un modèle d'hyperdocument en contexte pour la recherche d'information structurée sur le web. *Revue ISI (Ingéniérie des Systèmes d'Information, numéro spécial Recherche et filtrage d'information*, 7, 1997. 34

[Guarino *et al.*, 1994] N. Guarino, M. Carrara, et P. Giaretta. Formalizing ontological commitment. *Proceedings of National Conference on Artificial Intelligence (AAAI-94)*, pages 560–567, 1994. 67

[Guarino *et al.*, 1999] N. Guarino, C. Masolo, et G. Vetere. Ontoseek : Content-based access to the web. *IEEE Intelligent Systems*, 14(3) :70–80, 1999. 83

[Guarino, 1997] N. Guarino. Understanding, building, and using ontologies : A commentary to "using explicit ontologies in kbs development". *International Journal of Human and Computer Studies*, pages 293–310, 1997. 66

[Guarino, 1998] N. Guarino. *Formal ontology in information systems*. IOS press, 1998. 64

[Gövert & Kazai, 2002] N. Gövert et G. Kazai. Overview of the initiative for the evaluation of xml retrieval (inex) 2002. In INEX INEX [2002], pages 1–17. 52, 55

[G.W. Furnas & Lochbaum, 1988] S.T. Dumais T.K. Landauer R. Hrashman L.A. Streeter G.W. Furnas, S. Deerwester et K.E. Lochbaum. Information retrieval using singular decomposition model of latent semantic structure. In *Proceedings Of the 11th Annual ACM SIGIR Conference on research and development in information retrieval*, pages 465–480, 1988. 24

[Habert *et al.*, 1997] B. Habert, A. Nazarenko, et A. Salem. *Les linguistiques de corpus*. Armand Coli, Paris, 1997. 25, 130

[Habert *et al.*, 2001] B. Habert, N. Grabar, P. Jacquemart, et P. Zweigenbaum. Building a text corpus for representing the variety of medical language. *Corpus Linguistics*, 2001. 176

[Haemmerlé, 1995] O. Haemmerlé. *CoGITo : une plateforme de développement de logiciels sur les graphes conceptuels*. PhD thesis, Université Montpellier II, 1995. 78

[Halkidi *et al.*, 2003] M. Halkidi, B. Nguyen, I. Varlamis, et M. Vazirgiannis. Thesus : Organising web document collections based on semantics and clustering. *Journal on Very Large Databases, Special Edition on the Semantic Web*, 2003. 112

[Hamon, 2000] Thierry Hamon. *Variation sémantique en corpus spécialisés : Acquisition de relations de synonymie à partir de ressources lexicales.* PhD thesis, Université Paris-Nord, Villetaneuse, France, décembre 2000. 68

[Harris *et al.*, 1989] Z. S. Harris, M. Gottfried, T. Ryckman, P.Jr Mattick, A. Daladier, T.N Harris, et S. Harris. The form of information in science. analysis of an immunology sublanguage. *Boston studies in the Philosophy of Science*, 104 :391–407, 1989. 25

[Hayashi *et al.*, 2000] Y. Hayashi, J. Tomita, et G. Kikui. Searching text-rich xml documents. In *ACM SIGIR 2000 Workshop on XML and Information Retrieval*, 2000. 47

[Hearst, 1992] M.A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Fourteenth International Conference on Computational Linguistic*, Nantes, France, July 1992. 67

[Hearst, 1997] M. Hearst. Textiling :segmenting text info multi-paragraph subtopic passages. *Computational Linguistics*, 23(1) :33–64, 1997. 34

[Heflin & Hendler, 2000] J. Heflin et J. Hendler. Searching the web with shoe. In *Artificial Intelligence for Web Search. Papers from the AAAI Workshop.*, 2000. 98, 99

[Hernandez & Aussenac-Gilles, 2004] N. Hernandez et N. Aussenac-Gilles. Ontoexplo : Ontologies pour l'aide à une activité de veille ou d'exploration d'un domaine. In Foix, editor, *VIème Journées de l'innovation*, Janvier 2004. 83, 99

[Hernandez, 2004] N. Hernandez. *Description et Détection Automatique de Structures de Texte*. PhD thesis, LIMSI, Université Paris 11, France, december 2004. 131, 177

[Hirst & St-Onge, 1998] G. Hirst et D. St-Onge. Lexical chains as representations of context for the detection and correction of malapropisms. *WordNet : An electronic lexical database*, 1998. 107, 116

[Hirst & St-Onge, 2004] G. Hirst et D. St-Onge. Evaluating wordnet-based measures of lexical semantic relatedness. 2004. 115, 181

[Horrocks, 1998] I. Horrocks. Using an expressive description logic : Fact or fiction ? In Morgan Kaufmann Publishers, editor, *Sixth International Conference Knowledge Representation :Principles of Knowledge Representation and Reasoning*, pages 636–647, San Francisco, California, June 1998. 75

[INEX, 2002] INEX, editor. *Proceedings of the first Workshop of the INitiative for the Evaluation of XML retrieval*, 2002. 35, 51, 199, 202, 205, 207

[INEX, 2003] INEX, editor. *Proceedings of the second Workshop of the INitiative for the Evaluation of XML retrieval*, 2003. 35, 51, 200, 205, 206, 207

[INEX, 2004] INEX, editor. *Proceedings of the third Workshop of the INitiative for the Evaluation of XML retrieval*, 2004. 35, 51, 197, 200, 204, 205, 209, 210, 211

[J. Kahan & Swick, 2001] E. Prud'Hommeaux J. Kahan, M.R. Koivunen et R.R. Swick. Annotea : An open rdf infrastructure for shared web annotations. In *Proc. of the WWW10 International Conference, Hong Kong*, 2001. 98

[Jacquemin & Tzoukermann, 1997] C. Jacquemin et E. Tzoukermann. Nlp for term variant extraction : Synergy between morphology, lexicon and syntax. *Naturel Language Information Retrieval*, 1997. 15

[Jacquemin & Zweigenbaum, 2000] C. Jacquemin et P. Zweigenbaum. Traitement automatique des langues pour l'accès au contenu des documents. In *Le document Multimédia en Sciences du Traitement de l'Information*, pages 71–110. CÉPADUÈS-Éditions, Toulouse.edition ACM press, 2000. 13

[Jarmasz & Szpakowicz, 2003] M. Jarmasz et S. Szpakowicz. Roget's thesaurus and semantic similarity. In *Proceedings of Conference on Recent Advances in Natural Language Processing (RANLP'03)*, pages 212–219, 2003. 105

[Jarmasz, 2003] M. Jarmasz. Roget's thesaurus as a lexical resource for natural language processing. Master's thesis, School of Information Technology and Engineering, University of Ottawa, Canada, July 2003. 181

[Jiang & Conrath, 1997] J. Jiang et D. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings on International Conference on Research in Computational Linguistics*, 1997. 109, 111, 116, 181

[Jones *et al.*, 1998] K. Spark Jones, S. Walker, et S.E. Robertson. A probabilistic model of information retrieval : development and comparative experiments. Technical Report TR446, University of Cambridge Computer Laboratory, 1998. 21

[Kayser, 1997] D. Kayser. *La représentation des connaissances*. Hermès, 1997. 64

[Kazai *et al.*, 2005] G. Kazai, M. Lalmas, et A. de Vries. Reliability tests for the xcg and inex-2002 metrics dans advances in xml information retrieval. In *Lecture Notes in Computer Science 3493*, pages 60–72, 2005. 56

[Kazai, 2003] G. Kazai. Report of the inex 2003 metrics working group. In INEX INEX [2004], pages 184–190. 55

[Kent *et al.*, 1955] A. Kent, M. Berry, F.U. Leuhrs, et J.W. Perry. Machine literature searching viii : Operational criteria for designing information retrieval systems. *American Documentation*, 6(2) :93–101, 1955. 27

[Kilpeläinen, 1992] P. Kilpeläinen. *Tree Matching Problems with Applications to Structured Text Databases*. PhD thesis, University of Helsinki, 1992. 46

[Kim & Son, 2004] H. Kim et H. Son. Kyungpook national university at inex 2004 : Interactive track. In INEX INEX [2004]. 59

[Krovetz, 1997] R. Krovetz. Homonymy and polysemy in information retrieval. In *ACL/EACL*, 1997. 82

[Leacock & Chadorow, 1998] C. Leacock et M. Chadorow. The electronic lexical database. *Combining Local context and WordNet similarity for word sense identification*, pages 265–283, 1998. 116

[Lehmam, 2005] A. Lehmam. *Le résumé automatique des textes ; aspects linguistiques et computationnels.* Editions L'Harmattan, Paris, 2005. 30

[Lenat & Guha, 1990] D.B. Lenat et R.V. Guha. *Building Large Knowledge Based Systems.* Addison Wesley., 1990. 64

[Lerman *et al.*, 2001] K. Lerman, C. Knoblock, et S. Minton. Automatic data extraction from lists and tables in web sources. In *IJCAI-2001 Workshop on Adaptive Text Extraction and Mining*, 2001. 98

[Léger *et al.*, 2002] A. Léger, Y. Bouillon, P. Ecoublet, M. Bryan, R. Dieng, A. Persidis, A. Gomez-Perez, et M. Fernández López. Successful scenarios for ontology-based applications v1.0. deliverable2.2, 2002. 95

[Li *et al.*,] J. Li, L. Zhang, et Y. Yu. 98

[Lin, 1998] D. Lin. An information-theoretic definition of similarity. In *Proceedings of 15th International Conference On Machine Learning*, 1998. 106, 109, 110, 111, 112, 115, 116, 181

[Lovins, 1968] J. B. Lovins. Development of a stemming algorithm. *Computational Linguistic*, 11 :22–31, 1968. 15

[Lu & Keefer, 1994] X. A. Lu et R. B. Keefer. Query expansion/reduction and its impact on retrieval effectiveness. *Overview of the Third Text REtrieval Conference, NIST Special Publication*, pages 231–240, 1994. 83

[MacGregor & Bates, 1987] R. M. MacGregor et R. Bates. The loom knowledge representation langage. Manual ISI/RS-87-188, Information Sciences Institute, University of Southern California, 1987. 75

[Maedche & Staab, 2001] A. Maedche et S. Staab. Ontology learning for the semantic web. *Special Issue on the Semantic Web,IEEE Intelligent Systems*, 16(2), 2001. 67, 98

[Mahajan, 2004] A. Mahajan. Flexible retrieval in a structured environment. Master's thesis, University of Minnesota, 2004. xv, 45

[March, 2003] W3C Recommendation 25 March. Xpointer framework, 2003. 38

[Mass & Mandelbrot, 2003] Y. Mass et M. Mandelbrot. Retrieving the most relevant xml components. In INEX INEX [2003]. 43

[Mass & Mandelbrot, 2004] Y. Mass et M. Mandelbrot. Relevance feedback for xml retrieval. In INEX INEX [2004]. 59

[Mass *et al.*, 2002] Y. Mass, M. Mandelbrod, E. Amitay, D. Carmel, Y. Maarek, et A. Soffer. Juruxml - an xml retrieval system. In INEX INEX [2002]. 42

[Mayfield & Finin, 2003] J. Mayfield et T. Finin. Owlir : Information retrieval on the semantic web. In *SIGIR'2003 Semantic Web Workshop*, 2003. 96, 97

[Mihalcea & Moldovan, 1999] R. Mihalcea et D. Moldovan. A method for word sense disambiguation of unrestricted text. In *37th Annual Meeting of the Association for Computational Linguistics*, 1999. 82

[Miller & Charles, 1991] G.A. Miller et W.G. Charles. Contextual correlates of semantic similaity. In *Language and cognitive Processes*, 1991. 109, 114, 115, 180

[Minski, 1975] M. Minski. A framework for representing knowledge. *The Psychology of computer vision*, 1975. 72

[Moffat *et al.*, 1993] A. Moffat, R. Sacks-Davis, R. Wilkinson, et J. Zobel. Retrieval of partial documents. In *Text REtrieval Conference*, pages 181–190, 1993. 34

[Moldovan & Mihalcea, 2000] D.I. Moldovan et R. Mihalcea. Using wordnet and lexical operators to improve internet searches. *IEEE Internet Computing*, 2000. 83

[Moreau & Sébillot, 2005] F. Moreau et P. Sébillot. Contributions des techniques du traitement automatique des langues à la recherche d'information. Rapport de Recherche IRISA, No 1690, 2005. 14

[Morris & Hirst, 1991] J. Morris et G. Hirst. Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational Linguistics*, 17(1) :21–48, 1991. 105

[Mugnier & Chein, 1996] M. L. Mugnier et M. Chein. Représenter des connaissances et raisonner avec des graphes. *Revue d'Intelligence Artificielle*, 10-1 :7–56, 1996. 76, 78

[N. Kushmerick & Doorenbos., 1997] D. S. Weld N. Kushmerick et R. B. Doorenbos. Wrapper induction for information extraction. In *Intl. Joint Conference on Artificial Intelligence (IJCAI)*, 1997. 98

[Napoli, 1997] A. Napoli. Une introduction aux logiques de descriptions. Rapport de recherche N°3314, INRIA, December 1997. 73, 75

[Nazarenko, 2005] A. Nazarenko. Sur quelle sémantique reposent les méthodes automatiques d'accès au contenu textuel ? *Sémantique et Corpus*, 2005. 31

[Nebel, 1990] B. Nebel. Reasoning and revision in hybrid representation systems. *LNAI*, 422, 1990. 74, 75

[Nobécourt, 1999] J. Nobécourt. *De la modélisation à la représentation : MDos, un exemple de langage de modélisation multiformalismes.* PhD thesis, Université Paris 13, 1999. 69

[November, 1999] W3C Recommendation 16 November. Xml path language (xpath) version 1.0, 1999. 38, 49

[October, 2001] W3C Recommendation 15 October. Extensible stylesheet language (xsl) version 1.0, 2001. 38

[O'Keefe & Trotman, 2003] R.A. O'Keefe et A. Trotman. The simplest query language that could possibly work. In INEX INEX [2003]. 52, 57, 152

[Patel-Schneider *et al.*, 2002] P. Patel-Schneider, I. Horrocks, et F. van Harmelen. Reviewing the design of daml+oil : An ontology language for the semantic web. In R. Dechter, M. Kearns, et R. Sutton, editors, *Proceedings of the Eighteenth National Conference on Artificial Intelligence*, July 2002. 93

[Patwardham, 2003] S. Patwardham. Incorporating dictionary and corpus information in a measure of semantic relatedness. Master's thesis, 2003. 115

[Pedersen *et al.*, 2004] T. Pedersen, S. Patwardhan, et J. Michelizzi. Wordnet : :similarity - measuring the relatedness of concepts. In *Proceedings of the Nineteenth National Conference on Artificial Intelligence (AAAI-2004) (Intelligent Systems Demonstration)*, pages 1024–1025, 2004. 181

[Pepper & Moore, 2001] S. Pepper et G. Moore. Xml topic maps (xtm) 1.0. TopicMaps.Org, http ://www.topicmaps.org/xtm/1.0/, 2001. 92

[Piwowarski & Gallinari, 2003] Benjamin Piwowarski et Patrick Gallinari. Structure, recherche d'information et apprentissage. In *Journées francophones d'Extraction et de Gestion des Connaissances (EGC 2003)*, 2003. 48

[Piwowarski & Lalmas, 2004] B. Piwowarski et M. Lalmas. Interface pour l'évaluation de systèmes de recherche sur des documents xml. In *CORIA*, pages 143–160, 2004. 51, 53, 54

[Piwowarski *et al.*, 2002] B. Piwowarski, G.E Faure, et P. Gallinari. Bayesian networks and inex. In INEX INEX [2002]. 48, 49

[Piwowarski, 2003a] B. Piwowarski. *Techniques d'apprentissage pour le traitement d'informations structurées : application à la recherche d'information*. PhD thesis, Université Pierre et Marie Curie, 2003. 49, 54

[Piwowarski, 2003b] B. Piwowarski. Working group report : the assessment tool. In INEX INEX [2003]. 56

[Porter, 1980] M. F. Porter. An algorithm for suffix stripping. *Program*, 14 :130–137, 1980. 15

[Quillian, 1968] M.R. Quillian. Semantic memory. *Semantic information processing*, 1968. 65

[Rada *et al.*, 1989] R. Rada, H. Mili, E. Bicknell, et M. Blettner. Development and application of a metric on semantic nets. *IEEE Transaction on Systems, Man, and Cybernetics*, 19(1) :17–30, 1989. 106, 107

[Rajman & Bonnet, 1992] M. Rajman et A. Bonnet. New tools for text analysis : Corpora-based linguistics. In *The First Conference of the Association for Global Strategic Information*, 1992. 25

[Rajman *et al.*, 2000] M. Rajman, R. Besançon, et J-C. Chappelier. Le modèle dsir : une approche à base de sémantique distributionnelle pour la recherche documentaire. *revue Traitement Automatique des Langues (TAL)*, 41(2) :549–578, 2000. 25, 84

[Resnik, 1995] P. Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, 1995. 109, 110, 111, 115, 116, 181

[Resnik, 1999] P. Resnik. Semantic similarity in a taxonomy : An information based measure and its application to problems of ambiguity in natural language. *Journal of Artificial Intelligence Research*, 11, 1999. 115

[Rigau, 1998] G. Rigau. *Automatic Acquisition of Lexical Knowledge from MRDs*. PhD thesis, Departament de Llenguatges i Sistemes Informàtics.– Universitat Politècnica de Catalunya, 1998. 67

[Robertson & Jones, 1997] S.E. Robertson et K. Spark Jones. Simple proven approches to text retrieval. Technical report, City University, Department of Information Science, 1997. 18, 21, 49, 51

[Robertson & Walker, 1994] S. E. Robertson et S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *Proceedings of SIGIR 1994*, pages 232–241, 1994. 18, 51

[Roget, 1995] Roget. Roget's ii : The new thesaurus. 3rd ed. Boston : Houghton Mifflin, 1995. 105

[Rubenstein & Goudenough, 1965] H. Rubenstein et J.B. Goudenough. Contextual correlates of synonymy. In *Communications of the ACM*, 1965. 114, 115, 180

[Rubin *et al.*, 2002] D.L. Rubin, M. Hewett, D.E. Oliver, T.E. Klein, et R.B. Altman. Automatic data acquisition into ontologies from pharmacogenetics relational data sources using declarative object definitions and xml. In *Pacific Symposium on Biology*, Lihue, HI, 2002. 67

[Salton & Buckley, 1988] G. Salton et C. Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24 :513–523, 1988. 19

[Salton & McGill, 1983] G. Salton et M. J. McGill. *Introduction to Modern Information Retrieval.* New York : McGraw-Hill, 1983. 19

[Salton *et al.*, 1975] G. Salton, C. S. Yang, et C. T. Yu. A theory of term importance in automatic text analysis. *Journal of the American Society for Information Science and Technology*, 26(1) :33–44, 1975. 19

[Salton *et al.*, 1983] G. Salton, E.A. Fox, et H. Wu. Extended boolean information retrieval. In *Communications of the ACM*, pages 1022–1036, 1983. 21

[Salton *et al.*, 1996] G. Salton, A. Singhal, C. Buckley, et M. Mitra. Automatic text decomposition using text segments and text themes. In *UK Conference on Hypertext*, pages 53–65, 1996. 34

[Salton, 1971] G. Salton. *The SMART retrieval system : experiments in automatic document processing.* Prentice-Hall, 1971. 19

[Salton, 1984] G. Salton. The use of extended boolean logic in information retrieval. In *SIGMOD 84 : Proceedings of the 1984 ACM SIGMOD international conference on Management of data*, pages 277–285, New York, NY, USA, 1984. ACM Press. 21

[Saracevic, 1995] T. Saracevic. Evaluation of evaluation in information retrieval. In *SIGIR*, pages 138–146, 1995. 28

[Sauvagnat & Boughanem, 2004a] K. Sauvagnat et M. Boughanem. Etat de l'art, la recherche d'information dans des documents xml. Rapport interne IRIT/2004-4-1-R, 2004. 35

[Sauvagnat & Boughanem, 2004b] K. Sauvagnat et M. Boughanem. Using a relevance propagation method for adhoc and heterogeneous tracks in inex 2004. In INEX INEX [2004]. 58

[Sébillot, 2002] P. Sébillot. Traitement automatique des langues et recherche d'information. La recherche d'information sur les réseaux II (cours Inria)., 2002. 14

[Schenkel *et al.*, 2005a] R. Schenkel, A. Theobald, et G. Weikum. Efficient creation and incremental maintenance of the hopi index for complex xml document collections. In *21st International Conference on Data Engineering (ICDE'05)*, 2005. 39

[Schenkel *et al.*, 2005b] R. Schenkel, A. Theobald, et G. Weikum. Semanctic similarity search on semistructed data with xxl search engine. *Information Retrieval*, 8 :521–545, 2005. 84, 86

[Schlieder & Meuss, 2002] T. Schlieder et H. Meuss. Querying and ranking xml documents. *Special Topic Issue of the Journal of the American Society for Informations Systems (JASIS) on XML and Information Retrieval*, pages 489–503, 2002. 46, 130

[S.E. Robertson & Payne, 1995] M.M. Beaulieu M. Gatford S.E. Robertson, S. Walker et A. Payne. Okapi at trec-4. In *NIST Special Publication 500-236 : The Fourth Text REtrieval Conference (TREC-4)*, pages 73–96, 1995. 18, 21, 51

[Shah *et al.*, 2003] U. Shah, T. Finin, A. Joshi, R. Cost, et J. Mayfield. Information retrieval on the semantic web. In *In 10th International Conference on Information and Knowledge Management. ACM Press, 2003.*, 2003. 96

[Simon & Napoli, 1999] A. Simon et A. Napoli. *Un algorithme de fouille dans une représentation des données par objets : une application au domaine médical. Dans J. Charlet, M. Zacklad, G. Kassel, and D. Bourigaul, editors, Ingénierie des Connaissances, évolutions récentes et nouveaux défis*. Eyrolles, 1999. 79, 81

[Simon *et al.*, 2003] L. Simon, E. Desmontils, et C. Jacquin. Utilisation de techniques d'enrichissement d'ontologie pour améliorer le processus d'indexation structurée. In *Actes des journées francophones d'Ingénierie des Connaissances (IC'2003)*, 2003. 112

[Slodzian, 1999] M. Slodzian. Wordnet et eurowordnet :questions impertinentes sur leur pertinence linguistique. *Sémiotiques, Numéro spécial : Dépasser les sens iniques dans l'accès automatisé aux textes.*, 17 :51–70, 1999. 116

[Slodzian, 2000] M. Slodzian. Wordnet :what about its linguistic relevancy ? In *Proceedings de EKAW conference*, 2000. 116

[Sowa, 1984] J. F. Sowa. *Conceptual Structures : Information Processing in Mind and Machine*. Addison-Wesley, London, 84 edition, 1984. 76, 77

[Sowa, 2000] J. F. Sowa. *Knowledge Representation : Logical, Philosophical, and Computational Foundations*. Brooks/Cole Publishing Co, London, 2000 edition, 2000. 64, 72, 76, 78

[Staab & Maedche, 2001] S. Staab et A. Maedche. Knowledge portals - ontologies at work. *AI Magazine*, 2001. 98

[Stetina *et al.*, 1998] J. Stetina, S. Kurohashi, et M. Nagao. General word sense disambiguation method based on a full sentential context. usage of wordnet in natural language processing. In *COLING-ACL Workshop*, Montreal, 1998. 82

[Sussna, 1997] M. Sussna. *Text retrieval using inference in semantic metanetworks*. PhD thesis, University of California,San Diego, 1997. 111

[Swets, 1969] J.A. Swets. ffectiveness of information retrieval methods. *American Documentation*, 20 :72– 89, 1969. 28

[Szulman & Biébow, 2004] S. Szulman et B. Biébow. Owl et terminae. In *actes de IC'2004*, pages 41–52, 2004. 68

[Szulman *et al.*, 2002] S. Szulman, B. Biebow, et N. Aussenac-Gilles. Structuration de terminologies à l'aide d'outils de tal avec terminae. *Traitement Automatique des Langues*, 43, 2002. 68

[T. Hoppe & Fischer, 1995] J. Quantz A. Schmiedel T. Hoppe, C. Kindermann et M. Fischer. Back v5 tutorial and manual. Manual Projet KIT-BACK, Universitat berlin, Institut fur Software und Theorische Informatik, Allemagne, 1995. 75

[Tannier *et al.*, 2004] X. Tannier, J.J. Girardot, et M. Mathieu. Analysing natural language queries at inex 2004. In INEX INEX [2004]. 58

[Tombros *et al.*, 2004] A. Tombros, B. Larsen, et S. Malik. The interactive track at inex 2004. In INEX INEX [2004]. 59

[Tversky, 1977] A. Tversky. Features of similarity. *Psychological Review*, 84(4) :327–352, 1977. 106

[Uschold, 2002] M. Uschold. Where are the semantics in the semantic web ? *AI Magazine*, 2002. xv, 66, 67

[V. Brunie & Morizet-Mahoudeaux, 1999] B. Bachimont V. Brunie et P. Morizet-Mahoudeaux. *Modélisation des connaissances structurelles documentaires pour la conception d'un dossier médical hypertextuel*. Eyrolles Publisher, 1999. 178

[VanRijsbergen, 1979] C. J. VanRijsbergen. *Information retrieval.* London : Butterworths, 1979. 20, 29

[Veillard, 2000] D. Veillard. Les technologies associées à xml., 2000. xv, 36

[Ventos, 1997] V. Ventos. PhD thesis, Univesrité Paris Nord., 1997. 119, 120, 121

[Web, 2003] The Semantic Web. *M.C. Daconta and L.J. Obrst and K.T. Smith*. Wiley, 2003. 64

[Weigel *et al.*, 2004a] F. Weigel, H. Meuss, K.U. Schulz, et F. Bry. Content and structure in indexing and ranking xml. In *Proceedings of the 7th International Workshop on the Web and Databases (WebDB)*, 2004. 46

[Weigel *et al.*, 2004b] F. Weigel, K.U. Schulz, et H. Meuss. Ranked retrieval of structured documents with the s-term vector space model. In INEX INEX [2004]. xv, 46

[Weigel, 2002] F. Weigel. A survey of index techniques for semistructured documents. Project Thesis, University of Munich, Institute of Computer Science, 2002. 35

[Woodley & Geva, 2004] A. Woodley et S. Geva. Nlpx at inex 2004. In INEX INEX [2004]. 57

[WSM, 2003] *Journée « Web sémantique médical »*, 2003. 100

[Wu & Palmer, 1994] Z. Wu et M. Palmer. Verb semantics and lexical selection. In *Proceedings of the 32nd Annual Meetings of the Associations for Computational Linguistics*, pages 133–138, 1994. x, xv, 5, 103, 105, 108, 109, 111, 112, 113, 181

[Zadeh, 1965] L.A Zadeh. Fuzzy sets. *Information and control*, 8 :338–353, 1965. 21

[Zargayouna & Salotti, 2004a] H. Zargayouna et S. Salotti. Mesure de similarité dans une ontologie pour l'indexation sémantique de documents xml. In *Actes des journées francophones d'Ingénierie des Connaissances (IC'2004)*, 2004. 181

[Zargayouna & Salotti, 2004b] H. Zargayouna et S. Salotti. Mesure de similarité sémantique pour l'indexation de documents semi-structurés. In *12ème Atelier de Raisonnement à Partir de Cas*, 2004. 189

[Zargayouna *et al.*, 2001] H. Zargayouna, S. Salotti, et C. Golbreich. Raisonnement par similarité pour l'indexation et la recherche dans des documents multimédia. In *Complément des actes des journées francophones d'Ingénierie des Connaissances (IC'2001)*, 2001. 119

[Zargayouna, 2001] H. Zargayouna. Raisonnement par similarité pour l'indexation et la recherche dans des documents multimédia. Rapport interne LIMSI, N° 2001-12, 2001. 119, 124

[Zloof, 1977] M. Zloof. Query-by-example : A data base language. *BM Systems Journal*, 16(4) :324–343, 1977. 152

[Zweigenbaum, 1995] P. Zweigenbaum. Menelas final report. Deliverable report AIM-MENELAS17, 1995. 176