RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITÉ MOULOUD MAMMERI DE TIZI-OUZOU
FACULTÉ DE GÉNIE ELECTRIQUE ET D'INFORMATIQUE

DÉPARTEMENT D'INFORMATIQUE

MEMOIRE DE FIN D'ETUDES
En vue de l'obtention du diplôme de
Master en Informatique
Option : Conduite de Projets Informatiques

Thème :

Indexation sémantique du contenu des documents semi-structurés XML

Proposé et dirigé par :

Mr AMIROUCHE M.N.

Réalisé par :

Mlle KHIALI LYNDA

Présidente du jury :

Mme AMIROUCHE F.

Examinatrices :

Melle ILTACHE S.

Melle AIT ADDA S.

Promotion : 2014/2015

REMERCIEMENTS :

Au terme de ce travail,

Je tiens en premier lieu à remercier Allah pour le courage et la patience qu'il m'a donné afin de mener ce projet à terme.

Je tiens à exprimer ma profonde gratitude et sincères remerciements à mon promoteur Mr AMIROUCHE M. N., pour m'avoir proposé ce thème, pour la confiance qu'il m'a accordé, pour son suivi, sa disponibilité, ses orientations et ses remarques pertinentes et précieuses.

Je voudrai également remercier vivement les membres de jury qui ont aimablement accepté de juger notre travail.

A tous ceux qui ont contribué de prés ou de loin à l'élaboration de ce modeste travail, qu'ils trouvent ici l'expression de mes remerciements les plus sincères.

DÉDICACES :

Je dédie ce modeste travail

A mes chers parents qui m'ont toujours soutenu

A mes deux adorables frères Rabah et Rayan

A mes deux adorables sœurs Souad et Amel

A mes cousins Dahbia, Mohammed et Ikram

A mes chers grands parents

A toute ma grande famille, mes oncles, mes tantes et mes cousins

A tous mes merveilleux amis de la section B groupe 4

A mes meilleurs amis du labo 8

LYNDA

Table des matières

Table des matières

Table des matières

Table des matières

Liste des figures

Liste des figures

Liste des figures

Liste des tableaux

Liste des tableaux

Introduction générale

Introduction générale

L'Homme étend aujourd'hui son champ d'étude à un nombre croissant de domaines, ce qui conduit à l'augmentation incessante des connaissances et la prolifération du volume d'information produit.

L'apparition et la popularisation des ordinateurs ainsi que des supports de stockage a permis de réduire le coût d'archivage de cette gigantesque quantité d'information. Dés lors le problème n'est plus situé au niveau du stockage des documents, mais plutôt au niveau de la restitution des informations nécessaires aux utilisateurs en temps requis.

Ainsi, la conception et la mise en œuvre d'outils notamment les systèmes de recherche d'information (SRI), devient une nécessité absolue. Un SRI permet de retrouver, parmi une collection de documents, ceux qui répondent au besoin d'un utilisateur.

Contexte du travail :

Le contexte de notre travail se situe dans le domaine de la recherche d'information (RI).La RI est une branche de l'informatique qui concerne essentiellement l'acquisition, l'organisation, le stockage et la recherche de l'information.

Autrement dit, en quête d'information, l'utilisateur interroge une base documentaire par le biais d'une requête qui synthétise son besoin en information. Le rôle d'un système de RI est alors de retourner les documents jugés pertinents à sa requête.

Toute fois, le format des documents mis à disposition des utilisateurs a évolué de simples documents plats vers un nouveau format celui des documents structurés ou semi-structurés. L'apparition de langages de balisage tels que le standard XML, a permis de structurer le contenu informationnel des documents. Ajoutant ainsi, l'information de la structure à celle du contenu.

Dans le cadre de notre thèse nous nous sommes intéressés plus précisément à la recherche d'information structurée. Nous nous plaçons dans le cadre de documents semi-structurés XML, notre approche considère le contenu textuel par le sens des mots plutôt que par les mots eux mêmes, compris dans les documents.

Problématique :

Le but de tout système de recherche d'information est de satisfaire le besoin en information de l'utilisateur, en lui renvoyant les documents jugés pertinents à sa requête. Cependant, l'information pertinente renvoyée est souvent noyée au milieu d'information moins pertinentes ou encore non pertinentes, Dans ce cas l'utilisateur doit rechercher, dans les documents retournés, l'information requise.

Les documents XML, grâce à leur structure de balisage, permettent de traiter l'information textuelle avec une autre granularité que le document tout entier, l'unité d'information renvoyée à l'utilisateur n'est plus le document complet, mais un élément auto-explicite de celui-ci (paragraphe, section…).

Introduction générale

La communauté recherche d'information est confrontée alors à de nouvelles problématiques liées, d'une part, à la coexistence de l'information de contenu et celle de la structure dans les documents. Et d'une autre, au choix de la granularité de l'information à retourner à l'utilisateur. Ainsi, une adaptation des algorithmes et des procédures de recherche s'impose.

De nombreuses approches de recherche d'information structurée ont été proposées, ces approches présentent des limitations à différents niveaux, dont l'indexation qu'est une étape cruciale dans le processus de la RI.

En effet, l'indexation classique est basée sur les mots clés, où un élément est représenté par un sac de mots pondérés. Ainsi la pertinence d'un élément vis-à-vis d'une requête est souvent estimée en s'appuyant sur les fréquences d'apparition des mots de la requête dans ces mêmes éléments.

Les insuffisances de cette approche sont liées à la richesse du langage naturel. Puisque, un même mot peut posséder plusieurs sens et différents mots peuvent avoir une même signification. Ainsi, des éléments des documents bien qu'ils soient pertinents et contenant des mots sémantiquement équivalents mais lexicalement différents (synonymes) des mots de la requête, ne seront pas retrouvés. Par ailleurs, des éléments non pertinents, contenant des mots lexicalement identiques mais sémantiquement différents (homonymes) des mots de la requête seront retournés à l'utilisateur.

Pour pallier à ces limites, une approche par les sens des termes a été proposée. Elle correspond à l'indexation sémantique.

L'indexation sémantique permet de représenter l'élément par les sens des mots qu'il comprend plutôt que par les mots eux même. Elle se base sur des techniques de désambiguïsation pour identifier les sens des termes extraits généralement d'une ressource sémantique externe telle que l'ontologie WordNet.

Organisation du mémoire :

Notre mémoire traite de l'indexation sémantique du contenu des documents XML. Elle est organisée en quatre chapitres :

L'objectif du premier chapitre est de présenter la recherche d'information classique. Il définit les concepts de base liées a ce domaine, et décrit le processus en U de la RI, ainsi que ses étapes à savoir l'indexation, l'appariement et la reformulation de la requête. Il présente aussi les différents modèles d'appariement qui sont classé en trois catégories : les modèles basés sur la théorie des ensembles, les modèles algébriques et les modèles probabilistes. Et enfin, il expose les mesures d'évaluation des différents systèmes de recherche.

Le deuxième chapitre est scindé en deux parties. Dans la première partie, nous donnons un bref historique du standard XLM et nous présentons la structure générale d'un document

XML. Quand à la deuxième partie, elle est consacrée à l'état de l'art de la Recherche d'Information structurée (RIS), nous commençons d'abord par présenter les problématiques soulevées par la RIS. Nous décrivons ensuite les adaptations des méthodes d'indexation au format structuré de ces documents ainsi que des modèles d'appariements. Nous finissons par aborder les mesures d'évaluation des SRIS.

Dans le troisième chapitre, nous présentons les concepts fondamentaux de la RI sémantique et nous mettons l'accent sur les différents approches sémantiques en RI semi-structurée.

Dans le dernier chapitre, nous présentons l'approche que nous avons implémenté pour la RI sémantique dans les documents semi-structurés. Nous exposons également les expérimentations que nous avons réalisées afin de valider cette approche.

Chapitre I : Recherche d’information classique

Chapitre I : Recherche d'Information classique

Introduction :

La recherche d'information(RI) n'est pas un domaine récent, il remonte aux années 40, il est historiquement liée aux sciences de l'information et à la bibliothéconomie qui ont pour objectif d'établir les représentations des documents ainsi que des requêtes de l'utilisateur dans le but d'en récupérer des informations.

Un Système de Recherche d'Information (SRI), nécessite la combinaison de modèles et d'algorithmes. Ces derniers permettent la représentation, le stockage, la recherche et la visualisation des informations. L'objectif principal de ce système est de mettre en œuvre un processus de comparaison entre besoin utilisateur et documents d'une collection dans le but de retrouver ceux qui sont pertinents.

Depuis son apparition, la RI a connu de nombreuses avancées, qu'on va décrire dans ce premier chapitre. Nous commencerons par une présentation des concepts de base de la RI classique en section **I**, nous passerons ensuite à l'explication du processus en U de la recherche d'information en section **II**, la section **III** sera consacrée à la description des différents modèles de la RI. La dernière section, **IV**, sera dédiée aux mesures d'évaluation des SRI.

I. Notions de base de la recherche d'information :

La recherche d'information englobe des concepts clés nécessaires à la compréhension de ce travail dont voici les définitions :

- **La recherche d'information(RI)** :

La recherche d'information est une branche de l'informatique qui concerne essentiellement l'acquisition, l'organisation, le stockage et la recherche de l'information [Dinh et al., 2012].

- **Un système de recherche d'information(SRI) :**

Un système de RI est un ensemble de logiciels assurant l'ensemble des fonctions nécessaires à la recherche de l'information. Il offre des techniques et des outils permettant de localiser et de visualiser l'information pertinente relativement à un besoin en information, exprimé par un utilisateur sous forme de requête [Dinh et al., 2012].

- **Document :**

Un document est généralement défini comme le support physique d'une information structurée, sous forme de mots, de sons, d'images, une séquence vidéo, etc. Plus précisément on peut le définir comme un ensemble de données informatives présentes sur un support, d'une façon permanente et lisible par l'homme ou par une machine. En informatique, le mot document ou document électronique est généralement synonyme de fichier.

- **Collection de documents :**

Appelé aussi corpus, base ou fond documentaire, elle correspond à un ensemble de documents qui peuvent traiter de même domaine, dans ce cas il s'agit d'une collection de domaine (par exemple : collection de la génomique) et dans le cas contraire, c-à-d. Si ces

derniers ne traitent pas de même domaine on parlera alors d'une collection générique (par exemple : la collection des articles de presse).

- **Requête :**

Une requête constitue l'expression du besoin en information de l'utilisateur et fait office d'interface entre celui-ci et le SRI. Divers types de langages d'interrogation ont été proposés en RI pour formuler une requête. Parmi les plus répandus nous citons :

- langage booléen : L'utilisateur exprime sa requête sous forme d'un ensemble de termes reliés entre eux par des opérateurs booléens (ET, OU, NON).

- langage naturel ou quasi naturel : L'utilisateur exprime sa requête en langage libre (langage naturel) sous forme de mots clés. Le système se charge de traduire (analyser) ces mots clés en une requête de langage de base de données ou une autre forme interne utilisable par le système.

- langage graphique : Une interface d'aide à la formulation de la requête est proposée à l'utilisateur. En effet, une vue d'ensemble de la base d'information et en particulier une vue de termes représentant le contenu sémantique des documents, est donnée a l'utilisateur pour l'assister à formuler sa requête.

- **La pertinence** :

La pertinence est une notion fondamentale en RI et fait l'objet de tout SRI. Elle dénote une relation reliant une requête utilisateur à un document qui satisfait le besoin en information visé par cette dernière, plusieurs définitions [Saracevic, 1970] lui sont rattachées, on citera parmi celles-ci les suivantes :

- la correspondance entre un document et une requête, une mesure d'informativité du document à la requête.
- un degré de relation (chevauchement, relativité, …) entre le document et la requête.
- une mesure d'utilité du document pour l'utilisateur.

On distingue principalement deux types de pertinence : la pertinence système et la pertinence utilisateur.

- **Pertinence utilisateur** [Harter, 1992] [Mezzaro, 1997] [Saracevic, 1996] : se traduit par les jugements de pertinence de l'utilisateur sur les documents fournis par le SRI en réponse à une requête. La pertinence utilisateur est subjective, car pour un même document retourné en réponse a une même requête, il peut être jugé différemment par deux utilisateurs distincts (qui ont des centres d'intérêt différents). De plus, cette pertinence est évolutive, un document jugé non pertinent à l'instant **t** pour une requête peut être jugé pertinent à l'instant **t+1**, car la connaissance de l'utilisateur sur le sujet a évolué.

- **Pertinence Système** [Cleverdom, 1970] : est souvent présentée par un score attribué par le SRI afin d'évaluer l'adéquation du contenu des documents vis-à-vis de celui de la requête. Ce type de pertinence est objectif et déterministe.

Le but de tout système de recherche d'information est alors de faire correspondre au mieux la pertinence système avec la pertinence utilisateur [Denos, 1997].

II. Le processus de la RI :

Afin de répondre aux besoins en information de l'utilisateur, un système de recherche d'information intègre trois fonctions principales qui sont : l'indexation, l'appariement document-requête et la reformulation de la requête (processus non toujours présent mais important).

Nous pouvons représenter schématiquement un SRI, comme illustré par la figure 1.1, par ce qui est appelé communément le processus en U de recherche d'information.

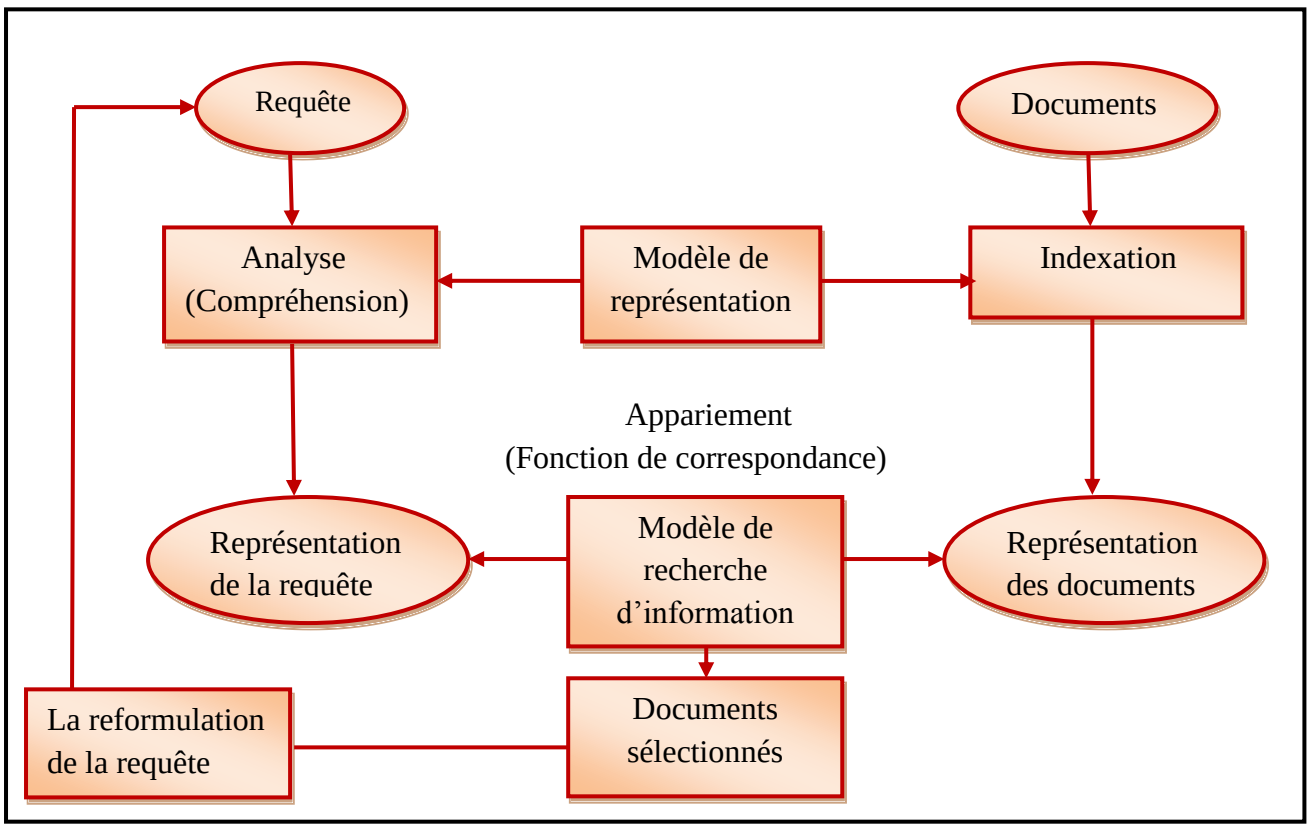


Figure 1.1 : processus en U de la recherche d'information[Belkin et al., 1992].

II.1. Le processus d'indexation :

L'indexation est une étape très importante dans le processus de RI. Elle consiste à associer à chaque document(ou requête) une liste de mots-clés appelés aussi descripteurs, susceptible de représenter au mieux son contenu sémantique. L'ensemble des descripteurs retenus sont regroupés dans des structures de données appelées index. Un index peut contenir soit des mots du document, des n-grammes (une séquence de N caractères consécutifs) ou des concepts (expressions pouvant contenir un ou plusieurs mots).

Le tableau ci-dessous, représente un exemple de ces différentes formes de descripteurs :

Texte		Modèles de la recherche d'information structurée
Mot	Origine	modèles, de, la, recherche, d, information, structurée
	Lemme	modèle, de, la, recherche, information, structure
	Racine	modèl, d, l, recherch, inform, structur
Concept		R.I
Bigramme		Mo;dè;le;s ;de; l;a ;re;ch;……..

Tableau 1.1 : Les différents types de descripteurs.

Le but général de l'indexation est d'identifier l'information contenue dans tout texte et de la représenter au moyen d'un ensemble d'entités (descripteurs) pour faciliter la comparaison entre la représentation d'un document et d'une requête

II.1.1. Langages d'indexation :

Le langage d'indexation est l'ensemble des descripteurs résultant de l'indexation d'une collection de documents. On distingue deux types de langage d'indexation : le langage contrôlé et le langage libre.

- **Langage libre :** le langage libre se compose d'éléments, qui sont extraits à partir du document ou de la requête à analyser.

- **Langage contrôlé :** Le langage contrôlé est un langage normalisé dont le vocabulaire est prédéfini (thésaurus ou bases terminologiques). Dans ce cas l'indexeur sélectionne un ou plusieurs descripteurs à partir de ce vocabulaire pour représenter le document.

II.1.2. Modes d'indexation :

L'indexation peut être effectuée selon trois modes différents : mode manuel, mode automatique et mode semi-automatique.

❖ **Mode manuel :**

En mode manuel, l'indexation est réalisée par un spécialiste ou un documentaliste qui effectue l'analyse du document et choisit les descripteurs les plus adéquats à sa représentation.

Ce type d'indexation est très souvent critiqué pour son coût. En effet, la personne chargée de l'analyse des documents doit posséder les connaissances minimales à la compréhension des centres d'intérêt du document, sous peine d'obtenir une indexation

incorrecte. L'indexation manuelle se caractérise aussi par un haut degré de subjectivité. Par ailleurs, pour un même document, des termes différents peuvent être affectés par des indexeurs différents, même si l'indexation s'appuie sur un langage contrôlé, ce qui entraîne une incohérence dans la base des index et diminue les performances du système de recherche.

❖ **Mode automatique :**

L'indexation automatique se fait à l'aide d'un processus entièrement automatisé. Elle présente l'avantage d'une régularité du processus, car elle fournit toujours le même index pour le même document. Elle constitue le mode le plus utilisé dans le processus de la RI.

❖ **Mode semi-automatique :**

Appelé aussi indexation assistée ou supervisée, ce mode d'indexation jumelle entre l'indexation manuelle et l'indexation automatique. Ainsi, une première représentation du document est fournit par le système, puis celle-ci est validée ou corrigée par un humain, ce qui veut dire que le choix final des descripteurs reste au spécialiste du domaine ou au documentaliste.

II.1.3. Les étapes de l'indexation automatique :

L'indexation automatique comprend les étapes suivantes :

- L'extraction des termes du document.
- La sélection des termes discriminatifs pour un document.
- La pondération des termes

II.1.3.1. L'extraction des termes du document :

L'extraction des termes du document est une tâche complexe réalisée par étapes. Ces étapes sont :

- **La tokenisation :** La tokenisation consiste à segmenter un texte en plusieurs unités atomiques, appelées jetons ou tokens, cette segmentation est basée en générale sur la ponctuation et sur une liste de séparateurs, le résultat de cette étape est un ensemble de mots.

- **L'élimination des mots vides :** Les mots vides sont des mots qui permettent de structurer une phrase (les articles, les conjonctions de coordination, les verbes auxiliaires, etc.). Ces mots ne portent pas de sens et ne peuvent pas constituer des termes d'indexation. On peut donc les éliminer, pour cela on utilise une liste qui regroupe tous les mots vides appelé stop liste ou anti-dictionnaire.

  Cependant, il ne faut pas oublier de tenir compte de certains mots vides qui auraient pour homographes des mots significatifs comme par exemple la conjonction de coordination « or » qui peut utilisé pour référer a «l'or » qui est un métal précieux.

- **La normalisation :** la normalisation consiste à représenter les différentes variantes d'un terme par un format unique appelé lemme ou racine par application d'un algorithme de racination ou de lemmatisation.
  - **Racinisation** : la racinisation, ou le stemming(en anglais), consiste à rechercher la forme tronquée d'un mot à partir de laquelle peuvent être reconstruites ses différentes variantes morphologiques. Par exemple : les mots : étude, étudiant et étudier, sont construits à partir de la racine étud.
    Parmi les algorithmes de racinisation qui existent on citera celui de Porter [Porter, 1980] pour l'anglais.
    L'avantage de la racinisation est qu'elle augmente le rappel, cependant elle cause une diminution de la précision.(les deux notion rappel et précision seront détaillées en section **IV**).

  - **Lemmatisation** : la lemmatisation consiste à remplacer un mot par son lemme. Un lemme représente la forme canonique d'un mot (infinitif pour les verbes, singulier pour les noms, etc.), il constitue en général les entrées dans un dictionnaire de cette langue.
    Par exemple : les mots port, portes et portera seront remplacés par leurs lemmes : port, porter ou porte selon le contexte et porter.
    La lemmatisation est une opération plus coûteuse que la racinisation car elle nécessite une analyse morphologique et syntaxique des phrases.

II.1.3.2. La sélection des termes discriminatifs pour un document :

Cette phase vise à réduire le nombre de termes issus de la première étape, en éliminant les non importants afin de ne garder que les termes pertinents pour la représentation du document. L'une des principales lois utilisées à cet effet est la conjecture de Luhn [Luhn, 1958] qui considère que les termes de fréquence très élevée (qui reviennent souvent) et ceux de faible fréquence (très rares) ne sont pas représentatif du contenu du document. Alors que les termes de fréquence intermédiaire sont les plus signifiants. Ainsi deux seuils de fréquence (seuil max et seuil min), sont fixés et seuls les termes entre ces deux seuils sont alors retenus dans l'index.

II.1.3.3. La pondération des termes :

La pondération permet d'affecter à chaque terme d'indexation une valeur qui représente son poids. De manière générale, la majorité des méthodes de pondération sont construites par la combinaison de deux facteurs. Un facteur de pondération local, noté TF (term frequency), quantifiant la représentativité locale d'un terme dans le document, et un second facteur de pondération globale, noté IDF (pour Inverse of Document Frequency), mesurant la représentativité globale du terme vis-à-vis de la collection des documents.

➢ **Pondération locale :**

La pondération locale permet de mesurer l'importance du terme dans le document, En ne tenant compte que des informations locales du terme qui ne dépendent que du document. Il existe plusieurs fonctions de pondération locales données par les formules suivantes :

- tfij : nombre d'occurrences du terme ti dans le document Dj.
- 0 ou 1 : le poids du terme vaut 1 si la fréquence d'occurrence du terme dans le document est supérieure ou égale à 1, et 0 sinon.
- $\alpha + \log(tf_{ij})$: avec α une constante.
- $\frac{tf_{ij}}{\max_{t_{i \epsilon D_j}} tf_{ij}}$ ou $0.5+0.5 \frac{tf_{ij}}{\max_{t_{i \epsilon D_j}} tf_{ij}}$: où $\max_{t_{i \epsilon D_j}} tf_{ij}$ représente la fréquence la plus élevée observée dans le document Dj. Ces deux formules dénotent la normalisation de la mesure TF permettant de réduire les différences entre les poids associées aux termes du document.

➢ **Pondération globale :**

La pondération globale mesure l'importance d'un terme dans toute la collection. L'idée sous-jacente est que les termes qui apparaissent dans peu de documents de la collection sont plus représentatifs du contenu de ces documents que ceux qui apparaissent dans tous les documents de la collection. Cette mesure est exprimée selon l'une des deux déclinaisons suivantes :

$IDF = \log \frac{N}{n_i}$

Ou

$IDF = \log\left[\frac{N-n_i}{N}\right]$

Où n_i est le nombre de documents contenant le terme t_i et N le nombre total de documents dans la collection

➢ **La mesure TF*IDF :**

La mesure TF*IDF combine les deux critères qu'on a vus: l'importance du terme pour un document (donné par TF), et le pouvoir de discrimination de ce terme (donné par IDF). Ainsi, un terme qui a une valeur de TF*IDF élevée doit être à la fois important dans ce document, et aussi il doit apparaître peu dans les autres documents. Parmi les formules résultantes de cette combinaison, nous citerons les suivantes :

- $TF*IDF = 0.5+0.5 \frac{tf_{ij}}{\max_{t_{i \epsilon D_j}} tf_{ij}} * \log \frac{N}{n_i}$
- $TF*IDF = tf_{ij} * \log \frac{N}{n_i}$

La mesure TF*IDF est une bonne approximation de l'importance d'un terme dans un document, particulièrement dans des corpus de documents de tailles homogènes. Cette mesure

a eu en revanche un succès très limité dans les corpus de tailles très variables puisque elle favorise les documents longs et augmente leurs similarité vis-à-vis à la requête. Le problème posé est que les termes appartenant aux documents longs apparaissent très fréquemment et emportent le poids sur les termes appartenant à des documents moins longs, les documents longs auront alors plus de chance d'être sélectionnés.

Pour remédier à ce problème une normalisation de cette mesure, par intégration de la taille du document dans les formules de pondération, a été proposée [Singhal et al., 1996] [Robertson et al., 1997].

Remarque :

Cependant, pour la requête il est difficile pour un utilisateur du SRI de pondérer ses termes. On choisit donc, en général, l'une des deux variantes suivantes :

w_{iq} = 1 si le terme t_i appartient au langage de l'indexation, 0 sinon .
w_{iq} = IDF_i (pondération globale du terme t_i) si le terme t_i appartient au langage d'indexation, 0 sinon.
Où wiq est le poids du terme ti dans la requête Q.

II.2. L'appariement document-requête:

Le processus d'appariement document-requête permet de mesurer la pertinence d'un document vis-à-vis d'une requête. De manière générale, à la réception d'une requête Q, le SRI lui crée une représentation conforme à celles des documents. Puis il calcule un score de similarité, noté RSV(D,Q) (Retrieval Status Value) , entre chaque document D contenu dans le corpus et cette requête Q, c'est ce score qui traduit la pertinence du document par rapport à la requête. A l'issue de cette étape, le système renvoie une liste de documents ordonnés selon leur degré de pertinence. L'expression de la fonction de similarité RSV(D,Q) est tributaire du modèle de RI choisi comme on va le voir en section **III**.

II.3. La reformulation de la requête :

La reformulation de la requête est un processus permettant la construction d'une nouvelle requête, plus à même de représenter les besoins en information de l'utilisateur, en modifier sa requête initiale soit par ajout de termes significatifs et/ou réestimation de leur poids. Les approches les plus utilisées à cet effet sont :

- **Reformulation par réinjection de pertinence :**

Appelé relevance feedback en anglais, selon cette méthode l'utilisateur soumet d'abord sa requête au système qui lui renvoie un ensemble de document, ensuit il doit indiquer ceux qui lui sont pertinent et ceux qui ne le sont pas.la requête est alors modifier automatiquement soit par repondération des termes ou par l'ajout (respectivement le retrait) de termes contenus dans les documents jugés pertinents (respectivement non pertinents) à la requête initiale.

Le jugement porté par l'utilisateur sur l'ensemble des documents retourné par le SRI peut se faire soit d'une manière explicite, dans ce cas l'utilisateur clique sur les documents en indiquant explicitement si il s'agit d'un document pertinent ou non, ou bien d'une manière implicite. L'approche implicite est basée sur l'hypothèse que le système doit déduire le jugement de l'utilisateur en interprétant son comportement (les cliques de souris sur les documents, le défilement sur une page Web, la sauvegarde des documents dans le marque-page et la durée de consultation des documents) sans lui demander d'autres actions supplémentaires ni d'efforts.

- **Reformulation par pseudo-réinjection de la pertinence :**

La reformulation par pseudo-réinjection de la pertinence (Blind Feedback ou encore Pseudo Relevance Feedback, notée PRF) utilise des techniques de réinjection automatique à l'aveugle pour construire la nouvelle requête. Dans cette approche l'intervention de l'utilisateur n'est pas requise, le SRI considère que les K premier document sont pertinents et les utilise pour transformer la requête. Le processus de la PRF consiste généralement à ajuster ou modifier le poids des termes (repondération) et ajouter les termes les plus pertinents qui sont extraits à partir de ces K premiers documents.

- **Expansion de la requête :**

L'expansion de la requête consiste à ajouter d'autres termes à ceux choisis par l'utilisateur pour l'interrogation. Ces nouveaux termes sont issus d'une ressource externe (ontologie, thésaurus ou dictionnaire) et sont proche sémantiquement de ceux contenus initialement dans la requête.

III. Modèles de la RI :

Tous SRI repose sur un modèle, dit modèle de recherche d'information ou modèle d'appariement document-requête, qui détermine son comportement vis-à-vis à un besoin utilisateur. Un modèle a pour fonction de créer une représentation interne pour un document ou pour une requête, basée sur les termes issues du processus d'indexation, Ainsi, que de définir une méthode de comparaison entre ces deux représentations (celle de la requête et du document) afin de déterminer leur degré de correspondance. Autrement dit, il fournit un cadre théorique pour la modélisation de la mesure de pertinence. [Salton et al., 1983].

On distingue principalement trois types de modèles: les modèles booléens, les modèles vectoriels et les modèles probabilistes.

III.1. Les modèles booléens :

Les modèles booléens sont les premiers modèles utilisés en RI [Salton, 1970], ils sont basés sur la théorie des ensembles et l'algèbre de Boole. Trois variations principales y sont distinguées : le modèle booléen, le modèle booléen étendu et le modèle booléen flou.

III.1.1.Le modèle booléen :

Ce modèle est caractérisé par sa simplicité, la rapidité de sa mise en œuvre et son caractère intuitif. Il se base sur l'hypothèse de présence/absence des termes de la requête dans le document.

Dans ce modèle un document est représenté par une conjonction logique de termes, par exemple : $d = t_1 \wedge t_2 \wedge \ldots \wedge t_n$ .Tandis que la requête est constituée d'un ensemble de termes séparés par les opérateurs booléens AND, OR, NOT (expression logique).Quant à l'appariement (RSV), c'est un appariement exact, autrement dit si un document implique au sens logique la requête alors le document est pertinent. Sinon, il est considéré non pertinent.

La correspondance RSV (D_j,Q_k), entre une requête Q_k et un document D_j est déterminée comme suit :

RSV(Dj,qi)=1 si qi ∈Dj ; 0 sinon,
RSV(Dj,qi∧qj)=1 si RSV(Dj,qi)=1 et RSV(Dj,qj)=1 ; 0 sinon,
RSV(Dj,qi∨qj)=1 si RSV(Dj,qi)=1 ou RSV(Dj,qj)=1 ; 0 sinon,
RSV(Dj,¬qi)=1 si RSV(Dj,qi)=0 ; 0 sinon,

Sachant que q_i et q_j sont des termes de la requête Q_k.

La réponse à une requête Q_K est l'ensemble des documents qui sont similaires à cette requête: $rep(Q_K) = \{Dj \in D \mid RSV(Dj, Q_K) = 1\}$ Où D est l'ensemble des documents constituant le corpus.

Cependant, le modèle booléen classique présente trois inconvénients majeurs qui sont :

- L'incapacité du modèle à trier les documents pertinents.
- Pondération des termes soit à 1(si le terme apparait dans le document) soit à 0 (si le terme est absent du document). Ainsi, tous les termes ont la même importance.
- Négligence des documents pertinents dont la représentation ne correspond qu'approximativement à la requête.

III.1.2.Le modèle booléen étendu :

Le modèle booléen étendu [Salton et al., 1983] est une extension du modèle booléen, il a été introduit afin de pallier au inconvénient de ce dernier. Moins stricte que le premier celui-ci autorise la sélection des documents qui ressemblent à la représentation de la requête et permet d'organiser ceux retenus par ordre d'importance. Pour ce faire les termes des

documents sont pondérés selon la fonction suivante :

$$w_{ij} = tf_{ij} * \frac{IDF(ti)}{\max IDF(ti)}$$

Où tf_{ij} est la fréquence du terme t_i dans dj, IDF (t_i) est la fréquence inverse de documents du terme ti calculée par :

$$IDF\ (t_i) = \frac{N_c}{|\{d : t_i \in d\}|}$$

Où N_c est le nombre total de document dans la collection et $|\{d : t_i \in d\}|$ est le nombre total de documents contenant le terme t_i.

Tandis que la requête demeure une expression booléenne classique. L'appariement requête-document est le plus souvent déterminé par les relations introduites dans le modèle p-norm basées sur les p-distances, avec $1 \leq p \leq \infty$. La valeur de p est indiquée au moment de la requête. Si m est le nombre de termes dans la requête, les fonctions de similarité se calculent comme suit :

$$RSV\ (d,Q_{ou}) = \left[\frac{x_1^p + x_2^p + \cdots + x_m^p}{m}\right]^{\frac{1}{p}}$$

$$RSV\ (d,Q_{et}) = \left[\frac{(1-x_1)^p + (1-x_2)^p + \cdots + (1-x_m)^p}{m}\right]^{\frac{1}{p}}$$

Où x_i représente le poids du terme ti dans le document d, sachant que t_i apparait dans la requête.

III.1.3.Le modèle booléen flou :

Le modèle booléen flou est lui aussi une extension du modèle booléen. Basé sur la théorie des ensembles flou (la logique floue), il vise à modéliser les notions 'imprécision', 'incertitude' de l'information [Paice, 1984][Dubois et Prade, 1988][Bosc et Prade, 1996] [Bordogna et Pasi, 2000].

Un ensemble flou est un ensemble dont les éléments sont affectés d'un degré d'appartenance. Dans notre cadre d'étude, chaque document correspond à un ensemble flou et les termes qui indexent se dernier représentent les éléments de cet ensemble. Quant au degré d'appartenance, il réfère au degré d'indexation d'un terme à un document. Autrement dit, il s'agit du poids de ce terme dans le document. Le score de pertinence du document d vis-à-vis de la requête q est calculé comme suit :

- sim(d, ti) = poids(ti, d)
- sim(d, qi∧ qj) = min(sim(d, qi), sim(d, qj))
- sim(d, qi∨ qj) = max(sim(d, qi), sim(d, qj))
- sim(d, ¬qi) = 1 − sim(d, qi)

Où q_i,q_j représente les termes de la requête q et poids(ti, d) représente le poids du terme t_i dans le document d

III.2. Les modèles vectoriels:

Appelés aussi modèles algébriques, ils reposent sur les bases mathématiques des espaces vectoriels.

Les modèles vectoriels constituent une classe de modèles de RI qui compte : le modèle vectoriel, le modèle vectoriel généralisé, le modèle LSI (Latent Semantic Indexing) et le modèle connexionniste.

III.2.1.Le modèle vectoriel :

Le modèle vectoriel a été introduit par Salton [Salton, 1971], concrétisé dans le cadre du système SMART. Il est l'un des modèles les plus utilisés et les plus étudiés en RI

Dans ce modèle, le document est représenté par un vecteur document. De même la requête est représentée par un vecteur requête constitué de N composantes (N étant le nombre total de termes issus de l'indexation de la collection des documents) comme suit :

$D_j = (w_{1j}, w_{2j}, \ldots, w_{nj})$
$Q = (w_{1q}, w_{2q}, \ldots, w_{nq})$

Où w_{ij} représente le poids du terme t_i dans le document D_j et w_{iq} représente le poids du terme t_i dans la requête. Quant aux composantes nulles, elles représentent les poids des termes qui n'apparaissent pas dans le document ou la requête alors que les composantes positives représentent les poids des termes significatifs.

Puisque la même représentation est utilisée pour les documents contenus dans la base et pour les requêtes des utilisateurs, il est possible de comparer directement les documents aux requêtes, et leur attribuer un degré de ressemblance. Ainsi, la pertinence système des documents s'en déduit suivant la règle : plus le document est proche d'une requête (plus il lui ressemble), plus il est pertinent. Voici un exemple dans la figure 1.2 suivante :

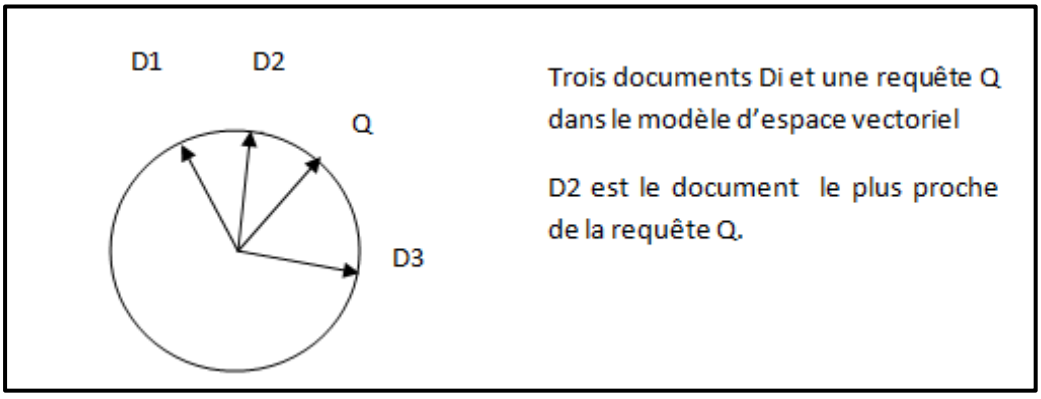


Figure 1.2 : Exemple d'une représentation du modèle vectoriel [Baziz et al., 2005].

L'évaluation de la distance entre les deux vecteurs, document et requête, est réalisée par l'une des fonctions suivantes :

Produit scalaire : $RSV(D_j, Q) = \sum_{i=1}^{n} w_{ij} * w_{iq}$

La formule de Dice : $RSV(D_j, Q) = \frac{2*\sum_{i=1}^{n} w_{ij}*w_{iq}}{\sum_{i=1}^{n} w_{ij}^2+\sum_{i=1}^{n} w_{iq}^2}$

La formule de Jaccard : $RSV(D_j, Q) = \frac{\sum_{i=1}^{n} w_{ij}*w_{iq}}{\sum_{i=1}^{n} w_{ij}^2+\sum_{i=1}^{n} w_{iq}^2-\sum_{i=1}^{n} w_{ij}*w_{iq}}$

la formule de cosinus : $RSV(D_j, Q) = \frac{\sum_{i=1}^{n} w_{ij}*w_{iq}}{\sqrt{\sum_{i=1}^{n} w_{ij}^2}*\sqrt{\sum_{i=1}^{n} w_{iq}^2}}$

L'un des avantages du modèle vectoriel réside dans sa simplicité conceptuelle et l'aisance de sa mise en œuvre. Ainsi que son habilité à ordonner les documents retournés selon leurs degrés de pertinence. Cependant, l'inconvénient majeur de ce modèle est son incapacité à modéliser les associations entre les termes d'indexation c'est-à-dire que chaque terme est considéré comme indépendant des autres. Pour remédier a cette limitation, plusieurs variantes du modèle vectoriel ont été proposées. Parmi elles on retrouve : le modèle vectoriel généralisé [Wong et al., 1985], le modèle LSI (Latent Semantic Indexing) [Foltz, 1990] [Furnas et al., 1987] et le modèle connexioniste [Boughanem, 1992] [Kwok, 1989].

III.3. Les modèles probabilistes :

Les modèles de recherche probabilistes sont fondés sur la théorie des probabilités, ils englobent le modèle probabiliste, les réseaux inférentiels bayésiens, et le modèle de langue.

III.3.1. Le modèle probabiliste :

Le premier modèle probabiliste a été proposé par Maron et Kuhns [Maron et al., 1960] au début des années 1960. Basé sur le calcule des probabilités ce modèle introduit deux probabilités conditionnelles qui sont :

P(R/D) : la probabilité de pertinence du document D sachant sa description.
P (NR/D) : la probabilité de non pertinence du document D sachant sa description.

Pour chaque requête utilisateur Q les deux probabilités sont estimées et le document D est sélectionné si la probabilité qu'il soit pertinent à Q est supérieure à la probabilité qu'il soit non pertinent à Q. le score d'appariement entre le document D et la requête Q est donné par :

$RSV(D, Q) = \frac{P(R/D)}{P(NR/D)}$

Par conséquent, le document D est dit pertinent si RSV (D, Q) > 1.

En appliquant le théorème de Bayes et après simplification, la fonction d'appariement est noté comme suit :

$RSV(D, Q) = \frac{P(R/D)}{P(NR/D)} \approx \frac{P(D/R)}{P(D/NR)}$

Avec P(D/R) (respectivement P(D/NR))est la probabilité d'observer le document D sachant qu'il est pertinent (respectivement non pertinent).

Il existe plusieurs méthodes pour l'estimation de ces différentes probabilités. La plus connue est celle du modèle BIR (Binary Independance Retrieval).

Le modèle BIR représente le document par un ensemble d'événements x_i indépendants, noté D(t1= x1, t2=x2, .. , tn= xn), soit égale à 0 si le terme ti est absent de D soit égale à 1 si le terme ti est présent dans D, les deux probabilités ,P(D/R) et P(D/NR), sont données par:

P(D/R) = P(t1= x1, t2=x2, .. , tn= xn /R)= $\prod_i P(t_i = x_i /R)$
P(D/NR) = P(t1= x1, t2=x2, .. , tn= xn /NR) = $\prod_i P(t_i = x_i /NR)$

En appliquant la distribution des termes selon la loi de Bernoulli, on obtient :

P(D/R) = $\prod_i P(t_i = x_i /R) = \prod_i P(t_i = 1/R)^{x_i} * P(t_i = 0/R)^{1-x_i}$
P(D/NR) = $\prod_i P(t_i = x_i /NR) = \prod_i P(t_i = 1/NR)^{x_i} * P(t_i = 0/NR)^{1-x_i}$

Et en posant :

pi = P(xi=1/R) et qi = P(xi=1/NR)

On déduit :

1-pi = P(xi=0/R) et 1-qi=P(xi=0/NP)

La fonction d'appariement peut s'écrire, après transformation, comme suit :

RSV(D,Q)= $\sum_{i,x_i=1} \log \frac{p_i(1-p_i)}{q_i(1-q_i)}$

Un des inconvénients de ce modèle est la nécessité d'une collection d'apprentissage pour l'estimation des probabilités p_i et q_i. Pour pallier cet inconvénient, Roberston [Roberston, 1994] a proposé le modèle 2-poisson basé notamment sur la notion de termes. Le résultat de ses travaux est la formule BM25 (Best Matching), largement utilisée dans les travaux actuels de RI.

Le modèle BM25 propose un schéma de pondération et une fonction d'appariement document-requête basée sur deux facteurs qui sont :

- La fréquence du terme dans le document.
- La longueur du document par rapport à la longueur moyenne des documents dans la collection.

La similarité, RSV(q, d), calculée pour une requête q et un document d est donnée par la formule suivante :

RSV (q, d) = $\sum_{t\epsilon\, q\cap d} \frac{(k_1+1).tf}{K+tf} . \frac{(k_3+1).qtf}{k_3+qtf} . w$

Où :

- k1et k3 détermine respectivement l'importance de la fréquence du terme dans le document et la requête.
- tf est la fréquence du terme dans le document d.
- qtf est la fréquence du terme dans la requête q.
- W représente la fréquence inverse de document (IDF), déterminé comme ci :

$W = \log_2 \frac{N - N_t + 0.5}{N_t + 0.5}$ où N_t représente le nombre de documents contenant le terme t et N le nombre de documents dans la collection.

- K est un facteur de normalisation de la langueur du document :

$K = k1.\left((1 - b) + b.\frac{dl}{avg\_dl}\right)$ où le paramètre b ∈ [0, 1] permet de déterminer l'effet de la normalisation de la longueur dl du document par rapport à la longueur moyenne des documents avg_dl dans la collection.

III.3.2. Les réseaux inférentiels bayésiens :

Un réseau inférentiel bayésien est un graphe de dépendances, orienté et sans cyclique. Dans ce graphe les nœuds représentent des variables propositionnelles ou également des constantes et les arcs des liens de dépendances entre les nœuds. Ainsi, si la proposition représentée par le nœud p cause ou implique la proposition représentée par le nœud q, on trace alors un arc de p vers q.

Dans le contexte de la recherche d'information [Turtle et al., 1991] les nœuds et les arcs sont définis comme suit :

- Les nœuds : représentent des concepts, des groupes de termes ou des documents.
- Les arcs : représentent les dépendances entre termes et entre termes et documents.

Des variables aléatoires sont associées aux termes de l'index, les documents et la requête. Une variable aléatoire associée à un document D représente l'événement d'observer ce document. Les arcs sont alors dirigés du nœud document vers ses nœuds termes : ainsi, l'observation d'un document est la cause d'une augmentation de la valeur des variables associées avec ses termes d'index. Quant à la variable aléatoire associées à la requête utilisateur, elle modélise l'événement que la requête a été vérifiée. La valeur de ce nœud requête est donnée en fonction des valeurs des nœuds associés aux termes de la requête. Ainsi, les arcs sont orientés des nœuds des termes de l'index vers le nœud de la requête. La figure suivante illustre un réseau inférenciel bayésien simple :

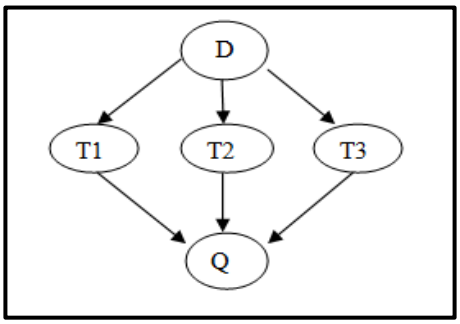


Figure 1.3 : Modèle de réseau inférentiel bayésien simple [Turtle et al., 1990].

La figure ci-dessus illustre un réseau inférentiel bayésien de pertinence d'un document vis-à-vis d'une requête composée de trois termes. L'événement « la requête est accomplie », Q=1, est réalisé si le sujet lié à un terme est vrai (T1=1,T2=1 ou T3=1), ou une combinaison de ces événement. Les trois sujet sont inférés par l'événement « le document est pertinent », D=1.Par l'enchaînement des règles de probabilité jointe des autres nœuds du graphe comme suit :

P(D,T1,T2,T3,Q)= P(D) P(T1|D) P(T2|D,T1) P(T3|D,T1,T2) P(Q|D,T1,T2,T3)

La direction des arcs indiquant les relations de dépendance entre les variables aléatoires, l'équation devient :

P(D,T1,T2,T3,Q)= P(D) P(T1|D) P(T2|D) P(T3|D) P(Q,T1,T2,T3)

La probabilité de réalisation de la requête pour un document D,P(Q=1|D=1) ,est utilisé comme score d'ordonnancement des documents, elle est donnée par la formule suivante :

$$P(Q=1|D=1) = \frac{P(Q=1,D=1)}{P(D=1)} = \frac{\sum P(D=1,T1=t_1,T2=t_2,T3=t_3,Q=1)}{P(D=1)}$$

III.3.3.Le modèle de langue :

Dans les approches de recherche d'information basées sur les modèles de langue, on considère que la pertinence d'un document face à une requête est en rapport avec la probabilité que la requête Q puisse être générée par le modèle de langue du document D, noté M_D. Autrement dit, ces modèles visent à déterminer la probabilité que la requête Q soit inférée ou générée par le modèle de langue du document. Dans ce cas la fonction de correspondance est égale à :

$RSV(D,Q) = P(Q|M_D)$

Où chaque document D est vu comme la représentation d'un langage auquel correspond un modèle de langue. Un modèle de langue se construit en utilisant « une fonction de probabilité P qui assigne une probabilité P(s) à un mot ou une séquence de mots s dans une langue ». Le modèle le plus utilisé en RI est le n-gramme sous sa forme la plus simple, c'est-à-dire avec des unigrammes où n=1. Ainsi le modèle M_D est donné par :

$M_D = \{(t_i, P(t_i))\}$ avec t_i les termes du document D.

La probabilité $P(t_i)$ est estimée par la fréquence du terme t_i dans le document, normalisée par la longueur du document :

$P(t_i) = \frac{tf_{t_i}}{\sum_{t_i \epsilon d} tf_{t_i}}$ où tf_{t_i} représente la fréquence du terme t dans le document D.

Dans le cas d'un modèle uni-gramme, la fonction de correspondance est donnée par le produit des probabilités individuelles des termes de la requête :

$RSV(d,q) = P(q|M_d) = \prod_{t \epsilon q} P(t|M_D)$

Où $P(t|M_D) = P(t)$ par abus d'écriture

Cependant, lorsqu'un terme de la requête est absent du document, la formule de probabilité P(t) lui attribue une probabilité nulle, et le produit des probabilités de tous les termes de la requête conduit à un score nul pour le document. Pour rendre ce score tolérant à l'absence de termes de la requête et pour avoir de meilleures estimations des probabilités, de nombreuses méthodes de lissage ont été proposées dont le lissage de Laplace. Le lissage de Laplace des probabilités consiste à ajouter une quantité faible identique à toutes les valeurs (généralement 1 dans le cas des modèles de langue pour la recherche d'information), si bien qu'aucune valeur n'est nulle.

IV. L'évaluation des SRI :

L'évaluation des systèmes de recherche d'information constitue une étape importante dans l'élaboration d'un modèle de recherche d'information. En effet, elle permet de caractériser le modèle et de fournir des éléments (critères) de comparaison entre modèles.

Plusieurs critères peuvent entrer en jeu dans le processus d'évaluation d'un SRI, tels que :

- Le temps de réponse.
- La présentation claire des résultats.
- L'effort fourni par l'utilisateur pour récupérer l'information pertinente.
- La pertinence des documents retournés.

La pertinence est le critère le plus important au quel doit répondre tout SRI c'est-à-dire qu'il doit être capable de retrouver, dans un corpus, tous les documents pertinents et rejeter tous ceux qui sont non pertinents à une requête utilisateur.

Les mesures les plus utilisées pour l'évaluation du critère de pertinence sont : le rappel et la précision.

➢ Le rappel :

Le rappel mesure la capacité du système à retrouver tous les documents pertinents répondant à une requête, il est donné par la formule suivante :

Rappel = $\frac{\text{nombre de documents pertinents selectionnés}}{\text{nombre de documents pertinents}}$

- **La précision :**

La précision mesure la capacité du système à rejeter tous les documents non pertinents à une requête, elle est donné par :

Précision = $\frac{nombre\ de\ documents\ pertinents\ selectionnés}{nombre\ de\ documents\ selectionnés}$

La courbe de Précision-Rappel :

Dans le cas d'un système idéal, le taux de précision est égal au taux de rappel (100%), c'est-à-dire, que le système a retourné tous les documents pertinents au besoin de l'utilisateur et seulement les documents pertinents.

Cependant, on peut obtenir en général un taux de précision et de rappel aux alentours de 30%. Les deux métriques rappel et précision ne sont pas indépendantes. Il y a une forte relation entre elles: quand l'une augmente, l'autre diminue. Comme illustré dans la figure 1.3.

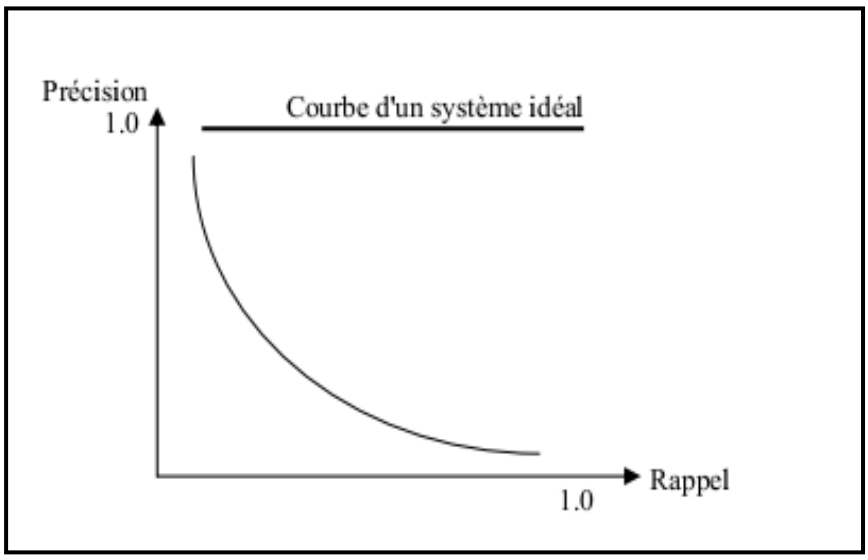


Figure 1.4 : Allure d'une courbe de rappel-précision [Baziz et al., 2005].

En plus du rappel et de la précision il existe d'autres mesures telles que :

- **Le bruit :**

La mesure d'évaluation bruit est une notion complémentaire à la précision, elle est défini par Bruit=1-Précision.

- **Le silence :**

La mesure d'évaluation silence est une notion complémentaire au rappel, elle est définie par Silence=1-Rappel.

- **Précision à k documents :**

La précision à k documents permet de mesurer la capacité d'un SRI à renvoyer les documents pertinents parmi les k premiers documents retournés en répondant à la requête q, sachant que la requête q admet k documents pertinents.

- **La précision moyenne :**

La précision moyenne (Mean Average Precision, dénotée MAP) mesure la capacité d'un SRI à pouvoir sélectionner les documents pertinents, en réponse à un ensemble de requêtes. Elle se calcule en deux étapes, la première étape consiste à calculer la précision moyenne pour une requête donnée, exprimée ainsi :

$AP_q = \frac{1}{N}\sum_{i=1}^{N} pr(d_i)$

Où N représente le nombre de documents pertinents pour la requête q et $pr(d_i)$ désigne la précision du $i^{ième}$ document pertinent calculée comme suit :

$$pr(d_i) = \begin{cases} \frac{r_{n_i}}{n_i} & \text{si } d_i \text{ est retrouvé} \\ 0 & \text{sinon} \end{cases}$$

Où n_i dénote le rang du document d_i et r_{n_i} le nombre de documents pertinents retrouvé au rang n_i

Dans la seconde étape, on calcule la précision moyenne pour un ensemble de requêtes de la manière suivante :

$MAP = \frac{1}{M}\sum_{j=1}^{M} AP_{qj}$

Où M représente le nombre de requête considérées et AP_{qj} la précision moyenne pour la requête j.

L'évaluation d'un SRI se fait à l'aide d'un corpus (ou collection) de test. Un corpus de test est constitué :

- d'un ensemble de documents(en nombre assez élevé).
- d'un ensemble de requêtes.
- d'une liste de documents pertinents pour chaque requête (jugement de l'utilisateur)

Le projet TREC(Text REtrieval Conference) est l'un des principaux fournisseurs des collections de test. Né des expérimentations du projet Cranfield dans les années de 1957 à 1967 [Cleverdon, 1967] et initié au tout début des années 90 par le NIST(National Institute of Standards and Technology) aux Etats-Unis, il est aujourd'hui co-sponsorisé par le NIST et le DARPA(Defense Advanced Research Projects Agency).La compagne TREC offre un cadre d'évaluation uniforme pour mesurer les performances et comparer les résultats des SRI. Elle est organisée annuellement et ses tâches changent d'une année à l'autre, mais elles reflètent bien les intérêts des chercheurs.

Conclusion :

Dans ce premier chapitre, nous avons présenté les différents concepts et notions de base de la recherche d'information. Nous avons aussi décrit les principales étapes d'un processus de RI, à savoir, l'indexation, l'appariement et la reformulation de la requête. Et enfin, nous avons présenté les modèles d'appariements existants dans la littérature.

Les différentes techniques et méthodes que nous avons abordé dans ce chapitre traitent des documents textuels plats qui sont considérés comme de simple sac de mots, Mais avec l'apparition de nouveaux standards de présentation des documents tel que le XML. Nous assistant a la prolifération des documents structurés ou semi-structurés qui apportent une autre information que le contenu textuel, il s'agit de la structure. D'où l'apparition de la recherche d'information structurée qui adapte les techniques de la recherche d'information classique pour la prise en compte de la dimension structurelle du document dans le processus de recherche. Le chapitre suivant sera donc consacré à la recherche d'information structurée.

Chapitre II : Recherche d’information semi-Structurée

Chapitre II : Recherche d'Information semi-Structurée

Introduction :

Un document représente le conteneur élémentaire de l'information exploitable par le système de recherche d'information. En ce sens, il est l'objet central du SRI. Lorsque celui-ci ne se compose que d'une suite de termes, on parle de document textuel plat. Dans le cas où le contenu textuel est structuré au travers de balises, on parle de document structuré ou semi-structuré.

La structuration des documents est rendue faisable grâce aux langages de balisage. De SGML (Standard Generalized Markup Language), au HTML (HyperText Markup Language), on arrive aujourd'hui à un format standard et universel d'échange de données connu sous le nom de XML (eXtensible Markup Language). Ce dernier langage fera l'objet de la première partie de notre chapitre.

L'évolution vers un nouveau type de documents, qui sont les documents structurés ou semi-structurés a nécessité le développement d'une nouvelle génération de SRI, qui permettent de prendre en compte la structuration du document, appelés systèmes de recherche d'information dans des documents structurés (SRIS).

La RI structurée soulève de nouvelles problématiques liées à la coexistence de l'information structurelle et de l'information de contenu, ce qui nécessite l'adaptation des techniques d'indexation, ainsi que des modèles d'appariement.

La deuxième partie de ce chapitre sera donc consacrée à la présentation des problématiques spécifiques de la RI structurée**,** ainsi que les différentes solutions proposées dans la littérature.

Ce chapitre est scindé en deux parties :

La première traite des documents XML, nous commencerons d'abord par définir les documents semi-structurés (section **I.1**) et la notion de structure (section **I.4**), nous présenterons ensuite la structure d'un document XML (section **I.6**) et la représentation graphique lui correspondante (section **I.8**).

Dans la seconde partie, nous présentons les différentes problématiques soulevées par la RI structurée (section **II.1**) et ses principales approches (section **II.3**), à savoir l'approche orientée données et l'approche orientée documents. Nous décrirons ensuite les techniques d'indexation des documents semi-structurés (section **II.4**), et les modèles d'appariement adaptés à la dimension structurelle de ces documents (section **II.7**). Nous aborderons enfin les approches utilisées pour l'évaluation des systèmes de recherche d'information structurée (section **II.8**).

Chapitre II : Recherche d'Information semi-Structurée

I. Présentation des documents semi-structurés : le langage XML

I.1. Documents semi-structurés :

Classiquement, un document semi-structuré est composé d'un ensemble de parties (le contenu), organisées de manière hiérarchique, aux quelles on peut associer des attributs externes. Un tel document est rédigé dans le but de communiquer une information aux lecteurs, tout en garantissant un degré de cohésion entre ces différentes parties [Mathias, 2002].

I.2. Langage XML :

Le langage XML dérive de SGML (Standard Generalized Markup Language) et de HTML (HyperText Markup Language). Comme ces derniers, il s'agit d'un langage orienté texte et formé de balises qui permettent d'organiser les données de manière structurée. Il est aussi bien utilisé pour le stockage de document que pour la transmission de données entre applications. Sa simplicité, sa flexibilité et ses possibilités d'extension ont permis de l'adapter à de multiples domaines.

I.3. Historique

L'ancêtre de XML est le langage SGML qui a été introduit en 1986 par C. Goldfarb. SGML a été conçu pour des documentations techniques de grande ampleur. Sa grande complexité a freiné son utilisation en dehors des projets de grande envergure. En 1991, T. Berners-Lee a défini le langage HTML pour le WEB. Ce langage est une version simplifiée à l'extrême de SGML, destinée à une utilisation très ciblée. XML est, en quelque sorte, intermédiaire entre SGML et HTML. Il évite les aspects les plus complexes de SGML tout en gardant suffisamment de souplesse pour une utilisation généraliste. La version 1.0 de XML a été publiée en 1998 par le consortium W3C[1] (World Wide Web Consortium). Une seconde version 1.1, qui est simplement une mise à jour pour les caractères spéciaux en lien avec Unicode, a, ensuite, été publiée en 2004. Ce standard a subi d'autre mises à jour qui ont donné naissances à d'autre version, nous citons : la version XML 1.1 deuxième édition publiée en 2006 et la version XML 1.0 cinquième édition publiée en 2008.

I.4. Notion de structure :

La structure des documents XML est définie par des balises encadrant les portions d'informations. Une balise (ou tag ou label) est une suite de caractères encadrés par "<" et ">", comme par exemple <nom_balise>. Un élément est une unité sémantique identifiée,

[1] Le World Wide Web Consortium, abrégé W3C, est un consortium fondé en octobre 1994 pour promouvoir la compatibilité des technologies du World Wide Web telles que HTML, XHTML, XML, RDF, CSS, PNG, SVG et SOAP. Le W3C n'émet pas des normes au sens européen, mais des recommandations à valeur de standards industriels.

formé d'une balise ouvrante, d'un contenu et de la balise fermante correspondante. La balise ouvrante prend la forme <nom_balise>. Quant à la balise fermante, elle prend la forme </nom_balise>.Le contenu d'un élément est formé de tout ce qui se trouve entre la balise ouvrante et la balise fermante [Chebili, 2011].

Comme par exemple : <ma_balise> texte </ma_balise>. De plus, les éléments peuvent être imbriqués, comme illustré dans l'exemple suivant :

```
< Document type = « livre »>
        <Titre> Introduction to modern information retrieval </Titre>
        <Auteur>
                <Nom>Gerard</Nom>
                <Prenom>Salton</Prenom>
        </Auteur>
        <Annee>1984</Annee>
</Document>
```

Figure 2.1 : Structuration des balises dans les documents XML.

Ainsi, un document XML peut se représenter sous la forme d'une arborescence d'éléments qui compte un seul élément racine (en anglais : document element) englobant tous les autres éléments.

Tous les éléments peuvent contenir un ou plusieurs attributs. Chaque élément ne peut contenir qu'une fois le même attribut. Un attribut est composé d'un nom et d'une valeur. Il ne peut être présent que dans la balise ouvrante de l'élément. Par exemple :

< ma_balise nom_attribut = valeur_ attribut > texte </ma_balise>.

I.5. Validation d'un document XML :

Pour qu'un document XML soit correct, il doit d'abord être bien formé et, ensuite, être valide. La première contrainte est de nature syntaxique. Un document bien formé doit respecter certaines règles syntaxiques propres à XML, qui compte principalement les suivantes :

- Il ne peut y exister qu'un seul élément racine qui englobe tous les autres éléments ;
- Pour chaque balise ouvrante doit correspondre une balise fermante ;
- Les balises doivent être bien imbriquées, et les chevauchements entre celles-ci sont interdits ;
- Le nommage des balises est libre, à condition qu'il commence par une lettre, tiret ou tiret bas et ne doit pas commencer par « XML ». Le nom peut ensuite être composé de lettres, chiffres, tirets, tirets bas et deux points, ainsi qu'il ne doit pas contenir de blancs. Si ce nom n'est constitué que d'un seul caractère, alors ce caractère doit être une lettre ;
- Les noms des éléments et des attributs sont sensibles à la casse ;

- Les valeurs d'attributs doivent être entre guillemets ;

La seconde contrainte est de nature structurelle. Un document valide doit respecter un modèle de document. Un tel modèle décrit de manière rigoureuse comment doit être organisé le document. Un modèle de documents peut être vu comme une grammaire pour des documents. Il existe plusieurs langages pour écrire des modèles de document. Nous retrouvons par ailleurs les DTD (Document Type Description), dont la syntaxe est héritée de SGML et les schémas XML dont la syntaxe est purement XML.

I.6. Structure d'un document XML :

Le langage XML est un format orienté texte. Un document XML peut être considéré comme une suite de caractères respectant quelques règles. Sa composition globale est immuable. Elle comprend toujours les constituants suivants :

- Prologue
- Corps du document (le contenu même du document).
- Commentaires et instructions de traitement :

Le document XML se découpe en fait en deux parties consécutives qui sont le prologue et le corps. Les commentaires et les instructions de traitement sont ensuite librement insérés avant, après et à l'intérieur du prologue et du corps.

La structure globale d'un document XML est la suivante :

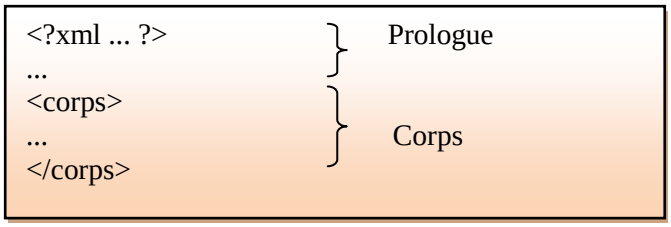


Figure 2.2 : Structure globale d'un document XML.

Le prologue contient deux déclarations facultatives mais fortement conseillées. La première déclaration est l'entête XML. Sa forme générale est :

<?xml version="..." encoding="..." standalone="..."?>.

L'attribut **version** précise la version d'XML utilisée. Les valeurs possibles actuellement sont 1.0 ou 1.1.le deuxième attribut, **encoding** , précise le codage des caractères utilisé dans le fichier. Les principales valeurs possibles sont US-ASCII, ISO-8859-1, UTF-8, et UTF-16.le dernier attribut, **standalone**, indique l'existence ou non d'une déclaration de type de document extérieure a celui-ci, les valeurs possible sont « no » ou « yes ».

La seconde déclaration est la déclaration du type du document (DTD) qui définit la structure du document. La référence à la DTD externe doit être placée au début du fichier.

Elle a la forme générale suivante, qui utilise le mot clé DOCTYPE : <!DOCTYPE ... >, On peut enrichir la DTD externe avec des déclarations locales comme on peut se contenter des déclarations locales et définir toutes les balises dans le document XML. Voici ci-dessous un exemple d'un document XML et la DTD (externe) qui lui correspond :

```
<?xml version= "1.0 " ?>
<!-- Exemple de fichier XML décrivant un article scientifique -->
<article annee="2003">
<en-tete>
    <titre> Recherche d'information sur le web : la grande révolution <titre>
    <auteur> André Dupont </auteur>
</en-tete>
<corps>
    <section>
        <sous-titre> Histoire de l'hypertexte : des pères fondateurs au World
        Wide Web </sous-titre>
        <par>Afin de maîtriser les enjeux des systèmes hypertexte,
        il convient, même si c'est une tâche ardue, d'essayer de les définir…
        </par>
    </section>
    <section>
        <sous-titre>Moteur de recherche</sous-titre>
        <par>On distingue plusieurs types de moteurs de recherche…</par>
        <par>Les annuaires…</par>
        <par>Les moteurs de recherche plein-texte…</par>
        <par>Les méta-moteur…</par>
    </section>
    <section>
        <sous-titre>L'analyse des liens</sous-titre>
        <par>…</par>
    </section>
</corps>
</article>
```

Figure 2.3 : Exemple de fichier XML article.xml

```
<?xml version= "1.0 " ?>
<!—DTD pour l'exemple d'article -->
< !ELEMENT article(en-tete, corps)>
< !ATTLIST article
        Annee CDATA #REQUIRED
>
< !ELEMENT en-tete(auteur+, titre>
< !ELEMENT auteur(#PCDATA)>
< !ELEMENT titre(#PCDATA)>
< !ELEMENT corps(section+)>
< !ELEMENT section(sous-titre, par>
< !ELEMENT sous-titre(#PCDATA)>
< !ELEMENT par(#PCDATA)>
```

Figure 2.4 : DTD correspondante à article.xml

Le corps du document est constitué de son contenu qui est organisé de façon hiérarchique. L'unité de cette organisation est l'élément. Chaque élément peut contenir du texte simple, d'autres éléments ou encore un mélange des deux, comme il peut être vide. Un élément vide est uniquement constitué d'une balise spécifique, dite « auto-fermante » dont la syntaxe est la suivante : <nom_balise/>.

La position des termes au niveau des imbrications permet de différencier les documents semi-structurés des documents structurés. Les documents semi-structurés tolèrent que des termes soient présents dans une balise qui contient elle-même une autre balise (on parle de contenu mixte) tandis que les documents structurés ne le permettent pas. On considère donc XML comme un format de documents semi-structurés.

Les commentaires en XML se déclarent de la même façon qu'en HTML, Ils commencent donc par "<!-- "et se terminent par "-->". Ils peuvent être placés à n'importe quel endroit tant qu'ils se trouvent à l'extérieur des balises et non imbriqués dans un autre commentaire.

Les instructions de traitement sont délimitées par les chaînes de caractères '<?' et '?>'. Ces instructions sont interprétées par l'application servant à traiter le document XML. Elles ne font pas totalement partie du document.

I.7. Importance de la technologie XML :

Le succès de XML est en grande partie du à ses qualités, parmi lesquels nous pouvons citer les suivants :

Séparation stricte entre contenu et présentation : Il faut bien distinguer le contenu d'un document et la présentation qui en est donnée. Un des premiers principes de XML est d'organiser le contenu de manière indépendante de la présentation. La règle est alors de choisir les balises pour organiser le document en privilégiant la structure de celui-ci par rapport à une éventuelle présentation.

Forte structuration : Une des particularités de XML est la structuration forte du document. Les balises dans un document ont un rôle sémantique. Ces dernières structurent les données textuelles et ajoutent ainsi de l'information, ce qui facilite le traitement de ces données.

Format texte avec gestion des caractères spéciaux : Un des atouts d'XML est sa prise en charge native des caractères spéciaux grâce à Unicode. De plus, on peut d'utiliser les différents codages (UTF-8, UTF-16, …) possibles puisque l'entête d'un document spécifie le codage.

Extensibilité et flexibilité : Tout type de données peut être décrit par XML pourvu que l'on fournisse une grammaire de la structure, il appartient donc, aux auteurs de documents de fixer les balises utilisées. Cette liberté dans les noms de balises permet de définir des vocabulaires particuliers adaptés aux différentes applications. Ces vocabulaires particuliers sont appelés dialectes XML. Il en existe des centaines voire des milliers pour couvrir tous les champs d'application de XML.

I.8. Représentation graphique d'un document XML:

L'organisation hiérarchique des éléments structurels des documents XML nous permet de représenter ces derniers sous la forme d'un arbre. Cette représentation arborescente correspond au modèle DOM.

DOM (Document Object Model, ou modèle objet de document) est une spécification du W3C [W3C, 2005]. Définissant la structure d'un document sous forme d'une hiérarchie d'objets (arbre) reliés entre eux, où chaque objet représente un atome du document XML.

La figure 2.5 schématise un extrait d'un document XML et l'arbre DOM lui correspondant. Ce type arbre se compose d'un seul nœud racine modélisant l'élément principal du document qui englobant tous les autres. D'un ensemble de nœuds internes qui correspondent à des éléments ou des attributs et enfin de nœuds feuilles qui réfèrent au contenu des éléments ou à la valeur des attributs. Quant aux arcs, ils reflètent la relation d'inclusion d'un élément dans un autre.

```
<Doc annee= ″2003 ″ >
    <Corps>
        <Section>
            <Titre>Recherche d'information traditionnelle</Titre>
            <Par>Le domaine de la recherche d'information…</Par>
        </Section>
        <Section>
            <Titre>Intelligence artificielle </Titre>
            <Par>Ce domaine…</Par>
        </Section>
    </Corps>
</Doc>
```

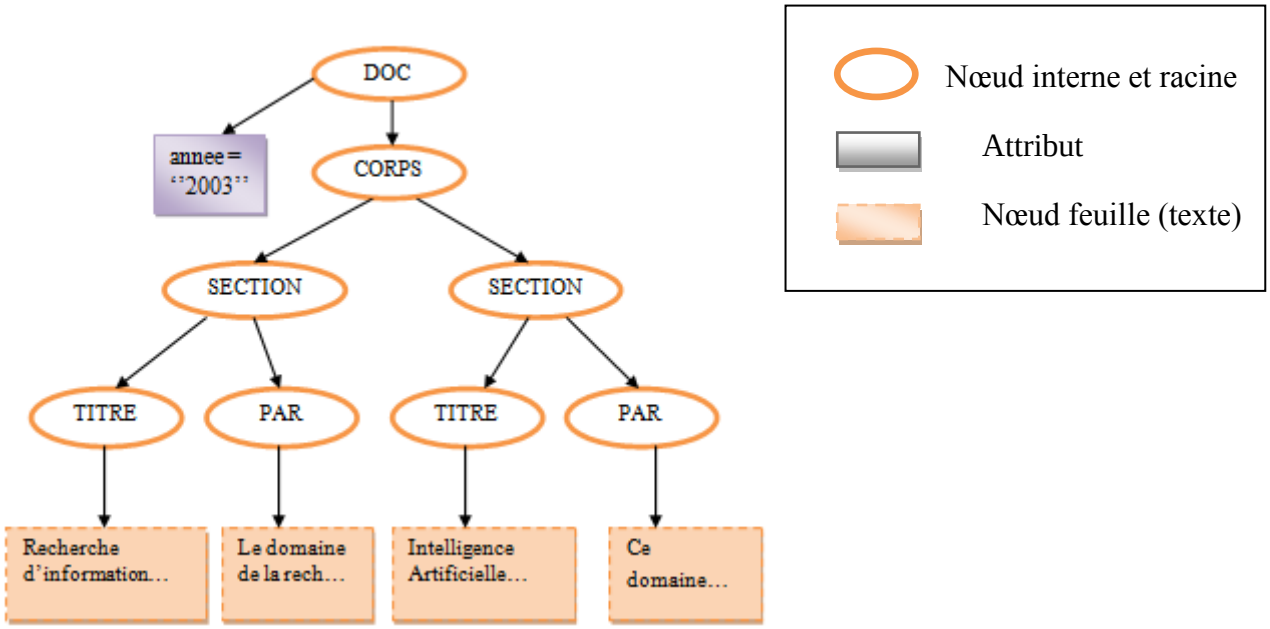


Figure 2.5 : Représentation arborescente (DOM) d'un extrait d'un document XML.

II. Recherche d'information structurée :

II.1. Problématiques spécifiques à la RI structurée :

Notre travail s'articule autour de la recherche d'information dans les documents XML semi structurés, l'utilisation de l'expression RI structurée n'est que par abus de langage.

La RI structurée soulève plusieurs problématiques relatives à chaque phase du processus de recherche dues à la dimension structurelle des documents dont elle traite.

Les problématiques au niveau de l'indexation peuvent se résumer en les questions suivantes : quelle unité doit-on indexer de la structure des documents ? Comment relier cette structure au contenu même du document ? En fonction de quelle dimension (niveau élément, documents, collection) doit-on pondérer les termes d'indexation ?

Au niveau de l'interrogation des documents, il s'agit d'offrir à l'utilisateur la possibilité d'exprimer des besoins diversifiés concernant le contenu des documents et/ou la structure, et ce de manière simple.

Enfin, au niveau des modèles de recherche, les systèmes de recherche doivent pouvoir décider de la granularité de l'information à renvoyer en réponse à une requête orientée contenu seulement ou une requête orientée contenu et structure

II.2. Granularité de l'information :

Contrairement à la RI traditionnelle, qui permet de renvoyer des résultats de recherche aux utilisateurs sous forme d'une liste de documents, la RI adaptée aux documents semi-structurés, qui permettent le balisage des contenus des documents, traite l'information avec une granularité plus fine. Le but des SRI dans les documents semi-structurés est alors d'identifier les parties des documents les plus pertinentes à une requête donnée. Ceci change le concept de granule (unité d'information) renvoyé à l'utilisateur.

Il est possible de distinguer trois différents types d'unités d'information :

- Le document entier : le système renvoie le document complet.
- L'élément (le nœud) : il peut s'agir de n'importe quel élément structurel du document.
- Un ensemble d'éléments : dans ce cas on procède à une recomposition d'information et le système renvoie un ensemble d'éléments organisés comme un document structuré.

Le but des SRI dans le contexte des documents structurés est alors de renvoyer des unités d'information auto-explicatives à l'utilisateur (élément ou ensemble d'éléments), et non des documents complets.

II.3. Principales stratégies en recherche d'information structurée :

La recherche d'information structurée compte deux approches principales qui sont [Fuhr et al., 2003] : l'approche orientée données et l'approche orientée documents. Chacune de ces approches définit des méthodes qui lui sont spécifiques pour l'indexation, l'interrogation, la recherche et le tri des documents XML.

L'approche orientée données : Le but est de développer des modèles de données permettant la représentation et l'interrogation en tenant compte à la fois du contenu et de la structure. Dans ce cas, les documents XML sont considères comme une base de données, dont les champs correspondraient aux éléments et attributs définis dans la DTD (ou le schéma) des documents. Elle utilise des techniques développées par la communauté des bases de données.
Au niveau de l'indexation, la communauté BD stocke toutes les informations textuelles et structurelles des documents au sein de tables dans une base de données. Le problème posé par cette approche se situe au niveau de la recherche sur le contenu textuelle des documents, puisque ce dernier est indexé entant que chaine de caractère, et non sous forme de termes indépendants. Des langages de requêtes associés ont été proposés par la communauté BD, leur syntaxe est généralement liés à celle du langage SQL, et permettent de spécifier des conditions sur la structure des documents. Au niveau de l'appariement, la pertinence est généralement calculée d'une manière booléenne stricte. De ce fait, seuls les éléments qui répondent exactement à la requête sont renvoyés.

L'approche orientée documents : Elle se focalise sur des applications considérant les documents structurés d'une manière traditionnelle, c'est à dire que les balises servent uniquement à décrire la structure logique[2] des documents. Cette approche a quant à elle été prise en charge par la communauté de la recherche d'information. Les mêmes techniques d'extraction des termes et d'indexation que de la RI classique sont maintenues pour l'indexation de l'information textuelle. D'autres approches spécifiques sont développées pour indexer l'information structurelle. Quant aux langages de requêtes, ils restent beaucoup plus simples que ceux proposés en BD en se rapprochant du langage naturel avec une extension pour exprimer les contraintes structurelles. Au niveau de l'appariement, un degré de pertinence entre la requête et les unités d'informations est évalué et un score de pertinence est attribué à ces derniers. Ce qui permet de sélectionner les unités d'informations qui répondent le mieux au besoin de l'utilisateur, et ensuite de les trier.

[2] La structure logique définit une organisation hiérarchique des données du document c'est-à-dire une organisation de l'information. Cette organisation s'établit autour d'abstractions représentant des parties du document: un document est composé d'un titre, d'une ou plusieurs sections, elles mêmes sont composées d'un titre, d'une ou plusieurs sous-sections, etc.

Le Tableau 2.1 récapitule les spécificités et les enjeux de la RI semi-structurée. Nous montrons les spécificités de la RI semi-structurée par sa comparaison avec la RI classique et la recherche des données (dans le sens de Bases de données).

	RI classique	***RI semi-structurée***	***Recherche de données***
Type de document	documents non structurés : documents plats	documents semi-structurés	documents structurés
Contenu	texte seulement	texte + structure	structure + données
Unité de recherche	document	élément	tuple
Type de requête	contenu	contenu et/ou structure	contenu et/ou structure
Langage de requête	langage libre	langages structurés	langages structurés : SQL
Interprétation	vague	vague	stricte

Tableau 2.1 : Récapitulatif de quelques caractéristiques de la RIS [Harrathi, 2010b].

D'une manière générale, les solutions proposées par la communauté de la RI peuvent être utilisées comme « surcouche » aux solutions orientées BD. Cette surcouche sert essentiellement à intégrer la notion de pertinence dans la recherche, en complétant les approches proposées par la communauté de BD pour le stockage et l'interrogation des documents [Sauvagnat, 2005].

II.4. L'indexation des documents semi-structurés :

La phase d'indexation est défini, par la communauté de la recherche d'information, comme étant un processus visant à identifier les descripteurs ou termes représentatifs du contenu du document. Cette définition reste valable pour le traitement des documents semi-structurés. Ainsi, la façon la plus simple d'indexer ces documents est de les considérer comme des fichiers plats. Dans ce cas aucune recherche sur la structure n'est plus possible et les documents existent uniquement dans leur intégralité.

D'où la nécessité d'adopter des schémas d'indexation qui devraient couvrir les deux aspects : contenu et structure. Ce qui permettra par ailleurs la reconstruction du document décomposé dans les structures de stockage, le traitement des expressions de chemin sur la structure, l'accélération de la navigation dans des documents, le traitement de prédicats vagues et précis sur le contenu de documents et la recherche par mots clés

Deux problématiques principales doivent alors être considérées :

– Trouver un moyen d'identifier et de stocker les descripteurs structurels,
– Lier efficacement les informations de structure avec les informations de contenu.

L'indexation d'un document semi-structuré doit ainsi passer par deux étapes qui sont : l'indexation du contenu et l'indexation de la structure. Bien que l'indexation des informations de contenu et des informations structurelles soient étroitement liées. Nous avons pris l'initiative de les aborder chacune à part.

II.4.1. L'indexation du contenu :

La problématique d'indexation de l'information textuelle reste d'actualité dans la RI structurée. Les approches orientées BD considèrent le contenu des nœuds feuilles comme étant l'unité textuelle d'indexation, alors que les approches orientées RI considèrent le terme.

Cependant, quelque soit l'unité textuelle choisie pour l'indexation, l'indexation du contenu ne doit pas se faire indépendamment de la structure du document car un terme figure à un emplacement précis dans le document et son importance est relative à cet emplacement (c'est-à-dire le nœud) ce qui n'est pas le cas pour la recherche d'information traditionnelle. On est alors confronté à deux problèmes : le premier est celui de la portée des termes et le second celui de l'importance des termes dans le document (la pondération).

II.4.1.1. Portée des termes d'indexation :

La portée des termes dans un document semi-structuré permet de le relier aux différents nœuds du document et de le considérer discriminant pour ceux la. La question est alors comment relier les termes de l'index à l'information structurelle du document, deux approches ont été proposées : une qui procède de manière à agréger le contenu des nœuds et une seconde qui indexe tous les contenus des nœuds séparément. Ces deux solutions correspondent aux approches d'indexation dites des sous-arbres imbriqués et des unités disjointes [Sauvagnat, 2005].

A. Sous-arbres imbriqués :

Dans cette approche, le texte complet de chaque nœud d'index est considéré comme un document atomique et propagent donc les termes des nœuds feuilles dans l'arbre des documents. Autrement dit, lorsqu'un terme indexe un nœud feuille il indexe aussi les nœuds internes ancêtres[3] de celui-ci. Cette approche permet ainsi d'indexer tous les sous-arbres des documents, Cependant l'index résultat contiendra de nombreuses informations redondantes. Un exemple de l'indexation des sous-arbres imbriqués est illustré dans la figure suivante :

[3] Un élément ou nœud XML est ancêtre d'un autre s'il contient celui-ci, quel que soit le nombre d'éléments ou nœuds situés entre eux.

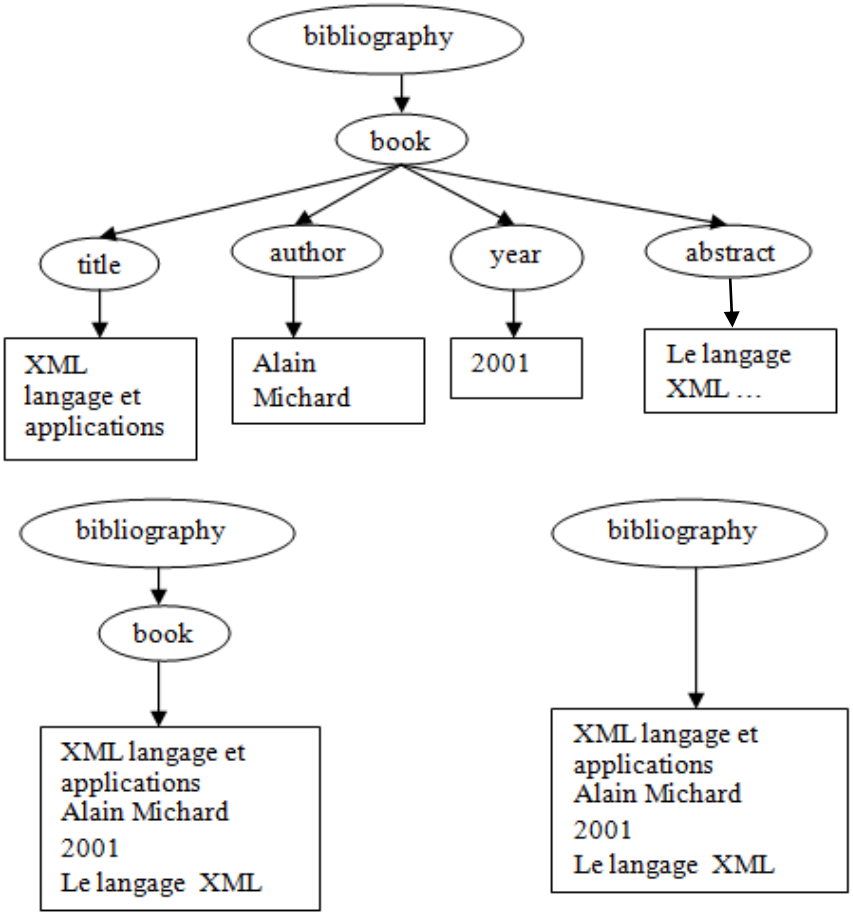


Figure 2.6 : Indexation de sous-arbres imbriqués.

Les termes « Alain Michard» sont d'abord associés au nœud /bibliography/book/author. Puis sont propagées, au nœud parent[4], /bibliography/book. Et enfin au nœud du niveau supérieur, /bibliography.

B. **Unités disjointes :**

Dans cette approche, le document est décomposé en unités disjointes liées par leurs relations hiérarchiques, ce qui permet de palier à l'inconvénient de redondance de l'information dans l'index, de la manière suivante :

- Les termes des nœuds feuilles sont uniquement reliés au nœud parent qui les contient. formant ainsi des ensembles de termes disjoints.
- Les autres nœuds internes, quant à eux ne seront reliés à aucun terme.

De cette façon, le texte de chaque nœud de l'index est l'union d'une ou plusieurs parties disjointes.

[4] Un élément ou nœud XML est parent d'un autre s'il contient celui-ci de façon directe, sans élément ou nœud situé entre eux.

L'indexation des unités disjointes de l'exemple précédent sera ainsi faite : les termes "XML langage et applications " seront uniquement reliés au nœud /bibliography/book/title, les termes "Alain Michard" au nœud /bibliography/book/author, le terme "2001" au nœud /bibliography/book/year et les termes "Le langage XML " au nœud /bibliography/book/ abstract. Les nœuds /bibliography/book et /bibliography ne sont quant à eux reliés à aucun terme.

II.4.2. L'indexation de l'information structurelle :

Les documents semi-structurés ajoutent une nouvelle métrique au processus de l'indexation qui est la structure. Différentes approches ont été proposées dans la littérature pour indexer l'information structurelle selon des granularités variées [Luk et al., 2002]. On distingue trois types d'approches pour l'indexation de l'information structurelle :

A. **Indexation basée sur des champs** :

Dans cette approche, un document est représenté comme un ensemble de champs (par exemple : titre, auteur, abstract) ,extraits généralement de la DTD ou du schéma XML lui correspondant, et du contenu associé à ces champs. Pour permettre une recherche restreinte à certains champs, les termes de l'index sont construits en combinant le nom du champ avec les termes du contenu comme illustré dans la figure suivante :

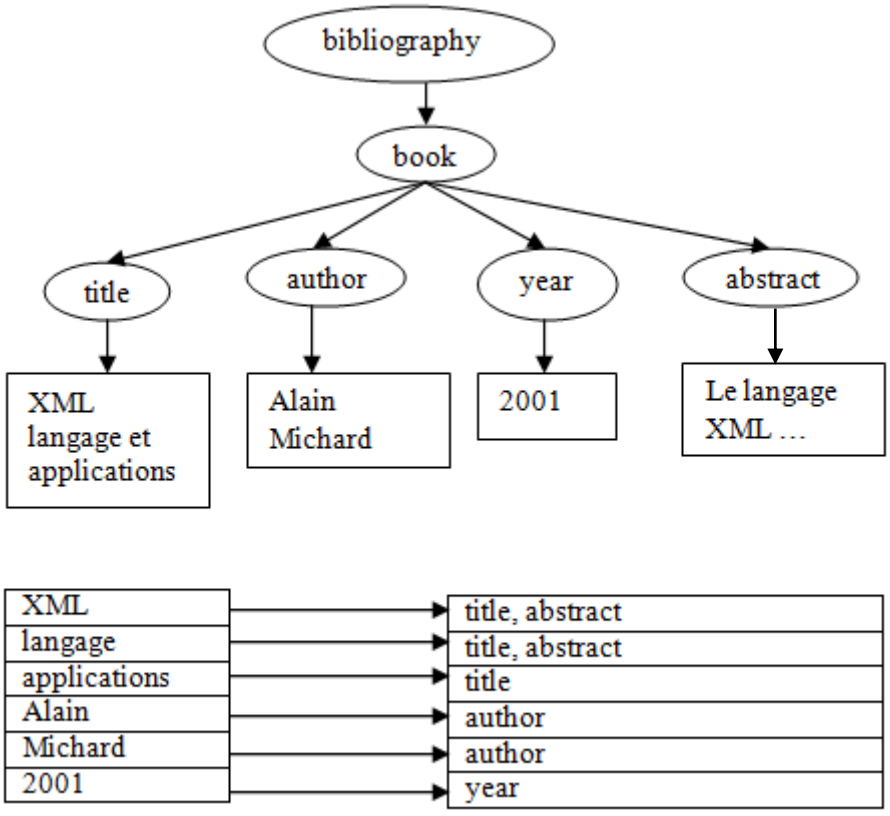


Figure 2.7 : Exemple d'indexation basée sur des champs.

On obtient ainsi des index élémentaires qui sont basés sur les champs, ignorant la notion de chemin ou de concepts structurels plus élevés. Le point fort de ces index est qu'ils n'occupent pas beaucoup d'espace en mémoire, cependant la réponse à des requêtes de structure avec les index élémentaires implique la reconstruction du chemin grâce à un ensemble de procédures de jointure assez compliquées.

B. Indexation basée sur des chemins :

Les techniques basées sur les chemins utilisent les chemins[5] de document entiers au lieu de nœuds comme unité de structure de base. Les index de chemins donnent pour chaque valeur répertoriée d'un chemin de balises la liste des documents répondants contenant un élément atteignable par ce chemin et ayant cette valeur, comme l'illustre la figure suivante :

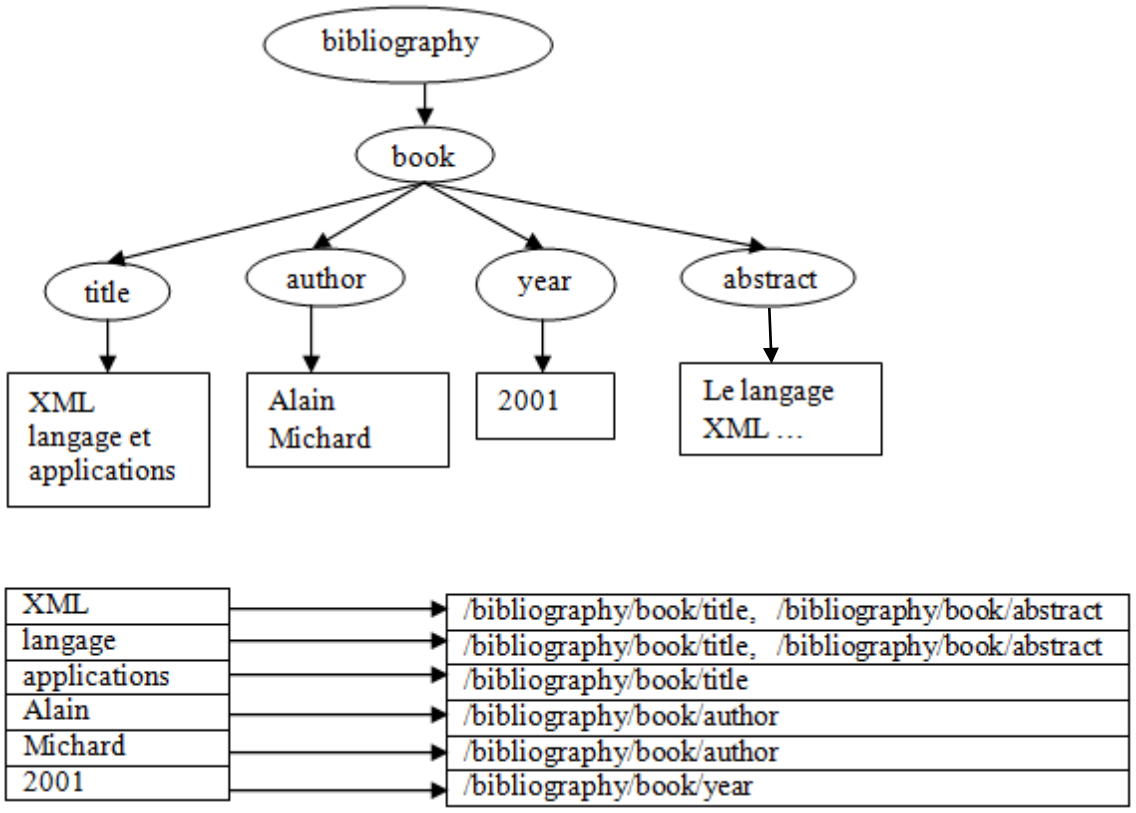


Figure 2.8 : Exemple d'indexation basée sur des chemins.

Cette technique facilite la navigation dans les documents en permettant la résolution des expressions XPath, elle permet aussi de retrouver des documents ayant des valeurs connues

[5] Imbrication des labels d'éléments de la racine jusqu'aux feuilles

pour certains éléments ou attributs. Cependant, elle souffre de la difficulté de retrouver les relations ancêtre-descendant entre les différents nœuds des documents.

C. Indexation basée sur des arbres :

Dans cette approche, chaque nœud de l'arbre est référencé grâce à un identifiant unique. Les termes sont donc associes à cet identifiant, ce qui permet de localiser de façon précise l'endroit ou ces termes sont apparus et de retrouver les relations hiérarchiques entre les éléments. Ces techniques basées sur des graphes sont les seules à pouvoir répondre aux requêtes sous forme d'arbre sans une perte de temps dans les jointures de multiples recherches. La figure suivante donne un exemple d'indexation basée sur les arbres :

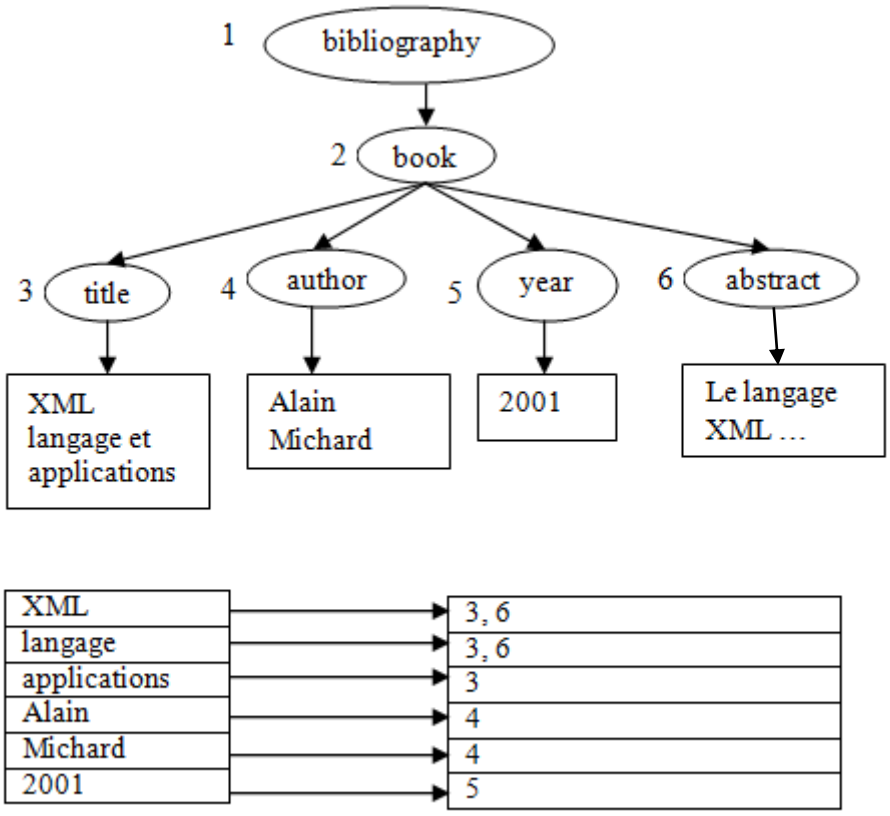
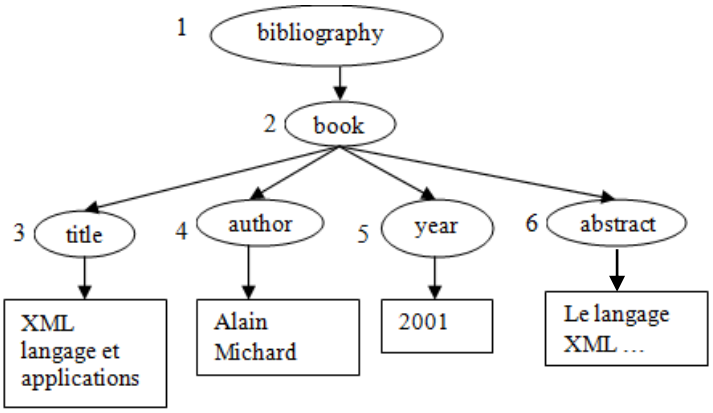


Figure 2.9 : Exemple d'indexation basée sur des arbres.

Cette approche a été adoptée dans plusieurs systèmes de recherche, parmi lesquels nous citons : EDGE et BINARY [Florescu, 1999].

Dans le système EDGE, la structure d'un document est stockée dans deux tables. La première appelée « Table EDGE » stocke pour chaque arc, l'identifiant du nœud source et cible, l'ordre d'apparition des nœuds, le nom du nœud cible et le type du nœud cible (interne ou feuille). La deuxième table appelée « Table Value String » stocke la valeur des nœuds feuilles. Comme illustré dans la figure suivante



source	ordinal	name	datatype	target
NULL	1	bibliography	ref	1
1	1	book	ref	2
2	1	title	ref	3
3	1	#texte	string	v1
2	2	author	ref	4
4	1	#texte	string	v2
2	3	year	ref	5
5	1	#texte	string	v3
2	4	abstract	ref	6
6	1	#texte	string	v4

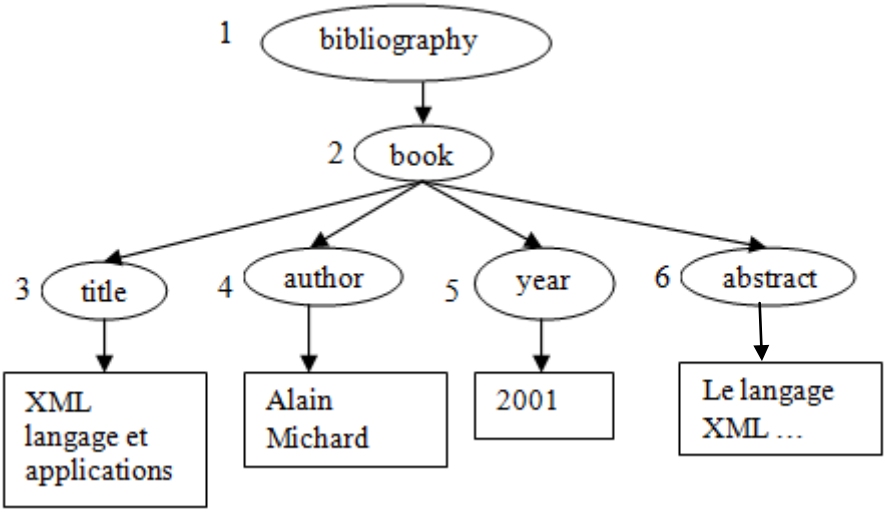
Table EDGE

Vid	value
v1	XML langage et applications
v2	Alain Michard
v3	2001
v4	Le langage XML

Table Value String

Figure 2.10 : Indexation d'un document XML avec l'approche EDGE.

De la même manière le système BINARY utilise des tables pour stocker la structure du document, cependant le nombre de tables accroit et dépend du nombre d'éléments distincts de l'arbre. Ainsi à chaque élément name correspond une table Bname. En d'autres termes, BINARY réalise une partition horizontale de la table EDGE en utilisant le nom de l'élément comme critère de partition, comme schématisé ci-dessous :



source	ordinal	name	datatype	target
NULL	1	bibliography	ref	1

B bibliography

source	ordinal	name	datatype	target
1	1	book	ref	2

B book

source	ordinal	name	datatype	target
2	1	title	ref	3

B title

source	ordinal	name	datatype	target
2	2	author	ref	4

B author

source	ordinal	name	datatype	target
2	3	year	ref	5

B year

source	ordinal	name	datatype	target
2	4	abstract	ref	6

B abstract

source	ordinal	name	datatype	target
3	1	#texte	string	v1
4	1	#texte	string	v2
5	1	#texte	string	v3
6	1	#texte	string	v4

Vid	value
v1	XML langage et applications
v2	Alain Michard
v3	2001
v4	Le langage XML

Table Value String

Figure 2.11 : Indexation d'un document XML avec l'approche BINARY.

Par comparaison à l'approche EDGE, les tables ont une taille moins importante, mais le nombre de jointures nécessaires pour des requêtes de chemin est important. Néanmoins cette approche s'est avérée très efficace pour des recherches sur un élément particulier.

II.5. Pondération des termes :

La pondération est une étape primordiale dans le processus d'indexation, elle permet d'affecter à chaque terme un poids. Alors qu'en recherche d'information classique, le poids d'un terme reflète son importance de manière locale au sein du document et de manière

globale au sein de la collection, s'ajoute en recherche d'information structurée l'importance du terme au niveau de l'élément.

De nouvelles formules permettant d'évaluer l'importance des termes au sein de l'élément, du document et de la collection s'avèrent plus appropriées, IDF (Inverse Document Frequency) utilisé en RI traditionnelle a été adapté pour la RI structurée sous le nom d'IEF (Inverse Element Frequency). Ainsi la formuleTF.IDF a été remplacée par TF.ITDF (Term Frequency-Inverse Tag and Document Frequency).

De nombreux auteurs ont proposés l'adaptation des formules de pondération traditionnellement utilisées en RI à la RIS.

Dans [Zargayouna, 2004], le calcul du poids des termes est influencé par le contexte (l'unité d'indexation) dans lequel ils apparaissent. Ce calcul de poids s'inspire de la méthode du TF-IDF qu'on applique aux balises. Ainsi la mesure TF-ITDF (T erm Frequency-Inverse T ag and Document Frequency), est définie de la manière suivante :

Soit T l'ensemble de tous les termes qui figurent dans le corpus,

B l'ensemble de tous les modèles de balises.

D l'ensemble de tous les documents du corpus.

TF-ITDF(t,b,d)= TF(t,b,d)*ITF(t,d)*IDF(t,b)

IDF(t,b) = log ($\frac{|B|_d}{TagF(t,b)}$) ITF = log ($\frac{|D|_b}{DF(t,b)}$)

Où

$|D|_b$ est le nombre total de documents où le modèle de balise b est présent dans leur structure

$|B|_d$ est le nombre total de balises dans le document d.

DF(t,b) : (Document Frequency) est le nombre de documents qui contiennent la balise b et dans laquelle le mot t apparaît au moins une fois.

TagF(t,b): (Tag Frequency) est le nombre de balises dans le document d et dans lesquelles le mot apparaît au moins une fois.

Cette formule permet de calculer la force discriminatoire d'un terme t pour une balise b relative à un document d

II.6. Interrogation des documents semi-structurés :

Dans la recherche des documents semi-structurés (documents XML), on distingue deux types de requêtes utilisateur :

- Requête orientée contenu : cette requête porte sur le contenu seul des unités d'information. Généralement la requête est exprimée par de simples mots clés et

c'est le SRI qui sélectionne l'unité d'information ou l'élément XML à renvoyer à l'utilisateur.

- Requête orientée contenu et structure : dans ce type de requête, l'utilisateur peut spécifier des contraintes structurelles pour indiquer le type des éléments à renvoyer par le système (on suppose donc que l'utilisateur à une connaissance au moins partielle de la collection qu'il interroge).

Les requêtes orientées structure seulement ne sont pas prises en compte par la communauté de recherche d'information. En effet ce type de requête correspond à une interrogation de type « base de données ».

II.7. Modèles d'appariement :

La majorité des approches orientées RI présentées dans la littérature sont des adaptations des modèles traditionnels. Les différents types de modèles : booléen, vectoriel et , probabiliste sont étendus de diverses façons pour tenir compte de l'information structurelle. On ajoute ainsi des paramètres supplémentaires pour ajuster les formules classiques comme le nombre d'enfants d'un élément, son type, la fréquence de ce type d'élément dans la collection et l'importance d'un terme dans les autres éléments du même type.

Dans ce qui suit, nous présentons les principales extensions faites sur les modèles de RI classique pour les adapter à la recherche dans les documents semi-structurés

II.7.1. Modèle booléen :

Une approche d'extension de ce modèle grâce au p-normes est proposé par dans [Hatano et al., 2002]. Dans cette approche seules les requêtes orientées contenu sont considérées, quant à l'unité d'information retournée, elle est définie comme tout élément contenant au moins un élément feuille. Pour ce faire, chaque élément feuille e_j(j=1,…N) est représenté par un vecteur : F(e_j)= $(w_{t_1}^{e_j}, w_{t_2}^{e_j}, \ldots, w_{t_n}^{e_j})$

Où $w_{t_i}^{e_j}$ représente le poids du terme t_i dans le nœud e_j.

La similarité entre un nœud feuille e_j (représenté par le vecteur F(e_j)) et une requête q(représenté par un vecteur dont les composantes sont 1 si le terme de la collection apparait dans la requête, 0 sinon) est calculée grâce au cosinus :

RSV (f(ej),q)=$\frac{f(e_j).q}{|f(e_j)||q|}$

Ils utilisent ensuite la p-norme[6] pour calculer le score final d'un élément en propageant les scores depuis les feuilles de l'arbre documentaire jusqu'à l'élément considéré, par la formule suivante:

$$RSV(q, e) = 1-\sqrt[\frac{1}{p}]{\frac{1}{|enf(e)|}\sum_{é \epsilon enf(e)}(1 - RSV(q, é))^p}$$

Où enf(e) représente les sous éléments de e.

II.7.2. Modèle vectoriel :

Les approches issues du modèle vectoriel représentent les éléments sous forme de vecteurs (obtenu par indexation des sous arbres imbriqués) de termes pondérés, puis calculent le degré de similarité entre chaque élément et la requête, et enfin, renvoient les éléments par ordre décroissant de leur pertinence à la requête. Nous présenterons dans ce qui suit les adaptations les plus importantes.

Une des premières adaptations du modèle vectoriel est proposé par Fuller dans [Fuller et al., 1993]. La formule proposé pou calculer la similarité d'un nœud n à une requête q={ t_1,t_2,..t_T} est la suivante :

$$Sim\ (q, n)= \alpha(T).cosm(q, n)+\sum_{k=1}^{s} \frac{cosm(q,n_k)}{\beta^{k-1}}$$

Où :

α (T) : Facteur représentant le type du nœud.

s : le nombre de nœuds descendants nk de n.

β : Paramètre permettant d'assurer que le nombre d'enfants n'introduit pas un biais dans la formule

La fonction cosm est définie de la manière suivante :

$$Cosm(q, n)= \sum_{i=1}^{T} \frac{w_i^q * w_i^n}{|n|}$$

Avec :
w_i^q: le poids du terme ti dans la requête q.

w_i^n : le poids du terme ti dans le nœud n.

|n| :le nombre de termes dans le nœud n.

[6] La norme est une extension de la valeur absolue des nombres aux vecteurs. Elle permet de mesurer la longueur commune à toutes les représentations d'un vecteur dans un espace.
Exemple : on note x un vecteur (x_1,…….x_n) de n composantes.
Sa norme p est donnée par la formule suivante : $\|x\|_p = (|x_1|^p + \cdots + |x_n|^p)^{\frac{1}{p}}$

Ce modèle peut être généralisé pour permettre le traitement des requêtes orientées contenu et structure. Par son application de manière récursive à chaque sous-arbre de la hiérarchie pour ensuite effectuer un agrégat des scores.

L'un des plus important modèles d'adaptation est ce lui proposé dans [Mass et al., 2002] [Mass et al., 2003] , le modèle JuruXML effectue une indexation d'élément par type (un index pour chaque type d'élément), et applique ensuite le modèle vectoriel pour la pondération des éléments. Il traite les deux types de requêtes les requêtes orientées contenu et les requête orientées contenu et structure.

Dans le premier cas, la requête est évaluée sur chacun des index et les résultats, qui ont été normalisés, sont ensuite fusionnés afin de fournir à l'utilisateur une liste unique de résultats.

Dans le deuxième cas, la requête q est d'abord décomposée en un ensemble de conditions de la forme (chemin c_i^q, terme t_i). On calcule ensuite pour un élément donné e une correspondance vague entre c_i^e le XPath du terme t_i dans e et c_i^q la condition de chemin pour le terme t_i dans q par la fonction suivante :

$$cr(c_i^q, c_i^e) = \begin{cases} \frac{1+|c_i^q|}{1+|c_i^e|} & \text{si } c_i^q \text{est une sous} - \text{séquence de } c_i^e \\ 0 & \text{sinon} \end{cases}$$

Par exemple [Sauvagnat, 2005], cr(article/bibl, article/bm/bib/bibl/bb) = 3/6 = 0.5.

La correspondance entre une requête q et un élément e est donnée par la formule suivant :

$$RSV(e, q) = \frac{\sum_{(t,c_i^q)\epsilon q} \sum_{(t,c_i^e)\epsilon e} w_q(t) * w_e(t) * cr(c_i^q, c_i^e)}{|q||e|}$$

Où :

$w_q(t)$,$w_e(t)$: représente respectivement les poids du terme t dans q et e.

|q| : le nombre de termes dans q.

|e| : le nombre de termes dans e.

II.7.3. Modèle probabiliste :

Kamps et al ont proposé dans [Kamps et al., 2004] une approche basée sur les modèles de langage pour traiter les requêtes orientées contenu. Dans sa forme classique ce modèle fait correspondre un modèle de langue pour chaque document. Dans l'approche structurée, il fait correspondre un modèle de langue pour chaque élément e de la collection. Puis, pour chaque requête q, les éléments sont triés par apport à la probabilité que le modèle de langue de l'élément génère la requête. Ceci revient à estimer la probabilité P(e,q) comme suit :

$P(e,q) = P(e) \cdot P(q|e)$

Où : P(e) : la probabilité à priori de l'élément e

P(q| e) : la probabilité que l'élément e génère la requête q.

La probabilité P(q|e) est calculée en supposant l'indépendance entre les termes de la requête $q=(t_1,t_2,\ldots t_n)$ et en utilisant une interpolation linéaire du modèle de l'élément e et celui de la collection, comme suit :

$P(t_1,\ldots t_n \mid e) = \prod_{i=1}^{n}(\lambda . P(t_i \mid e) + (1-\lambda).P(t_i))$

Où : $P(t_i \mid e)$: la probabilité d'observer le terme t_i dans l'élément.

$P(t_i)$: la probabilité d'observer le terme t_i dans la collection.

λ : paramètre de lissage entre le modèle de l'élément e et celui de la collection.

Une approche basée sur les réseaux bayésiens est proposée dans [Piwowarski , 2002], La structure de réseau bayésien utilisée reflète directement la hiérarchie des documents. Dans ce modèle, la variable aléatoire associée à chaque élément de la hiérarchie du document peut prendre trois valeur dans l'ensemble V = {N, G, E}, où :

N : signifie qu'un élément est non pertinent.

G : signifie qu'un élément est peu spécifique.

E : signifie qu'un élément est très spécifique.

Deux autres types de variables aléatoires sont considérés, la première est la requête, représentée sous forme d'un vecteur de fréquences de termes. La deuxième est associé aux modèles de pertinence utilisés pour évaluer la similarité locale d'un l'élément à la requête et peut prendre deux valeurs : pertinent ou non pertinent.

Pour une requête donnée, un score local de pertinence est calcule pour chaque élément. Ce score dépend uniquement de la requête et du contenu de l'élément. Pour calculer ce score local, deux modèles sont utilises. Le premier (M_1) calcule une valeur S_1 pour chaque élément, comme suit :

$$S_1(e) = \frac{\sum_t \ tf_q(t)\frac{tf_e(t)}{tf_{par}(t)}}{\sum_t tf_q(t)}$$

Où :

$Tf_e(t)$: la fréquence du terme t dans l'élément e

$Tf_{par}(t)$: la fréquence du terme t dans l'élément parent de e.

$Tf_q(t)$: la fréquence du terme dans la requête q.

Le second (M_2), calcule une valeur S_2 pour chaque élément, comme suit :

$S_2(e) = \frac{S_1(e)}{\log(20+length(e))}$ Où length (e) représente le nombre de mots contenu dans e et ses descendants.

La probabilité qu'un élément soit pertinent pour le premier (respectivement le deuxième) modèle M_1 (respectivement modèle M_2) est définie par :

$P(M_i = R|\text{requête, contenu de l'élément}) = S_i$ avec $i \in \{1, 2\}$

La probabilité qu'un élément soit dans l'état N , G ou E dépend de l'état de l'élément parent, et du jugement du modèle sur l'élément (pertinent ou non pertinent) comme le montre la figure suivante :

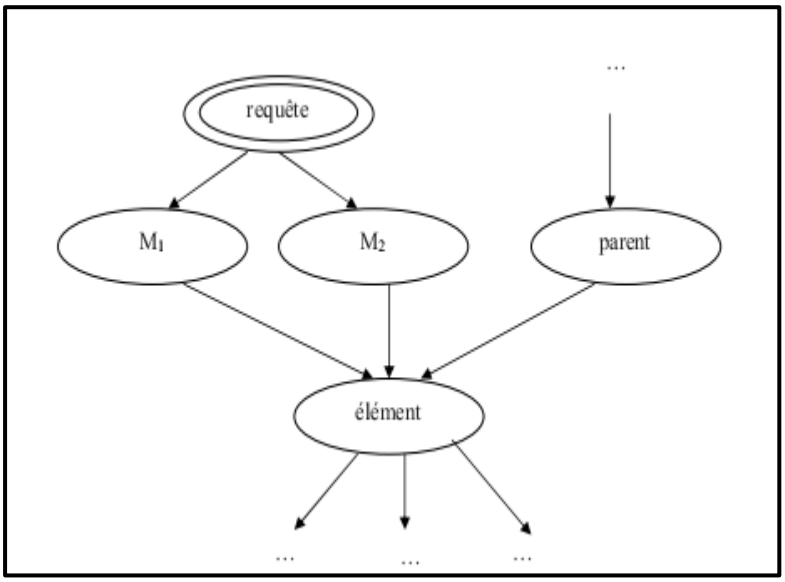


Figure 2.11 : Modèle de réseau bayésien. L'état de l'élément dépend de l'état du parent et de la pertinence de l'élément pour les modèles M_1 et M_2

Cette probabilité est calculée en utilisant la formule suivante :

$$P(e = v|q) = \sum_{v_p \in V} \sum_{r_1, r_2 \in (R, \neg R)} \theta_{c(e), v, v_p, r_1, r_2} \times P(e \text{ parent } = v_p) \times P(M_1 = r_1|q) \times P(M_2 = r_2|q)$$

Où $v \in V$, q est une requête composée de simples termes, et θ est un paramètre obtenu par apprentissage. Il dépend des différents états des 4 variables aléatoires (état de l'élément, état du parent, pertinence des modèles de base M1 et M2), et de la catégorie c(e) de l'élément.

La probabilité P (e = E|q) donne le score de pertinence final de l'élément, qui permet ensuite de classer les éléments selon leur degré de pertinence. Ces scores sont calculés récursivement en commençant par les éléments les plus grands et ensuite les plus petits.

II.8. Evaluation des systèmes de recherche d'information structurée:

L'évaluation des SRI est une phase très importante pour comparer leurs performances. Comme TREC(Text Retreival Conference) pour la recherche d'information dans les documents textuelles plats, INEX est une compagne d'évaluation des SRI dans des documents XML.Elle est mise en place chaque année depuis 2002. Elle fournit un corpus de documents XML, un ensemble de requêtes et leurs jugements de pertinence, dans le but de promouvoir l'évaluation de la recherche sur des documents XML.

II.8.1. Jugements de pertinence :

L'évaluation des systèmes de recherche d'information structurée dans INEX vise principalement à mesurer la capacité de ce système à retrouver des éléments à la fois exhaustifs et spécifiques au sujet de la requête. Dans [Sauvagnat, 2005], le principe de recherche d'information dans les documents semi-structurés est définit comme suit : « un système devrait toujours retrouver l'unité d'information exhaustive et spécifique répondant à une requête ».

- **Exhaustivité :**

L'exhaustivité décrit à quel point l'élément traite du sujet de la requête, c'est-à-dire qu'elle tient compte de la présence ou de l'absence de l'information recherchée dans un élément. Nous distinguerons quatre niveaux de d'exhaustivité [Zargayouna, 2005] :

- Non exhaustif (0) : l'élément ne traite pas du sujet de la question ;
- Faiblement exhaustif (1) : l'élément traite marginalement le sujet de la question ;
- Moyennement exhaustif (2) : le sujet de la question est en grande partie traité dans l'élément ;
- Totalement exhaustif (3) : le sujet de la question est traité exhaustivement (tout ou la majorité) dans l'élément.

- **Spécificité :**

La spécificité décrit à quel point l'élément se focalise sur le sujet de la requête. Nous distinguerons à nouveau quatre niveaux [Zargayouna, 2005] :

- Non spécifique (0) : le sujet de la requête n'est pas un thème de l'élément ;
- Faiblement spécifique (1) : 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 (2) : le sujet de la requête est un thème majeur de l'élément (l'élément peut contenir quelques informations non pertinentes) ;
- Totalement spécifique (3) : le sujet de la requête est le seul thème de l'élément.

Les jugements de pertinence portés à l'intérieur d'un même document obéissent à un certain nombre de règles qui permettent de vérifier la consistance de ces jugements [Piwowarski et Lalmas, 2004] :

- Si tous les enfants d'un élément sont non pertinents, alors cet élément est non pertinent.
- L'exhaustivité d'un élément est toujours supérieure ou égale à l'exhaustivité d'un de ses fils. Il est en effet impossible de trouver plus d'informations pertinentes au sein d'un élément que dans la totalité de l'élément.
- La spécificité d'un élément est inférieure ou égale au maximum de la spécificité de ses fils : un élément qui est par exemple totalement spécifique ne peut pas contenir que des éléments qui ne sont que moyennement, faiblement ou pas du tout spécifiques.

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 [Sauvagnat, 2005].

F_{quant} : exhaustivité * spécificité $\longrightarrow$ [0, 1]

Qu'on notera : (e, s) $\longrightarrow F_{quant}$ (e; s)

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.

$$F_{strict}(e, s) = \begin{cases} 1 & \text{si } e = 3 \text{ et } s = 3 \\ 0 & \text{sinon} \end{cases}$$

- La fonction $F_{generalised}$ est utilisée pour pouvoir prendre en compte les différents degrés d'exhaustivité et spécificité.

$$F_{generalised}(e, s) = \begin{cases} 1 & \text{si } (e,s) = (3,3) \\ 0.75 & \text{si } (e,s) \in \{(2,3),(3,2),(3,1)\} \\ 0.5 & \text{si } (e,s) \in \{(1,3),(2,2),(2,1)\} \\ 0.25 & \text{si } (e,s) \in \{(1,2),(1,1)\} \\ 0 & \text{si } (e,s) = (0,0) \end{cases}$$

La fonction $F_{generalised}$ favorise les éléments jugés exhaustifs, elle leur assigne un score élevé même si leur spécificité est faible. Une autre fonction, notée F_{sog} (Specificity Oriented Generalised), orientée spécificité a été définie comme suit :

$$F_{sog} = \begin{cases} 1 & si\ (e,s) = (3,3) \\ 0.9 & si(e,s) = (2,3) \\ 0.75 & si\ (e,s) \in \{(1,3),(3,2)\} \\ 0.5 & si\ (e,s) = (2,2) \\ 0.25 & si\ (e,s) \in \{(1,2),(3,1)\} \\ 0.1 & si\ (e,s) \in \{(2,1),(1,1)\} \\ 0 & si\ (e,s) = (0,0) \end{cases}$$

Deux autres classes de fonctions ont été définies :

- Les fonctions favorisant les éléments spécifiques, c'est-à-dire, qui retournent les éléments ayant le plus haut degré de spécificité, déterminées ainsi :

$$F_{s3\_e321}(e, s) = \begin{cases} 1 & \text{si } e \in \{3,2,1\} \text{ et } s = 3 \\ 0 & \text{sinon} \end{cases}$$

$$F_{s3\_e32}(e, s) = \begin{cases} 1 & \text{si } e \in \{3,2\} \text{ et } s = 3 \\ 0 & \text{sinon} \end{cases}$$

- Les fonctions favorisant les éléments exhaustifs, autrement dit, qui retournent les éléments ayant le plus haut degré d'exhaustivité, déterminées ainsi :

$$F_{e3\_s321}(e, s) = \begin{cases} 1 & \text{si } s \in \{3,2,1\} \text{ et } e = 3 \\ 0 & \text{sinon} \end{cases}$$

$$F_{e3\_s32}(e, s) = \begin{cases} 1 & si\ s \in \{3,2,\} \ et\ e = 3 \\ 0 & sinon \end{cases}$$

II.8.2. Mesures d'évaluation :

La compagne INEX fournit aussi des procédures d'évaluation. Ces procédures d'évaluation sont basées sur les mesures de rappel et précision. Dans ce qui suit, nous présenterons un aperçu sur les mesures d'évaluation utilisées [Hlaoua, 2007].

- **Le gain cumulé (xCG) :**

La mesure xCG cumule les scores de pertinences des éléments de la liste des résultats. Etant donnée une liste triée d'éléments par ordre décroissant dans laquelle, les éléments sont présentés par leur score de pertinence, le gain cumulé au rang i, noté xCG[i], est calculé comme la somme des pertinences jusqu'à ce rang :

$$xCG[i] = \sum_{j=1}^{i} xG[j]$$

Où xG[j] représente le score du document au rang j.

Pour chaque requête, on calcule ensuite un vecteur de gain idéal xCI à partir de la base de rappel, afin de le comparer au xCG. On définit ainsi le gain cumulé normalisé, noté nxCG, qui reflète le gain relatif de l'utilisateur accumule jusqu'à un rang i, compare à ce qu'il aurait du atteindre si le système avait produit une liste triée optimale. Il est définit comme suit :

$$nxCG[i] = \frac{xCG[i]}{xCI[i]}$$

- **L'effort-précision (ep(r)) :**

Cette mesure représente l'effort (en nombre de liens à visiter) qu'un utilisateur doit fournir pour parvenir à un gain donné r. La valeur 1 correspond à une performance idéale,

pour laquelle l'utilisateur effectue un minimum d'effort pour atteindre le niveau de gain r. Elle est définit par la formule suivante :

$$ep\ [r] = \frac{e_{idéal}}{e_{run}}$$

Tel que :

e_{ideal} représente le rang pour lequel le gain cumulé est atteint par la courbe idéale

e_{run} représente le rang pour lequel le gain cumulé est atteint par le système.

- **Le gain-rappel (gr(i)):**

Le gain-rappel est le rapport entre le gain cumulé obtenu en une position i de la liste et le gain total auquel on peut parvenir, donné par la fonction suivante :

$$gr\ [i] = \frac{xCG[i]}{xCI[n]}$$

Où n est le nombre total d'éléments pertinents.

L'effort-précision à une valeur donnée de gain-rappel mesure l'effort d'un utilisateur pour atteindre un gain relatif au gain total qu'il peut obtenir.

Conclusion :

Dans ce chapitre, nous avons présenté les différents éléments de la RI structurée. En premier lieu, nous avons donné un aperçu sur les documents semi-structurés. En second lieu, nous avons présenté les enjeux soulevés par la RI structurée relatifs à l'aspect structurel de ces documents. La dimension structurelle apportée au contenu textuel des documents permet de considérer l'information avec une autre granularité que le document tout entier. Le but des SRI traitant des documents semi-structurés est alors d'identifier des parties des documents les plus pertinentes à une requête donnée. Nous avons ainsi présenté les principales approches d'indexation et d'appariement développées en RIS. Nous avons également détaillé les nouveaux concepts d'évaluation des systèmes de recherche en RIS.

Les approches actuelles dans la recherche des documents semi-structurés (documents XML) sont basées sur une indexation par mots clés ou termes. Ces approches ne prennent pas en considération le sens des mots, or la sémantique de la structuration ainsi que celle du contenu est importante à prendre en compte. De nouvelles approches d'indexation basée sur le sens des termes ont été introduites, il s'agit de l'indexation sémantique et conceptuelle que nous allons détailler dans le chapitre suivant.

Chapitre III : Recherche d'Information sémantique dans les documents semi-Structurés

Chapitre III : Recherche d'Information sémantique dans les documents semi-Structurés

Introduction :

Dans les deux chapitres précédents, nous avons présenté le processus de recherche d'information appliqué aux documents plats ainsi qu'aux documents semi-structurés (document XML). L'une des phases les plus importantes de ce processus est l'indexation.

L'indexation vise principalement à sélectionner les descripteurs susceptibles de représenter au mieux le contenu des documents, les techniques d'indexation déjà présentées sont basées sur des mots simples extraits directement des documents en question. Cette approche a montré son insuffisance devant le problème de l'ambiguïté sémantique des mots du langage naturel. En effet les utilisateurs des 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. D'ou la nécessité de développer des approches qui exploitent la sémantique des textes dans la représentation de l'information. Ce type d'indexation passe du niveau des mots au niveau des concepts ou des sens des mots pour décrire le contenu. Ainsi on parle d'indexation sémantique ou conceptuelle.

Dans ce qui suit, nous présenterons d'abord les problématiques liées à l'indexation classique (section **I**), nous donnerons ensuite la définition de quelques concepts clés (section **II**), puis nous définirons l'indexation sémantique (section **III**) et nous exposerons quelques ressources sémantiques externes utilisées pour l'identification des sens des mots (section **IV**), et enfin nous décrirons les approches d'indexation sémantique dans les documents semi-structurés (section **V**).

I. Problématique liée à l'indexation classique :

Les problèmes liés à l'indexation classique, basée sur des mots-clés, se présentent dans l'ambiguïté des mots et leur disparité.

➢ L'ambiguïté des mots :

L'ambiguïté des mots, ou ambiguïté lexicale, se rapporte à des mots lexicalement identiques et portant des sens différents. Elle se divise en deux catégories : l'ambiguïté syntaxique et l'ambiguïté sémantique.

- L'ambiguïté syntaxique : elle se rapporte à des différences dans la catégorie syntaxique. Par exemple, « run » peut apparaître en tant que nom ou verbe.
- L'ambiguïté sémantique : elle se rapporte à des différences dans la signification, et est décomposée en homonymie et polysémie. La polysémie est le fait qu'un terme possède plusieurs sens. Quant à l'homonymie, c'est le fait que deux mots ont la même forme orale ou écrite mais des sens différents.

Le problème d'ambiguïté implique que des documents non pertinents, contenant les mêmes mots que la requête sont retrouvés, ce qui influe négativement sur la précision.

- **La disparité des mots** :

Elle se réfère à des mots lexicalement différents mais portant un même sens, autrement dit c'est le phénomène de synonymie.

Le problème de disparité implique qu'il est impossible de trouver des documents représentés par un mot M1 synonyme d'un mot M2, où M2 est présent dans la requête.

II. Terminologie :

Voici dans ce qui suit les définitions de quelques concepts, nécessaire pour la compréhension de ce chapitre**.**

- **Mot :** Un mot est défini comme, « la plus petite unité signifiante qui peut exister de façon autonome dans une phrase; dans un texte écrit, il est délimité par des blancs ou par des signes de ponctuation »[Harrathi, 2010b]. Le mot est une unité sémantique indécomposable (minimal) qui évoque toujours un sens donné, ou le plus souvent plusieurs sens, selon son contexte.

- **Terme :** Un terme est une expression linguistique, un mot ou un groupe de mots. On distingue donc deux types de terme : les termes simples composés d'un seul mot, et les termes complexes composés de plusieurs mots.

- **Objet :** Un objet est défini comme étant, « un élément de la réalité qui peut être perçu ou conçu, les objets peuvent être matériels (par exemple : l'oiseau) ou immatériels (par exemple : la liberté) » [Harrathi, 2010b].

- **Sens :** Le sens, en linguistique, est la signification cognitive d'une expression (mot, phrase, énoncé etc). Il désigne le contenu conceptuel de l'expression ou la manière avec laquelle on exprime quelque chose. Un mot ayant plusieurs sens est appelé polysémique. Son sens dépend du contexte dans lequel il est utilisé. Un mot peut être utilisé dans son sens le plus courant (on parle alors de sens propre) ou dans un sens plus imagé (on dit qu'il est au sens figuré).

- **Concept :** Un concept est une représentation générale et abstraite de la réalité d'un objet, d'une situation ou d'un phénomène. C'est une unité de pensée qui désigne un sens, une idée conçue par l'esprit, de façon non ambiguë. Ainsi le concept est souvent assimilé à la signification d'un terme. Un concept est exprimé par un terme simple ou

complexe. Ce terme peut être préféré ou non-préféré. Chaque concept a un terme préféré unique qui est souvent le nom standard du concept et plusieurs termes non-préférés.

III. L'indexation sémantique (Sense Based Indexing) :

L'indexation sémantique s'intéresse principalement à la représentation des documents et requêtes par les sens des mots qu'ils contiennent plutôt que par les mots eux mêmes. Elle consiste, lors de l'analyse d'un document (ou de la requête), à rattacher chaque mot à un concept sous-jacent qui représente son sens. Ainsi, par exemple, pour le mot "jaguar", il faut déterminer s'il s'agit du félin, de la voiture ou de l'avion.

L'indexation sémantique s'intéresse à deux points principaux [Boubekeur, 2008], qui sont :

- Retrouver le sens correct de chaque mot dans le document ou la requête.
- Représenter ce document ou cette requête à l'aide des sens identifiés.

Ainsi, lors du processus d'indexation, pour chaque descripteur extrait du document, un sens lui est affecté. Si le descripteur possède plusieurs sens, celui-ci est désambiguïsé. La désambigüisation consiste à sélectionner, parmi les différents sens correspondant à un descripteur le plus adéquat selon son contexte.

L'usage des ressources terminologiques externes (Dictionnaire, thésaurus, ontologies,…) est l'une des principales caractéristiques de l'indexation sémantique. En effet, les différents sens des unités d'informations du contenu textuel sont identifiés grâce à ces ressources.

IV. Les ressources exploitées pour l'indexation sémantique :

IV.1. Dictionnaire :

Un dictionnaire est un recueil des mots d'une langue, des termes d'une science, d'un art, rangé par ordre alphabétique, avec leurs significations. Un dictionnaire de la langue indique la définition, l'orthographe, les sens et les emplois des mots de cette langue.

Un dictionnaire automatisé est un dictionnaire sous forme électronique qui peut être interrogé via un logiciel d'application.

IV.2. Thésaurus :

Un thésaurus constitue un dictionnaire hiérarchisé des vocabulaires contrôlés. Ce vocabulaire est normalisé, il présente les termes génériques ou spécifiques à un domaine de connaissance. Ces termes sont organisés de manière conceptuelle (ils dénotent des concepts) et reliés entre eux par des relations sémantiques. Les relations courantes dans un thésaurus sont les suivantes:

- Relation hiérarchique (spécialisation et généralisation) : un concept générique désigne les entités ou concepts généraux en référence aux autres concepts et au domaine considéré. Un concepts spécifique précise et identifie les entités ou concepts plus précis à l'intérieur du champ sémantique d'un terme générique donné.
- Relation d'équivalence (synonymie ou quasi-synonymie) : relie les termes représentant un même concept.
- Relation d'association : relie les concepts selon un autre axe de type : sujets connexes, proche-de, relié-à, etc.
- Relation d'appartenance : appartenance d'un concept à un groupe de concepts. le regroupement de concept se fait selon un critère spécifique.

IV.3. Ontologies :

La définition la plus citée présente une ontologie comme étant « une spécification explicite et formel d'une conceptualisation partagée » [Harrathi, 2010b]. La conceptualisation se réfère ici à l'élaboration d'un modèle abstrait d'un domaine du monde réel en identifiant et en classant les concepts pertinents décrivant ce domaine. La formalisation consiste à rendre cette conceptualisation exploitable par des machines.

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. Elles sont classées selon deux critères : la structure de la conceptualisation et le sujet de la conceptualisation.

La classification des ontologies selon la structuration de la conceptualisation a fait émerger trois catégories à savoir :

- Les ontologies terminologiques (lexiques, glossaires…).
- Les ontologies d'information (schéma d'une BD).
- Les ontologies des modèles de connaissances.

Quand à leur classification selon le sujet de la conceptualisation, elle a fait ressortir quatre types d'ontologie, qui sont :

- Les ontologies d'application: elles contiennent toutes les informations nécessaires pour modéliser les connaissances pour une application particulière.
- Les ontologies de domaine: elles fournissent un ensemble de concepts et de relations décrivant les connaissances d'un domaine spécifique.
- Les ontologies génériques (dites aussi de haut niveau) : elles sont similaires aux ontologies de domaine, mais les concepts qui y sont définis sont plus génériques et décrivent des connaissances tel que l'état, l'action, l'espace et les composants. Généralement, les concepts d'une ontologie de domaine sont des spécialisations des concepts d'une ontologie de haut niveau.

- Les ontologies de représentation (méta-ontologies) : elles fournissent des primitives de formalisation pour la représentation des connaissances. Elles sont généralement utilisées pour écrire les ontologies de domaine et les ontologies de haut niveau. c'est-à-dire qui permettent de spécifier dans un langage formel les concepts d'un domaine et leurs relations.

L'une des ontologies la plus utilisée en recherche d'information est WordNet, nous avons choisi pour notre part de l'exploiter afin d'implémenter notre approche, elle vous sera décrite dans le chapitre suivant.

IV.4. Taxonomie :

La taxinomie est la forme la plus simple des vocabulaires contrôlés, elle se présente sous la forme d'une hiérarchie simple de termes. Les liens hiérarchiques dans une taxonomie correspondent à des liens de spécialisation /généralisation.

V. L'indexation sémantique des documents semi-structurés :

Les documents semi-structurés, comme déjà définis, sont caractérisés par la coexistence de l'information structurelle et de l'information de contenu. Les approches d'indexation sémantique proposées pour palier aux problèmes de l'indexation classique de ce type de documents se divisent en trois catégories selon la prise en considération de la structure, du contenu ou de la structure et du contenu.

Dans ce qui suit nous illustrerons ces trois approches à travers les différents travaux d'indexation sémantique recensés dans la littérature.

V.1. Approche orientée contenu :

Dans ce premier type d'approche, seul le contenu est pris en considération, on proposant d'indexer celui-ci par des concepts. Quant à la structure, elle n'est pas prise en compte.

Dans [Harathi, 2010a], l'auteur propose d'utiliser le modèle vectoriel sémantique. A cet effet le document XML est représenté par un arbre DOM composé de deux types de nœud : les nœuds texte (notés N_T) et les nœuds élément (notés N_E). Chaque nœud texte est représenté par un vecteur dans l'espace d'indexation. Les dimensions de l'espace d'indexation sont l'ensemble des concepts d'une ontologie Ω. Ainsi, dans un espace conceptuel d'indexation $C_\Omega = \{c_1,\ldots c_n\}$ où les c_i sont les concepts d'indexation, un nœud texte N_T^j est représenté par un vecteur des poids des concepts.

$$N_T^j = (w_{1j},\ldots\ldots w_{kj},\ldots\ldots\ldots w_{nj})$$

La première étape dans le processus d'extraction des concepts consiste à repérer les séquences des mots susceptibles d'être des labels de concepts dans l'ontologie Après cette étape seuls les termes qui ont une correspondance dans l'ontologie sont retenus,

Si le terme dénote plusieurs concepts, il nécessite une désambiguïsation, le procédé de désambiguïsation s'appuie sur le contexte du terme, ce contexte peut représenter une phrase, un paragraphe ou un élément logique de la structure logique (les nœuds texte dans les documents XML). Soit le mot t_k à désambiguïser, on définit son contexte d'apparition CA comme suit :

$$CA = \{t_1, \ldots t_k, \ldots . t_n\}$$

On note $C_\Omega(t_k)$ l'ensemble des concepts de l'ontologie Ω ayant comme label le terme t_k.

$$C_\Omega(t_k) = \{c_k^1, \ldots\ldots c_k^i, \ldots\ldots c_k^m\}$$

Avec c_k^i le ième concept dénoté par le terme t_k et m le nombre de concept dénoté par le terme t_k

De la même façon on identifie pour chaque terme du contexte CA l'ensemble des concepts qu'il dénote, puis on calcule pour chaque combinaison possible des différents concepts attachés à chacun des termes du contexte CA une valeur de similarité globale.

La désambiguïsation s'appuie sur l'hypothèse qui considère que les mots qui apparaissent dans des contextes similaires tendent à avoir des sens proches. De cette façon, la combinaison des concepts dans laquelle les concepts sont très proches est sélectionnée, autrement dit la combinaison des concepts dont la valeur de similarités entre les concepts est maximale.

La pertinence d'un nœud vis-à-vis d'une requête est déterminée selon que les concepts de la requête soient proches des concepts du nœud. A cet effet, le nœud texte et la requête sont représenté par un graphe pondéré dont les nœuds sont les concepts. Chaque arête de ce graphe est affectée d'un poids représentant la similarité sémantique entre les concepts.

La figure 3.1 décrit un exemple d'un graphe sémantique :

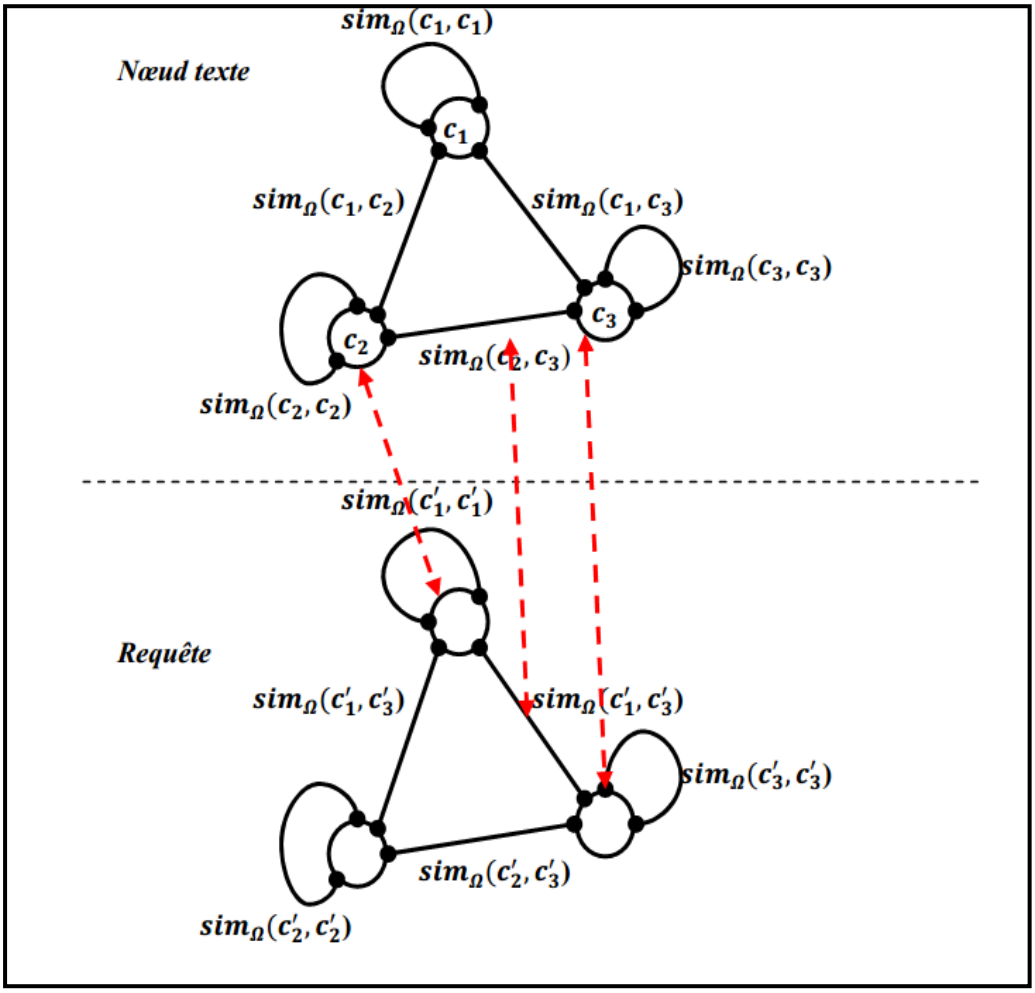


Figure 3.1 : Graphe sémantique d'une requête et d'un nœud texte.

Ainsi, le calcul de score de pertinence d'un nœud texte vis-à-vis d'une requête revient à mesurer à quel point le graphe sémantique du nœud est proche du graphe sémantique de la requête et cela en utilisant la mesure de cosinus.

Dans cette approche seuls les nœuds de type texte sont indexés par des vecteurs sémantiques de concepts. Comme les documents semi-structurés possèdent une structure arborescente, les index des nœuds sont imbriqués les uns dans les autres et par conséquent, l'index d'un nœud de type élément contient les index de ses nœuds descendants de type texte. Ainsi, les concepts des nœuds de type texte sont propagés dans l'arbre des documents.

V.2. Approche orientée structure :

Les approches orientées structure proposent d'indexer uniquement la structure, Ils ont été introduits afin de palier au problème de l'interrogation des documents semi-structurés hétérogènes. Ces documents suivent des DTD différentes présentant ainsi une structuration distincte pour un contenu similaire. Voici ci-dessous un exemple de deux DTD différentes décrivant un même domaine (deux DTD représentées sous la forme d'un arbre) :

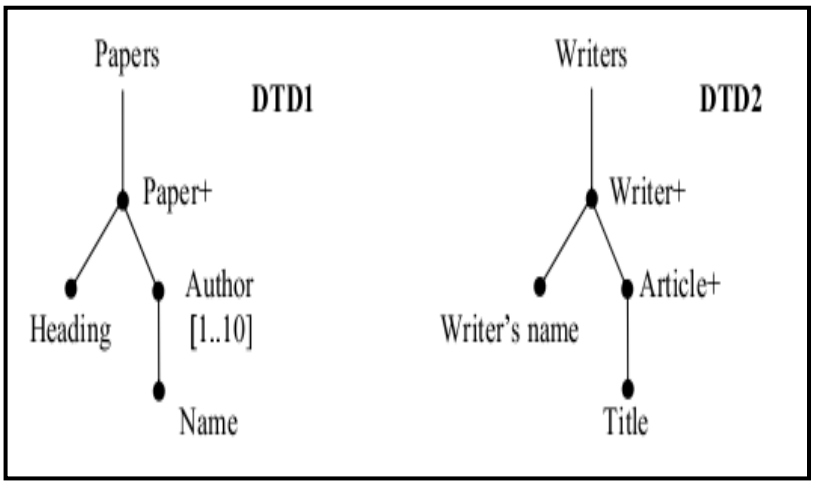
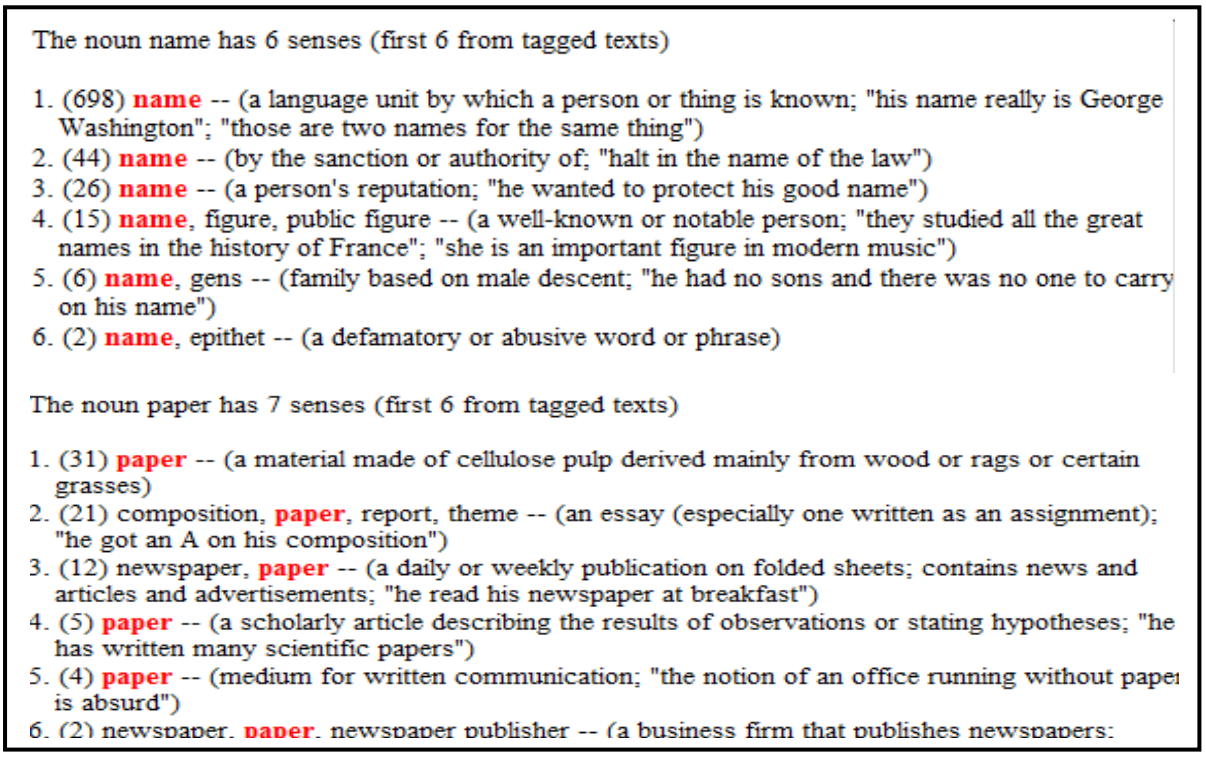


Figure 3.2 : Deux DTD différentes décrivant le même domaine.

Lors de l'interrogation d'un corpus de documents hétérogènes, l'utilisateur pourrait exprimer des conditions de structure qui ne correspondent pas aux DTD des documents, qui sont pertinents à sa requête, ces derniers ne seront donc pas retournés par le SRI. Ainsi, pour interroger un document dont le schéma est nouveau, un système doit pouvoir soit adapter la requête posée au document, soit pouvoir adapter le document pour lui appliquer la requête.

Différentes approches traitent de l'indexation sémantique de la structure des documents, Dans son approche Bouidghaghen [Bouidghaghen, 2007] exploite la sémantique portée par les balises XML en utilisant l'ontologie WordNet.

La première étape dans cette approche est d'extraire les concepts candidats pour chaque balise identifiée dans la collection de documents par projection sur WordNet. Chaque nom de balise peut avoir plusieurs sens et correspondre à plusieurs concepts de l'ontologie. Reprenons l'exemple de la figure 3.1, La balise « name » possède 6 sens, la balise « paper » 7 sens. Comme le montre la figure suivante :



The noun name has 6 senses (first 6 from tagged texts)

1. (698) **name** -- (a language unit by which a person or thing is known; "his name really is George Washington"; "those are two names for the same thing")
2. (44) **name** -- (by the sanction or authority of; "halt in the name of the law")
3. (26) **name** -- (a person's reputation; "he wanted to protect his good name")
4. (15) **name**, figure, public figure -- (a well-known or notable person; "they studied all the great names in the history of France"; "she is an important figure in modern music")
5. (6) **name**, gens -- (family based on male descent; "he had no sons and there was no one to carry on his name")
6. (2) **name**, epithet -- (a defamatory or abusive word or phrase)

The noun paper has 7 senses (first 6 from tagged texts)

1. (31) **paper** -- (a material made of cellulose pulp derived mainly from wood or rags or certain grasses)
2. (21) composition, **paper**, report, theme -- (an essay (especially one written as an assignment); "he got an A on his composition")
3. (12) newspaper, **paper** -- (a daily or weekly publication on folded sheets; contains news and articles and advertisements; "he read his newspaper at breakfast")
4. (5) **paper** -- (a scholarly article describing the results of observations or stating hypotheses; "he has written many scientific papers")
5. (4) **paper** -- (medium for written communication; "the notion of an office running without paper is absurd")
6. (2) newspaper, **paper**, newspaper publisher -- (a business firm that publishes newspapers;

Figure 3.3 : Les différents sens des mots « name » et « paper » extraits de WordNet.

La deuxième étape est la désambiguïsation qui permettra de sélectionner pour chaque balise le meilleur sens. Le principe de la désambiguïsation consiste à supposer que, parmi les différents concepts candidats (sens) pour une balise donnée, le plus adéquat est celui qui a le plus de liens avec les autres concepts du même contexte. Ce contexte peut être formé de toutes les balises de la DTD à laquelle appartient cette balise ou peut être restreint à l'ensemble formé de sa balise mère et éventuellement la liste de ses balises filles et attributs.

Soit à désambiguïser une balise bi représentée par l'ensemble des concepts qui lui sont associés de cardinalité n, noté Si = $\{S_1^i, S_2^i, \ldots\ldots S_n^i\}$. On affecte à chaque concept candidat (ou sens d'une balise) un score (C_score) égale à la somme des valeurs de similarité qu'il a obtenu avec les autres concepts candidats des balises de son contexte sauf ceux qui sont dans le même ensemble de sens que le sien.

Pour la balise b_i, le score de son Kiéme sens est calculé comme suit :

$$\text{C\_score}\ (S_k^i) = \sum_{\substack{j\epsilon\ [1..m], j\neq i \\ l\epsilon[1..n]}} Sim\ (S_k^i, S_l^j)$$

Où m représente le nombre de balises formant le contexte d'une balise et n le nombre de sens propres à chaque balise b_i.

Le concept à retenir est celui qui maximise le C_score :

$$S_i = \text{Best\_score}\ (b_i) = \text{Max}_{k=1..n} \left\| \text{C\_score}(S_k^i) \right\|$$

En tenant compte de l'exemple illustré par la figure 3.4, voici quelques valeurs de similarité calculées pour la balise « name » et « title » :

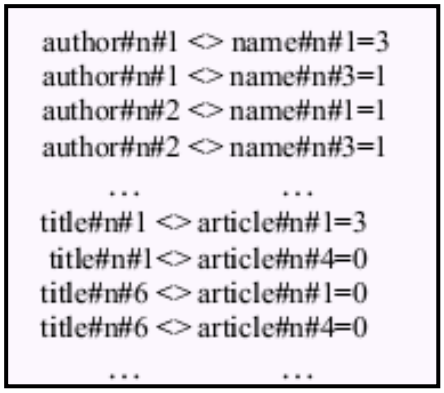


Figure 3.4 : les mesures de similarité calculées entre les concepts.

Suivant le cumul de similarité calculé pour chaque sens d'une balise donnée, le concept qui maximise le score de similarité sera retenu comme sens de la balise. Les résultats de l'exemple de la figure 3.1 est illustré ci-dessous :

Figure 3.5 : le meilleur score cumulé des concepts retenus.

La dernière étape de cette approche est de construire un dictionnaire contenant les concepts retenus lors de l'étape de désambiguïsation. Pour chaque concept, le dictionnaire comprendra aussi la liste des balises le référençant (liste de ses balises synonymes). Par exemple la balise "writer" est identifiée comme synonyme de la balise "author", elles correspondent au même concept de l'ontologie, qui est "writer#n#1". Le dictionnaire construit correspondant à l'exemple de la figure 3.1 est le suivant :

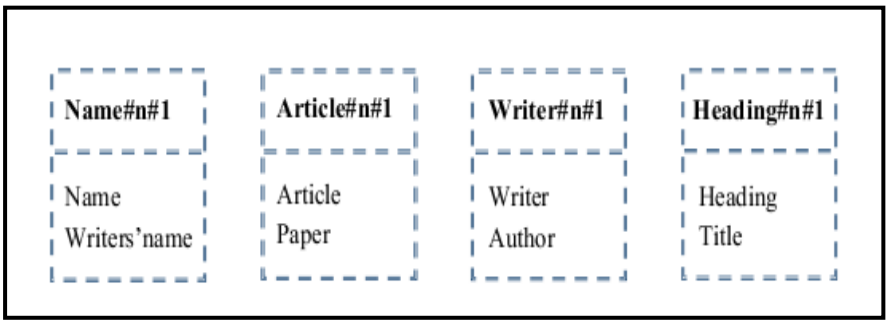


Figure 3.6 : Ensemble des concepts insérés dans le dictionnaire des synonymes

Ainsi, pour chaque requête d'un utilisateur contenant des conditions de structure formulées dans les termes d'une quelconque DTD de la collection, il sera possible de chercher pour chaque balise qui y figure, le concept correspondant dans le dictionnaire et d'identifier la liste des balises synonymes pour étendre la requête aux autre documents de la collection qui suivent d'autres DTD et les inclure dans la recherche.

V.3. Approche orientée contenu et structure :

Cette troisième approche propose d'indexer la structure et le contenu par des concepts tenant ainsi compte des deux dimensions du document XML.

Zargayouna[Zargayouna, 2005] dans son approche prend en considération la dimension sémantique, tant au niveau des termes que de la structure. Cette approche modélise le document XML comme un arbre étiqueté où chaque élément (et attribut) correspond à un nœud sans distinction entre les éléments et les attributs. Cet arbre est ensuite réduit en regroupant ensemble les nœuds ayant le même label et le même chemin à partir de la racine, pour ne garder que des chemins uniques. Comme illustré dans la figure ci-dessous :

```
<patient id=  "p1">
    <age> Patient agé de 70 ans </age>
    <antecedents>
        <antecedent> un patient qui a constitué un infractus ambulatoire
        </antecedent>
        <antecedent> Il présentait un angor  d'effort
        </antecedent>
        <traitements>
            <traitement> Les trois lésions ont pu être  traitées avec un petit trait de dissection
            localisé …
            </traitement>
            <traitement>……
            </traitement>
        <traitements>…
    </antecedents>
</patient>
```

Figure 3.7 : Extrait d'un document XML

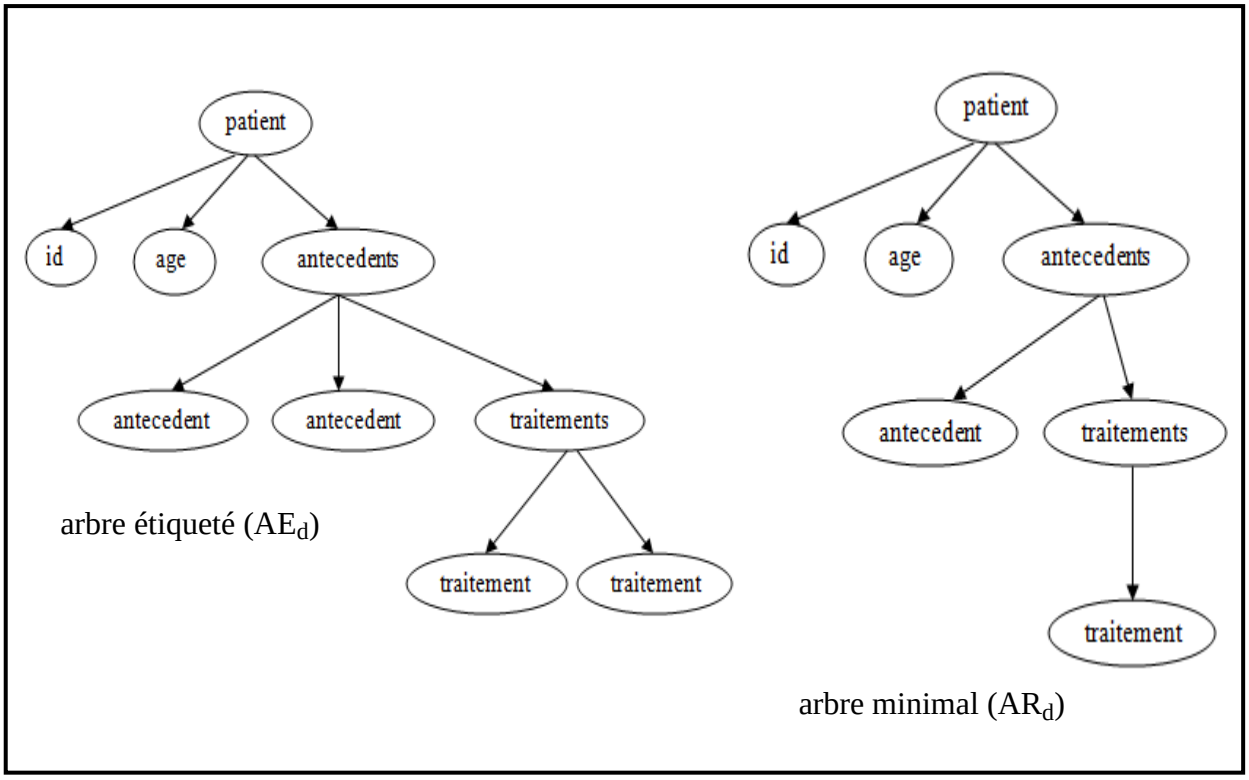


Figure 3.8 : Représentation en arbre et en arbre réduit d'un extrait d'un document XML.

Pour chaque nœud n de l'arbre réduit, est associé un modèle d'élément, noté E_e^Θ
Tel que :

e = label(n), label(n) correspond à l'étiquette de l'élément n.

Θ = label(n_0), label(n_1),…,label(n_{n-1}),label(n) où n_0,n_1,…n_{n-1},n représente le chemin de la racine n_0 jusqu'au nœud n.

L'indexation de la structure décompose ensuite l'arbre du document en un ensemble de contextes structurels représentés par les éléments feuilles[7] et les éléments mixtes[8]. Un contexte structurel, noté C_e^Θ, correspond à un modèle d'élément E_e^Θ porteur de texte. Comme illustré dans la figure suivante :

[7] Les éléments porteurs de texte.
[8] Les éléments internes (non feuilles) qui comportent aussi bien un contenu textuel qu'un contenu structurel.

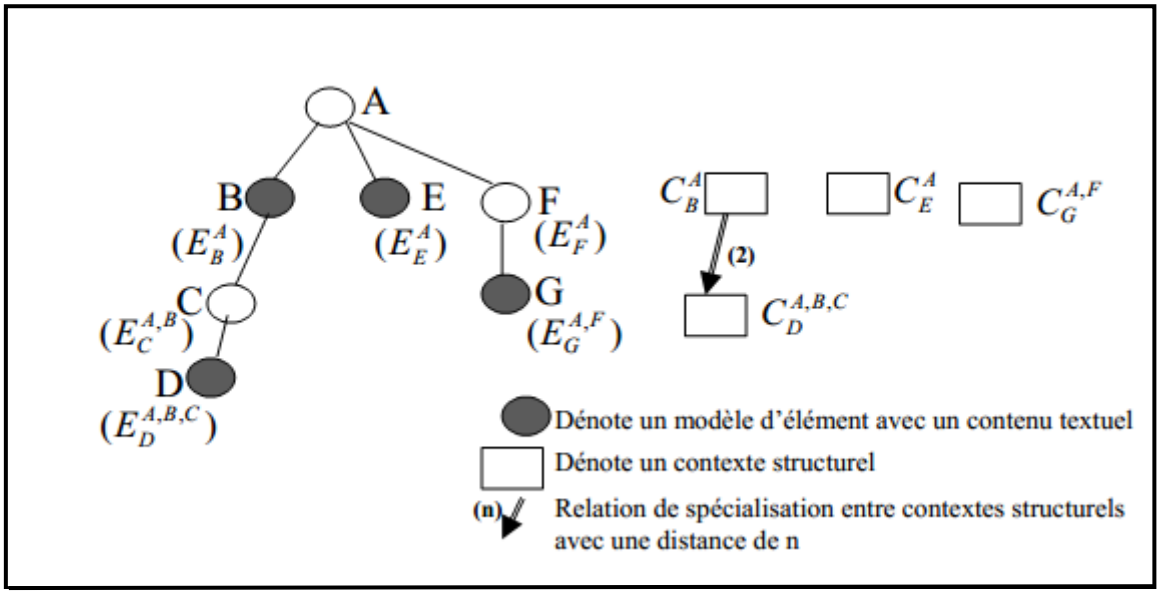


Figure 3.9 : Exemple d'arbre réduit et des contextes associés avec leur relation.

A l'issue de cette étape, un index structurel est construit. Il comprend la liste des contextes identifiés, 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 3.9 schématise la structure d'un index contenant quatre documents représentés par leurs contextes structurels. Nous pouvons remarquer que les documents d_1,d_2,d_3 partagent les mêmes contextes structurels (le contexte C_B^A ainsi que le contexte$C_D^{A,B,C}$, apparaissent dans les trois documents), quant au document d_4, il présente une structuration différente.

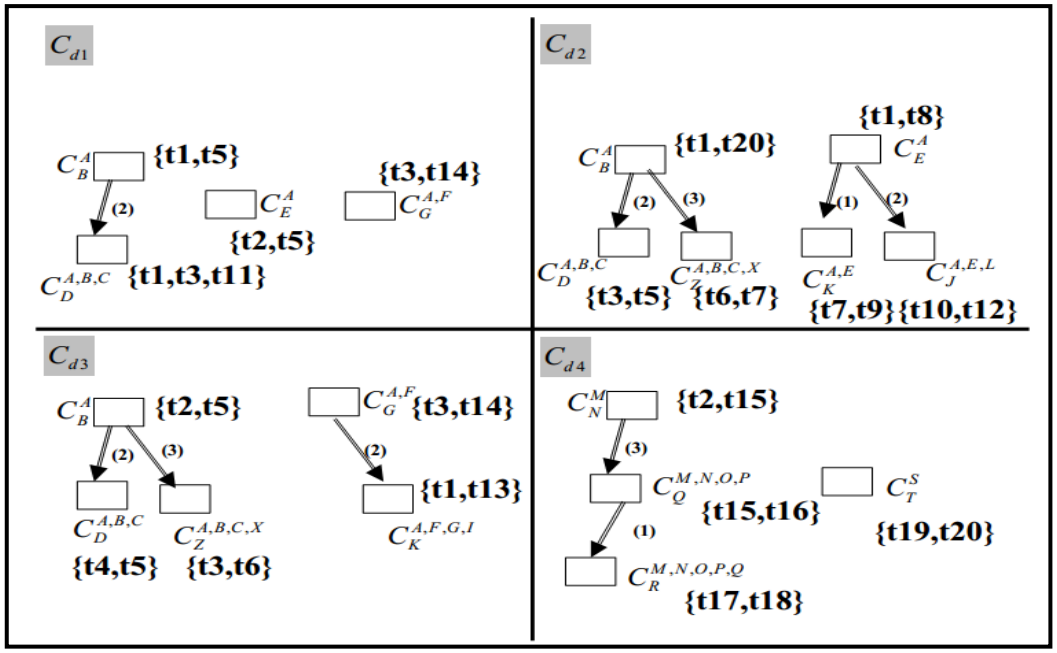


Figure 3.10 : Exemple d'un index structurel de quatre documents.

Les termes associés aux contextes structurels sont ensuite rattachés à leurs concepts par projection sur l'ontologie WordNet, Ces concepts dénotent l'ensemble de sens possibles du terme. Puis Ils sont désambiguïsés a fin de ne garder qu'un seul concept pour chaque terme. 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. C'est-à-dire choisir pour chaque terme le sens qui maximise la similarité sémantique avec les autres sens des autres termes de même contexte. Soit par exemple à désambiguïser le terme "mouse" apparaissant dans deux contextes, comme représenté dans la figue suivante :

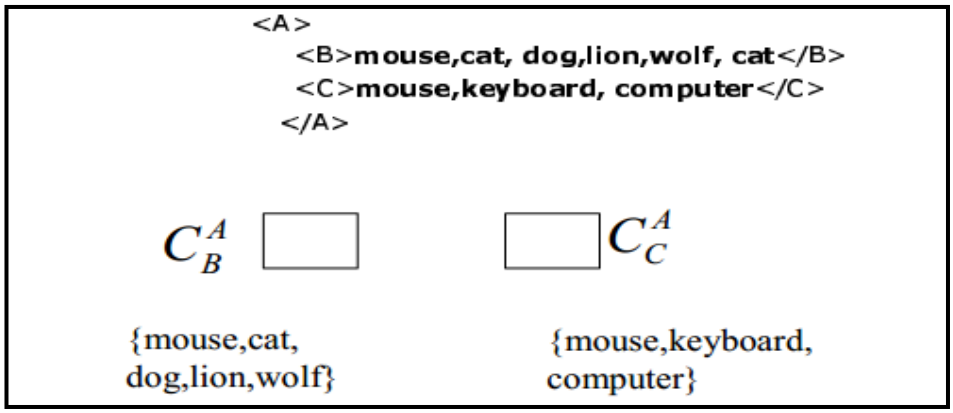


Figure 3.11 : Exemple de contextes avec un terme polysémique.

Dans le contexte C_B^A le sens retenu pour le terme mousse est celui d'un rongeur, quant au contexte C_C^A c'est celui d'une unité électronique périphérique.

De même, les balises du corpus sont reliées à l'ontologie. Ainsi si les deux modèles E_B^A et E_F^G font référence au même concept elles sont considérées comme équivalents et on une similarité sémantique de 1.Pour chaque balise, on identifié les balise équivalentes. Deux balises sont équivalentes si leur similarité sémantique dépasse un certain seuil. Ce qui servira à créer des classes d'équivalence lors de la phase d'interrogation et permettra de substitué un contexte par un autre.

Cette approche propose une adaptation du modèle vectoriel permettant de représenter chaque contexte par un vecteur des termes qui constituent son vocabulaire, les termes sont ensuite pondérés en fonction de leurs fréquences, leurs représentativités et leurs pouvoirs discriminatoires. Après désambiguïsation les poids des termes sont enrichis par la similarité conceptuelle des termes co-occurrents dans le même contexte.

L'auteur propose SemIR (Semantic Information Retrieval), 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.

Ce système permet de retrouver l'unité d'information minimale à retourner à l'utilisateur. Il permet aussi 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.

Chapitre III : Recherche d'Information sémantique dans les documents semi-Structurés

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 ainsi 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.

Conclusion :

La recherche d'information sémantique et conceptuelle ont été introduites afin de palier au problème de l'ambiguïté des mots du langage naturel.

Ce chapitre nous a permis d'introduire la recherche d'information sémantique comme une approche basée sur les sens des mots, qui sont extraits des ressources sémantiques telles que les ontologies, thésaurus et taxonomies. Nous avons aussi présenté les différentes approches d'indexation sémantique adaptée aux documents semi-structurés à savoir l'approche orientée contenu, l'approche orientée structure et l'approche orientée contenu et structure.

Le chapitre suivant sera dédié à la présentation de notre approche, nous vous décrirons l'ontologie Word Net qui sera exploitée pour l'identification des sens des mots ainsi que l'approche d'indexation implémentée.

Chapitre IV: Approche de recherche d’information sémantique dans les documents semi-structurés et expérimentations

Chapitre IV : Approche de recherche d'information sémantique dans les documents semi-structurés et expérimentations

Introduction :

Dans les chapitres précédents, nous avons présenté le processus de recherche d'information appliqué aux documents plats ainsi que son adaptation à la RI structurée. Nous avons aussi présenté l'indexation sémantique qui prend en considération l'aspect sémantique du contenu informationnel des documents et permet une indexation par le sens des mots.

Dans ce chapitre nous présenterons notre approche pour le traitement des documents semi-structurés XML. Nous décrirons le modèle de représentation des documents XML (section **I**), nous présenterons ensuite l'architecture de notre système (section **II**). Ainsi que le les étapes de l'indexation classique (section **III**) et la procédure de pondération des termes (section **IV**) adopté. Nous décrirons aussi notre schéma d'indexation sémantique (section **V**) et l'ontologie WordNet (section **V.1**) utilisée pour l'identification des sens. Puis nous présenterons le schéma d'appariement nœud-requête (section **VI**). Enfin nous exposerons le corpus de test qui nous permettra d'évaluer notre approche, ainsi que les résultats de nos expérimentations (section **VII**).

I. Modèle de représentation d'un document :

Un document XML est représenté sous forme d'un arbre, défini par les ensembles N,F et L [Sauvagnat, 2005] :

- N représente l'ensemble des nœuds internes, N={n_1,n_2,……}
- F représente l'ensemble des nœuds feuilles, F= {nf_1,nf_2,……}
- L représente l'ensemble des arcs orientés, Un arc orienté est une paire (u, v) formé de deux éléments des ensembles N ou F tels que :
  - u est le prédécesseur de v
  - chaque nœud interne, ni ∈ N, apparait au moins une fois dans une pair (u, v) en tant que premier composant.
  - chaque n_i∈ N, nf_i∈ F excepté le nœud racine apparait une et une seule fois dans une pair (u, v) en tant que second composant, en d'autre terme un nœud admet un et un seul parent.

Voici ci-dessous un exemple de document XML suivi de sa représentation en arbre :

```
<?xml version= ”1.0 ” ?>
<article>
<header>
<title>An American in Paris</title>
<id>309</id>
      …..
</header>
<bdy>
       …
<p>Gershwin composed the piece on commission from the New York Philharmonic …</p>
<p>Gershwin collaborated on the original program notes with the critic and composer...</p>
<sec>
<p>  An American in Paris is scored for 3 flute …</p>
</sec>
<sec>
<p>An American in Paris has been frequently recorded over the years …</p>
<p>In 1951, MGM released a musical comedy, An American in Paris,… </p>
<p>A part of the symphonic composition is also …</p>
</sec>
<sec>
<p>An American in Paris: listen to the whole symphonic poem …</p>
</sec>
</bdy>
</article>
```

Figure 4.1 : Extrait d'un fichier XML.

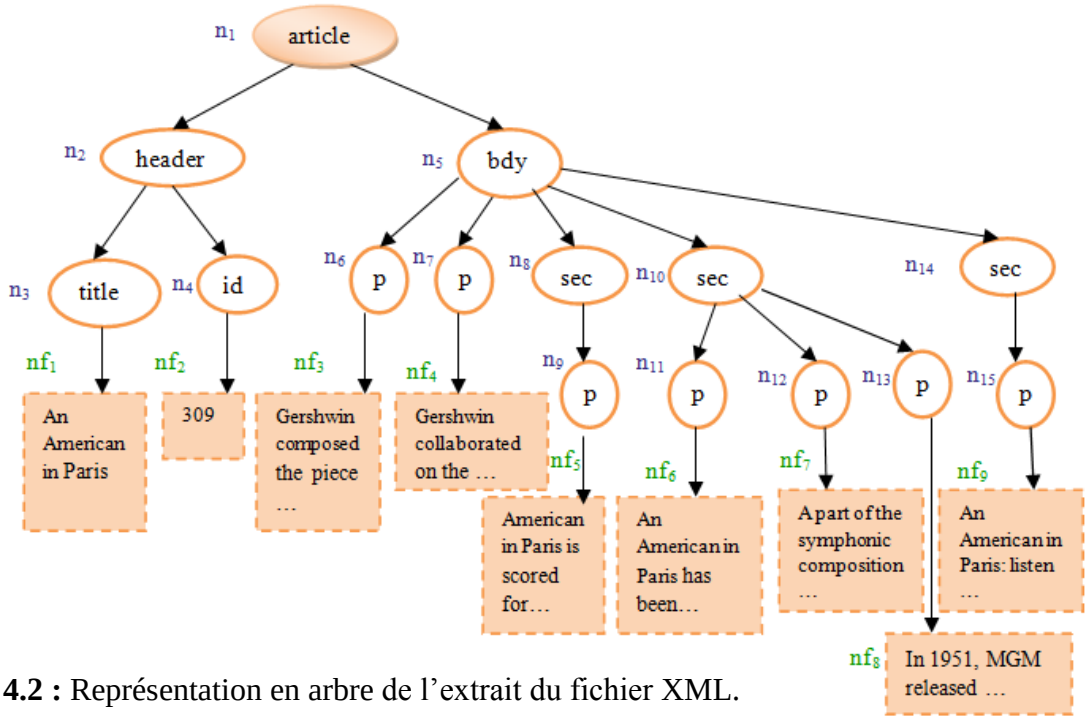


Figure 4.2 : Représentation en arbre de l'extrait du fichier XML.

Dans l'exemple ci-dessus, les ensembles N, F et L sont ainsi formés :

- N= $\{n_1,\ldots\ldots n_{15}\}$
- F = $\{nf_1,\ldots\ldots\ldots nf_9\}$
- L = {(n1,n2),(n1,n5),….(n3,nf1),…..}

II. Architecture de notre système :

L'architecture de notre système est représentée dans la figure suivante :

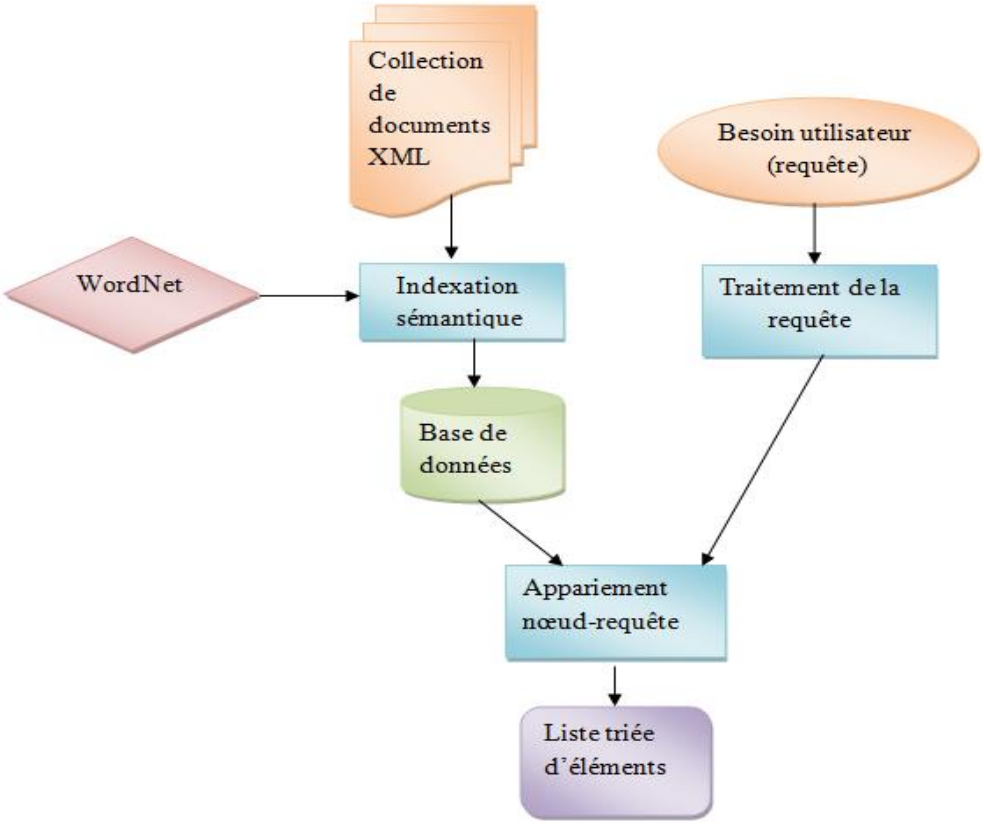


Figure 4.3 : Architecture générale de notre système.

Les principaux modules schématisés dans la figure ci-dessus sont :

- Le module d'indexation : qui parse les documents XML, tokenise le contenu textuelle de chaque nœud feuille, élimine les mots vides et racinise les mots utiles. Puis projette les descripteurs résultants dans l'ontologie afin d'extraire leur sens. Le résultat de cette étape est ensuite stocké dans une base de données.
- Le module de traitement de la requête : permet d'analyser la requête utilisateur pour générer une représentation exploitable par le SRI.

- Le module d'appariement nœud-requête : permet de comparer les représentations des nœuds à celle de la requête afin de renvoyer à l'utilisateur une liste triée d'éléments répondant à sa requête.

III. Les étapes de l'indexation classique :

L'indexation est une tâche sélective permettant de représenter le document par l'ensemble de ses descripteurs, qui permettent de véhiculer sa sémantique. Dans le cas des documents semi-structurés XML, l'indexation est appliquée aux nœuds feuilles étant donné que l'information textuelle est située à leur niveau.

Chaque document est d'abord parsé pour identifier les nœuds feuilles. Ensuite pour chaque nœud feuille identifié, on procède à la tokenisation de son contenu textuelle produisant une liste de termes. La prochaine étape est l'élimination des mots vides, pour cela on a utilisé une liste de mots vides. On ne garde ainsi, que les mots utiles, l'étape suivante est l'identification des collocations, les collocations sont définies comme des associations de mots récurrentes, et cela grâce a une projection des termes dans WordNet.

Dans cette étape, on commence par identifier les groupes de mots qui seront projetés. Une fenêtre de mots de dimension prédéterminée est alors fixée. Après avoir analysé les entrées de notre ontologie, la dimension de la fenêtre a été fixée à trois mots, ce qui permettra d'identifier les collocations composées de trois ou bien de deux mots.

Le principe est de sauvegarder l'ordre d'apparition des mots dans les nœuds feuilles, puis de projeter chaque groupe de trois mots et deux mots identifié dans l'ontologie, afin de déterminer s'il correspond à une entrée dans cette ontologie.

A l'issue de cette étape on obtient deux ensembles de termes :

- Les termes simples.
- Les collocations correspondantes à des entrées dans WordNet.

La dernière phase est la racinisation. Pour notre approche, on a utilisé l'algorithme de Porter pour l'anglais [Porter, 1980].

IV. Procédure de pondération :

La pondération permet d'attribuer pour chaque terme un poids traduisant son importance dans le nœud où il apparait ainsi que dans la collection de nœuds. La formule TF.IDF pour la pondération des termes dans les documents plats a été redéfinie pour s'adapter à l'aspect structurel des documents XML.

Deux facteurs sont définis comme suit :

- tf_j^{nfi} : La fréquence du terme t_j dans le nœud feuille nf_i, qui permet de rendre compte de l'importance locale du terme tj dans un nœud.
- $ief_j = \log \frac{|F_c|}{|nf_i|}$

Où :

$|F_c|$ est le nombre total de nœuds feuilles de la collection
$|nf_i|$ est le nombre de nœuds feuilles de la collection contenant le terme t_j

Le facteur ief_j permet de rendre compte de l'importance globale du terme dans la collection de nœuds.

La combinaison de ces deux facteurs [Sauvagnat, 2005] permet de déterminer le poids d'un terme tj dans un nœud nf_i comme suit :

$$W_j^{nf_i} = tf_j^{nfi} * ief_j$$

Ainsi, un nœud feuille est représenté par un ensemble de termes et de leur poids, comme suit :

$$nf_i = \{(t_1, W_1^i), (t_2, W_2^i), \dots . \} = \{(t_j, W_j^i)\}$$

V. Indexation sémantique :

L'indexation sémantique propose de représenter le document par le sens des mots plutôt que par les mots eux même. Pour cela, on use des ressources externes telles que les ontologies et les thésaurus qui nous permettent d'identifier le sens des mots.

Avant de décrire notre approche d'indexation sémantique, nous vous présenterons la ressource sémantique exploitée pour l'extraction des sens des mots, qui est l'ontologie WordNet.

V.1. Présentation de WordNet :

WordNet[9] est une base de données lexicale développée de puis 1985 par des linguistes du laboratoire des sciences cognitives de l'université de Princeton, sous la direction des deux professeur George A. Miller et Christiane Fellbaum. La première version diffusée remonte à juin 1991.Son but est de répertorier, classifier et mettre en relation de diverses manières le contenu sémantique et lexical de la langue anglaise.

[9] http://www.cogsci.princeton.edu

A l'origine, les concepteurs de WordNet ne prétendaient construire ni une structure conceptuelle, ni une ontologie, mais bien une ressource lexicale rendant compte de l'usage des mots et de leur mise en relation dans la langue. Ce n'est qu'ensuite que le réseau lexical de WordNet a été perçu comme une représentation conceptuelle qui pourrait tenir lieu d'ontologie.

V.1.1. Contenu deWordNet:

WordNet couvre la grande majorité des noms, verbes, adjectifs et adverbes de la langue anglaise. C'est un réseau de 144 684 termes organisés en 109 377 concepts. Le tableau suivant donne des statistiques sur le nombre de mots et de concepts dans WordNet :

Catégorie	Mots	Concepts
Nom	107 930	74 488
Verbe	10 806	12 754
Adjectif	21 365	18 523
Adverbe	4 583	3 612
Total	144 684	109 377

Tableau 4.1 : Nombre de mots et de concepts dans WordNet.

V.1.2. Notion de synset :

Le synset (set of synonyms) est la composante atomique sur laquelle repose WordNet, chaque synset dénote un concept, il comprend :

- un ensemble de mots quasi-synonymes, sorte de « classe d'équivalence » sémantique désignant un concept particulier, séparés par des virgules.
- la description du sens du concept (glose), mise entre parenthèse.
- éventuellement, un ou plusieurs exemples d'utilisation entre guillemets.

Les mots ayant plusieurs sens appartiennent à plusieurs synsets. Par exemple le mot « mouse » figure dans 6 synsets. Tel que présenté ci-dessous :

1. mouse -- (any of numerous small rodents typically resembling diminutive rats having pointed snouts and small ears on elongated bodies with slender usually hairless tails)
2. shiner, black eye, mouse -- (a swollen bruise caused by a blow to the eye)
3. mouse -- (person who is quiet or timid)
4. mouse, computer mouse -- (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 ; "a mouse takes much more room than a trackball")
5. sneak, mouse, creep, pussyfoot -- (to go stealthily or furtively ; "..stead of sneaking around spying on the neighbor's house")
6. mouse -- (manipulate the mouse of a computer)

V.1.3. Relation sémantique dans WordNet:

La relation de base entre les termes dans WordNet est la synonymie, les termes désignant un même concept forment une classe d'équivalence. Quand aux synsets, ils sont liés par des relations telles que spécifique-générique : hyponyme-hyperonyme (is-a) et la relation de composition meronymie-holonymie (part-of), définit ci-dessous :

- Hyperonymie désigne une classe de concepts englobant des instances de classes plus spécifiques : Y est un hyperonyme de X si X est un type de Y. Par exemple, "fruit" est un hyperonyme de "pomme" et de "cerise".
- Hyponymie désigne un membre d'une classe de concepts : X est un hyponyme de Y si X est un type de Y. Par exemple, "France" est hyponyme de "pays", "cheval" est hyponyme de "animal".
- Holonymie est le nom de la classe globale dont les noms méronymes font partie. Y est un holonyme de X si X est une partie de Y. Par exemple, "corps" est un holonyme de "bras", de même que "maison" est un holonyme de "toit".
- Méronymie est le nom d'une partie constituante, substance de ou membre d'une autre classe : X est un méronyme de Y si X est une partie de Y. Par exemple, "voiture" a pour méronymes "porte", "moteur", "roue", etc.

Ces quatre relations sont schématisées dans la figure suivante :

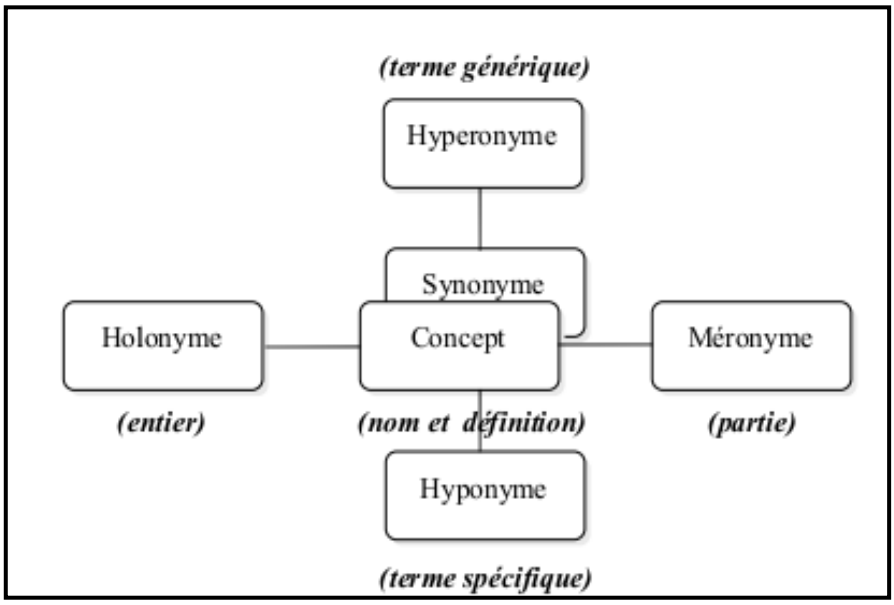


Figure 4.4 : Principales relations sémantiques dans WordNet.

V.2.Schéma d'indexation sémantique:

Le résultat de l'indexation classique est une liste de mots représentant chaque nœud feuille de la collection. La prochaine étape est l'identification des sens correspondants à ces mots.

Cette étape est décomposée en trois principales phases, qui sont :

- Identification des termes candidats.
- Identification des sens candidats pour chaque terme.
- Désambiguïsation des termes.

V.2.1. Identification des termes candidats :

On dénote par termes candidats, les termes qui correspondent a des entrés dans l'ontologie WordNet. A l'issue de cette étape seuls les termes (collocation et mots simple) qui ont une correspondance dans WordNet sont retenus.

V.2.2. Identification des sens candidats pour chaque terme :

Dans cette phase, les termes retenus dans l'étape précédente sont projeté dans l'ontologie WordNet, afin d'identifier pour chaque termes l'ensemble de sens lui correspondants.

Si le terme ne correspond qu'un seul sens dans l'ontologie celui-ci est retenu. Dans le cas contraire, si le terme correspond à plusieurs sens, on le désambiguïse afin de ne retenir qu'un seul sens.

V.2.3. Désambiguïsation des termes et mesure de similarité :

Les termes polysémiques (qui possèdent plusieurs sens) sont désambiguïsés. La désambiguïsation permet de retenir pour chaque terme un seul sens.

La désambiguïsation des termes est faite selon leur contexte d'apparition, c'est-à-dire selon les termes co-occurrents ou les termes voisins. On considère donc que les termes apparaissant ensemble (dans un même contexte) sont proche sémantiquement.

Le processus de désambiguïsation d'un terme t_k comprend les principales étapes suivantes [Harrathi, 2010b] :

- Identification du contexte du terme t_k
- Extraction des concepts correspondants aux termes du ce contexte de l'ontologie.
- Identification des différentes combinaisons possibles des concepts dans ce contexte.
- Calcule du score de similarité entre concepts pour chaque combinaison.
- Sélection de la combinaison qui maximise ce score de similarité.

Le contexte est définit comme un ensemble de termes qui apparaissent dans un même nœud. Il est noté comme suit :

Contexte= $\{t_1,t_2,\ldots\ldots t_k,\ldots t_n\}$

Où : n représente le nombre de terme dans le nœud considéré.

L'ensemble de sens correspondants à un terme t_k extraits de la ressource Ω est noté :

$Concept_{\Omega}(t_k) = \{c_k^1, \ldots\ldots c_k^i, \ldots\ldots c_k^m\}$

Où : c_k^i désigne le ième concept (sens) dénoté par le terme t_k et m le nombre de concept dénoté par le terme t_k (m=| $Concept_{\Omega}(t_k)$ |)

La désambiguïsation consiste à sélectionner une seul combinaison des concepts (sens) parmi les combinaisons possibles des concepts dans un contexte donné.

Combinaison = $\{c_1^{i_1}, \ldots\ldots, c_k^{i_k}, \ldots. c_n^{i_n}\}$ i_k= 1....| $concept_{\Omega}(t_k)$ |

Où :

$c_k^{i_k}$ désigne le i_k-ième concept dénoté par le terme t_k.

Le nombre de combinaisons N_c possibles pour un contexte contenant n terme est :

$N_c = \prod_1^{k=n} |concept_{\Omega}(t_k)|$

Le processus de désambiguïsation consiste donc à retrouver pour un terme dans un contexte donné le sens associé qui est sémantiquement le plus proche des sens des autres termes qui apparaissent dans ce contexte. Ainsi on choisit parmi les combinaisons de sens possible la combinaison vérifiant la condition du rapprochement sémantique. Pour cela on calcule pour chaque combinaison un score de similarité, puis on sélectionne la combinaison maximisant le score de similarité.

Soit CB= $\{c_1,c_2,\ldots\ldots c_n\}$une combinaison de concepts, on définit la similarité sémantique entre les concepts de CB comme la moyenne des similarités entre tous les concepts de CB.

$$Moyenne\_Sim(CB) = \frac{2.\sum_{i=1}^{i=n}\sum_{j=i+1}^{n} sim_{\Omega}(C_i,C_j)}{n(n-1)}$$

La combinaison retenue est CB_{max} qui a le maximum de moyenne de similarité entre ses concepts.

Max= ArgMax(Moyenne_Sim(CB_i)) avec $1 \leq i \leq \prod_1^{k=n} |concept_\Omega(t_k)|$

Où :

CB_i est l'i_ième combinaison de concepts du contexte.

t_k est le k-ème terme dans le contexte.

n est le nombre des termes dans le contexte.

Le problème de désambiguïsation est de nature combinatoire. Le nombre de scores de similarités à calculer dépend du nombre de combinaisons identifiées. Il faut donc déterminer le nombre minimal de termes les plus proches du terme à désambiguïser qui permettrons de sélectionner parmi ses sens le plus approprié selon son contexte. Ces termes forment une fenêtre du contexte. Une fenêtre peut être définie par m termes avant et après le terme cible.

L'utilisation d'une fenêtre permet de réduire la complexité de l'algorithme de désambiguïsation dans le cas où les termes présentent une ambiguïté élevée et la taille du contexte est très grande.

Pour une fenêtre de taille 3, la désambiguïsation des termes du contexte {t_1, t_2, t_3, t_4, t_5} se déroule de la façon suivante :

- Identification des concepts {c_1, c_2} à partir des combinaisons de la fenêtre {$concept_\Omega(t_1)$, $concept_\Omega(t_2)$, $concept_\Omega(t_3)$ }.
- Identification du concept {c3} à partir des combinaisons de la fenêtre {c_2, $concept_\Omega(t_3)$, $concept_\Omega(t_4)$ }.
- Identification des concepts {c_4, c_5} à partir des combinaisons de la fenêtre {c3, conceptΩ (t4), conceptΩ (t5) }.

Les travaux réalisés dans ce domaine démontrent qu'il n'existe pas de taille optimale fixe qui soit adaptée à tous les termes. Certaines études indiquent que désambiguïser en utilisant une fenêtre de ± 2 termes donne les mêmes résultats que de le faire en utilisant toute la phrase [Harrathi, 2010b].

Le processus de désambiguïsation repose essentiellement sur les scores de similarité calculés entre les concepts. Pour deux concepts c1 et c2, leur score de similarité reflète le degré de rapprochement sémantique entre eux. La fonction sim_Ω est définie comme suit :

$$sim_\Omega : \begin{cases} C_\Omega * C_\Omega \longrightarrow [0,1] \\ (c_1 * c_2) \longrightarrow sim_\Omega(c_1, c_2) \end{cases}$$

Où :

Ω représente la ressource sémantique.

C_Ω dénote l'ensemble des concepts de Ω.

$c_1$et c_2 deux concepts de C_Ω.

Le score de similarité est un réel compris entre 0 et 1. La valeur 0 signifie que c_1 n'est pas similaire à c_2. Quand à la valeur 1, elle signifie que c_1 est fortement similaire à c_2.

Parmi les mesures de similarité proposées dans la littérature, on retrouve la mesure de Wu-Palmer [Wu et al., 1994]. Cette mesure définit la similarité entre deux concepts par rapport à la distance qui sépare ces deux derniers dans la hiérarchie de l'ontologie, ainsi que par leur position par rapport à la racine. Elle propose la fonction suivante :

$$Sim_{WPalmer}(c_1, c_2) = \frac{2*prof(c)}{dist(c_1,c)+dist(c_2,c)+2*prof(c)}$$

Tel que :

c : représente le concept le plus spécifique ancêtre des deux concepts c_1 et c_2.

prof(c) : représente le nombre d'arcs qui sépare c de la racine.

$dist(c_i,c)$: représente le nombre d'arc qui séparent c_i de c.

Voici ci-dessous un exemple qui nous permet d'illustrer cette mesure :

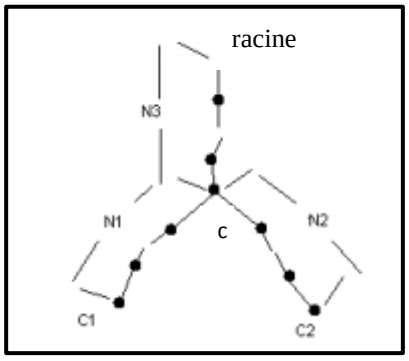


Figure 4.5 : Deux concepts et leur concept commun le plus spécifique dans une taxinomie

La formule précédente est ainsi réécrite :

$$Sim_{WPalmer}(c_1, c_2) = \frac{2*N_3}{N_1+N_2+2*N_3}$$

Une fois les termes du nœud feuille désambiguïsés, nous retenons un ensemble de concepts représentatifs du contenu sémantique de celui-ci. Ces concepts sont ensuite pondérés en utilisant la formule de pondération déjà présentée en section IV. Ainsi, un nœud feuille est représenté par un ensemble de concepts et de leur poids, comme suit :

$$nf_i = \{(c_1, W_1^i), (c_2, W_2^i), \dots . \} = \{(c_j, W_j^i)\}$$

Où : W_j^i représente le poids de j-ème concept dans le nœud nf_i.

VI. Appariement nœud-requête :

La phase d'appariement permet de sélectionner parmi les nœuds de la collection ceux qui répondent à une requête utilisateur, elle consiste à comparer la représentation de chaque nœud à celle de la requête et de lui associer un score de pertinence qui reflète le degré similarité entre ce nœud et la requête

Pour chaque document de la collection, on considère d'abord les nœuds feuilles. Un nœud feuille est représente par deux vecteurs:

Un vecteur concept :

$$nfc_i = \{wc_1^i, wc_2^i, \dots, wc_j^i, \dots wc_n^i\}$$

Où :
wc_j^i représente le poids du concept c_j dans le nœud nf_i
n représente le nombre de concepts associés au nœud feuille nf_i.

Un vecteur terme :

$$nft_i = \{wt_1^i, wt_2^i, \dots, wt_j^i, \dots wt_m^i\}$$

Où :
wt_j^i représente le poids du terme t_j dans le nœud nf_i
m représente le nombre de terme associés au nœud feuille nf_i.

De même une requête Q est représentée par deux vecteurs:

Un vecteur concept :

$$Qc = \{wc_1, wc_2, \dots, wc_j, \dots, wc_n\}$$

Où :
n représente le nombre de concepts associés à la requête Q.
wc_j représente le poids du concept c_j dans la requête Q.

Un vecteur terme :

$$Qt = \{wt_1, wt_2, \dots, wt_j, \dots, wt_m\}$$

Où :
m représente le nombre de termes associés à la requête Q.
wt_j représente le poids du terme t_j dans la requête Q.

Pour le calcul des poids des concepts (respectivement termes) apparaissant dans la requête, on a utilisé la formule de pondération brute qui est proportionnelle au nombre d'occurrences des concepts (respectivement termes) dans la requête, elle retourne pour chaque concept (respectivement terme) son nombre d'occurrences.

Le score de pertinence, noté RSV (nf_i, Q), est obtenu en utilisant la formule de cosinus. Nous calculons d'abord le score de pertinence entre les deux vecteurs concepts, comme suit :

$$RSVc\ (nf_i, Q) = \frac{\sum_{j=1}^{n} wc_j^i * wc_{jq}}{\sqrt{\sum_{i=1}^{n} wc_j^{i\,2}} * \sqrt{\sum_{i=1}^{n} wc_{jq}^2}}$$

Le principe de la formule de cosinus est de multiplier le poids de chaque concept c_j de la requête par son poids dans le nœud feuille, ainsi si le concept c_j ne figure pas dans le nœud feuille son poids prendrai la valeur 0. Cependant, l'intérêt de la recherche d'information sémantique est de sélectionner des nœuds feuilles qui ne compte pas nécessairement les mêmes concepts que la requête, mais des concepts proches.

A cet effet, nous proposons d'étendre la mesure de cosinus de la façon suivante :

$$RSVc\ (nf_i, Q) = \frac{\sum_{j=1}^{n} wc_{jq} * wc(\max(c_{jq}))}{\sqrt{\sum_{i=1}^{n} wc_j^{i\,2}} * \sqrt{\sum_{i=1}^{n} wc_{jq}^2}}$$

Où wc ($\max(c_{jq})$) représente le poids du concept c_k^i dans le nœud feuille nf_i qui maximise la similarité avec le concept c_{jq}. $\max(c_{jq})$ est ainsi déterminé [Harrathi, 2010b] :

$\max(c_{jq}) = \max(\ sim(c_{jq}, c_k^i)\ )$ et $sim(c_{jq}, c_k^i) >$ seuil, $1 < k < n$

La valeur du seuil est fixé à 0.8 afin d'inclure que les documents ayant des concepts fortement similaires aux concepts de la requête.

Puis, nous calculons le score de pertinence entre les deux vecteurs termes, comme suit :

$$RSVt\ (nf_i, Q) = \frac{\sum_{j=1}^{m} wt_j^i * wt_{jq}}{\sqrt{\sum_{i=1}^{m} wt_j^{i\,2}} * \sqrt{\sum_{i=1}^{m} wt_{jq}^2}}$$

Le score de pertinence RSV (nf_i, Q), est enfin calculer en sommant les deux scores obtenus précédemment, comme suit :

$RSV\ (nf_i, Q) = RSVc\ (nf_i, Q) + RSVt\ (nf_i, Q)$

Une fois que le score de pertinence des nœuds feuilles est calculé, il est propagé dans l'arbre du document XML vers les nœuds internes (ancêtres). On se base alors sur deux hypothèses qui sont :

- Les nœuds internes qui possèdent le plus grand nombre de nœuds feuilles pertinents, sont les plus pertinents.
- Plus grande est la distance entre un nœud feuille et son ancêtre, moins il contribue à sa pertinence

La valeur de pertinence d'un nœud interne n_i est calculée selon la formule suivante [Sauvagnat, 2005] :

$$RSV(n_i, Q) = \sum_{nf_k \epsilon F_n} \alpha^{dist(n_i, nf_k)-1} * RSV(nf_k, Q)$$

Tel que :

Fn représente l'ensemble des nœuds feuilles nf_k descendant du nœud interne n_i.

$\alpha \epsilon [0,1]$ est un paramètre qualifiant l'importance de la distance séparant les nœuds dans la formule de propagation.

dist (n_i,nf_k) représente le nombre d'arc séparant les deux nœuds.

VII. Evaluation et expérimentation :

Afin d'évaluer notre approche d'indexation sémantique des documents semi-structurés XML, nous avons eu recours à la compagne d'évaluation INEX. Cette campagne fournit une plate forme d'évaluation des systèmes de recherche d'information structurée. Elle fournit un corpus de documents, un jeu de requêtes et les jugements de pertinences associés à ces dernières.

VII.1. INEX 2009 :

VII.1.1. Collection de test:

Dans le but de valider notre travail, nous avons testé notre approche sur la collection de test INEX 2009[10]. Cette collection est dérivée des articles XML de Wikipédia (version anglaise), elle traite différentes entités telles que : les personnalités, les villes, les films … .

La collection INEX 2009 a été créée depuis le 8 Octobre 2008 et se compose de 2666190 articles ayant une taille totale de 50.7 GB.

La figure ci-dessous décrit un exemple d'un article de la collection :

[10] inex.otago.ac.nz

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE article SYSTEM "../article.dtd">
<articlexmlns:xlink="http://www.w3.org/1999/xlink">
<property  confidence="0.8" wordnetid="104916342">
<manner  confidence="0.8" wordnetid="104928903">
<header>
<title>Cardiac yoga</title>
<id>8317024</id>
<revision>
<id>190295680</id>
<timestamp>2008-02-10T01:31:49Z</timestamp>
<contributor>
<username>DumZiBoT</username>
<id>6085301</id>
</contributor>
</revision>
<categories>
<category>Yoga styles</category>
</categories>
</header>
<bdy>
<image width="150px" src="cardyo001.jpg" type="thumb">
<caption>
Cardiac yoga sample exercise
</caption>
</image>
<image width="150px" src="cardyo002.jpg" type="thumb">
<caption>
Cardiac yoga sample exercise
</caption>
</image>
……
<p>
<b>Cardiac yoga</b> is a system of <link xlink:type="simple" xlink:href="../475/255475.xml">
stress management</link> and <link xlink:type="simple" xlink:href="../172/3158172.xml">
health promotion</link> designed specifically to focus on the needs of a heart patient.  Cardiac yoga is basically artery gentle <link
xlink:type="simple" xlink:href="../258/34258.xml">
yoga</link> exercises tailored to the special needs of people who have various cardiac problems, live with a cardiac condition or recover
from cardiac diseases, and their families.
</p>
….
</bdy>
</manner>
</property>
</article>
```

Figure 4.6 : Extrait d'un document article.XML (INEX 2009).

La collection INEX 2009 contient un très grand nombre de balises. Nous avons donc sélectionné manuellement un petit ensemble T de type de doxels : T= {bdy, sec, p}.

Un doxel représente une partie structurelle d'un document XML (paragraphe, section, etc.). Le type t d'un doxel D correspond au nom de la balise XML de l'élément D. Considérons l'exemple suivant :<A> This is an <B>example</B> of <C> XML </C> document </A>

Ce document contient 3 doxels : le premier est délimité par la balise A, il est donc de type A, les deux autres sont de type B et C.

VII.1.2. Jeu de requête :

Autre la collection de test, INEX 2009 fournit un ensemble de requête (ou topic) composé de 115 requêtes. Les requêtes d'INEX se divisent en deux catégories :

- Les requêtes CO (Content Only) : Une requête CO est composée de simples mots clés, elle ne contient aucune indication de structure permettant de déterminer la granularité de l'information à renvoyer.
- Les requêtes CAS (Content And Structure) : Une requête CAO contient à la fois des mots clés et des contraintes sur la structure des documents.

Un exemple d'une requête INEX 2009 est illustré dans la figure suivante :

```
<topic id="2009022" ct_no="207">

<title>Szechwan dish food cuisine</title>

<castitle>

//article[about(.,cuisine) or about(.,dish) or about(.,food)]//sec[about(.,Szechwan) or
about(.,Sichuan)]

</castitle>

<phrasetitle>"Szechwan dish" "Szechwan food" "Szechwan cuisine"</phrasetitle>

<description>I want to find some famous Szechwan dishes.</description>

<narrative>

I am a Chinese, and I like the Szechwan dish very much. I want to find some famous
Szechwan dishes, so the information about the dishes of Szechwan food is relevant. But
the information about the chef, the restaurant or other dishes is irrelevant.

</narrative>
```

Figure 4.7 : Exemple de requête de la campagne INEX 2009.

Pour chaque requête, différents champs permettent d'expliciter le besoin de l'utilisateur : le champ 'title' donne la définition simplifiée de la requête, le champ 'castitle' donne la forme structuré de la requête (mots clés ainsi que contraintes sur la structure), le champ 'phrasetitle' contient un ensemble de mots clé, quant aux deux champs 'description' et 'narrative' explicités en langage naturel, indiquent les intentions de l'auteur.

VII.1.3. Jugements de pertinence :

La campagne INEX 2009 fournit en plus de la collection de documents et du jeu de requêtes, les jugements de pertinence qui font correspondre chaque requête à ses nœuds pertinents. Dont voici un extrait :

```
2009001 Q0 3260094 4213 4436 144 144:4213
2009001 Q0 21201 24903 33106 137 137:542 704:1871 2578:2506 5089:19984
2009001 Q0 52502 23232 39116 197 197:2085 2287:21147
2009001 Q0 19653466 16901 30413 288 288:2150 2458:14751
2009001 Q0 141921 22899 39637 124 124:11553 11685:11346
2009001 Q0 3260076 0 5071
2009001 Q0 80144 0 22137
```

Figure 4.8 : Extrait du fichier jugement INEX 2009.

Chaque ligne du fichier jugement est organisé en champs, tel que:

- Le premier champ représente l'identifiant de la requête.
- Le deuxième champ est inutilisable, il prend toujours la valeur Q0.
- Le troisième champ représente l'identifiant du document.
- Le champ quatre dénote le nombre de caractère pertinent présent dans le document.
- Le cinquième champ dénote le nombre de caractère du document.
- L'avant dernier champ dénote la position du premier caractère pertinent pour la requête.
- Le dernier champ est une série de passage pertinent retrouvée dans le document désigné par la position du premier caractère pertinent dans le passage ainsi que sa longueur, noté [<offset>:<length>]+.

VII.2. Environnement technologique :

L'implémentation de notre approche a été entièrement réalisée en java. Le choix de se langage de programmation a été effectué en raison du nombre d'API existantes que sa soit les API d'analyse des documents XML, ou bien les API d'accès a l'ontologie WordNet.

Voici dans ce qui suit une brève définition de chacune des API utilisées :

L'API JAWS : JAWS[11](Java API for WordNet Searching) est une API simple et rapide. Elle permet d'accéder à la base de données de WordNet donnant ainsi la possibilité aux applications java de récupérer des données à partir de cette base. Cette API est compatible avec les versions 2.1 et 3.0 des fichiers de base de données de WordNet. Nous avons donc intégré à notre modèle la version 2.1 de WordNet.

[11]http://lyle.smu.edu/~tspell/jaws/

L'API JWS : JWS[12] (Java Wordnet Similrity) est la version java d'une API initiale, Wordnet Similrity, implémenté en perl. Elle permet de mesurer la similarité entre deux concepts dans WordNet. Les mesures de similarité implémentés par l'API sont au nombre de dix dont la mesure de Wu_Palmer déjà définie plus haut.

L'API SAX : SAX[13] (Simple API for XML) est un parseur de documents XML. Il permet d'analyser le document afin d'en extraire le contenu des nœuds feuille. Contrairement à l'analyseur DOM qui nécessite la construction de l'arbre du document en mémoire, SAX est un parseur événementiel qui réagit à des événements tels que l'ouverture et la fermeture des balises puis renvoie leur contenu.

VII.3. Résultats expérimentaux :

Nos expérimentations ont portés sur 60 documents de la collection INEX 2009 répondant à 12 requêtes. Pour évaluer notre approche, nous avons recensé le nombre de doxels pertinents retournés par notre système.

VII.3.1. Comparaison entre l'indexation classique et l'indexation sémantique :

Dans cette section, nous comparons notre approche d'indexation sémantique à une approche d'indexation classique basée sur les mots clés. La comparaison est donnée par les histogrammes suivants qui schématisent l'évolution du nombre de doxels pertinents retournés par le système (Voir figures 4.9, 4.10, 4.11).

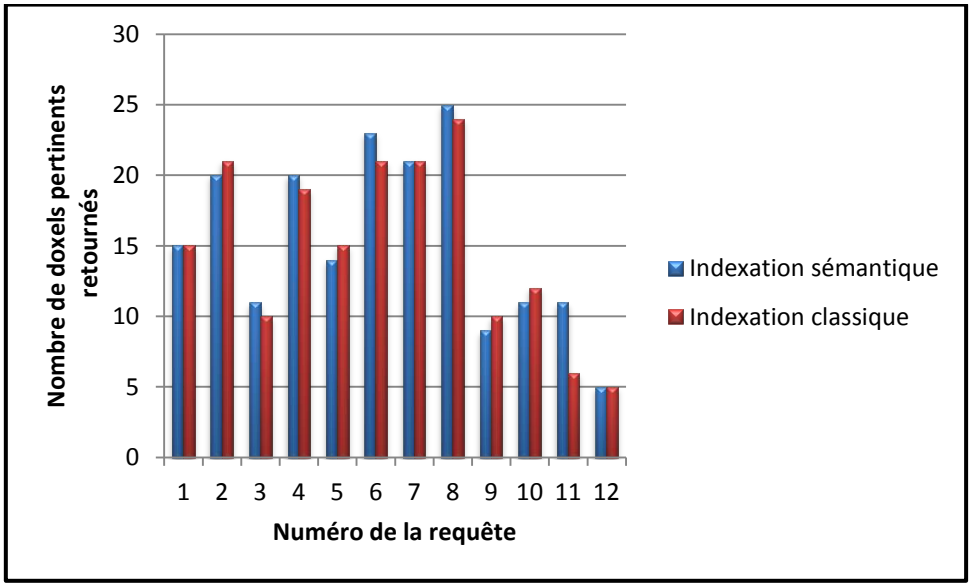


Figure 4.9 : Graphe d'évolution du nombre de doxels pertinents pour les 25 premiers doxels.

[12]https://code.google.com/p/wordnet-topic-dumper/source/browse/trunk/lib
[13]http://www.saxproject.org/

Nous constatons que le nombre de doxels pertinents retournés pour les requêtes 3,4,6,8 et 11 dans le cas de l'indexation sémantique est supérieur à celui retournés dans le cas de l'indexation classique. Quant aux requêtes 2, 5, 9 et 10, le nombre de doxels pertinents retourné dans le cas de l'indexation classique est supérieur à celui retourné dans le cas de l'indexation sémantique. Enfin pour les requêtes 1,7 et 12, le nombre de doxels pertinents retourné est le même dans le cas des deux approches.

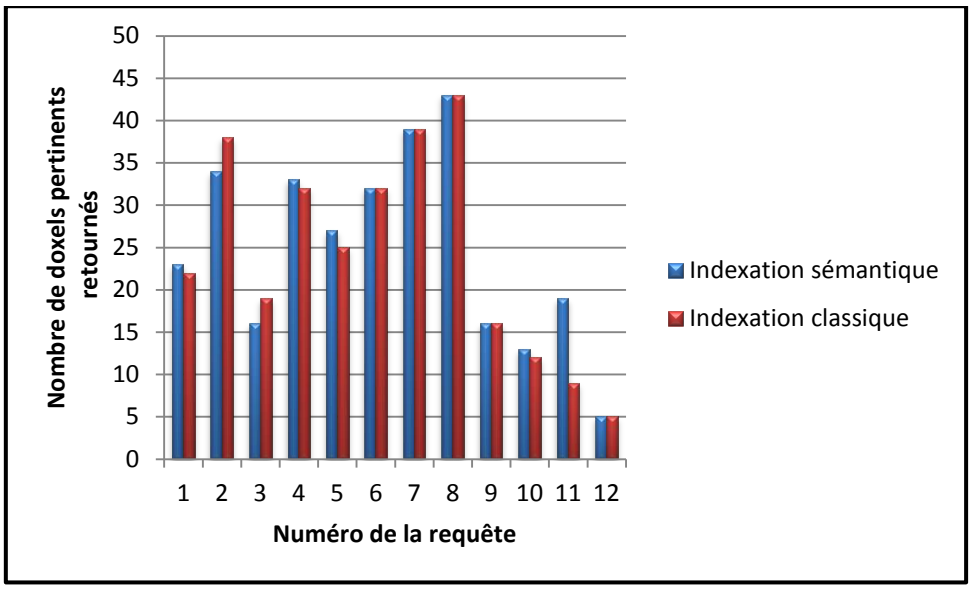


Figure 4.10 : Graphe d'évolution du nombre de doxels pertinents pour les 50 premiers doxels.

Nous remarquons que le nombre de doxels pertinents retourné pour les requêtes 1,4,5, 10 et 11 dans le cas de l'indexation sémantique est supérieur à celui retourné dans le cas de l'indexation classique. Quant aux requêtes 2 et 3 le nombre de doxels pertinents retourné dans le cas de l'indexation classique est supérieur à celui retourné dans le cas de l'indexation sémantique. Enfin pour les requêtes 6,7,8,9 et 12 le nombre de doxels pertinents retourné est le même dans le cas des deux approches.

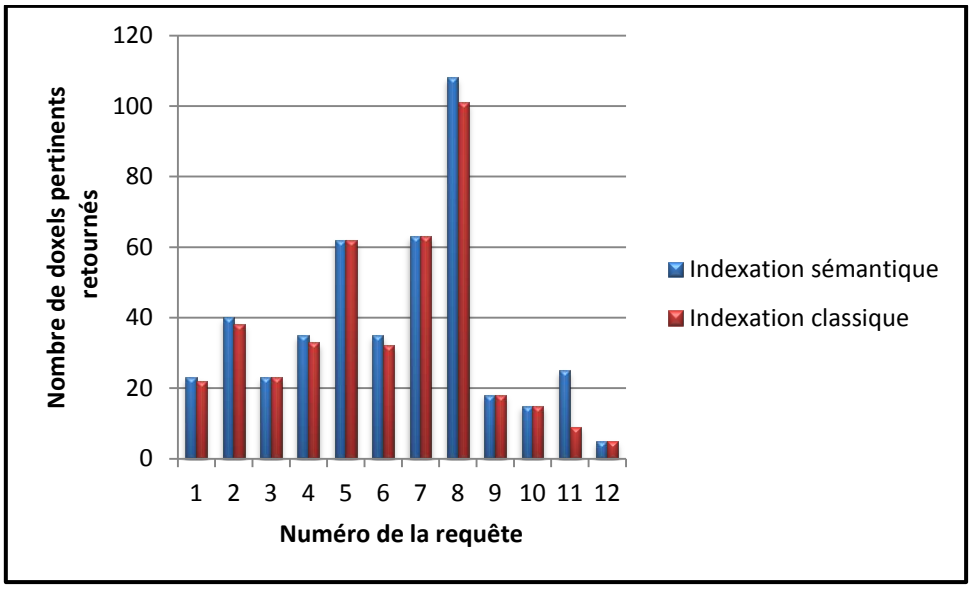


Figure 4.11 : Graphe d'évolution du nombre de doxels pertinents pour la totalité des doxels retournés.

Nous remarquons que les résultats obtenus et illustrés dans la dernière figure 4.11 dénotent une performance meilleure dans le cas de l'approche basée sur la sémantique des doxels et cela pour 50% des requêtes (soit 6 requêtes).

VII.3.2. Evaluation de l'impact du facteur dist(n_i, nf_k) (voir pages 81) **:**

Une fois le score des nœuds feuilles calculé, celui-ci est propagé dans un document aux nœuds internes en prenant en considération la distance entre un nœud interne et ses nœuds feuilles descendants. A cette fin, un paramètre α est introduit (défini dans la section **VI** de la page 81). Pour effectuer l'évaluation, nous faisons varier la valeur du paramètre α dans un intervalle [0.5, 0.9]. La valeur 0.5 dénote une forte importance du facteur distance, cette importance diminue en augmentant la valeur de α. Les résultats obtenus sont représentés dans les figures 4.12 et 4.13 :

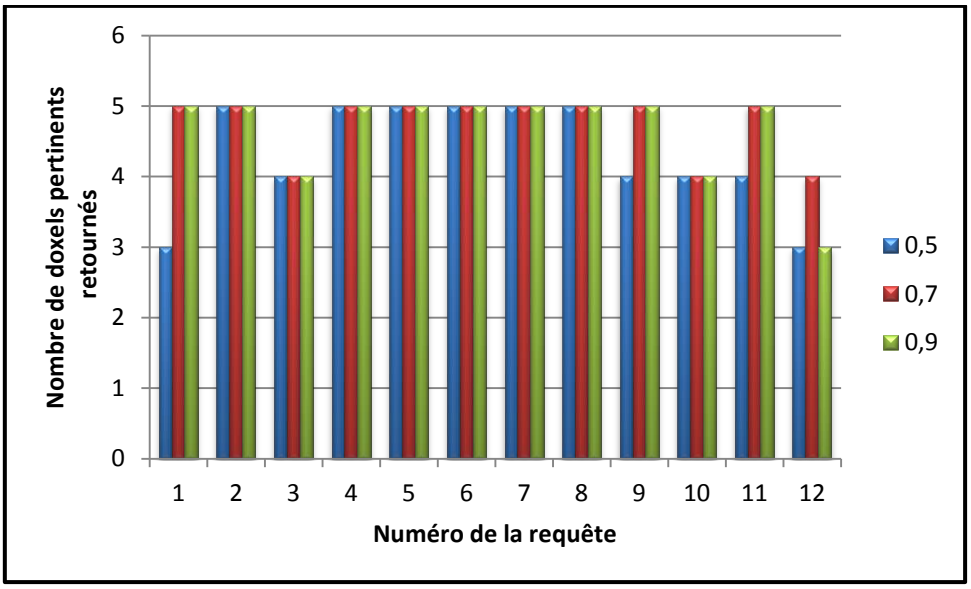


Figure 4.12 : Graphe d'évolution du nombre de doxels pertinents aux 5 premiers doxels en fonction de α.

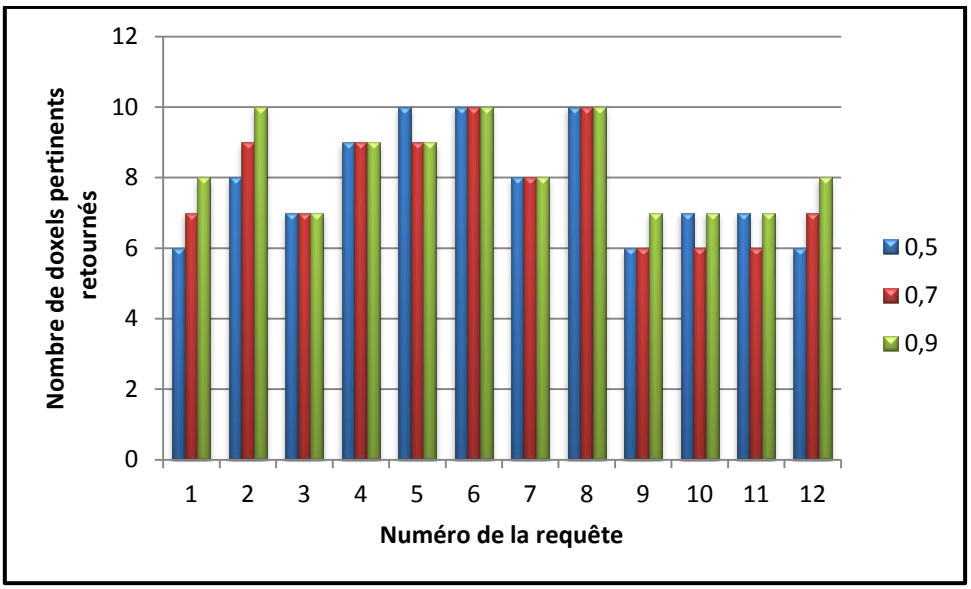


Figure 4.13 : Graphe d'évolution du nombre de doxels pertinents aux 10 premiers doxels en fonction de α.

Nous remarquons dans les deux figures ci-dessus que les performances de notre système restent invariables dans le cas des requêtes 3,4,6,7 et 8. Ce qui ne nous permet pas de tirer une conclusion sur l'impact du facteur distance car cela nécessite des expérimentations sur un jeu de requêtes plus important.

Conclusion :

Ce chapitre a été dédié à la présentation de notre système. Nous avons décrit la représentation en arbre des documents XML, le schéma d'indexation classique, qui permet de sélectionner les descripteurs de chaque nœud feuille de la collection. Ainsi que le schéma d'indexation sémantique qui permet de faire correspondre chaque descripteur avec son sens extrait de l'ontologie WordNet. Nous avons aussi présenté un schéma d'appariement nœud-requête qui permet de sélectionner parmi les nœuds de la collection de test, ceux qui répondent au besoin de l'utilisateur.

Nous avons enfin, présenté l'environnement technique de développement de notre système ainsi que les résultats de nos expérimentations.

Conclusion générale

Conclusion générale

Notre mémoire dont le thème est « indexation sémantique du contenu des documents XML » se situe dans le contexte général de la recherche d'information, et plus particulièrement dans le cadre de la recherche d'information semi-structurée.

A la différence de la recherche d'information classique qui manipule des documents plats, la recherche d'information semi-structurée (RIS) manipule des documents semi-structurés caractérisés par la structuration de leur contenu textuel.

Les documents semi-structurés permettent de mieux cerner le besoin de l'utilisateur, car le but d'un système de recherche d'information semi-structurée (SRIS) est de renvoyer à l'utilisateur des unités d'information (nœuds) focalisées sur son besoin, et non plus des documents entiers. A cet effet de nouvelles méthodes concernant l'indexation, le stockage et l'interrogation des documents doivent être développées.

Nous nous intéressons dans notre mémoire aux méthodes d'indexation proposées pour les documents semi-structurés XML. Nous avons d'abord exposé la problématique liée à l'indexation classique se basant sur les mots clés ainsi que ses limitations.

Afin de s'affranchir des limites de l'indexation classique, l'indexation sémantique a été introduite. Elle se base sur la représentation par les sens des mots extraits d'une ressource externe (ontologie WordNet). L'approche présentée dans ce mémoire est décomposée en trois étapes :

- La première étape consiste à identifier les termes possédant des entrées dans l'ontologie WordNet.
- La deuxième étape consiste à identifier les sens de chaque terme retenu.
- La dernière étape est la phase de désambiguïsation.

La désambiguïsation s'applique aux termes polysémiques possédant plusieurs sens et vise à sélectionner un seul sens, le plus adéquat au contexte du terme en se basant sur les mesures de similarité qui calculent un score de similarité (rapprochement) entre deux sens.

Nous avons aussi proposé une fonction d'appariement. Elle aussi basée sur la même mesure de similarité et qui permet de renvoyer à l'utilisateur une liste de doxels triée par ordre décroissant de pertinence.

Nous avons enfin testé notre approche en se basant sur la collection INEX 2009. Nous avons comparé notre approche sémantique à une approche classique, ce qui a démontré que le nombre de doxels pertinents retournés par notre système est supérieur au nombre de doxels pertinents retournés dans le cas de l'indexation classique.

Les perspectives de notre travail sont les suivantes :

- Effectuer nos expérimentations sur une collection plus volumineuse afin d'affiner les résultats obtenus et tester notre approche en utilisant différentes mesures de similarité.

Conclusion générale

- Indexation sémantique de la structure des documents XML, ce qui permettra à l'utilisateur d'exprimer des requêtes orientées contenu et des requêtes orientées « structure et contenu ».
- Exploiter les concepts de la fouille de donnée dans le domaine de la recherche d'information tel que la classification par caractéristique qui permet de catégoriser les documents en classes en se basant par exemple sur la sémantique véhiculée par chaque document.

Annexe

Annexe I

Technologies XML :

De nombreuses technologies et langages se sont développés autour de XML. Ceux-ci enrichissent les outils pour la manipulation des documents XML. Voici ci-dessous les principaux langages qui font partie de l'environnement XML.

- **XSL :** eXtensible Stylesheet Language est un langage de feuilles de style extensible développé spécialement pour XML. Recommandé par le W3C pour effectuer la représentation des données de documents XML. Ce langage permet la transformation d'un document XML en n'importe quel type de fichier texte (PDF, HTML..) ou encore en un autre document XML.

- **XSLT :** Une feuille de style XSLT (eXtensible StyleSheet Language Transformation) contient des règles qui décrivent des transformations. Ces règles sont appliquées à un document source XML pour obtenir un nouveau document XML résultat. Le langage XSLT est aussi souvent utilisé pour réaliser des transformations simples sur des documents. Il s'agit, par exemple, de supprimer certains éléments, de remplacer un attribut par un élément ou de déplacer un élément.

- **XPath :** XPath est un langage d'expressions permettant d'écrire des chemins dans l'arbre d'un document XML. Ces chemins décrivent des ensembles de nœuds du document qu'il est ainsi possible de manipuler. La version 2.0 de XPath a considérablement enrichi le langage. Il est devenu un langage beaucoup plus complet capable, par exemple, de manipuler des listes de nœuds et de valeurs atomiques.

- **XPointer :** XML Pointer est langage qui permet d'adresser des éléments sélectionnés dans la structure des documents XML c'est-à-dire faire correspondre une URL à un fragment de document XML. XPointer utilise la syntaxe XPath, enrichie d'options permettant de désigner des portions de document.

- **XLink :** est une spécification du W3C .Cette technologie permet de créer des liens entre documents XML ou portions de documents XML (grâce à XPointer). Contrairement aux liens entre fichiers HTML, XLink permet de créer des liens liant plus de deux documents.

- **XQuery :** XML Query ou XQuery est un langage de requête permettant donc d'extraire des informations d'un document XML. Proche sémantiquement de SQL, ce langage utilise la syntaxe XPath pour adresser des parties spécifiques d'un document XML.

- **SVG :** SVG (Scalable Vector Graphics) est un dialecte de XML pour le dessin vectoriel. Qui permet de décrire des graphes à deux dimensions en XML.

- **XHTML** : Extensible HyperText Markup Language est un langage de balisage servant à l'écriture de pages du World Wide Web. Conçu pour succéder au langage HTML,

Annexe I

XHTML est fondé sur la syntaxe définie par XML, plus récente et plus simple que la syntaxe définie par SGML sur laquelle repose HTML.

L'analyseur des documents XML :

XML permet de définir la structure du document uniquement. Toutefois la récupération des données encapsulées dans le document nécessite un outil spécifique appelé analyseur syntaxique, ou parseur. L'analyseur syntaxique est un outil logiciel permettant de parcourir un document et d'en extraire les informations qu'il contient.

Dans le cas de XML on parle alors de parseur XML, On distingue deux types de parseurs XML :

- les parseurs validants (validating) permettant de vérifier qu'un document XML est conforme à sa DTD
- les parseurs non validants (non-validating) se contentant de vérifier que le document XML est bien formé (c'est-à-dire respectant la syntaxe XML de base)

Les analyseurs XML sont également divisés selon l'approche qu'ils utilisent pour traiter le document. On distingue actuellement deux types d'approches :

- Les API utilisant une approche **hiérarchique** : les analyseurs utilisant cette technique construisent une structure hiérarchique contenant des objets représentant les éléments du document, et dont les méthodes permettent d'accèder aux propriétés. La principale API utilisant cette approche est **DOM** (Document Object Model)
- Les API basés sur un mode **événementiel** : elles permettent de réagir à des événements, comme le début d'un élément ou sa fin, et de renvoyer le résultat à l'application utilisant cette API. **SAX** (Simple API for XML) est la principale interface utilisant l'aspect événementiel.

Parcours d'un arbre :

Il existe plusieurs manières de parcourir les arbres. Les plus utilisées sont :

– Le parcours en pré-ordre (ou préfixe) : chaque nœud est visité avant chacun de ses fils parcourus de gauche à droite.
– Le parcours en post-ordre (ou postfixe) : chaque nœud est visité après chacun de ses fils parcourus de la gauche vers la droite.
– Le parcours en in-ordre (ou infixe) : chaque nœud est visité depuis ses frères disponibles de gauche à droite.

Annexe I

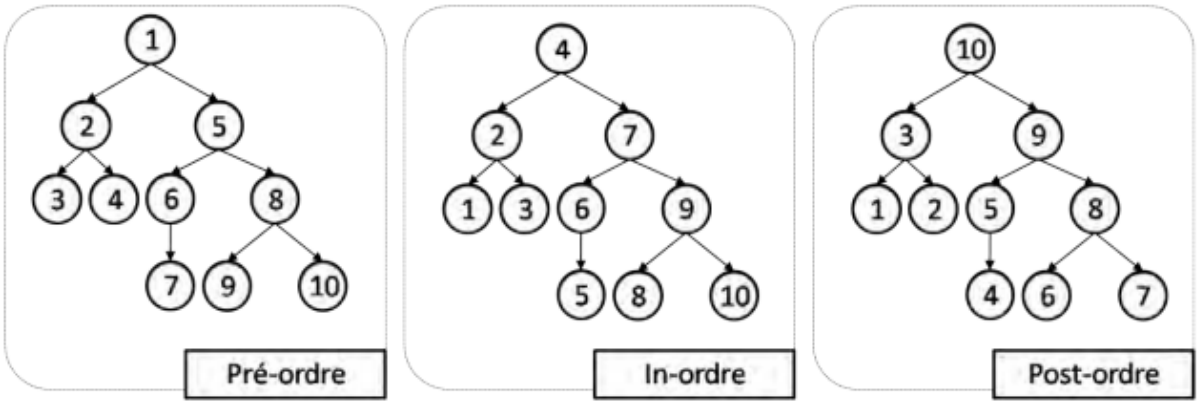


Figure I.1 : schéma des trois approches de parcours d'arbres les plus répandues.

Bibliographie

Bibliographie

[Baziz et al., 2005] : Baziz M., Indexation Conceptuelle Guidée Par Ontologie Pour La Recherche d'Information. Thèse de Doctorat en Informatique de l'Université Paul Sabatier de Toulouse (Sciences). Décembre 2005.

[Belkin et al., 1992] : Nicholas J. Belkin, Peter Ingwersen, Annelise Mark Pejtersen. « Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval ». Copenhagen, Denmark, June 21-24, 1992 ACM 1992.

[Boubekeur, 2008] : Boubekeur F. "Contribution à la définition de modèles de recherche d'information flexibles basés sur les CP-Nets", thèse de doctorat en informatique, Université Paul Sabatier. 2008.

[Boughanem, 1992] : M. Boughanem ."les Systèmes de Recherche d'Information: d'un modèle classique à un modèle connexionniste", Thèse de Doctorat de l'Université Paul Sabatier, 1992.

[Bouidghaghen, 2007] : Bouidghaghen O, Prise en compte de l'hétérogénéité structurelle en recherche d'information semi-structurée. Thèse de Magister, Université M'hamed Bougara-Boumerdes. 2007.

[Bordogna et Pasi, 2000] : Bordogna, G. et Pasi, G. Flexible querying of structured documents. In FQAS'00, pages 350–361. 2000.

[Bosc et Prade, 1996] : Bosc, P. et Prade, H. (1996). An Introduction to the Fuzzy Set and Possi-bility Theory-Based Treatment of Soft Queries and Uncertain Or Imprecise Databases. In Uncertainty Management in Information Systems.

[Chebili, 2011] : Hicham CHEBILI. « Agrégation des résultats dans la recherche d'information Semi-Structurée». Mémoire de Magister. Ecole supérieure d'informatique (E.S.I) Oued-Smar Alger, Ecole doctorale nationale en sciences et technologies de L'information et de la communication –STIC, 2011.

[Cleverdon, 1967] : Cleverdon, C. (1967). The Cranfield tests on index language devices. In Aslib Proceedings, pages 173–194.

[Cleverdom, 1970] : Cleverdom, C. «Progress in documentation .Evaluation of information systems », Journal of Documentation, 1970.

[Dinh et al., 2012] : Ba-Duy Dinh, Lynda Tamine-Lechani. « Accès à l'information biomédicale : vers une approche d'indexation et de recherche d'information conceptuelle basée sur la fusion de ressources termino-ontologiques», Thèse de Doctorat en Informatique de l'Université Paul Sabatier, Toulouse, 2012.

Bibliographie

[Denos, 1997] : Denos, N. « Modélisation de la pertinence en recherche d'information: modèle conceptuel, formalisation et application ». Thèse de Doctorat de l'Université Joseph Fourier-Grenoble I, 1997.

[Dubois et Prade, 1988] : Dubois, D. et Prade, H. (1988). Possibility theory : an approach to compu-terized processing of uncertainty. Plenum press.

[Florescu, 1999] : D. Florescu and D. Kossmann. Storing and querying XML data using an RDBMS. IEEE Data Engineering Bulletin, 22(3) : p.27–34, 1999.

[Foltz, 1990] : P. W. Foltz, Using Latent Semantic Indexing for information filtering.CACM, pp. 40-47, 1990

[Foltz, 1990] : P. W. Foltz, Using Latent Semantic Indexing for information filtering.CACM, pp. 40-47, 1990.

[Fuller et al., 1993] : M. Fuller, E. Mackie, R. Sacks-Davis, and R. Wilkinson. Structural answers for a large structured document collection. In Proceedings of ACM SIGIR 1993, Pitthsburgh, pages 204–213, 1993.

[Fuhr et al., 2003] : N. Fuhr and K. Grossjohann. XIRQL : a query language for information retrieval in XML documents. In In Proceedings of SIGIR 2001, Toronto, Canada, 2003.

[Furnas et al., 1987] : Furnas, G.W., Landauer, T.K., Gomez, L.M., and S.T. Dumais.: The Vocabulary Problem in Human-System Communication, Communications of the ACM 30 (1987) 964-971.

[Harrathi, 2010a] : Harrathi R, Calabretto S. « Une approche de recherche sémantique dans les documents semi-structurés». Lyon, 2010.

[Harrathi, 2010b] : Rami Harrathi. Recherche d'information conceptuelle dans les documents semi-structurés. Thèse de doctorat, Institut National des Sciences Appliquées de Lyon. 2010.

[Hatano et al., 2002] : K. Hatano, H. Kinutani, M. Yoshikawa, and S. Uemura. Information Retreival System for XML Documents. 2002.

[Harter, 1992] : Harter, S. «Psychological relevance and information science », Journal of the American Society for information Science (JASIS),1992.

[Hlaoua, 2007] : Hlaoua L. Reformulation de Requêtes par Réinjection de Pertinence dans les Documents Semi-Structurés. Thèse de doctorat, Université Paul Sabatier.2007.

Bibliographie

[Kwok, 1989] : K.L. Kwok, A neural network for probabilisticinformation retrieval. 12th International ACM SIGIR Conference on Research and Developpement in Information Retrieval, pp 21-30, 1989.

[Kamps et al., 2004] : J. Kamps, M. de Rijke. And B. Sigurbjornsson, Length normalization in XML retrieval. In Proceedings of SIGIR 2004, Sheffield, England, pages 80-87, 2004.

[Luhn, 1958] : Luhn, H. « The automatic creation of literature abstracts ». IBM Journal of Research and Development 24, 2 (1958), 159–165.

[Luk et al., 2002] : R. W. Luk, H. Leong, T. S. Dillon, A. T. Chan, W. B. Croft, and J. Allan. A survey in indexing and searching XML documents. Journal of American Society for Information Science and T e chnolo gy (JASIST), 2002.

[Maron et al., 1960] : Maron, M., and Kuhns, J. On relevance, probabilistic indexing and information retrieval. Journal of the Association for Computing Machinery 7 (1960), pages 216–244.

[Mass et al., 2002] : Y. Mass, M. Mandelbrod, E. Amitay, D. Carmel, Y. Maarek, and A. Sof-fer. JuruXML- an XML retrieval system at INEX'02. In Proceedings of INEX 2002, Dagstuhl, Germany, pages 73–80, 2002.

[Mass et al., 2003] : Y. Mass and M. Mandelbrod. Retrieving the most relevant XML components. In Proceedings of INEX 2003, Dagstuhl, Germany, 2003.

[Mathias, 2002] : Mathias G. « Indexation et interrogation de chemins de lecture en contexte pour la Recherche d'Information Structurée sur le Web ». Thèse de Doctorat, Université JOSEPH FOURIER - GRENOBLE I, 2002.

[Mezzaro, 1997] : Mezzaro, S. « Relevance, the whole (hi) story » Journal of American Society for information Science,1997.

[Paice, 1984] : Paice, C. D. (1984). Soft evaluation of Boolean search queries in information retrieval systems. Inf. Technol. Res. Dev. Appl., 3:33–41.

[Piwowarski, 2002] : B. Piwowarski, G.-E. Faure, and P. Gallinari. Bayesian networks and INEX. In Proceedings in the First Annual Workshop for the Evaluation of XML Retrieval (INEX), December 2002.

[Piwowarski et Lalmas, 2004] : B. Piwowarski et M. Lalmas. Interface pour l'évaluation de systèmes de recherche sur des documents xml. Université Paris 6, France. In CORIA, 2004.

[Porter, 1980] : Porter, M. F. An Algorithm for Suffix Stripping. Program 14(3),1980 : 130-137.

Bibliographie

[Roberston, 1994] : Robertson, S., Walker, S., Jones, S., Hancock-Beaulieu, M. et Gat-ford, M. (1994). Okapi at TREC-3. In Proceedings of the Third Text REtrieval Conference, TREC'94.

[Robertson et al., 1997] : S. E. Robertson and S. Walker. « On relevance weights with little relevance information ». In Proceedings of the 20th annual international ACM SIGIR conference on Research and development in information retrieval, pages 16–24. ACM Press, 1997.

[Salton, 1970] : Salton, G.« Evaluation problems in interactive information retrieval. Information Storage and Retrieval », 6(1):29–44. 1970.

[Salton, 1971] : G. Salton, The SMART retrieval system : Experiments in automatic document processing. Prentice Hall, 1971.

[Salton et al., 1983] : Salton, G., E.A. Fox, H.Wu. « Extended Boolean information retrieval System ». CACM 26(11), pp. 1022-1036, 1983.

[Saracevic, 1996] : Saracevic, T. « Relevance reconsidered ».Conception of Library and InformationScience, 1996.

[Sarasevic, 1970] : Saracevic, T. « The concept of "relevance" in information science : a historical review », dans T Saracevic (dir), Introduction to Information science. R.R. Bowker, New York, 1970.

[Sauvagnat, 2005] : Karen Sauvagnat. Modèle flexible pour la Recherche d'Information dans des corpus de documents semi-structurés. Thèse de doctorat, IRIT, Université Paul Sabatier de Toulouse. 2005.

[Singhal et al., 1996] : A. Singhal, C. Buckley, M. Mitra. « Pivoted document length normalization ». In Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval . Zurich, Switzerland .Pages: 21 - 29 . 1996.

[Turtle et al., 1990] : H. Turtle, W. B. Croft, Inference networks for document retrieval. Proceedings of ACM SIGIR 90, pages: 1-24, 1990.

[Turtle et al., 1991] : H.Turtle et W.B.Croft , Evaluation of an Inference Network Based Retrieval Model, ACM transaction on Information Systems July, pages : 70-77, 1991.

[Wong et al., 1885] : Wong, S., Ziarko, W. etWong, P. Generalized vector spaces model in information retrieval. In Proc. of the 8th ACM-SIGIR conference, pages 18-25. Montreal, Quebec. 1985.

Bibliographie

[Wu et al., 1994] : Wu Z. & Palmer M. Verb Semantics and Lexical Selection, Proceedings of the 32nd Annual Meetings of the Associations for Computational Linguistics, pages 133-138. 1994.

[W3C, 2005] : Document object model (dom). Rapport technique, 2005.

[Zargayouna, 2004] : Zargayouna H. Contexte et sémantique pour une indexation de documents semi-structurés. LIMSI/CNRS-Université Paris 11, 2004.

[Zargayouna, 2005] : Zargayouna H. Indexation sémantique de documents XML. Thèse de doctorat, Université Paris XI, UFR scientifique d'Orsay. 2005.