N° d'ordre: 10/2012-D/INF

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE DES SCIENCES ET DE LA TECHNOLOGIE HOUARI BOUMEDIENE
FACULTE D'ELECTRONIQUE ET D'INFORMATIQUE

THESE

Présentée pour l'obtention du grade de DOCTEUR
En : INFORMATIQUE
Spécialité : Intelligence artificielle et Bases de Données avancées

Par : ***Bessai Fatma Zohra née Mechmache***

Sujet

Modèle possibiliste pour la recherche d'informations agrégée dans des corpus de documents semi structurés

Soutenue publiquement, le 12/12/2012, devant le jury composé de :

Mme. AISSANI MOKHTARI Aicha	*Professeur à l'USTHB*	*Présidente*
Mme. ALIMAZIGHI Zaia	*Professeur à l'USTHB*	*Directrice de thèse*
M. BOUGHANEM Mohand	*Professeur à l'IRIT Toulouse*	*Co-Directeur de thèse*
Mme. ABED Hafida	*Professeur à l'U. Blida*	*Examinatrice*
M. KECHID Samir	*Maître de conférences, USTHB*	*Examinateur*
M. AHMED OUAMER Rachid	*Maître de conférences, U. Tizi Ouzou*	*Examinateur*

Résumé

Nous nous sommes intéressés, dans ce travail, à la recherche d'information XML qui vise à récupérer non pas un ensemble de documents pertinents, mais un ensemble d'éléments (parties du document) pertinents par rapport à une requête.
Grâce à la théorie des possibilités et plus particulièrement au réseaux possibilistes, nous proposons un nouveau modèle de recherche d'information XML qui permet une sélection automatique d'un ensemble d'éléments pertinents provenant de différentes parties du document XML. Les relations termes-éléments et éléments-document sont modélisées par des mesures de possibilité et de nécessité. Dans ce modèle, la requête de l'utilisateur déclenche un processus de propagation pour retrouver des portions de documents nécessairement ou au moins possiblement pertinents par rapport à la requête.

Ce modèle permet de revisiter la granularité de l'unité d'information retournée. En effet, au lieu de retourner une liste d'éléments séparément, comme il se fait habituellement dans la plupart des systèmes de recherche d'information XML, notre modèle essaye de générer la meilleure agrégation des éléments pertinents susceptible de répondre au mieux au besoin de l'utilisateur formulé à travers une liste de mots clés.

Mots clés : Recherche d'information XML, Document XML, Théorie des possibilités, Réseaux possibilistes, agrégation des éléments, Recherche d'informations agrégée.

Abstract

We were interested, in this work, in XML information retrieval which aims to retrieve not a set of relevant documents, but a set of elements (parts of document) relevant to a query.
Thanks to possibilistic theory and more especially to possibilistic networks, we propose a new XML information retrieval model, which allows an automatic selection of a set of relevant elements from various parts of XML document. Relations terms-elements and elements-document are modeled through possibility and necessity. In this model, the user's query starts a process of propagation to recover parts of document necessarily or at least possibly relevant.

This model revisits the granularity of the unit of information returned. Indeed, instead of returning a list of elements separately, as it is usually done in most XML information retrieval systems, our model tries to build the best aggregation of relevant elements which is likely to be relevant to a query composed of key words.

Keywords: XML Information Retrieval, XML document, Possibilistic Theory, Possibilistic Network, Elements Aggregation, Aggregated search.

A mes parents, mon mari, mes enfants,
mes sœurs et mon frère.

Cherchons comme cherchent ceux qui doivent trouver et trouvons comme trouvent ceux qui doivent chercher encore. Car il est écrit, celui qui est arrivé au terme ne fait que commencer.

Saint Augustin

Remerciements

Je suis heureuse de pouvoir exprimer mes vifs remerciements au Professeur Nadjib Badache, Directeur du CERIST, pour m'avoir encouragé et soutenu dans la réalisation de cette thèse. Qu'il trouve ici l'expression de mon profond respect.

Je tiens tout particulièrement à exprimer toute ma gratitude et mes vifs remerciements à mes Directeurs de thèse Madame Zaia Alimazighi et Monsieur Mohand Boughanem pour m'avoir encadré durant ces années de thèse et m'avoir aidé à mener à bout ce travail, mais aussi pour leurs précieux conseils et réflexions critiques sur mes propositions qui ont été d'un grand apport pour la finalisation de ce travail. Qu'ils soient assurés de mon très grand respect.

Je remercie également les membres du jury :
Mme Aissani Mokhtari Aicha, professeur à l'université des Sciences et de la Technologie Houari Boumediene pour m'avoir fait l'honneur de présider ce jury; Mme Bouarfa Abed Hafida, professeur à l'université Saad Dahlab de Blida ; Mr Kechid Samir, maître de conférences à l'université Houari Boumediene et Mr Ahmed Ouamer Rachid, maître de conférences à l'université Mouloud Mammeri de Tizi Ouzou pour avoir accepté d'évaluer mon travail.

Pour finir, je remercie mes parents, toute ma famille, mon mari, mes enfants chéris Selma, Mohamed Cherif, Zahir et Nazim ainsi que ma chère amie et sœur Bensefia Hassina pour leur amour et leur soutien.

J'adresse aussi mes remerciements à tous mes amis et collègues du Centre de Recherche sur l'Information Scientifique et Technique (CERIST) pour m'avoir encouragé durant toutes ces années.

TABLES DES MATIERES

Deuxième Parie: Un Modèle Possibiliste pour la Recherche d'Information Structurée

LISTE DES FIGURES

LISTE DES TABLEAUX

Introduction générale

Nous vivons dans un monde où l'information est omniprésente, que ce soit dans les journaux, dans les magazines, à la télévision ou sur Internet, l'information est disponible dans une multitude de formats tous plus accessibles les uns que les autres.
Augmenter la quantité d'information qui déferle sur tout un chacun peut sembler une amélioration de la qualité de vie, mais au delà d'une certaine limite, des outils s'avèrent nécessaires pour pouvoir l'analyser, en tirer profit et même en rejeter lorsque cela s'avère nécessaire. Parmi ces outils, on trouve les Systèmes de Recherche d'Information (SRI) dont le rôle est de fournir des techniques et des modalités permettant de sélectionner, de manière automatique, l'information pertinente répondant à des besoins en information de l'utilisateur. Les travaux présents dans ce mémoire rentrent précisément dans le domaine de la Recherche d'Information (RI).

Contexte de travail

La recherche d'information automatisée permet à un utilisateur de formuler un besoin d'information, à l'aide d'une requête, pour obtenir une réponse issue d'un ensemble de documents. L'idée est apparue dès la naissance des premiers ordinateurs. Depuis, avec la gigantesque augmentation des connaissances produites et conservées numériquement, elle est devenue un secteur stratégique pour beaucoup d'entreprises et elle a indiscutablement bouleversé, par l'intermédiaire de l'Internet, le rapport de tout un chacun avec l'information. Aussi, vu la croissance sans cesse de la quantité d'information et des utilisateurs qui tentent de l'exploiter, les enjeux de la recherche d'information sont devenus considérables.
Un des aspects importants auquel nous nous intéressons dans le cadre de ce travail, est que l'information est de plus en plus formulée de façon structurée. Cette structure est formalisée explicitement dans le texte lui même par des balises, à l'aide de langages de représentations spécifiques. Parmi ces langages, XML (eXtended Markup Language) est désormais devenu un standard universellement utilisé. Il permet l'échange et le stockage de l'information avec une certaine flexibilité qui l'a rendu extrêmement populaire. Les documents obtenus sont dits semi

structurés, car ils sont en quelque sorte un intermédiaire entre les documents classiques et les bases de données.

La problématique engendrée, en RI, par ce type de document est liée à la nature de leur contenu. En effet, comme ces documents comportent de l'information (du texte) et des contraintes structurelles (des balises), ils ne peuvent pas être efficacement exploités par les techniques classiques de RI, qui considèrent le document comme un granule d'information indivisible. Aussi, dans un document XML toute partie (élément dans le jargon XML) du document peut être considérée comme réponse potentielle à la requête de l'utilisateur. La partie concernée peut être spécifiée directement dans la requête de l'utilisateur ou calculée automatiquement par le système de recherche d'information. Les requêtes dans les systèmes de Recherche d'Information Structurée (RIS) de type XML peuvent en effet avoir deux formes : une forme « contenu seulement », la requête est dans ce cas composée que de mots clés et une forme combinant la structure et le contenu.

Afin d'exploiter au mieux l'ensemble des informations disponibles, les méthodes existantes de recherche d'information doivent être adaptées ou de nouvelles méthodes doivent être proposées. C'est dans ce contexte de recherche d'information structurée que se situent nos travaux. Notre objectif est de proposer un modèle permettant de sélectionner automatiquement l'élément (ou l'ensemble d'éléments) du document qui répond le mieux au besoin de l'utilisateur formulé à travers une liste de mots clés (contenu seulement 'CO'). De plus, afin de mieux prendre en compte la structure hiérarchique d'un document XML, les dépendances entre éléments, ainsi que l'incertitude liée à la notion de pertinence ; le modèle que nous proposons trouve ses fondements théoriques dans la théorie des possibilités et les réseaux possibilistes. La structure réseau fournit une manière naturelle de représenter les liens entre un document, ses éléments (balises) et son contenu. Quant à la théorie des possibilités, elle permet de quantifier de manière qualitative et quantitative les différents liens sous jacents. Elle permet notamment d'exprimer le fait qu'un terme est certainement ou possiblement pertinent vis-à-vis d'un élément et/ou d'un document et de mesurer à quel point un élément (ou un ensemble d'éléments) peut nécessairement ou possiblement répondre à la requête de l'utilisateur.

Problématique

La RI classique (dans les documents « plats ») permet de renvoyer des résultats de recherche aux utilisateurs sous forme d'une liste de documents et c'est à l'utilisateur de chercher l'information qu'il désire à l'intérieur de chaque document.

La RI structurée offre à un utilisateur la possibilité d'avoir des réponses sous une autre forme plus significative. Les réponses aux requêtes des utilisateurs sont des unités d'information auto explicatives [125]. Cela signifie que l'information contenue ne dépend pas d'une autre pour être comprise. Les unités d'information sont des sous arbres des documents XML.

De par leur structure, l'utilisateur interrogeant des corpus de documents XML peut formuler deux types de requêtes, selon sa connaissance du corpus :

- des requêtes portant uniquement sur le contenu des unités d'information : ces requêtes sont composées de simples mots-clés, l'utilisateur laisse le SRI décider de la granularité de l'information à renvoyer,
- des requêtes portant sur la structure et le contenu des unités d'information, dans lesquelles l'utilisateur spécifie des besoins précis sur certains éléments de structure. Dans ce type de requêtes, l'utilisateur peut utiliser les conditions de structure pour indiquer le type des éléments qu'il désire voir renvoyer, mais aussi plus simplement pour préciser son besoin.

La dimension structurelle des documents structurés a soulevé plusieurs problématiques. Parmi ces problématiques, nous citons:

1. La problématique spécifique à l'indexation de documents semi structurés. L'indexation doit prendre en compte l'information structurelle qui s'ajoute au contenu. Quant à ce contexte plusieurs questions se posent [125]: Que doit-on indexer de la structure des documents? Comment relier cette structure au contenu même du document? Comment pondérer les termes d'indexation, c'est à dire comment évaluer l'importance d'un terme au sein de l'élément, du document et de la collection?

2. Pour la modélisation de la pertinence. Les modèles énoncés dans la littérature calculent la pertinence d'un document (ou élément) comme un score d'appariement entre le document et la requête. Ce score de pertinence est une valeur unique, il est calculé à partir des poids des termes de la requête et ceux des documents, ces poids sont considérés comme des données certaines. Ce

constat nous amène à poser la question suivante: Est-ce que la prise en compte d'une pertinence vue comme une variable binaire peut couvrir toute la sémantique de cette notion ?
Cette question découle du fait que les modèles de RI, mesurent la pertinence d'un document en réponse à une requête utilisateur en se basant sur une unique valeur dite « score de pertinence ». De nombreux travaux de la littérature se sont penchés sur la notion de pertinence [26, 30, 84]. Ces travaux s'accordent à dire qu'il n'existe pas une définition précise de la pertinence, et que cette notion est dynamique, multidimensionnelle et dépend de la perception de l'utilisateur (un utilisateur peut juger les mêmes documents restitués en réponse à une requêté donnée d'une manière différente entre deux instants de temps successifs). Ainsi, dans notre démarche, nous estimons qu'il n'est pas judicieux d'affecter une unique valeur à la pertinence supposée englober la totalité de la sémantique de cette notion. L'attribution d'une valeur unique de pertinence est une démarche limitatrice, dans le sens où il est difficile de couvrir toute sa sémantique. La décision de restituer un document, donc d'évaluer sa pertinence, en se basant sur une seule valeur ne peut pas être fiable ni basée sur une certitude.

3. Le traitement du contenu est également problématique. En effet, le contenu textuel n'apparaît souvent que dans les nœuds feuilles. Cependant, un nœud interne peut être pertinent même s'il ne contient aucun terme d'indexation, la pertinence ne provient pas seulement de la structure. Afin de rétablir l'ordre entre les nœuds pour que les feuilles ne soient pas privilégiées par rapport aux nœuds internes, ceux-ci doivent avoir un score relatif au contenu. On distingue dans la littérature deux principes courants, la plupart des modèles de RI utilisent la propagation des scores : on propage le score d'un nœud pertinent à une requête (en terme de contenu) à ses ancêtres. Les autres modèles se basent sur la propagation du contenu : au lieu de la propagation du score, on propage le contenu d'un nœud à un autre via les liens de descendance et on calcule son score indépendamment.

4. Les documents semi structurés ont réactualisé la problématique de la granularité de l'information à renvoyer, et ils permettent de traiter l'information avec une granularité modulable qui peut être identifiée en faisant intervenir la structure du document. Pour résoudre cette problématique, la majorité des modèles proposés dans la littérature [91, 92, 102,106, 133] tentent de renvoyer les unités d'informations sous forme d'une liste d'éléments disjoints. Nous

considérons que les unités pertinentes ne sont pas nécessairement des éléments disjoints, mais peuvent être une agrégation d'éléments du document.

Contribution

Notre contribution dans le cadre de la recherche d'information structurée concerne les requêtes portant uniquement sur le contenu. Cette contribution se situe à plusieurs niveaux et tente de répondre aux problématiques présentées dans la section précédente :

1. Notre première contribution porte sur l'indexation et le stockage des documents. Nous proposons un modèle de représentation des données générique qui conserve la structure arborescente du document. Ce modèle permet de traiter des documents semi structurés possédant des structures hétérogènes, sans avoir de connaissance a priori sur cette structure. Nous nous sommes également intéressés au problème de la pondération des termes d'indexation, et nous proposons de tenir compte de l'importance locale (au niveau de l'élément) et globale (au niveau de la collection) des termes.

2. Dans la littérature les modèles de RI sont catégorisés en fonction de la définition qu'ils donnent à la pertinence. À notre sens, une seule valeur de pertinence ne couvre pas l'incertitude intrinsèque à la notion de pertinence. Nous cherchons à traduire la pertinence en essayant de modéliser différents aspects auxquels elle est reliée. Les travaux que nous proposons s'inscrivent dans la définition d'un nouveau modèle de RIS permettant notamment une nouvelle modélisation de la pertinence. Par conséquent, nous proposons de modéliser la pertinence dans un cadre possibiliste. La théorie des possibilités et la logique possibiliste proposent des mesures duales traitant l'incertitude liée à l'information de manière flexible.

Un des objectifs de ce travail est donc de proposer une approche moins restrictive pour la modélisation de la pertinence. En effet, nous donnons deux sens différents mais complémentaires à la notion de pertinence. Nous proposons une pertinence certaine et une pertinence plausible d'un élément (ou agrégation d'éléments) étant donnée une requête. Le cadre possibiliste est plus à même de prendre en compte l'ignorance partielle qui peut affecter les informations utilisées dans les différents calculs en particulier, le poids des termes dans les éléments et la pertinence d'un élément. Plus précisément, notre modèle sépare les raisons de sélectionner un élément pertinent de celles de le rejeter, en utilisant deux mesures : la nécessité et la possibilité. La possibilité de

pertinence tente d'éliminer les éléments non pertinents. La nécessité de pertinence met l'accent (le "focus") sur les éléments qui semblent très pertinents.

3. Nous traitons le contenu des éléments XML en faisant appel aux fondements de la recherche d'information traditionnelle par adaptation à la nature des documents semi structurés. Or, dans les documents semi structurés le contenu textuel n'apparaît souvent que dans les nœuds feuilles. Cependant, un nœud interne peut être pertinent même s'il ne contient aucun terme d'indexation, la pertinence ne provient pas seulement de la structure. Afin de rétablir l'ordre entre les nœuds pour que les feuilles ne soient pas privilégiées par rapport aux nœuds internes, ceux-ci doivent avoir un score relatif au contenu. Pour propager le texte se situant dans les nœuds feuilles vers leurs ancêtres, contrairement aux modèles de RI utilisant la propagation des scores, notre modèle se base sur la propagation du contenu (termes) d'un nœud vers un autre nœud à travers les liens de descendance permettant ainsi de calculer le score relatif au contenu de chaque nœud de manière indépendante.

4. Les systèmes d'accès aux documents structurés sont confrontés au problème de l'unité d'information à sélectionner dans le cas où la requête de l'utilisateur ne comporte que des mots clés. En effet, les techniques classiques de RI (plein texte) considèrent le document entier comme un granule d'information indivisible, or dans le cas des documents XML tout élément peut être une réponse potentielle à la requête de l'utilisateur. Le défi à relever est alors d'arriver à identifier automatiquement l'unité d'information, en l'occurrence les parties du document XML, pertinente par rapport à la requête de l'utilisateur. Pour cela, la majorité des modèles proposés dans la littérature [91, 92, 102, 106, 133] renvoient des unités d'informations sous forme d'une liste d'éléments disjoints. Nous considérons que les unités pertinentes ne sont pas nécessairement des éléments disjoints, mais peuvent être une agrégation d'éléments du document. Prenons un document avec la structure suivante: (document (titre) (chappter1 (section1) (section 2)) chapitre2 (...)); si l'information pertinente se trouve dans le "titre" et la "section1", la plupart des systèmes de recherche d'informations XML retournent l'ensemble du document comme unité pertinente. Dans notre cas, nous considérons que, la seule unité à être retournée est un élément fictif appelé agrégat et formé par deux éléments: "titre" et "section1". Pour atteindre cet objectif, nous proposons un modèle permettant de sélectionner automatiquement l'agrégation d'éléments

non redondants qui répondent le mieux au besoin de l'utilisateur formulé à travers une liste de mots-clés. Le modèle que nous proposons trouve ses fondements théoriques dans les réseaux possibilistes. La structure du réseau fournit une représentation naturelle des liens entre le document, ses éléments et son contenu. Quant à la théorie des possibilités, elle permet de quantifier qualitativement et quantitativement les différents liens sous-jacents; et mesure à quel point un ensemble d'éléments peut nécessairement ou possiblement répondre à la requête de l'utilisateur.

Afin de vérifier la faisabilité de ces propositions, un prototype a été développé, et les propositions ont été évaluées sur un échantillon de la collection INEX (*INitiative for the Evaluation of XML retrieval*); dans le but de situer nos travaux par rapport à des travaux similaires dans le domaine.

Organisation de la thèse

Cette thèse est organisée en deux grandes parties: la première partie présente le contexte dans lequel se situent nos travaux, à savoir la recherche d'information et plus précisément la recherche d'information dans des documents semi structurés; la seconde partie décrit notre contribution qui consiste en la proposition d'un modèle de recherche d'information structurée basé sur les réseaux possibilistes.

L'objectif de la première partie, **Recherche d'Information et Structure**, est de présenter les approches proposées dans la littérature pour la recherche d'information traditionnelle dans des documents texte "plats" ainsi que pour la recherche d'information dans des documents semi structurés.

Le chapitre 1, **Recherche d'Information**, présente les concepts de base de la recherche d'information. Nous décrivons le processus d'indexation. Nous rappelons la définition de la pertinence ainsi que les techniques de pondération des termes d'indexation. Nous décrivons le processus de recherche d'information tout en présentant les différents modèles d'appariement document-requête. Nous présentons enfin les mesures utilisées pour l'évaluation des systèmes de recherche d'information.

Le chapitre 2, **Recherche d'Information Structurée**, s'intéresse aux approches proposées dans la littérature pour la recherche d'information dans des documents semi structurés de type

XML. Nous donnons une description du document XML et les différentes technologies qui s'y rapportent. Nous présentons les nouvelles problématiques soulevées par la recherche d'information structurée et les solutions qui ont été proposées dans la littérature pour y faire face. Différents modèles de recherche proposés dans la littérature sont présentés. La plus part de ces modèles de recherche sont des adaptations, faites aux modèles de RI classiques pour pouvoir exploiter l'information additionnelle de structure portée par les documents semi structurés et permettre des requêtes portant sur le contenu et/ou la structure des documents. Enfin, de nouvelles mesures d'évaluation de la performance des systèmes de recherche d'information structurée sont présentées.

La seconde partie de cette thèse, intitulée **Un Modèle Possibiliste pour la Recherche d'Information Structurée**, présente nos travaux, à savoir le modèle de recherche que nous proposons et les évaluations que nous avons effectuées pour valider notre approche.

Le chapitre 3, **Théorie des Possibilités** décrit sommairement la théorie des possibilités et les réseaux possibilistes qui constituent la base de notre approche.

Le chapitre 4, **MPRIX : Modèle Possibiliste pour la Recherche d'Informations XML agrégée** présente le modèle que nous proposons pour répondre aux différentes problématiques de la recherche d'information structurée. Nous rappelons nos motivations et nous donnons une présentation détaillée de notre modèle. MPRIX est basé sur les réseaux possibilistes. Les relations document-éléments et éléments-termes sont modélisées par des mesures de possibilité et de nécessité. Dans ce modèle, la requête de l'utilisateur déclenche un processus de propagation pour retrouver des portions de documents nécessairement ou au moins possiblement pertinents par rapport à la requête. Un exemple d'une telle recherche est proposé de façon à illustrer notre approche.

Le chapitre 5, **Expérimentations et Résultats,** décrit l'architecture du prototype que nous avons développé pour valider notre approche ainsi que le protocole d'évaluation et présente les résultats que nous avons obtenus grâce à nos expérimentations.

En conclusion, nous dressons un bilan de nos travaux, en mettant en exergue nos propositions. Nous finissons par la proposition des perspectives possibles de nos travaux.

Première partie

Recherche d’Information et Structure

Chapitre I
Recherche d'Information

I.1 Introduction

La Recherche d'Information (RI) [10, 117, 123] est une branche de l'informatique qui s'intéresse à l'acquisition, l'organisation, le stockage, la recherche et la distribution de l'information. Un Système de Recherche d'Information (SRI) permet de sélectionner à partir d'une collection de documents des informations pertinentes, répondant à des besoins utilisateurs, exprimés sous forme de requêtes.

Dans ce chapitre, nous présentons les concepts de base de la recherche d'information, puis nous passons en revue les principaux modèles de RI.

I.2 Concepts de base de la recherche d'information

I.2.1 Définition d'un système de recherche d'information

Plusieurs définitions ont été proposées pour un Système de Recherche d'Information qui sont plus ou moins proches les unes des autres.

Tomek Strzalkowski donne la définition suivante [137]:

«La tâche typique de la recherche d'information, est de sélectionner des documents dans une base de données, en réponse à une requête de l'usager, et leur rangement par ordre de pertinence».

Alan Smeaton a proposé cette définition [135]:

«Le but d'un système de recherche d'information est de retrouver des documents en réponse à une requête des usagers, de manière à ce que les contenus des documents soient pertinents au besoin initial d'information de l'usager».

Pour Gerard Salton [123]**:**

«Un système de recherche d'information manipule un ensemble de documents sous forme d'unités informationnelles ou conteneurs sémantiques, non décomposables. L'objectif d'un système de recherche d'information et d'aiguiller la recherche dans le fonds documentaire, en

direction de l'information pertinente relativement à un besoin en information exprimé par une requête utilisateur. A cet effet, le système assure les fonctionnalités de communication, stockage, organisation et recherche d'information».

I.2.2 Fonctions d'un SRI

Les principales fonctions d'un SRI sont :

- Indexation des documents de la collection.
- Appariement document-requête.
- Reformulation de la requête.

Ces fonctions sont résumées dans la figure **I.1** suivante [33] :

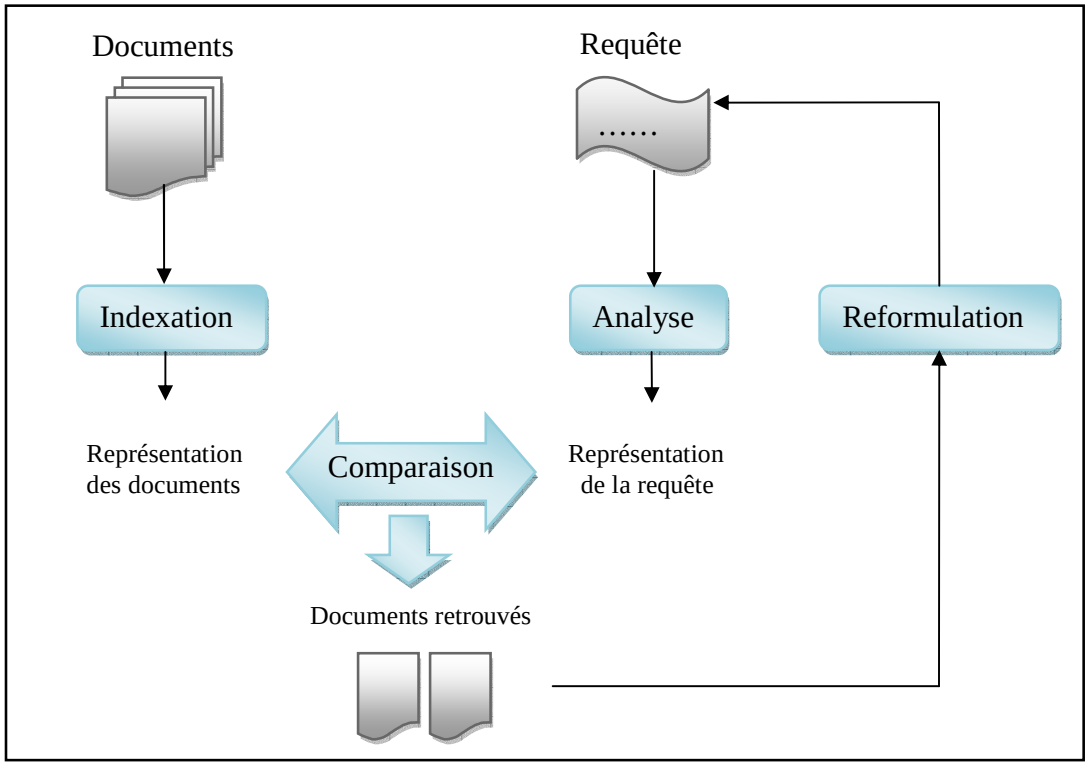


Figure I.1 *- Processus en U de la RI*

I.2.3 Collection de documents

La collection de documents (ou fonds documentaire, corpus) constitue l'ensemble des informations exploitables et accessibles. Elle est constituée d'un ensemble de documents. Dans le cas général et pour un souci d'optimalité, la collection (base) constitue des représentations

simplifiées mais suffisantes pour ces documents. Ces représentations sont étudiées de telle sorte que la gestion (ajout et/ou suppression d'un document) ou l'interrogation (recherche) de la base se font dans les meilleures conditions de coût d'accès et de stockage [11].

I.2.4 Besoin en information

Le besoin en information de l'utilisateur est exprimé par une requête portant sur le contenu sémantique des documents recherchés. La qualité de la requête exprimée par l'utilisateur varie considérablement avec sa connaissance du domaine et avec son aptitude à définir son besoin, on note qu'il y a souvent une importante perte d'information entre le besoin et son expression en requête [59].

I.2.5 Représentation des documents et des requêtes

Pour effectuer une recherche documentaire, les documents et les requête passent d'abord par un processus de représentation afin de les traduire d'une description brute vers une description structurée qui couvre au mieux le contenu sémantique. Ce processus de conversion est appelé indexation. Les descripteurs issus de l'indexation sont des mots clés associés généralement à des poids afin de différencier leur degré de représentativité du contenu sémantique de l'unité en question.

I.2.6 Pertinence

La pertinence est une notion centrale dans la recherche d'information car toutes les évaluations s'articulent autour de cette notion. Les travaux de recherche [26, 100] s'accordent sur la difficulté de la définition de cette notion. Voici quelques définitions proposées de la pertinence [124]:

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

Nous constatons que la pertinence n'est pas une relation isolée entre un document et une requête; elle est généralement liée à la perception de l'utilisateur, de plus elle est multidimensionnelle et évolue durant le temps de recherche [33].

I.2.7 Appariement document - requête

La comparaison (ou la correspondance) entre un document 'D' et une requête 'Q', afin de juger la pertinence du document par rapport à la requête, est réalisée par le calcul d'un score, appelé degré de pertinence. Ce score est calculé à partir d'une fonction ou d'une probabilité de similarité notée **RSV (Q, D)** (Retrieval Status Value). Cette mesure tient compte du poids des termes dans les documents, déterminé en fonction d'analyses statistiques et probabilistes [125].
La correspondance peut être binaire (document pertinent ou non pertinent), nous parlons alors d'appariement exact. Dans le cas contraire, les documents retournés sont ordonnés selon leur degré de pertinence.

I.3 Processus d'indexation

Nous définissons le processus d'indexation comme la transformation de l'information contenue dans un document vers un autre espace de représentation manipulable par un système de recherche d'information. Les objets (documents ou requêtes) sont représentés par des descripteurs présentés sous forme d'une liste de mots clés et de poids qui leurs sont associés [121].
Dans les systèmes de recherche d'information, deux types de langage d'indexation sont définis: langage libre ou contrôlé. Le type du langage choisi influence la manière dont sont choisis les descripteurs d'index [121].
Langage libre : les termes d'indexation sont généralement issus du document même, aucune contrainte n'est spécifiée. Ce type est utilisé dans la technique d'indexation plein texte.
Langage contrôlé (langage normalisé) : une liste de termes d'indexation est définie pour éviter les problèmes de langue (polysémie, synonymie, …).

Techniquement, l'indexation peur être manuelle, automatique ou semi-automatique

a- Indexation manuelle:

L'analyse est laissée au spécialiste du domaine ou documentaliste pour identifier et construire une représentation du contenu d'un document. Cette méthode assure la meilleure pertinence des réponses apportées par le SRI par apport aux deux autres méthodes d'indexation [125], mais elle est fréquemment critiquée pour sa variabilité, car elle repose sur la subjectivité de l'interprétation humaine. En fait, un indexeur à deux moments différents peut présenter deux

termes distincts pour représenter le même concept et deux indexeurs différents peuvent présenter des termes différents pour indexer un même document. Un autre inconvénient de l'indexation manuelle est son coût très élevé en temps de traitement.

b- Indexation automatique:

La première approche d'indexation automatique fut introduite à Internationnal Conference on Scientific Information (ICSI) en 1958 par Luhn [95]. L'indexation automatique est réalisée à l'aide d'un processus entièrement automatisé qui consiste à déterminer les unités syntaxiques ou mots-clés représentatifs des concepts et des thèmes exprimés dans le document. Les méthodes existantes vont d'une simple extraction de mots simples (unitermes) à des analyses linguistiques et/ou statistiques permettant d'établir des relations sémantiques entre ces mots. La puissance de l'indexation automatique réside dans le fait qu'elle fournit le même index pour un document donné (régularité du processus d'indexation).

c- Indexation semi-automatique:

Cette méthode n'est qu'une combinaison des deux méthodes précédentes. Elle est appelée aussi indexation assistée, elle revient le plus souvent à faire valider ou corriger par un humain une représentation du document proposé par le système d'indexation [121].

I.3.1 Critères d'une bonne indexation

Les performances d'un SRI dépendent fortement de la qualité de son processus d'indexation. L'indexation est jugée sur deux critères : la cohérence et l'adéquation entre les représentations des requêtes et des documents [121]:

- La cohérence (régularité) :

La cohérence d'une indexation est mesurée par la distance qui existe entre un document et sa représentation (contenu sémantique). Une bonne indexation doit être cohérente, c'est-à-dire que deux textes traitant du même sujet et utilisant des vocabulaires différents, doivent être indexés par les mêmes descripteurs.

- L'adéquation entre les représentations :

Cela revient au fait que les descripteurs des documents doivent appartenir au même vocabulaire que celui utilisé pour décrire les requêtes.

I.3.2 Quelques techniques d'indexation

Nous présenterons, dans ce qui suit, quelques techniques d'indexation utilisées par les SRI.

I.3.2.1 Indexation lexicale

C'est la méthode la plus utilisée grâce à sa simplicité, elle consiste à extraire les mots clés en éliminant les mots vides (inutiles) en utilisant un anti-dictionnaire ou une stop liste. Pour tout document et la liste des termes qu'il contient, un fichier inverse est créé pour dresser pour chaque terme, la liste des documents qui le contiennent. Le fichier inverse contient tous les mots décrivant les documents du corpus. Ainsi à chaque terme du fichier inverse, peuvent correspondre le nombre de références, le numéro de document, le code du paragraphe, le numéro de la phrase où ce terme figure. Le fichier inverse est généralement trié par ordre alphabétique pour accélérer la recherche.

L'inconvénient de cette technique est le fait qu'elle ne prenne pas en compte les problèmes linguistiques (polysémie, synonymie, homographie, …).

I.3.2.2 Les méthodes linguistiques

Ces méthodes se basent sur des analyses lexicale, syntaxique et sémantique afin d'indexer les documents par leurs sens [57]. Ces méthodes ont pour objectifs de retourner des documents pertinents même si ces derniers ne contiennent aucun terme de la requête. La notion de pertinence est basée aussi bien sur les idées apportées par le document ou la requête que sur les termes contenus.

La mise en œuvre des méthodes linguistiques nécessite la disponibilité d'analyseurs et de bases de connaissances (dictionnaires électroniques, des liste de domaines, …).

1- **Analyse morphologique :**

1-1 Segmentation en unités linguistiques : elle consiste à découper le document en sections, paragraphes, phrases et mots. Dans cette étape, on traite les ambiguïtés des séparateurs et des délimiteurs de texte (par exemple un point s'utilise également pour marquer une abréviation).

1-2 Racinisation : c'est l'étape de normalisation des mots, regrouper les mots sémantiquement proches à partir des ressemblance graphiques (mots de forme apparentée), les mots sont stockés sous forme canonique. Cela est réalisé par les fonctions de désuffixation, recodage et lemmatisation.

2- Analyse syntaxique:

Associer à chaque mot une étiquette pour indiquer sa catégorie syntaxique (nom, verbe, adjectif, …).

3- Analyse sémantique:

Associer à chaque mot une étiquette qui définit sa catégorie sémantique générale (événement, mouvement, …) ou bien un sens de mot (par exemple artère –> vaisseau sanguin).

Les travaux de désambiguïsation sémantique sont très récents et sont généralement moins maîtrisés que les autres traitements automatiques.

4- Analyse pragmatique:

Comment les phrases peuvent être interprétées selon leur contexte d'énonciation (interlocuteurs, phrases précédentes, connaissance commune du monde, . . .)

I.3.2.3 Les méthodes probabilistes

Ces méthodes se basent sur la théorie des probabilités, la décision de sélectionner ou non un terme d'indexation est analysée comme un processus de décision, sous l'hypothèse implicite que le contenu sémantique d'un document peut être extrait des mots qui le composent [66].

Afin de modéliser l'apparition des occurrences d'un mot dans un document, nous pouvons admettre que ces occurrences sont distribuées de manière complètement aléatoire, et on modélise la probabilité qu'un mot apparaisse **k** fois dans un document par une loi de Poisson:

$$P(K) = \frac{e^{-\lambda}.\lambda^{k}}{k!}$$

λ : indique la moyenne des occurrences de ce mot dans le document.

Ce modèle stochastique a prouvé que l'apparition des mots outils (la, de, par, …) est purement aléatoire. Par contre, la distribution des occurrences des mots spécifiques à un document s'écartait de ce modèle (les occurrences de ces mots ne sont pas indépendantes dans leurs apparitions), ils ont tendance à apparaître en groupes.

Donc pour choisir les mots d'indexation, on distinguera premièrement les mots spécifiques à une collection par le fait que leur distribution s'écarte de celle de Poisson en recourant par exemple à un test de χ^2, puis un terme sera sélectionner pour indexer un document si ce terme apparaît fréquemment dans le document que dans un autre document choisi au hasard, cela revient à tester la validité de cette dernière hypothèses.

I.4 Pondération des termes

La pondération est l'une des fonctions importantes du processus d'indexation (voir de la RI). Elle permet de mesurer l'importance d'un terme dans un document. La majorité des modèles et approches proposés en RI se base souvent sur des considérations et interprétations statistiques (ou parfois linguistiques) pour calculer cette importance. Ces méthodes tirent leurs origines de la loi de Zipf et de la conjecture de Luhn.

I.4.1 Loi de Zipf

La loi de Zipf [162] est une loi empirique énoncée en 1949 par G. K. Zipf. Elle est basée sur le principe du moindre effort (*Principle of Least Effort*), Zipf considère qu'il est plus facile pour un auteur d'un document de répéter certains termes que d'en utiliser de nouveaux. Elle décrit la répartition statistique des fréquences d'apparition des différents éléments d'un ensemble, comme les mots d'un texte. Dans un texte, selon Zipf la fréquence d'un mot dans un document est inversement proportionnelle à son rang de classement dans la liste des mots classés par ordre décroissant des fréquences. Formellement, cette loi s'exprime comme suit :

*rang * fréquence = constante*

La loi de distribution des termes suit alors la courbe présentée dans la figure **I.2**

Il existe une étape importante avant la pondération des termes qui consiste à ne garder que les termes d'indexation représentatifs (ou significatifs) du document. Pour cela un autre concept est introduit, il s'agit de la conjecture de Luhn [95].

I.4.2 La conjecture de Luhn

Luhn s'est inspiré de la loi de Zipf pour déterminer les termes représentatifs d'un document [95]. Luhn émet une hypothèse sur l'information contenue dans les termes (informativité ou significativité ou représentativité) d'un document. Ainsi la sélection des termes

d'indexation consiste à éliminer respectivement les termes de fréquences très faibles car ils sont rarement utilisés dans une requête et les termes de fréquences très élevés car ils ne sont pas discriminants entre le document et la collection (Par *discriminant* on désigne les termes appartenant à un document et n'appartenant pas aux autres documents de la collection).

Par exemple, dans un corpus de documents informatiques, le mot *ordinateur* est très fréquent et ne va pas permettre de différencier les documents entre eux. Pour éliminer les termes dont l'informativité est jugée faible, deux seuils (seuil minimal et seuil maximal) sont considérés (arbitrairement). Seuls les termes entre ces deux seuils sont intéressants pour représenter les documents. La conjecture de Luhn est illustrée dans la figure **I.2** suivante [125]:

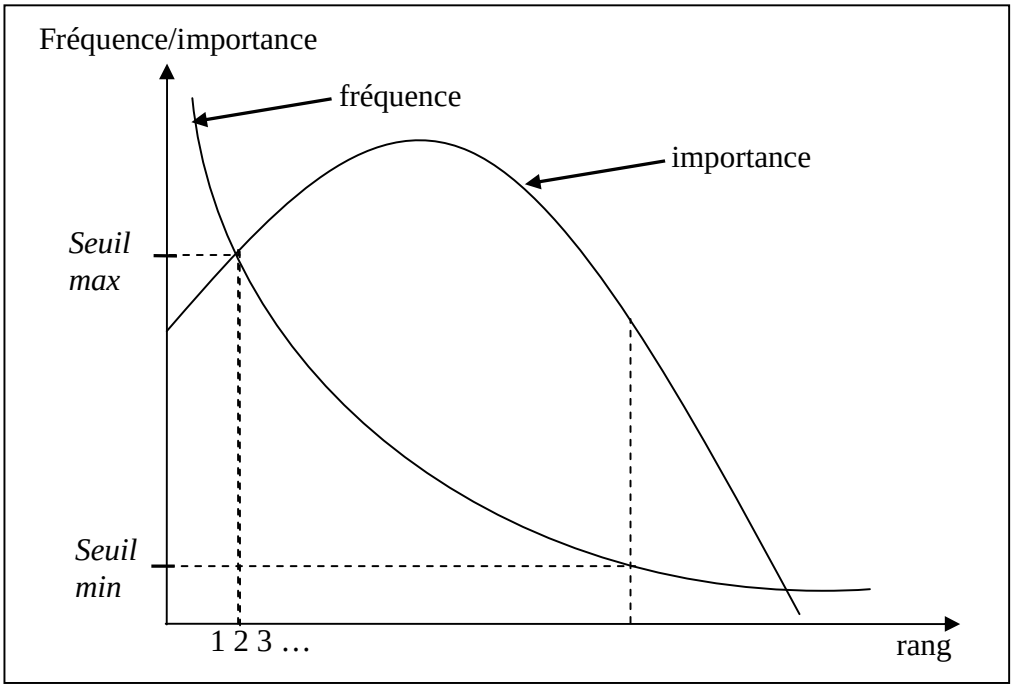


Figure I.2 *- Schéma illustrant les mots significatifs*

Cette approche est utilisée pour réduire la taille de l'index de la collection.

I.4.3 Pondération en tf_idf

La technique de pondération utilisée joue un rôle fondamental dans les SRI. La majorité des techniques de pondération sont construites par la combinaison de deux facteurs :

Un facteur de pondération locale désigné par **tf** (*term frequency*) permet de mesurer la représentativité locale d'un terme. Il prend en compte le contenu informatif d'un terme par rapport à un document (il est proportionnelle à la fréquence du terme dans le document). Son utilisation peut prendre plusieurs expressions (fonction brute en tf_{ij}, $\log(tf_{ij})$, … Où $\mathbf{tf_{ij}}$: désigne le nombre d'occurrences du terme i dans le document j).

Un facteur de pondération globale désigné par idf (inverse of document frequency) permet de mesurer la représentativité globale du terme vis-à-vis de la collection des documents. Il prend en compte le contenu informatif du terme par rapport à l'ensemble des documents de la collection. Ce facteur peut être utilisé selon les deux formules suivantes :

$$idf = \text{Log}\ (N / n_i)$$

Ou

$$idf = \text{Log}\ ((N - n_i) / N)$$

Où $\mathbf{n_i}$ **:** désigne le nombre de documents contenant le terme i

N : désigne la taille de la collection (nombre de documents de la collection).

I.5 Reformulation de la requête

L'utilisateur, d'un SRI, exprime sa requête par des termes qui lui sont propres, mais qui ne correspondent pas forcément à ceux utilisés pour indexer les documents pertinents de la base d'indexation du système de recherche d'information. Cette requête initiale est généralement traitée comme un essai pour apporter de l'information à l'utilisateur, le système retourne une liste de documents par ordre de pertinence décroissante. Afin d'affiner la recherche précédente, et de coordonner le langage de recherche (utilisé par l'utilisateur dans la requête) et le langage d'indexation du système de recherche d'information, la notion de *reformulation de la requête* est utilisée. Cette dernière, consiste à générer une requête plus adéquate que celle initialement formulée par l'utilisateur, son principe est de modifier la requête initiale par ajout de termes significatifs et/ou correction de leurs poids.

Nous distinguons principalement deux approches de reformulation de la requête:

I.5.1 Reformulation automatique

Cette méthode est basée sur les techniques d'associations de termes. L'extension de la requête est faite à partir d'un **thésaurus** qui définit les relations entre les différents termes de l'index et permet de sélectionner de nouveaux termes à ajouter à la requête [125].

Rocchio a publié le premier travail majeur sur l'enrichissement automatique de requête, il a proposé de mettre à jour le vecteur requête en fonction d'un ensemble de documents pertinents et d'un autre ensemble de documents non pertinent. Du fait de cette mise à jour, des mots nouveaux peuvent apparaître [10, 73].

Thesaurus : Outil permettant de représenter la proximité au voisinage sémantique entre termes de la collection. La construction d'un thesaurus est l'établissement des liens sémantiques entre les termes. Cette construction peut être manuelle ou automatique.

I.5.2 Reformulation manuelle (réinjection de pertinence ou relevance feedback)

Méthode plus populaire que la première, son principe fondamental est d'utiliser la requête initiale afin d'amorcer la recherche, puis modifier celle-ci à partir des jugements de pertinence et/ou de non pertinence de l'utilisateur sur les documents restitués, soit pour re-pondérer les termes de la requête initiale, soit pour y ajouter (respectivement supprimer) d'autres termes contenus dans les documents pertinents (respectivement non pertinent).

La nouvelle requête obtenue à chaque itération feedback, permet de corriger la direction de la recherche dans les documents pertinents.

Nous citons ici quelques avantages de cette méthode [125]:

- L'utilisateur n'a pas à s'occuper des détails de la reformulation, d'où la simplicité d'exécution.
- Elle offre un meilleure contrôle du processus de recherche en augmentant le poids des termes importants et en diminuants celui des termes non importants.

I.6 Les modèles connus de la RI

Le modèle de RI joue un rôle central dans la RI, c'est lui qui détermine le comportement d'un SRI. Etant donné un ensemble de termes pondérés issus de l'indexation d'un contenu d'un document et d'une requête, le modèle doit alors accomplir les rôles suivants [141]:

- Créer une représentation interne pour le document et pour la requête basée sur ces termes.
- Définir une méthode de comparaison entre une représentation du document et une représentation de la requête afin de déterminer leur degré de correspondance (ou similarité).

Nous distinguons trois classes principales de modèles de RI :

- Les modèles booléens
- Les modèles vectoriels
- Les modèles probabilistes

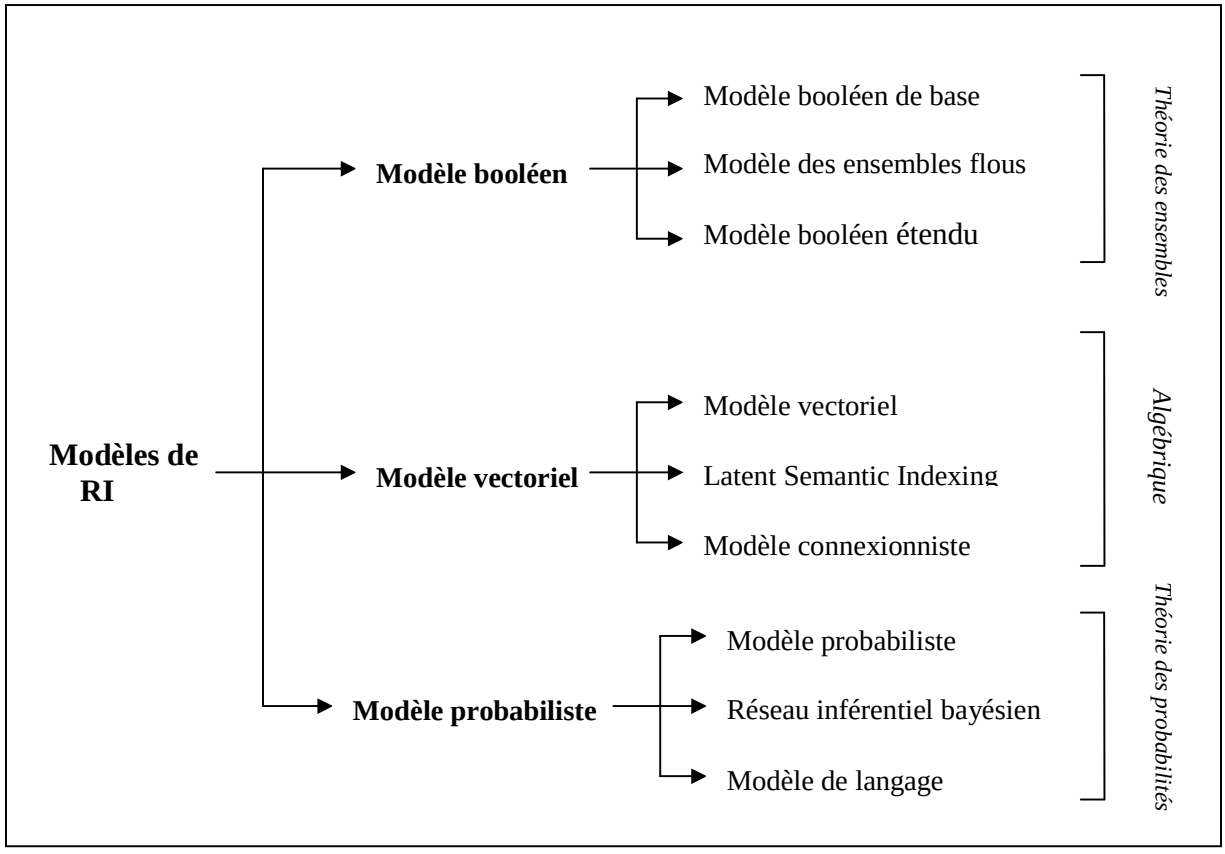


Figure I.3 *- Modèles de recherche d'information*

I.6.1 Les modèles booléens

I.6.1.1 Le modèle booléen de base

Le modèle booléen [73] est le modèle le plus ancien et le plus simple de tous les modèles de RI conventionnels. Il est basé sur la théorie des ensembles et l'algèbre de Boole. Il propose de représenter la requête sous forme d'une expression logique composée de termes assemblés par les connecteurs logiques *ET* ($\wedge$), *OU* ($\vee$) et *NON* ($\neg$) (Par exemple $q = (t_1 \wedge t_2) \vee (t_3 \wedge \neg t_4)$) et de représenter le document comme une conjonction logique de termes non pondérés (Par exemple $d = t_1 \wedge t_2 \ldots \wedge t_n$).

Le processus de recherche d'information consiste à effectuer des opérations logiques sur les ensembles de documents, définis par l'occurrence ou l'absence de termes d'indexation, afin de réaliser un appariement avec l'équation de la requête. Ce modèle se base sur le mode d'appariement exact, un document ne sera retourné que s'il répond exactement aux termes de la requête formulée.

Pour que le document **d** corresponde à une requête **q**, il faut que l'implication suivante soit valide :

d => q

La correspondance RSV (d, q) entre une requête et un document est déterminée de la façon suivante :

$RSV(d, t_i) = 1$ si $t_i \in d$; 0 sinon.

$RSV(d, q_1 \wedge q_2) = 1$ si $RSV(d, q_1) = 1$ et $RSV(d, q_2) = 1$; 0 sinon.

$RSV(d, q_1 \vee q_2) = 1$ si $RSV(d, q_1) = 1$ ou $RSV(d, q_2) = 1$; 0 sinon.

$RSV(d, \neg q_1) = 1$ si $RSV(d, q_1) = 0$; 0 sinon.

Sachant que t_i est un terme de la requête q et d un document du corpus.

La mise en œuvre de ce modèle est facile grâce au fichier inverse.

$\mathbf{t}_1$: D_1, D_2, D_5, D_7

$\mathbf{t}_2$: $D_3, D_7, D_5,$

$\mathbf{t}_3$: D_1, D_2, D_3, D_6, D_7

$\mathbf{t}_4$: D_1, D_2

Figure I.4 *- Exemple d'un fichier inverse*

Malgré la simplicité du modèle, il a plusieurs inconvénients dont nous citons les principaux :

- La correspondance entre un document et une requête est soit 0 ou bien 1, En conséquence le système retourne un ensemble de documents dans un ordre quelconque comme réponse à la requête. Donc il est impossible de dire qu'un document est plus pertinent qu'un autre, ce qui revient à laisser la tâche à l'utilisateur

de fouiller dans les documents retournés afin de sélectionner les documents les plus pertinents, et dans le cas où plusieurs documents correspondent à la requête, la tâche devient très difficile.

- Les termes d'indexation ne sont pas pondérés, il est donc difficile de distinguer les termes les plus importants.
- Les documents pertinents dont la représentation ne correspondent qu'approximativement à la requête ne sont pas sélectionnés.
- La formulation des requêtes nécessite une maîtrise parfaite des opérateurs booléens, cependant un problème en pratique est que les utilisateurs manipulent mal ces opérateurs logiques, d'où des problèmes dans la représentation de leurs besoins en information.

I.6.1.2 Le modèle basé sur les ensembles flous

C'est une extension du modèle booléen de base, il est basé sur la théorie des ensemble flous proposée par **Zadeh** [159], au niveau de la représentation de la requête il n'y a pas de changement (elle reste toujours une expression logique). Un document est représenté comme un ensemble de termes pondérés comme suit :

$D = \{(t_1, a_1),\ldots, (t_i, a_i), \ldots\}$

Où a_i est le degré d'appartenance du terme t_i au document D.

La correspondance RSV entre une requête **q** et un document **d** peut prendre plusieurs formes.

La plus classique est la suivante :

$$RSV(d, t_i) = a_i$$

$$RSV(d, q_1 \wedge q_2) = \min\mathbf{(}\ RSV(d, q_1)\mathbf{,}\ RSV(d, q_2)\ \mathbf{)}$$

$$RSV(d, q_1 \vee q_2) = \max\mathbf{(}\ RSV(d, q_1)\mathbf{,}\ RSV(d, q_2)\ \mathbf{)}$$

$$RSV(d, \neg q_1) = 1 - RSV(d, q_1)$$

Cette évaluation classique n'est pas parfaite à cause des fonctions *min* et *max*, alors d'autres évaluations ont été proposées afin de la corriger. Nous citons ci-dessous l'évaluation définie par **Lukaswicz** qui permet aux deux parties de jouer un rôle dans l'évaluation.

$$RSV(d, t_i) = a_i$$

$$RSV(d, q_1 \wedge q_2) = RSV(d, q_1) * RSV(d, q_2)$$

$$RSV(d, q_1 \vee q_2) = RSV(d, q_1) + RSV(d, q_2) - RSV(d, q_1) * RSV(d, q_2)$$

$$RSV(d, \neg q_1) = 1 - RSV(d, q_1)$$

L'objectif principal atteint par ce modèle est le fait qu'il mesure le degré de correspondance entre un document et une requête, ce qui permet alors d'ordonner les documents selon leur ordre de pertinence.

I.6.1.3 Le modèle booléen étendu

Ce modèle a été introduit par **Salton** en 1983 [123], il est aussi appelé modèle P_Norm, son principe de base est d'attribuer aux termes de la requête et des documents des poids, et contrairement au modèle booléen de base, il interprète les opérateurs de l'équation de la requête comme des distances entre requête et documents. Par conséquent, ce modèle accepte l'appariement approché et il mesure le degré de pertinence document- requête.

La fonction de similarité est décrite sous forme conjonctive ou disjonctive de la façon suivante :

Opérateur OU : $$RSV(D_j, Q) = \left[\frac{\sum_{i=1}^{m} q_i^{\,p} \times d_{ij}^{\,p}}{\sum_{i=1}^{m} q_i^{\,p}} \right]^{1/p}$$

Opérateur ET : $$RSV(D_j, Q) = \left[\frac{\sum_{i=1}^{m} q_i^{\,p} \left(1 - d_{ij}^{\,p}\right)}{\sum_{i=1}^{m} q_i^{\,p}} \right]^{1/p}$$

Avec: **p** une constante fixée

$D = \{d_1, d_2, d_3 \ldots, d_m\}$

$Q = \{q_1, q_2, q_3 \ldots, q_m\}$

Et $0 \leq d_i, q_i \leq 1$ sont des poids du document D_j et de la requête Q associé au terme t_i.

Ce modèle corrige les inconvénients du modèle booléen de base, mais il reste rarement utilisé dans la pratique.

I.6.2 Les modèles vectoriels

I.6.2.1 Le modèle vectoriel

Ce modèle est utilisé dans le système **SMART** (Salton's Magical Automatic Retriever of Text) [122], il fait partie des modèles statistiques, les requêtes et les documents sont représentés dans un espace vectoriel engendré par les termes d'indexation, cet espace est de dimension **m** égale au nombre de termes d'indexation de la collection de documents.

On définit ce modèle formellement comme suit :

L'espace vectoriel est engendré par tous les termes de l'index, soit: $\{t_1, t_2, t_3, \ldots, t_m\}$

Chaque document et chaque requête sont représentés respectivement par un vecteur document et un vecteur requête dans l'espace vectoriel précédent comme suit :

$D = \{d_1, d_2, d_3, \ldots, d_m\}$

$Q = \{q_1, q_2, q_3, \ldots, q_m\}$

Où d_i et q_i correspondent aux poids du terme t_i dans le document et dans la requête.

La pondération des termes de la requête (q_i) peut être laissée à l'utilisateur afin de souligner les termes les plus importants dans sa requête, ou bien la pondération est la même que celle utilisée pour les documents.

Le processus de recherche de ce modèle est basé sur la détermination des vecteurs documents qui s'approchent du vecteur requête en calculant le degré de similarité entre ces deux vecteurs.

Plusieurs méthodes ont été proposées pour calculer la similarité entre deux vecteurs, nous citons ci-dessous trois d'entre elles qui sont fréquemment utilisées :

Produit scalaire : $RSV(Q, D_j) = \sum_{i=1}^{m} q_i \times d_{ij}$

Mesure de Jaccard : $RSV(Q, D_j) = \frac{\sum_{i=1}^{m} q_i \times d_{ij}}{\sum_{i=1}^{m} q_i^2 + \sum_{i=1}^{m} d_{ij}^2 , - \sum_{i=1}^{m} q_i \times d_{ij}}$

La mesure cosinus : $RSV(Q, D_j) = \frac{\sum_{i=1}^{m} q_i \times d_{ij}}{\left[\sum_{i=1}^{m} q_i^2\right]^{1/2} \times \left[\sum_{i=1}^{m} d_{ij}^2\right]^{1/2}}$

- Des études sur le modèle vectoriel ont montré que son processus de recherche d'information est plus performant lorsque les vecteurs requêtes et documents sont normalisés, nous citons parmi les algorithmes de normalisations celui de **Rocchio** [119].
- Ce modèle restitue les documents qui correspondent approximativement à la requête, et enfin il ordonne les documents selon leur degré de similarité avec la requête.
- L'inconvénient de ce modèle et le fait qu'il ne prend pas en compte les relations entre les termes de l'index (il suppose l'indépendance entre les termes de l'index).

I.6.2.2 Le modèle LSI (Latent Semantic Indexing)

En considérant que dans un texte les idées sont plus reliées aux concepts qu'elles décrivent qu'aux termes utilisés pour leur description, l'objectif fondamental de ce modèle [29, 41] est d'aboutir à une représentation conceptuelle des documents (ou représentation sémantique).
Comparativement au modèle vectoriel, dans ce modèle la dimension de l'espace représentant les documents est réduite, et ce, en atténuant l'effet de variation d'usage des termes dans la collection. Ainsi, les documents et les requêtes sémantiquement similaires sont plus proches avec cette représentation par concepts qu'avec la représentation par mots clés. Ceci permet au système de sélectionner des documents pertinents même s'ils ne contiennent aucun mot de la requête.
Cependant, le modèle **LSI** se base sur la décomposition en valeurs singulières, désignée par **SVD** (*Singular Value Decompsition*) de la matrice termes-documents (matrice de dimension (t*d) représentant les poids des termes dans les documents de la collection). Cette technique permet de

réduire l'espace des termes d'indexation, et d'implanter partiellement une recherche sémantique (orientée concepts).

Le principal inconvénient de ce modèle est qu'il n'est pas souple pour certaines applications comme le filtrage d'information. De plus, il n'est pas performant pour des bases documentaires de petite taille [50].

I.6.2.3 Le modèle connexionniste

Ce modèle se base sur le formalisme des réseaux de neurones [27, 28, 89, 90, 1001]. La construction d'un réseau de neurone formel est faite à partir des représentations initiales des documents et de la requête. Le mécanisme de recherche d'information se fait par propagation des signaux (valeurs) de la couche d'entrée vers la couche de sortie en passant par les couches internes (couches cachées) à travers les connexions du réseau. Chaque neurone de la couche d'entrée reçoit une valeur d'activation et transmet une valeur de sortie aux neurones de la couche suivante, ces derniers transmettent à leur tour une valeur de sortie aux neurones de la couche suivante…et ainsi de suite jusqu'à arriver à la couche de sortie dont les valeurs de sortie servent de critères de décision (pertinence des documents et expansion de requête). Ce processus est illustré par la figure **I.5**.

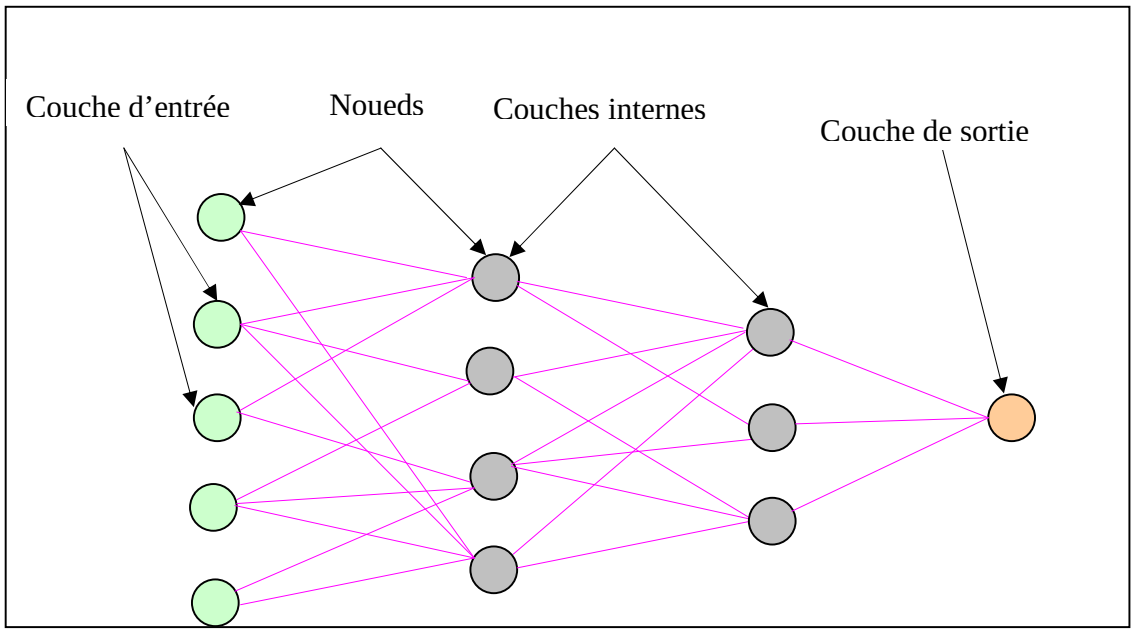


Figure I.5 *- Un réseau de neurones à couches*

La notion du réseau est intéressante pour représenter les relations et associations qui existent entre les différents éléments d'un SRI. Cependant, la définition d'un réseau de neurones en RI est à la charge du constructeur du modèle qui doit définir :

- les différentes couches du réseau et les neurones de chaque couche.
- La fonction d'entrée et de sortie de chaque neurone.
- Les liens entre les neurones et leurs poids associés.

L'avantage de l'approche connexionniste est sa capacité d'apprentissage, ce qui rend les SRI plus adaptatifs.

I.6.3 Les modèles probabilistes

I.6.3.1 Le modèle probabiliste

Le premier modèle probabiliste fut introduit au début des années 1960 par Maron et Kuhns [97], il utilise un modèle mathématique basé sur la théorie des probabilités.

Le processus de recherche se base sur un calcul de proche en proche, de la probabilité de pertinence d'un document vis-à-vis d'une requête. Pour cela, le processus de décision complète le procédé d'indexation probabiliste en utilisant deux probabilités conditionnelles :

- *P(d_{ij}/pert) :* Probabilité que le terme t_i apparaît dans le document D_j sachant que ce dernier est pertinent pour la requête.

- *P(d_{ij}/Non pert) :* Probabilité que le terme t_i apparaît dans le document D_j sachant que ce dernier n'est pas pertinent pour la requête.

Le calcul d'occurrence des termes d'indexation dans les documents est basé sur l'application d'une loi de distribution du type loi de poisson sur un échantillon représentatif de documents d'apprentissage.

En posant les hypothèses suivantes :

- la distribution des termes dans les documents pertinents est la même que leur distribution par rapport à la totalité des documents.
- Les variables documents pertinents et les variables documents non pertinents, sont indépendantes.

Le degré de pertinence d'un document D_j est obtenue par le calcul de probabilité de pertinence, notée **$P(Pert/ D_j)$.**

$$P(Pert/ D_j) = \sum_{i=1}^{m} \log \frac{P(d_{ij}/Pert)}{P(d_{ij}/Non\ Pert)}$$

I.6.3.2 Le modèle de réseau inférentiel bayésien

Ce modèle a été proposé par Turtle [147]. Les réseaux bayésiens sont des graphes acycliques. Dans le graphe, les nœuds représentent les variables aléatoires associées aux termes de l'index, les documents et les requêtes de l'utilisateur.

Une variable aléatoire associée avec un document dj représente l'évènement d'observer ce document. Les arcs sont 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. La variable aléatoire associée à la requête de l'utilisateur modélise l'évènement que la requête a été vérifiée. La valeur de ce nœud requête est une 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 I.6, issue de [147], illustre un réseau inférentiel bayésien simple de pertinence d'un document vis-à-vis d'une requête composée de trois termes. L'évènement '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 évènements.

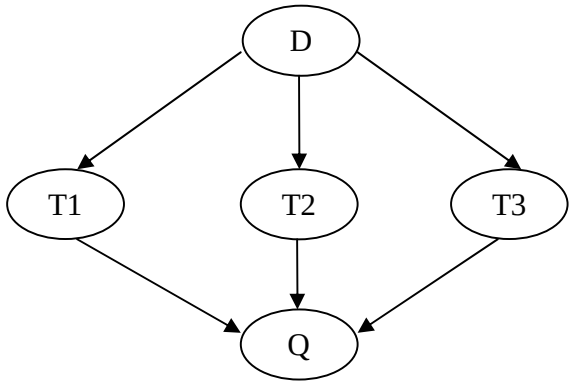


Figure I.6 *- Un réseau inférentiel bayésien simple*

Les trois sujets sont inférés par l'événement "le document est pertinent" (D=1). Par l'enchaînement de règles de probabilités, la probabilité jointe des autres nœuds du graphe est :

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)(T3/ D)P(Q/ T1, T2, T3)

La probabilité de réalisation de la requête P(Q =1/D=1) peut être utilisée comme score d'ordonnancement des documents:

$$P(Q=1/\mathrm{D}=1)=\frac{P(Q=1,D=1)}{P(D=1)}$$

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

Le modèle nécessite la connaissance de P(D = [0|1]), P(Ti = [0|1]/D = [0|1]) et P(Q = [0|1] / (T1, T2, . . ., Tn) ϵ $\{0, 1\}^n$), cette dernière étant la plus difficile à trouver car le nombre de probabilités à spécifier augmente exponentiellement avec le nombre de termes de la requête. Pour résoudre ce problème, Turtle [147] a identifié quatre formes canoniques de P(Q/ T1, T2, . . .Tn): and, or , sum et wsum.

Le modèle inférentiel bayésien a été mis en oeuvre dans le système INQUERY [5]. Le cadre probabiliste dans lequel se situe INQUERY peut être utilisé pour formuler des requêtes simples basées sur des mots clés, des requêtes booléennes, des requêtes basées sur des phrases ou bien une combinaison des trois types. Pour ce faire, INQUERY propose des opérateurs de moyenne et de moyenne pondérée, des opérateurs booléens probabilistes ou stricts (on conserve alors les probabilités), des opérateurs de proximité et de synonymie. Une procédure d'analyse de la requête permet de générer une forme inférentielle prête à être évaluée. INQUERY propose également une expansion de requête. La figure I.7 représente un réseau bayésien utilisé par

INQUERY.

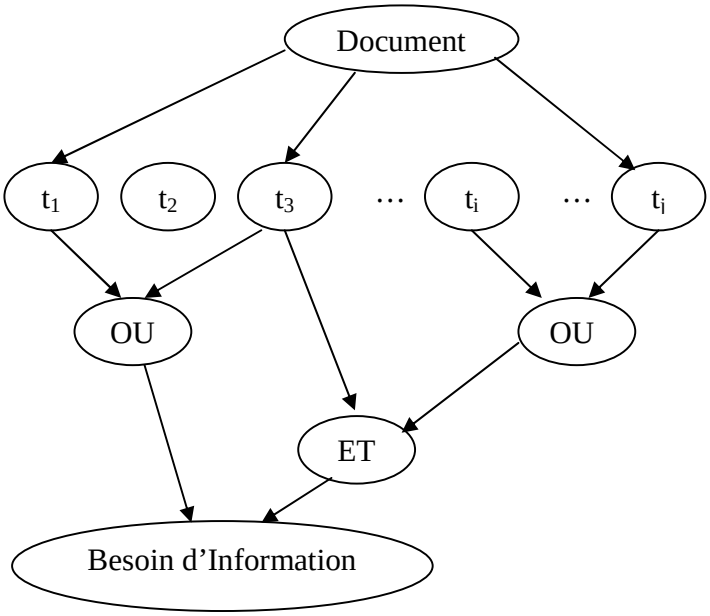


Figure I.7 *- Un réseau bayésien utilisé par INQUERY*

L'inconvénient principal des réseaux bayésiens reste le calcul des probabilités, qui demande un temps exponentiel au nombre de termes dans la requête même si l'introduction des quatre formes canoniques dans [147] résout partiellement le problème.

I.6.3.3 Le modèle de langue

Le premier modèle de langue introduit par Ponte et Croft [114] en RI était en 1998. Ce modèle considère la pertinence d'un document face à une requête comme la probabilité que la requête puisse être générée par le modèle de langue du document.

P (Q/ D_j) : La probabilité que la requête puisse être générée par le modèle de langue.

Le principe de ce modèle est donc, d'évaluer la probabilité **P (Q/D_j)** puisque :

$$RSV(Q, D_j) = P(Q/ D_j).$$

Pour estimer P (Q/ D_j), l'ensemble des hypothèses suivantes est posé :

- Les termes de la requête sont indépendants.
- Tous les termes d'indexations sont considérés lors de l'estimation de P (Q/ D_j), soit m le nombre de ces termes.

- La requête Q est composée des termes q_1,…, q_n et les termes q_{n+1}, …, q_m sont absents de Q.

On a alors :

$$RSV(Q, D_j) = P(Q/ D_j) = \prod_{q_1 \in Q} P(q_i/ D_j) \times \prod_{q_1 \notin Q} (1 - P(q_i/ D_j))$$

Plusieurs approches ont été développées pour estimer la probabilité d'observer le terme q_i dans le document D. Parmi ces approches, nous trouvons celle de Ponte & Croft [114], Song & Croft [136] et Hiemstra [70].

I.7 Evaluation des systèmes de recherche d'information

L'évaluation d'un système de recherche d'information, consiste à définir des outils de mesures pour tester ces performances. Pour Cleverdan [40], l'évaluation d'un système peut être menée en mesurant six quantités principales :

- L'univers de discours de la collection : le degré auquel le système inclut l'information pertinente.
- temps de réponse : temps moyen entre la formulation de la requête et la réponse du système.
- La présentation des réponses (ordre de pertinence).
- L'effort demandé à l'utilisateur.
- Le rappel
- La précision.

L'évaluation se fait selon deux aspects : aspect efficience et aspect efficacité. L'aspect efficience peut être résumé aux quatre premiers points cités par Cleverdan. L'aspect efficacité consiste en la faculté du système de recherche à retourner un maximum de documents pertinents et un minimum de documents non pertinents, les mesures de rappel et précision sont les mesures les plus simples et les plus utilisées pour prendre en compte ces deux contraintes.

I.7.1 Les mesures de Rappel/Précision

L'efficacité de tout SRI est mesurée principalement par le rappel et la précision. La définition de ces mesures est centrée autour de la notion de la pertinence d'un document étant

donnée une requête. Il est donc important pour pouvoir mesurer l'efficacité des SRI sur des collections de documents, d'avoir des ensembles de requêtes pour lesquelles les pertinences sont connues. Pour répondre à la question qui consiste à savoir si le SRI restitue tous les documents pertinents de la collection, des collections de tests ont été mises en place par différentes organisations. Dans ce contexte, la campagne d'évaluation, probablement la plus connue dans le domaine de RI, est la campagne TREC (Text REtrieval Conference) qui a débuté dans les années 90 et qui est une série de conférences soutenues par le **DARPA** (Defense Advanced Research Projects Agency) [93]. TREC a permis de réunir des recherches communes sur l'ensemble des méthodologies de RI et offre des moyens de comparaisons entre différents systèmes. Différents aspects sont évalués tels que le temps de réponse des SRI, les ressources mises en place par les SRI, la capacité des SRI à restituer des documents pertinents, etc. Lorsque les collections ne sont pas trop volumineuses, des experts jugent pour des requêtes données les pertinences des documents manuellement. Lorsqu'elles sont volumineuses, d'autres approches sont utilisées comme par exemple l'échantillonnage, le pooling ou la recherche. Le pooling consiste à considérer l'union des sous ensembles des n premiers documents restitués par différents systèmes comme pertinents. L'échantillonnage consiste à estimer la taille des ensembles de documents réellement pertinents. L'approche basée sur la recherche consiste à opérer une recherche guidée qui prend fin lorsque l'expert juge que tous les documents pertinents ont été trouvés.

Nous allons détailler ci-dessous les notions de bruit, silence, précision et rappel
Soient **P** l'ensemble des documents pertinents dans le corpus et **R** l'ensemble des documents restitués, pour une même requête.
Nous définissons le tableau de contingence de pertinence suivant :

	Pertinent	**Non Pertinent**	
Restitué	$P \cap R$	$\overline{P} \cap R$	R
Non Restitué	$P \cap \overline{R}$	$\overline{P} \cap \overline{R}$	$\overline{R}$
	P	$\overline{P}$	

Tableau I.1 *- Contingence de la pertinence*

Le silence : mesure l'ensemble de documents pertinents non retournés par le système

Le bruit : mesure l'ensemble de documents non pertinents (parasites) retournés par le système

$$Silence = P \cap \overline{R}$$

$$Bruit = \overline{P} \cap R$$

$$Taux\_précision = \frac{\text{nombre de documents pertinents restitués}}{\text{nombre de documents restitués}} = \frac{|P \cap R|}{|R|}$$

$$Taux\_rappel = \frac{\text{nombre de documents pertinents restitués}}{\text{nombre de documents pertinents}} = \frac{|P \cap R|}{|P|}$$

La précision : mesure le taux de documents pertinents parmi tous les documents retrouvés par le système de recherche.

Le rappel : mesure le taux de documents pertinents retrouvés par le système de recherche parmi l'ensemble des documents pertinents de la collection.

Une méthode pour évaluer et comparer les systèmes de recherche d'information consiste à tracer les courbes *précision_rappel* de chaque système puis faire des études comparatives sur ces courbes, si une courbe d'un système est au dessus d'une autre, nous pouvons alors affirmer que le premier système est plus efficace que le second. La courbe *précision_rappel* est dessinée à partir des coordonnées (*taux_rappel, taux_précision*) pour l'ensemble des requêtes soumises à la collection de test.

Dans le cas d'un système idéal, les valeurs de précision et de rappel seront égales à 1, c'est-à-dire que tous les documents pertinents et rien que ceux-ci, sont sélectionnés, on aurait alors une droite. En pratique, les valeurs de rappel et de précision évoluent inversement ce qui fait que la courbe *précision_rappel* est décroissante. Ainsi, plus la courbe d'un système est élevée, plus il est performant.

I.7.2 Les mesures combinées

L'inconvénient principal des mesures de rappel et de précision est qu'elles représentent des aspects différents de l'ensemble des documents sélectionnés. Des mesures agrégatives ont donc été proposées pour synthétiser l'information contenue dans ces deux mesures. Nous présentons les plus importantes :

• **Moyenne harmonique**

Une moyenne harmonique combine le rappel et la précision en un nombre compris entre 0 et 1, elle est définit dans [131]:

$$Harmonique = \frac{2}{\frac{1}{rappel} + \frac{1}{précision}}$$

Elle vaut 0 lorsque aucun document pertinent n'est sélectionné et 1 lorsque tous les documents sélectionnés sont pertinents. La valeur de la moyenne harmonique est élevée lorsque les valeurs de précision et de rappel sont élevées, garantissant ainsi un compromis entre ces deus mesures.

• **La E-mesure**

C'est une extension de la mesure harmonique, proposée par [117]. Elle permet à l'utilisateur de spécifier s'il est plus intéressé par le rappel ou par la précision:

$$E - mesure(b) = 1 - \frac{1 + b^2}{\frac{b^2}{rappel} + \frac{1}{précision}}$$

Le paramètre b désigne l'importance relative de la précision et du rappel. Si b>1, on privilégie la précision et si b<1 on privilégie le rappel. Malgré leur caractère hautement subjectif, les mesures de rappel et de précision ont permis aux modèles de RI de se développer depuis les années 1960, les courbes de *précision_rappel* et la précision moyenne se sont imposées et sont utilisées dans les compagnes d'évaluation.

D'autres mesures moins conventionnelles peuvent aussi servir à évaluer la performance d'un système de recherche d'information. Mizzaro dans [100], fait une étude complète des différentes mesures d'évaluation utilisées en RI.

I.8 Conclusion

Nous avons présenté dans ce chapitre, une vision globale du domaine de la recherche d'information. Nous y avons développé les principales étapes d'un processus de recherche d'information, à savoir, la représentation des documents et des requêtes, l'appariement document-requête, l'indexation avec la présentation de certaines techniques d'indexation, la reformulation de requête, les modèles de recherche d'information les plus connus, la notion de pertinence ainsi que l'évaluation des systèmes de recherche d'information.

Avec le développement des technologie de l'information et de la communication tel que le réseau Internet, les corpus de documents ont évolué vers des représentations structurées (ou semi-structurées), or les systèmes de recherche d'information traditionnels sont conçus uniquement pour exploiter le contenu sémantique (le texte) des documents et donc ne sont pas adaptés au traitement simultané de la structure et du contenu. Pour cela, nous assistons aujourd'hui au développement de systèmes de recherche d'informations semi structurées qui cherchent à tirer profit de la structuration de l'information en la combinant avec l'information de contenu.

Dans le chapitre suivant, nous nous intéressons particulièrement à ces nouveaux systèmes pour la recherche d'information semi structurée. Les principales problématiques qu'ils cherchent à résoudre (interrogation, indexation, identification des unités d'information pertinentes) sont comparables à celles soulevées dans la RI traditionnelle, mais doivent être abordées en ajoutant la dimension structurelle à la dimension de contenu.

Chapitre II
Recherche d'Information Structurée

II.1 Introduction

Le développement du document électronique et du Web [99] ont vu émerger puis s'imposer des formats de données structurés, tels que le SGML (Standard Generalized Markup Language) [60] et le XML (eXtensible Markup Language) [32, 151], conçus à l'origine pour faciliter l'échange et la standardisation des données. Aujourd'hui, l'utilisation du langage XML est de plus en plus répondue dans de nombreuses applications dans le cadre de la représentation, du stockage et de l'échange de documents et de données, en particulier sur le Web. Par conséquent, le type des documents mis à la disposition des utilisateurs a évolué du simple document texte "plat" aux documents structurés ou semi structurés, permettant de représenter l'information sous une forme plus riche que le simple contenu et adaptée à des besoins spécifiques. En effet, les documents semi structurés permettent de représenter conjointement l'information textuelle et l'information de structure d'un document. La connaissance de la structure des documents est une ressource additionnelle qui devrait être exploitée pendant la recherche d'information afin de mieux exprimer un besoin d'information. Dans le contexte de la RI, la question majeure soulevée par ce type de document concerne la manière de manipuler efficacement la structure et le contenu du document pour mieux répondre aux besoins de l'utilisateur. Ces besoins peuvent être formulés par le biais de requêtes formées que de mots clé ou par des requêtes comportant des mots clés et des contraintes structurelles (des balises).

Le domaine de la Recherche d'Information Structurée (RIS) est un domaine qui répond à la recherche d'information dans les documents semi structurés généralement de type XML. Nous assistons aujourd'hui à l'expansion de ce domaine, il s'est principalement développé ces dernières années notamment avec l'apparition de l'initiative INEX[1][74] et de plusieurs Workshop dans différentes conférences de RI (SIGIR [2] [132] par exemple).

[1] INEX : Initiative for the evaluation of XML Retrieval
[2] SIGIR : Special Interest Group of Information Retrival

Les approches proposées dans le cadre de la recherche d'information XML peuvent être classées en deux principales catégories : (i) l'***approche orientée données*** utilise des techniques développées par la communauté des Bases de Données (BD), (ii) l'***approche orientée documents*** est prise en charge par la communauté recherche d'information. Les *approches orientées données* s'intéressent davantage à la structure du document. Plusieurs langages [34] ont été définis, LOREL [1], XML-QL [94], QUILT [38], XML-GL [36]. Les *approches orientées documents* considèrent les documents XML comme une collection de documents textes comportant des balises (éléments) et des relations entre ces balises. Les balises sont utilisées comme moyen pour mieux identifier la pertinence d'une partie de document vis –à- vis d'une autre partie. La majorité des travaux ont en fait adapté les modèles de RI reconnus pour traiter les documents XML [52, 69, 91, 102, 106, 125, 130].

Si les approches orientées BD permettent de traiter efficacement la structure des documents XML, elles sont cependant limitées pour le traitement de la partie textuelle des documents. Les mots clés sont en effet traités de façon binaire (présent /absent), or il a été démontré en RI textuelle que la prise en compte des poids des mots-clés dans un document est primordiale, voire nécessaire [123]. Ceci permet de mesurer un degré de pertinence d'un document (ou d'une partie de document) vis-à-vis d'une requête et donc de renvoyer à l'utilisateur une liste triée de résultats, comme le proposent les approches de RI.

Les enjeux actuels pour le domaine de la recherche d'information structurée, sont le développement de modèles de RI pour une prise en compte simultanée de la structure et du contenu, ainsi que la définition et l'évaluation de nouvelles problématiques spécifiques aux données semi structurées.

Dans ce chapitre, nous introduisons la notion de document semi structuré avec les notions de contenu et de structure, puis nous développons les différentes problématiques soulevées par la recherche d'information structurée ainsi que les solutions apportées. Ensuite, nous passons en revue les différents modèles permettant d'effectuer la recherche d'information structurée et nous terminons par la présentation des mesures d'évaluation proposées dans le cadre de la campagne INEX, afin de pouvoir estimer la performance des différents systèmes et de stimuler la recherche dans ce domaine.

II.2 Les documents semi structurés

Les documents semi structurés sont des documents électroniques représentées selon un format structuré, c'est-à-dire un format qui utilise des balises pour décrire la structure logique des documents [15]. Cette structure logique d'un document correspond à un type de structuration permettant de présenter de façon cohérente et logique le contenu des documents. Souvent, la structure logique d'un document sera sa division en parties, chapitres, sections, etc., de même que certaines autres unités telles des notes de bas de page ou des références bibliographiques.

Décrire la structure d'un document, consiste à identifier et décrire chacun des éléments textuels – ou non textuels – qui le constituent. Ceci dit, cette description peut prendre plusieurs formes. En effet, nous distinguons, en général, deux types de structure : la structure physique et la structure logique.

Si l'on s'attache à retrouver ***la structure physique*** d'un document, on décrira sa mise en page, on définira les différentes zones de texte, leur agencement les unes par rapport aux autres, ainsi que l'ensemble de leurs caractéristiques typographiques : police, couleur, gras, italique, etc.

Si l'on s'intéresse maintenant à ***la structure logique***, nous décrirons plutôt le rôle et la nature de chaque élément d'un document ainsi l'ensemble des liens hiérarchiques et/ou logiques qui les lient les uns aux autres (titre, auteur, introduction, résumé, mots clés, chapitre, sous chapitre, section, sous section, conclusion).

Nous noterons qu'une structure physique bien conçue aura pour principale qualité de rendre lisible la structure logique du document ce qui peut amener à confondre les deux. Cependant, il est important de bien faire la distinction entre ces deux types de structuration, car même si elles peuvent conduire toutes deux à un découpage du document équivalent, la manipulation des éléments du document et l'accès à chacun d'eux, sont très différents.

En langage XML [32, 72], la structure logique des documents est définie par des balises (tags ou labels) encadrant des portions d'informations. Ce type de document représente un compromis entre les données fortement structurées issues de la communauté bases de données (données relationnelles par exemple) et les données faiblement structurées issues des communautés document numérique et RI (documents plats, images etc.…).

Le tableau suivant est un récapitulatif des principales caractéristiques des documents plats, semi structurés et structurés dans le cadre de la recherche d'information [140].

	Documents plats	**Documents semi structurés**	**Documents structurés**
Contenu	texte seulement	texte + structure	structure + données
Besoin d'information	général		précis
L'unité recherchée	document	élément (doxel)	(tuple)
Demande	texte seulement	contenu et/ou structure	
Requête	mots clés	mots clés et/ou langages de requête structurés	langages de requête structurés
Interprétation	vague		stricte

Tableau II.1 *- Résumé des caractéristiques des documents plats, semi structurés et structurés*

Le format XML est aujourd'hui le format de représentation le plus utilisé. Par conséquent, dans la suite nous nous focalisons sur les documents semi structurés de type XML et les technologies qui s'y rapportent.

II.2.1 Notions de structure

Nous présentons ici les notions de base liées à la structure, s'appliquant aussi bien à des documents SGML que des documents XML:

- Une balise (ou tag ou label) est une suite de caractères encadrés par "<" et ">", comme par exemple <balisename>.

- Un élément est une unité sémantique identifiée, délimitée par des balises de début < b > et de fin < /b >, comme par exemple <balisename> texte </balisename>. Les éléments peuvent être imbriqués (mais ne doivent pas se recouvrir) formant ainsi une structure hiérarchique comme le montre l'exemple suivant:

```
<Ouvrage>
        <Titre > Recherche d'Information </Titre >
        <Résumé>Devant la masse croissante d'information … </Résumé>
….
         <Chapitre>
            <Titre chapitre> Indexation </titre chapitre>
            <Paragraphe> L'indexation est le processus … </Paragraphe>
          </Chapitre>
</Ouvrage>
```

Nombreuses sont les technologies reliées au format de données XML. Parmi les plus connus et les plus susceptibles d'aider à la recherche d'information (voir figure II.1), citons les DTD (Document Type Definition), les Schémas XML et les mécanismes de base pour adresser des éléments dans des documents XML (XPath), pour traiter les documents XML (DOM et SAX), pour présenter et transformer les contenus XML (XSL), les espaces de nom, XLink et XPointer pour la gestion des liens, etc.

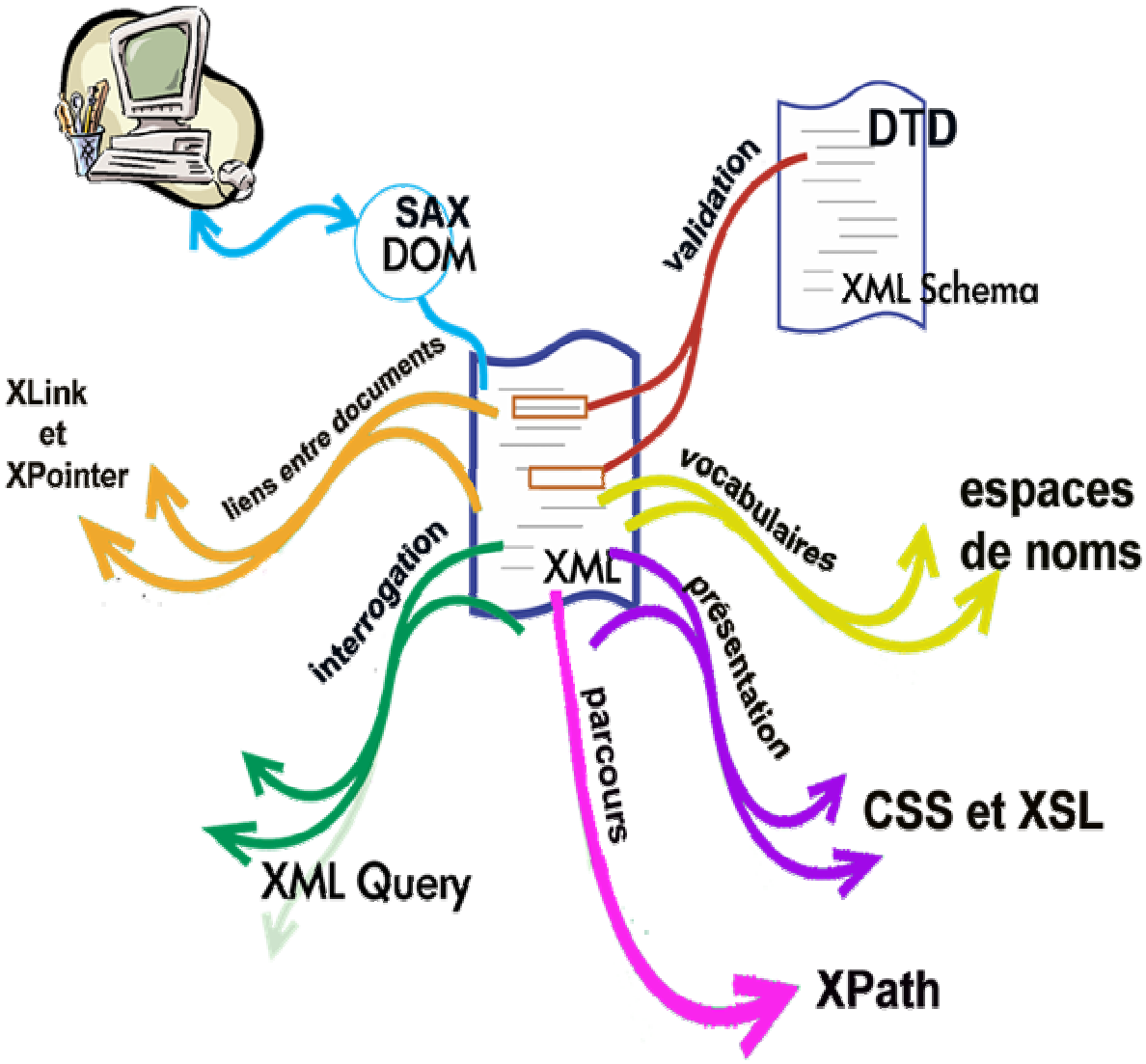


Figure II.1 *- La galaxie XML*

II.2.2 DTD et Schémas XML

Les documents se conformant aux règles syntaxiques définies par les recommandations XML du W3C, sont dits "bien formé". Cette propriété garantit qu'un analyseur XML pourra analyser le document XML avec succès. La correction syntaxique des documents XML n'est cependant pas suffisante, il est intéressent de pouvoir exprimer des contraintes sur leur structure en fonction du vocabulaire utilisé. Ces contraintes sont exprimées par des langages ou grammaires qui permettent de décrire des classes de documents. D'où vient la notion de document XML valide, c'est-à-dire se conformant à une grammaire, à savoir les DTD (Document Type Definition) ou les Schémas XML. On dira alors qu'une classe de documents possède une structure générique définie par la DTD (ou le schéma XML) alors qu'un document instance de cette classe possède une structure spécifique.

II.2.2.1 Les DTD (Document Type Definition)

La DTD associée au document décrit la structure générique du document. Elle contient l'ensemble des balises qu'il est possible d'inclure, ainsi que des relations de composition entre ces balises [72]. Les DTD sont nécessaires pour la définition d'entités générales utilisées dans les instances des documents, mais il n'est pas obligatoire d'associer une DTD à un document XML. La DTD ne gère pas les espaces de noms et son système de typage est pauvre (le contenu des nœuds textes -#PCDATA- ne peut pas être typé) [67], mais il est toutefois possible d'assigner des valeurs par défauts aux attributs et de définir les valeurs possibles par une énumération. Les contraintes sur les modèles de contenu sont : la séquence ",", le choix "|" et les opérateurs de cardinalité:? (1|0), + (au moins un), * (0 ou plus). La figure suivante illustre un exemple de DTD:

```
< ?xml version="1.0" ?>
<!--DTD pour document XML  de type 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, paragraphe)>
<!ELEMENT sous-titre (#PCDATA)>
<!ELEMENT paragraphe (#PCDATA)>
```

Figure II.2 *- Exemple d'une DTD représentant un article*

II.2.2.2 Les Schémas XML

Le Schéma XML présente de nombreuses améliorations par rapport aux DTD, notamment une plus grande flexibilité et un typage plus important des données [50]. C'est une recommandation du W3C depuis 2001 [153] pour remplacer les DTD jugées trop peu expressives. Elle est divisée en plusieurs sous recommandations: *XML Schema Part 0* qui décrit l'utilisation des Schémas XML, *XML Schema Part 1* qui décrit les structures et *XML Schema Part 2* qui décrit les types de données. Elles adoptent la syntaxe XML, et proposent un système de typage très riche permettant à l'utilisateur de définir ses propres types de données ainsi que des mécanismes d'héritage et de sous typage. L'occurrence des éléments peut être contrôlée d'une façon très fine, les espaces de noms sont gérés et des mécanismes d'importation et d'inclusion sont proposés. Un exemple d'un schéma XML est donné dans la figure suivante:

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            targetNamespace="http://www.library.org"
            xmlns="http://www.library.org"
            elementFormDefault="qualified">

    <xsd:element name="livre">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="titre_livre" type="xsd:string"/>
          <xsd:element name="auteur" type="xsd:string"/>
          <xsd:element name="annee_edition" type="xsd:string"/>
          <xsd:element name="maison_edition" type="xsd:string"/>
          <xsd:element name="langue" type="xsd:string"/>

           <xsd:element name="titre_chapitre" type="xsd:string"/>
             <xsd:element name="titre_paragraphe" type="xsd:string"/>
             <xsd:element name="contenu_paragraphe" type="xsd:string"/>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
</xsd:schema>
```

Figure II.3 - *Exemple d'un Schéma XML représentant un livre*

II.2.3 Espaces de noms

Les espaces de noms (namespaces) est recommandation du W3C [157] qui permet de combiner dans un seul document des vocabulaires XML (éléments et attributs) définis par plusieurs applications. Comme les vocabulaires ont été définis indépendamment les uns des autres, le risque de collision (un nom d'élément ou d'attribut utilisé par deux applications

différentes) est grand. Les espaces de noms résolvent alors ce genre de problème : ils nomment de manière unique un élément ou un attribut en associant un domaine à un ensemble de noms. En pratique, on préfixe l'objet de l'espace de nom correspondant. Les espaces de nom sont identifiés par des URIs (Uniform Resource Identifiers), mais l'on précise pour chacun d'eux un label qui servira de préfixe aux balises concernées.

```
<livres xmlns:eda =" editeur -A"  xmlns:edb =" editeur -B">
      <eda : livre >
           <eda: auteur >... </ auteur >
           <eda: titre >... </ titre >
      </eda:livre >
      <edb: livre titre ="..." auteur ="..."/ >
</livres >
```

Figure II.4 *- Exemple d'utilisation des espaces de noms*

II.2.4 DOM (Document Object Model)

L'API (Application Programming Interface) DOM est une spécification du W3C [152] qui fournit un moyen pour gérer des documents XML dans un programme. Par exemple, il offre la possibilité de créer des documents et des fragments de documents, de naviguer dans les documents, etc. Le DOM s'intercale généralement entre le parseur XML et l'application ayant besoin des informations du document. Ce qui signifie que le parseur lit les données du document pour les diriger vers le DOM qui est alors utilisé par une application de niveau supérieur. L'API DOM est basée sur une structure d'objets pour représenter un document balisé. L'analyseur génère un arbre d'objets reliés entre eux, chaque objet représentant un atome du document XML.

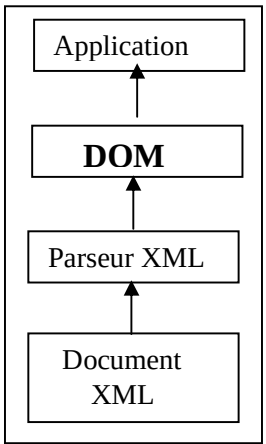


Figure II.5 *- Position du DOM*

Le traitement de documents XML à l'aide du DOM monopolise une grande quantité de mémoire vive car le DOM crée tous les objets en mémoire associant chacun d'eux à un nœud dans le document XML [67].

II.2.5 SAX

L'API DOM n'est pas le seul outil pour analyser les documents et en extraire des informations. L'autre API s'appelle SAX (Simple API for XML) et s'avère efficace dans les opérations où le DOM présente des faiblesses (à l'inverse, le DOM comble les lacunes de SAX). Les deux approches sont complémentaires et indispensables au développeur XML [72].

L'API SAX a été développée pour optimiser l'analyse de documents XML volumineux. Le problème avec DOM est qu'il doit créer une image importante du document en mémoire avant que l'on puisse lire celui-ci. Cette opération nécessite beaucoup de ressources. Elle n'est pas adaptée aux extractions d'une partie du document.

Si le document doit être traité entièrement, DOM produit d'excellents résultats mais l'opération est longue. Par contre, si le document doit être traité partiellement, SAX est plus approprié car orienté évènement.

SAX possède les avantages suivants [72]:

- Elle est peu exigeante en ressources : le document est chargé partiellement en mémoire. Bien entendu, si SAX s'utilise pour construire l'image mémoire du document, il est préférable de prévoir un espace équivalent à celui que réserverait le DOM.
- Elle est rapide, car le document n'a pas à être chargé en totalité pour que son analyse commence.
- Elle met l'accent sur le contenu plus que sur la mise en forme du document.
- Ses qualités de filtrage permettent de se concentrer sur le sous-ensemble de données qui nous intéresse.

Pourquoi n'est-elle donc pas utilisée dans toutes les circonstances ? Parce que SAX possède aussi quelques inconvénients :

- Les données sont extraites séquentiellement. Il n'est pas possible de contrôler l'ordre de recherche du parseur. Autrement dit, les données sont reconstituées à partir d'événements spécifiques, ce qui peut poser un problème pour les recherches complexes.
- SAX est en lecture seule : il est impossible de modifier les éléments du document. La seule façon de générer un nouveau document est d'intercepter tous les événements, de copier le texte d'origine et d'y intégrer les modifications.

II.2.6 XPath

XPath est un langage qui permet d'adresser les parties d'un document XML afin de permettre d'en extraire avec précision les informations voulues, à l'aide de spécification d'expressions régulières décrivant un chemin ou une composition de chemins dans une arborescence XML. Xpath 1.0 est une recommandation du W3C [39]. Il possède une syntaxe simple et non ambiguë qui permet d'écrire des expressions qui réfèrent à tout élément, sous élément, attribut, instruction de traitement, etc.… d'un document XML. XPath indique les nœuds par leur position absolue, position relative, type, contenu ou autres critères.

Un document XML a une structure arborescente: il est composé de nœuds. Pour XPath, il existe sept types de nœuds:

- Le nœud racine
- Les nœuds d'éléments
- Les nœuds de texte
- Les nœuds d'attributs
- Les nœuds de commentaires
- Les nœuds d'instructions de traitement
- Les nœuds d'espaces de noms.

II.2.7 XPointer

XPointer définit la syntaxe pour les fragments des documents XML [65]. Le but est de pouvoir adresser d'une manière précise des parties d'une ressource XML et de la représenter. XPointer est une extension de XPath, il réutilise en grande partie les mêmes concepts et syntaxe, il est aussi possible avec XPointer de faire des recherches avec des chaînes de caractères.

II.2.8. XQuery

XQuery est un langage de requête [155, 156], pour l'interrogation des documents XML, proposé par le W3C comme un standard qui prend avantage à la fois du contenu et de la structure spécifiés dans la collection, pour retourner les éléments XML qui satisfont strictement les conditions logiques spécifiées dans la requête. Le but est de combiner la flexibilité des moteurs de recherche plein texte, tout en tirant le maximum d'avantages de la connaissance de la structure comme dans les bases de données.

II.2.8 XLink

XLink fournit les constructions hypertexte de XML. Il permet d'effectuer des liens simples ou étendus (multisource, multicible) et des annotations (ressources contenant d'autres liens) [45]. De plus, chaque élément peut devenir un lien.

II.2.9 XSL (eXtensible Style sheet Language)

XSL est un langage de feuilles de styles. Il est composé de deux parties principales [6] :

- XSLT (XSL Transformation) : Conçu au départ comme le composant de transformation du langage de formatage XSL du W3C, XSLT est devenu un langage de transformation de documents XML indépendant. Il permet la transformation de documents XML en documents XSL-FO (XSL Formatting Objects) mais aussi vers n'importe quel autre vocabulaire XML (XHTML, SVG, etc . . .) ainsi que vers d'autres modèles comme le texte pur ou LaTeX.

- XSL/FO : langage qui permet de formater l'affichage et/ou l'impression d'un document XML (boîtes, positionnement, ordonnancement et propriétés d'affichage). Il s'agit d'une extension de CSS (Cascading Style Sheet), associée aux documents HTML. XSLT 1.0 et XSL/FO 1.0 sont des recommandations du W3C [154].

II.3 Les spécificités de la Recherche d'Information Structurée

II.3.1 L'unité d'information pertinente

Si les systèmes de RI classiques considèrent les documents dans leur globalité, les systèmes de RIS cherchent plutôt à identifier des parties de documents les plus pertinentes à une requête. L'introduction d'une information structurelle dans le codage des documents bouleverse complètement la notion d'unité d'information. Cette unité était jusqu'à présent le document, ce n'est pas le cas aujourd'hui. Ce changement quant à la notion d'unité d'information est à l'origine de l'apparition de nouvelles problématiques spécifiques aux documents semi structurés. Certaines de ces nouvelles problématiques sont aujourd'hui particulièrement étudiées dans le cadre de la RI sur les documents semi structurés notamment au travers de l'initiative INEX. Ainsi, le concept de "granule ou d'unité d'information" renvoyée à l'utilisateur n'est plus un document dans sa totalité. Il correspond plutôt à un nœud (sous arbre) dans la représentation arborescente du document semi structuré dont la pertinence vis-à-vis d'une requête est calculée selon deux nouvelles notions : "exhaustivité" et "spécificité". On parle alors d'unité d'information exhaustive à une requête, si elle contient toutes les informations requises par la requête, et d'unité d'information spécifique à une requête, si tout son contenu concerne la requête. De plus, la partie du document renvoyée par le système se doit être auto explicative, c'est-à-dire de ne pas dépendre du reste du document pour être comprise. Par ailleurs, les résultats renvoyés doivent se suffire à eux-mêmes. L'application stricte du principe d'auto-explicativité conduirait à une perte bien trop importante de spécificité; donc, pour satisfaire le principe d'auto-explicativité et le principe de spécificité, le système de recherche doit trouver les sous arbres de taille minimale pertinents à la requête. Par conséquent, le principe de RI dans les documents semi structurés est défini comme suit : "Un système de recherche d'information devrait toujours renvoyer l'unité d'information la plus exhaustive et spécifique répondant à une requête", ce qui revient à trouver les sous arbres de taille minimale pertinents à la requête.

De par leur structure, l'utilisateur interrogeant des corpus de documents XML peut formuler deux types de requêtes, selon sa connaissance du corpus:

- Les requêtes portant sur le contenu, composées de simples mots clés (la structure n'est pas spécifiée) et cela dans le cas où l'utilisateur ne connaît pas précisément la structure des documents ou bien son besoin d'information est trop vague. Dans ce cas, c'est le système qui se

charge de trouver l'unité d'information la plus pertinente et qui décide du granule d'information à retourner à l'utilisateur.

- Les requêtes portant sur le contenu et la structure, où l'utilisateur peut spécifier des besoins précis sur certains éléments de structure qu'il désire voir retourner en utilisant des contraintes structurelles dans la requête.

Afin de répondre à ces nouvelles notions, il y a lieu d'adapter les anciens systèmes pour pouvoir répondre aux problématiques liées à la modification de l'unité d'information recherchée que cela soit au niveau de l'indexation, l'interrogation ou même la recherche et le tri des unités d'information renvoyées.

II.3.2 La problématique d'indexation

La problématique dans le cadre de l'indexation se situe principalement dans l'information structurelle [125]. Dans le cas des documents plats, le processus d'indexation consiste à collecter et à pondérer les mots clés permettant un ciblage sémantique du document. Dans le cas des documents semi structurés, la dimension structurelle s'ajoute au contenu. Donc, la vision du processus d'indexation change puisqu'elle sera guidée par la structure du document, ce qui engendre les questions suivantes :

- Que doit-on indexer de la structure des documents ?
- Comment relier cette structure au contenu même du document ?
- Comment pondérer les termes d'indexation, c'est-à-dire, comment évaluer l'importance d'un terme au sein de l'élément, du document et de la collection ?

L'objectif du processus d'indexation dans ce cas, est alors de tenir compte de la structure et du contenu pour permettre d'effectuer des recherches portant sur le contenu et/ou la structure des documents.

II.3.3 La problématique d'interrogation

Il s'agit de permettre à l'utilisateur d'exprimer ces besoins d'information diversifiés (concernant le contenu des documents et/ou la structure) de manière simple via un langage de requête (langage naturel, mots clés, …), et de sélectionner les unités d'informations jugées pertinentes à ces besoins.

L'évaluation d'une information vis-à-vis d'une requête se complique et implique d'autres questions dans le cadre des documents XML, principalement en ce qui concerne la structure :

- les requêtes orientées contenu, imposent au SRI de décider la granularité appropriée de l'information à retourner.
- les requêtes orientées contenu et structure : l'utilisateur spécifie les éléments à retourner et précise la granularité de l'information pertinente, alors la dimension de la spécificité n'a plus réellement de sens. La pertinence des informations structurelles doit pouvoir être évaluée, et l'arbre de la requête et l'arbre du document doivent pouvoir être comparés de façon non stricte.

Afin de répondre à ces problématiques soulevées par la notion de structure, les systèmes de recherche d'information traditionnels doivent être adaptés ou bien de nouvelles méthodes doivent être proposées pour l'indexation et l'interrogation des documents semi structurés.

II.4 Techniques d'indexation des documents semi structurés

Une façon simple d'indexer des documents XML est de les considérer comme des fichiers plats, et le processus d'indexation dans ce cas-là est similaire à celui utilisé en RI traditionnelle, c'est-à-dire qu'il consiste à identifier et à extraire les termes importants du contenu textuel des documents. Cependant, aucune recherche sur la structure n'est plus possible, et l'unité d'information renvoyée à l'utilisateur est le document dans sa totalité.

Un schéma d'indexation de documents XML doit couvrir certains aspects, comme de permettre la recherche par mots clés, le traitement des expressions de chemin sur la structure XML et d'autoriser le traitement de prédicats vagues et précis sur le contenu de documents XML [125].

Nous allons décrire ci-dessous l'indexation de l'information de contenu (textuelle) et l'indexation de l'information structurelle tout en citant les principales techniques citées dans la littérature.

Les exemples illustrant l'indexation des documents semi structurés porteront sur le document XML de la figure II.6 suivante **:**

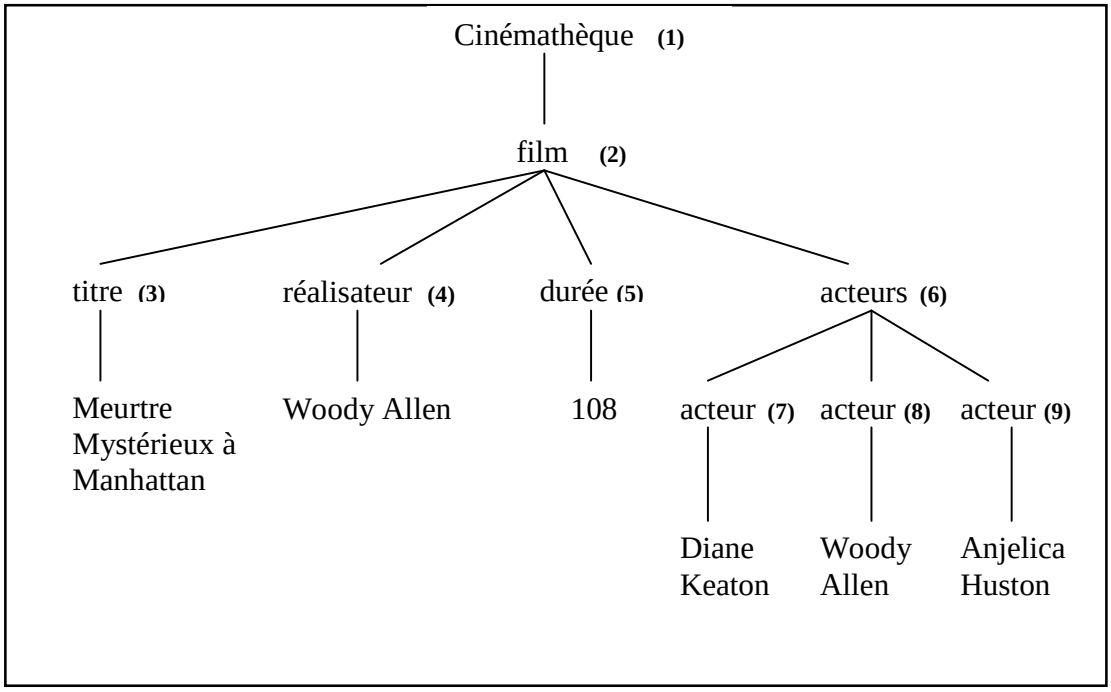


Figure II.6 *- Exemple de document XML représenté sous forme arborescente*

II.4.1 Indexation et pondération de l'information textuelle

II.4.1.1 Indexation de l'information textuelle

Le processus d'indexation, consiste comme dans le cas de la recherche d'information traditionnelle à extraire les termes importants des documents, mais pour la recherche d'information structurée, le problème de la portée des termes d'indexation se pose et la question est: comment rattacher les termes à l'information structurelle? Doit-on chercher à agréger le contenu des nœuds ou au contraire à indexer tous les contenus des nœuds séparément? Ces deux solutions correspondent aux approches d'indexation dites des sous arbres imbriqués et des unités disjointes.

II.4.1.1.1 Approche des sous arbres imbriqués

Dans cette approche les termes d'un nœud feuille sont rattachés à tous ses nœuds ancêtres dans le document [3, 79, 133]. Cependant, comme les documents XML possèdent une structure hiérarchique, les nœuds de l'index sont imbriqués les uns dans les autres et l'index contient de nombreuses informations redondantes.

II.4.1.1.2 Approche des unités disjointes

Dans cette approche le document XML est décomposé en unités disjointes, de manière que le texte de chaque nœud de l'index est l'union d'une ou plus de ces parties disjointes [8, 31, 54, 61, 80, 102, 120]. Les termes des nœuds feuilles sont uniquement reliés au nœud parent qui les contient.

Par exemple, dans l'approche des sous arbres imbriqués le terme 'Meurtre' du document XML de la figure II.6 est relié aux nœuds /cinémathèque/film/titre, /cinémathèque/film et /titre. Et dans l'approche des unités disjointes, le terme 'Meurtre' est uniquement relié au nœud /cinémathèque/film/titre.

II.4.1.2 Pondération des termes d'indexation

En recherche d'information traditionnelle, le poids d'un terme cherche à rendre compte de 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 qui le contient.

Les occurrences des termes ne suivent plus forcément la loi de Zipf [64, 162]. Le nombre de répétitions des termes peut être réduit dans les documents XML et l'utilisation d'idf (Inverse Document Frequency) n'est pas forcément appropriée [125]. Par conséquent, l'utilisation d'ief (Inverse Element Frequency) a été proposée par de nombreux auteurs [63, 150]. Nous trouverons des exemples d'adaptation des formules de pondération traditionnellement utilisées en recherche d'information à la recherche d'information structurée dans [145].

Dans [161], 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 tf-idf qu'on applique aux balises. Ainsi, les auteurs définissent le tf-itdf (Term Frequency- Inverse Tag and Document Frequency), qui permet de calculer la force discriminatoire d'un terme 't' pour une balise 'b' relative à un document 'd'. Le TF-ITDF est définit de la façon suivante:

Soient T l'ensemble de tous les termes du 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)

Avec :

$$\mathbf{ITF(t,d) = \log(|D|_b / DF(t,b))}$$

$$\mathbf{IDF(t,b) = \log(|B|_d / TagF(t,b))}$$

$|D|_b$: nombre de documents où le modèle de balise 'b' figure dans leur structure

$|B|_d$: nombre de balises dans le document 'd'

DF(t,b) : nombre de documents qui contiennent la balise 'b', dont laquelle le mot 't' figure au moins une fois

TagF(t,b) : nombre de balises dans le document 'd' et dans lesquelles le mot 't' figure au moins une fois.

II.4.2 Indexation de l'information structurelle

La structure peut être indexée selon des granularités variées [96] et donc ce n'est pas forcément toute l'information structurelle qui est utilisée dans le processus d'indexation. Parmi les approches proposées dans la littérature, nous distinguons trois types d'approches pour l'indexation de l'information structurelle: indexation basée sur des champs, indexation basée sur des chemins, et enfin indexation basée sur des arbres.

II.4.2.1 Indexation basée sur des champs

Dans cette approche, tout terme d'indexation est associé au nom de la balise dans laquelle il apparaît. Cette méthode a l'avantage de permettre de savoir dans quel contexte l'information textuelle est énoncée.

Par exemple, si nous appliquons cette méthode aux termes 'Meurtre', 'Allen' et '108' nous aurons les résultats figurant dans le tableau **II.2** qui illustre aussi la notion de propagation citée dans l'indexation de l'information textuelle:

Terme	***Sans propagation***	***Avec propagation***
Meurtre	Titre	cinémathèque, film, titre
Allen	réalisateur, acteur	cinémathèque, réalisateur, film, acteurs, acteur
108	Durée	cinémathèque, film, durée

Tableau II.2 - *Exemple d'indexation basée sur des champs*

II.4.2.2 Indexation basée sur des chemins

Il s'agit dans cette méthode d'indiquer le chemin d'accès logique pour tout terme d'indexation. Cette méthode permet de retrouver rapidement des documents ayant des valeurs connues pour certains éléments ou attributs et facilite la navigation de façon à résoudre efficacement des expressions XPath. Par conséquent, l'indexation produit des index de chemins, c'est-à-dire des index donnant pour chaque valeur répertoriée d'un chemin de balises (de type Xpath) la liste des documents répondants contenant un élément atteignable par ce chemin et ayant cette valeur.

Voici un exemple illustrant l'indexation basée sur des chemins :

Terme	***Indexation***
Meurtre	/cinémathèque/film/titre
Allen	/cinémathèque/film/réalisateur /cinémathèque/film/acteurs/acteur
108	/cinémathèque/film/durée

Tableau II.3 - Exemple d'indexation basée sur des chemins

II.4.2.3 Indexation basée sur des arbres

Un identifiant unique (UID) est associé à chaque nœud de l'arbre représentant le document XML (voir figure II.6) de sorte à pouvoir reconstruire la structure arborescente du document, pour la navigation et le traitement d'expression XPath. Les termes sont attribués à cet identifiant, ce qui permet de repérer précisément l'endroit où ces termes apparaissent et de retrouver les relations hiérarchiques entre les éléments.

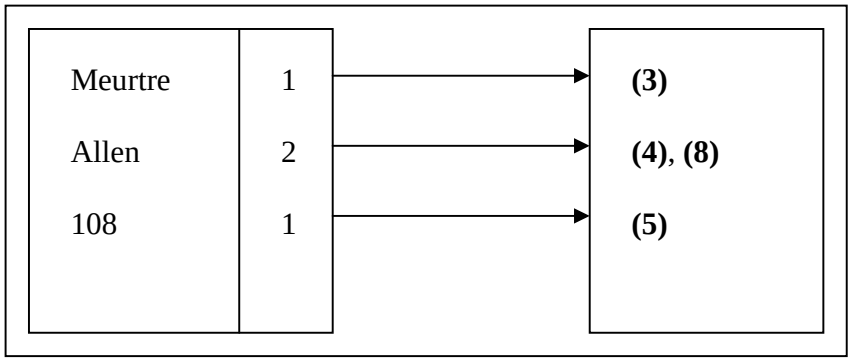


Figure II.7 *- Exemple d'indexation basée sur des arbres*

II.5 Langages de requêtes

Différents langages de requêtes sont proposés dans la littérature pour l'interrogation de documents XML. Ils offrent à l'utilisateur la possibilité d'exprimer des besoins diversifiés concernant le contenu et/ou la structure.

Dans [56] sont définies les fonctionnalités que doit supporter un langage de requête XML pour être à la fois un langage de requête pour les systèmes documentaires et pour les bases de données. Parmi ces fonctionnalités nous citons:

- Sélection des arbres sur critères multiples,
- Possibilité d'effectuer toutes les opérations des types de base,
- Tri des résultats,
- Imbrication de requêtes,
- Possibilité d'utilisation des agrégats et fonctions associées,
- Préservation de structures,
- Construction de structures nouvelles.

De plus, l'intégration de fonctions des systèmes documentaires nécessite la prise en compte de requêtes par liste de mots-clés du type : CONTAINS (<élément>, collection de mots clés).

Parmi les premiers langages spécifiés pour l'interrogation des documents semi structurés, nous pouvons citer LOREL [1], XML-QL [94] ou XQL [118]. Ce dernier a été développé par Microsoft et qui au lieu d'utiliser des canevas XML, il propose d'étendre les URL pour interroger des collections de documents XML avec des expressions Xpath. Le langage QUILT [38] a été

développé pour interroger des documents et des bases de données. XML-GL [36] est un langage de requêtes graphique s'utilisant sur des graphes XML. Il présente le principal avantage d'être très ergonomique. XQuery [155] est le langage de requêtes pour XML proposé par le W3C. XQuery tire très fortement ses constructions et caractéristiques du langage QUILT, lui-même dérivé de Xpath, XQL, XML-QL, et LOREL. XQuery peut être perçu comme un sur ensemble du langage SQL. Les fonctionnalités de SQL sur les tables (collections de tuples) sont étendues pour supporter des opérations similaires sur des collections d'arbres. Ces extensions ont conduit à intégrer les fonctions suivantes : projection d'arbres sur des sous-arbres, sélection d'arbres et de sous-arbres en utilisant des prédicats sur les valeurs des feuilles, combinaison des arbres extraits de collection en utilisant des jointures d'arbres, réordonnancement des arbres, imbrication des requêtes, calculs d'agrégats, etc. XQuery supporte des fonctions orientées RI : en particulier, un prédicat 'contains' est intégré pour la recherche par mots-clés [16, 17, 21]. Pour faciliter la recherche dans des structures mal connues, XQuery permet enfin d'exprimer des chemins indéterminés ou partiellement connus, tout comme Xpath. Nous trouverons dans le tableau suivant un exemple de requête XQuery.

```
FOR $p IN fn : distinct-values (fn :doc (« bibliotheque.xml »)// editeur)
        LET $a := fn :doc(« bibliothèque.xml »)//livre[editeur =$p]/prix
        WHERE fn : count ($a) >100
        RETURN <grandediteur>
                        <nom >$p/text() </ nom >,
                        < prixmoyen > fn : avg( $a )</ prixmoyen >
                </ grandediteur >
```

Tableau II.4 *- Exemple de requête XQuery : lister les noms des éditeurs qui ont publié plus de 100 livres et la moyenne des prix des livres édités pour chacun*

Une comparaison des langages de requêtes les plus importants, sur la base des principales caractéristiques est faite dans [125], elle est donnée dans le tableau suivant :

	LOREL	XML-QL	XQL	XML-GL	XQuery
Navigation	Oui	Oui	Oui	Oui	Oui
Support Xpath	Limité	Limité	Oui	/	Oui
Sélection	Oui	Oui	Oui	Oui	Oui
Jointure	Oui	Oui	Non	Oui	Oui
Tri	Oui	Oui	Non	Oui	Oui
Construction	Non	Oui	Non	Oui	Oui
Mot-clé	Non	Non	Non	Non	Oui
Fonction	Oui	Oui	Non	/	Oui
Imbrication	Oui	Oui	Oui	Oui	Oui
Agrégation	Oui	proposée	Non	/	Oui

Tableau II.5 - *Comparaison de différents langages de requêtes pour XML*

Comme nous pouvons le voir, de très nombreux langages de requêtes ont été proposés dans la littérature. La plupart sont très puissants, mais leur syntaxe, souvent dérivée de SQL, est difficilement accessible pour les novices. Des interfaces adaptées peuvent être associées, mais leur feraient perdre de leur puissance. De plus, ces langages proposent le prédicat 'contains' pour exprimer des conditions sur le contenu des éléments. Cependant, ce type de prédicat ne permet pas de mesurer la similarité entre les unités d'informations et les conditions de contenu (requête). Pour intégrer ce dernier aspect et permettre une recherche orientée sur la pertinence (c'est-à-dire avec une pondération des résultats), des langages issues de la communauté recherche d'information ont apparu tel que le langage NEXI qui a été défini dans [143, 144] pour répondre aux besoins de la campagne d'évaluation INEX, le langage du modèle XFIRM [125] et enfin le W3C a proposé un Working Draft [156], qui a pour but d'étendre les caractéristiques de recherche de XQuery à la recherche plein texte. Le langage TexQuery [7] en est une application.

II.6 Modèles de recherche d'information structurée

Les systèmes de recherche d'information traditionnels (présentés dans le chapitre **I** section **7**) ne prennent pas en considération l'information structurelle. Cette dernière apporte une information supplémentaire à la définition du besoin de l'utilisateur, elle est accessible directement à travers les nouveaux formats de données. Donc, il est indispensable d'adapter les systèmes de recherche traditionnels principalement pour les deux raisons suivantes :

1- Les systèmes de recherche d'information traditionnels sont conçus pour des documents linéaires plats et donc ne prennent pas en compte l'information structurelle. De plus, les

corpus de documents sont composés de documents hétérogènes aussi bien dans leurs formes que dans leurs contenus. Paradoxalement, les systèmes de recherche traditionnels ont été conçus surtout pour des documents homogènes [42].

2- L'apparition des documents semi structurés a entraîné l'apparition de nouvelles problématiques spécifiques de recherche d'information comme l'unité d'information pertinente (voir section II.3). La structure est une nouvelle source d'information non négligeable, dans [130] il est même affirmé que «*Ignorer la structure du document revient à ignorer sa sémantique* », elle facilite la représentation des documents, ainsi que leur interprétation et leur exploitation, afin d'apporter des réponses aux nouveaux besoins en information. Par exemple, l'utilisateur peut choisir la granularité d'information à retourner en utilisant des contraintes structurelles sur les éléments à restituer.

Les différentes approches développées tendent d'une part de séparer la structure d'un document de son apparence et d'autre part de lier la structure logique et le contenu. L'accès simultané à la structure des documents et à leur contenu permet d'envisager de nouveaux modes d'exploitation de l'information textuelle. Celle-ci nécessite la création de nouveaux outils et de nouveaux modèles capables d'exploiter de telles données [106].

Nous présentons dans la suite les principales extensions faites sur les modèles de RI traditionnels pour les adapter à la recherche dans les documents semi structurés ainsi que d'autres approches spécifiques pour le traitement des conditions de structure.

II.6.1 Modèle booléen étendu

Les auteurs dans [68] se sont basés sur des approches portant sur l'extension des modèles booléens aux documents structurés grâce aux p-normes. Leur travail se basant essentiellement sur la définition de l'unité d'information renvoyée à l'utilisateur, ne considère que les requêtes orientées contenu. L'unité d'information est définie comme étant tout élément contenant au moins un élément feuille. Pour calculer le score d'un élément vis-à-vis d'une requête, les auteurs propagent des scores, depuis les feuilles de l'arbre documentaire jusqu'à l'élément considéré. Pour ce faire, chaque élément feuille e_j (j=1, …, N) est représenté par un vecteur:

$$F\ (e_j) = (w_{t_1}^{e_j}, w_{t_2}^{e_j}, ..., w_{t_n}^{e_j})$$

Où n représente le nombre de termes indexés dans la collection et $w_{t_i}^{e_j}$ représente le poids d'un terme t_i dans un élément feuille qui est donné par:

$$w_{t_i}^{e_j} = \frac{tf\ (t_i, e_j)}{\sum_{j=1}^{N} \sum_{k=1}^{n} tf\ (t_k, e_j)} \cdot \log \frac{N_d}{df\ (t_i)}$$

Où $tf(t_k, e_j)$ est la fréquence du terme t_k dans l'élément e_j, N_d est le nombre de documents dans la collection et $df(t_i)$ est le nombre de documents contenant le terme t_i.

La similarité entre un élément feuille e_j (représenté par le vecteur $F(e_j)$) et une requête q (représentée par un vecteur dont les composantes sont 1 si un terme de la collection apparaît dans la requête, 0 sinon) est calculée comme suit:

$$sim\ (F\ (e_j), q) = \frac{F\ (e_j).q}{|F\ (e_j)|.|q|}$$

Ils utilisent ensuite la p-norme pour calculer le score final d'un élément. Ce score est calculé de manière récursive car il prend en compte le score des sous éléments :

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

Où p est généralement choisi avec une valeur égale à 2, enf(e) représente les sous éléments de e. Lorsqu'un élément e est une feuille :

$$RSV\ (q,e) = sim\ (F\ (e_j), q)$$

II.6.2 Modèle vectoriel étendu

Dans cette approche, les éléments sont représentés par vecteurs de termes pondérés. Généralement, les approches développées utilisant ce modèle utilisent l'indexation basée sur des arbres, elles propagent les termes des nœuds feuilles dans l'arbre du document [125].

Il se base sur le calcul de la similarité de chaque élément à la requête, puis les éléments sont renvoyés selon leur ordre de similarité.

Dans [51] la mesure suivante permet de calculer la similarité entre un nœud n et une requête $q = \{q_1, q_2, .., q_T\}$:

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

Où α (T) est le facteur permettant de prendre en compte le type de nœud, s le nombre d'enfants n_k de n, β le paramètre permettant d'assurer que le nombre d'enfants n'introduit pas un biais dans la formule.

Et la fonction cosm(q, n) est définie comme suit :

$$\text{cosm}\ \ (q, n) = \sum_{i=1}^{T} \frac{w_i^{\,q} \cdot w_i^{\,n}}{|n|}$$

Où $w_i^{\,q} et\ w_i^{\,n}$ les poids du terme t_i respectivement dans la requête q et dans le nœud n, |n| est le nombre de termes dans le nœud n.

Le modèle vectoriel est adopté dans plusieurs approches, nous citons Zargayouna [161], qui définit un modèle d'indexation de documents XML implémentant le modèle vectoriel, dans lequel le document n'est pas considéré comme un vecteur, mais comme une matrice de termes et de modèles de balise.

Les auteurs dans [35] proposent le modèle JuruXML qui est une adoption du modèle vectoriel. Ce modèle indexe les éléments selon leur type (un index par type d'élément) et applique ensuite le modèle vectoriel pour la pondération des éléments. Les requêtes orientées contenu sont évaluées sur chacun des index, et les résultats (normalisés) sont ensuite fusionnés afin de fournir à l'utilisateur une liste unique de résultats. Une requête structurée est quant à elle évaluée en trois

phases. Tout d'abord, la requête originale est décomposée en un ensemble de conditions de la forme (*chemin, terme*). Ensuite, une correspondance vague entre les chemins est calculée par la fonction suivante :

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

Où c_i^q est la condition de chemin pour un terme t_i de la requête q et c_i^e le chemin XPath de ce terme dans l'élément e.

Enfin, nous avons:

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

Où $w_q(t)$ et $w_e(t)$ sont les poids du terme t respectivement dans la requête q et l'élément e, et |q| et |e| sont les nombres de termes respectivement dans q et e.

La fonction de similarité entre deux chemins *cr* permet de favoriser les termes qui apparaissent dans des endroits proches (au sens de la structure), par exemple :

cr (/book/title, /book/title) = 1,

et cr (/book/title, /book/sections/title) = 0,5

II.6.3 Modèles probabilistes

Les auteurs dans [79] proposent une approche basée sur les modèles de langage pour traiter les requêtes orientées contenu. Les auteurs considèrent que comme n'importe quel élément XML peut potentiellement être renvoyé à l'utilisateur, chaque élément doit être traité comme une unité d'indexation à part entière. Par conséquent, pour chaque élément, le texte qu'il contient ainsi que le texte contenu dans ses descendants est indexé. Un modèle de langage est ensuite estimé pour chaque élément de la collection. Pour une requête donnée, les éléments sont triés par rapport à la probabilité que le modèle de langage de l'élément génère la requête. Ceci revient à estimer la probabilité P(e,q), où e est un élément et q une requête :

$$P(e, q) = P(e) . P(q/e)$$

Deux probabilités doivent donc être estimées : la probabilité a priori de l'élément P(e) et la probabilité qu'il génère la requête P(q/e). Pour la seconde probabilité, les auteurs considèrent que les termes de la requête sont indépendants et utilisent une interpolation linéaire du modèle d'élément et du modèle de collection pour estimer la probabilité d'un terme de la requête. La probabilité d'une requête q, composée des termes t_1, t_2,...t_n, est calculée de la façon suivante :

$$P(q/e) = P(t_1 ,\dots, t_n/e) = \prod_{i=1}^{n}(\lambda.p(t_i/e) + (1-\lambda)p(t_i))$$

Où $P(t_i/e)$ est la probabilité d'observer le terme t_i dans l'élément e, $P(t_i)$ est la probabilité d'observer le terme t_i dans la collection et λ paramètre de lissage.

Le calcul des probabilités peut être réduit à la formule de calcul des scores ci-dessous, pour un élément e et une requête q composée des termes t_1, t_2,...t_n.

$$s(e, t_1, \dots t_n) = \beta \cdot \log\Big(\sum_t tf(t, e)\Big) + \sum_{i=1}^{n} \log\Big(1 + \frac{\lambda \cdot tf(t_i, e) \cdot (\sum_t df(t))}{(1-\lambda)\, df(t_i) \cdot (\sum_t tf(t, e))}\Big)$$

Où tf(t, e) est la fréquence du terme t dans l'élément e, df(t) est le nombre d'éléments contenant t, λ est le poids donné au modèle de langage de l'élément en lissant avec le modèle de la collection, et β est un paramètre servant à combler le fossé entre la taille de l'élément moyen et la taille de l'élément moyen pertinent.

Dans [150], l'utilisation de la fréquence inverse d'élément ief est proposée pour faciliter les pondérations par élément : un nouveau poids probabiliste pour les termes est alors formulé en utilisant ief et freq(t,e) la fréquence du terme t dans l'élément e. Les poids des termes de la requête peuvent être étendus avec des conditions sur l'appartenance du terme à un certain élément ou chemin.

L'utilisation des réseaux bayésiens (formalisme probabiliste de représentation graphique des dépendances conditionnelles entre des variables aléatoires) [37, 76] à la recherche d'information structurée se trouve dans [106, 107, 108]. Dans ce modèle, la structure de réseau bayésien» utilisée reflète directement la hiérarchie des documents, c'est-à-dire que les auteurs considèrent que chaque élément de la hiérarchie possède une variable aléatoire associée. La variable aléatoire associée à un élément structurel peut prendre trois valeurs différentes dans l'ensemble V = {N,G,E}, avec N indiquant que l'élément n'est pas pertinent, G que l'élément est peu spécifique et E que l'élément possède une forte spécificité.

Dans ce modèle, pour chaque élément e et pour une requête donnée q, 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.

Deux autres types de variables aléatoires sont considérés :

- Une variable aléatoire est associée à chaque requête, représentée sous forme de vecteur de fréquences de termes.
- Une variable est associée à chaque modèle de pertinence utilisé pour évaluer la similarité locale d'un élément à une requête. Elle prend deux valeurs : pertinent ou non pertinent. Un score de pertinence locale d'un élément à une requête, qui ne dépend que de l'élément et de la requête, peut être calculé en utilisant plusieurs modèles : la fréquence des termes de la requête dans la requête, dans l'élément, dans le parent de l'élément, la longueur de l'élément,…

La probabilité qu'un élément soit dans l'état N, G ou E dépend ensuite de l'état de l'élément parent, et du jugement par le(s) modèle(s) de pondération utilisé(s) que l'élément est pertinent ou non pertinent, comme le montre la figure II.8 suivante :

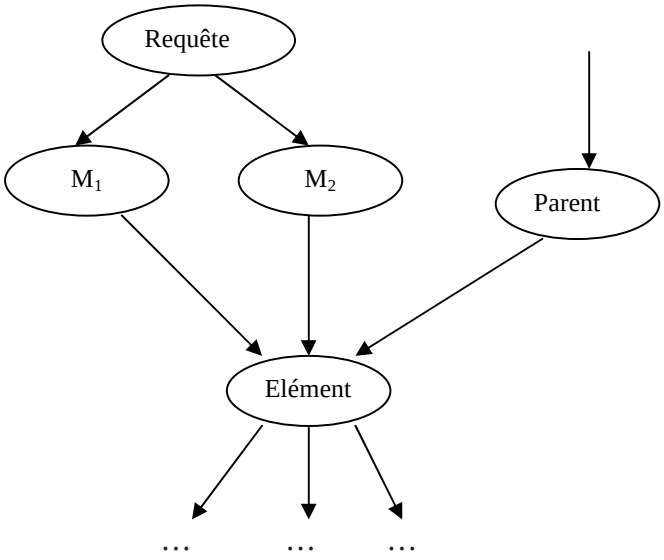


***Figure II.8** - Modèle de réseau bayésien. L'état de l'élément dépend de l'état du parent et de la pertinence de l'élément pour les modèles M_1 et M_2*

On a alors (si on considère deux modèles de base M_1 et M_2 pour le calcul du score local d'un élément e) :

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

Où v ϵ V, q est une requête composée de simples termes et θ un paramètre obtenu par apprentissage. Il dépend des différents états des quatre variables aléatoires (état de l'élément, état du parent, pertinence des modèles de base M_1 et M_2), et de la catégorie c(e) de l'élément.

Les scores de pertinence sont calculés récursivement dans le réseau bayésien en commençant par la racine des documents. Le modèle est étendu au traitement des requêtes orientées contenu et structure dans [148].

II.6.3.1 Modèle d'inférence probabiliste

Une approche d'extension du modèle probabiliste inférentiel aux documents semi structurés consiste en la prise en compte de l'information structurelle dans les probabilités conditionnelles de jointure, par exemple : P(d/t) devient P(d/p contains t) où d représente un document ou une partie du document, t est un terme et p un chemin dans l'arbre structurel de d. Une méthode

d'augmentation basée sur le modèle probabiliste est proposée dans [54, 61]. Elle est basée sur le langage de requêtes XIRQL, et a été implémentée au sein du moteur de recherche HyRex. L'approche adopte une indexation par unités disjointes (sauf pour les nœuds feuilles d'une granularité trop fine, dont les termes sont propagés au nœud indexable le plus proche dans la hiérarchie). De plus pour préserver des unités disjointes, on ne peut associer à un nœud que les termes non reliés à ses descendants. Les calculs de pertinence se font comme suit :

- Dans le cas des requêtes orientées contenu, le poids de pertinence d'un nœud est calculé par propagation des poids des termes les plus spécifiques dans l'arbre du document. Les poids sont de plus diminués par multiplication par un facteur d'augmentation.

- Dans le cas des requêtes orientées contenu et structure, on calcule d'abord les probabilités d'apparition de chaque terme de la condition de contenu dans les éléments répondants aux conditions de structure. Puis, des sommes pondérées de ces probabilités sont effectuées.

Considérons l'exemple de la structure de document de la figure II.9 [54], contenant un certain nombre de termes pondérés (par leur probabilité d'apparition dans l'élément), et la requête orientée contenu «XML».

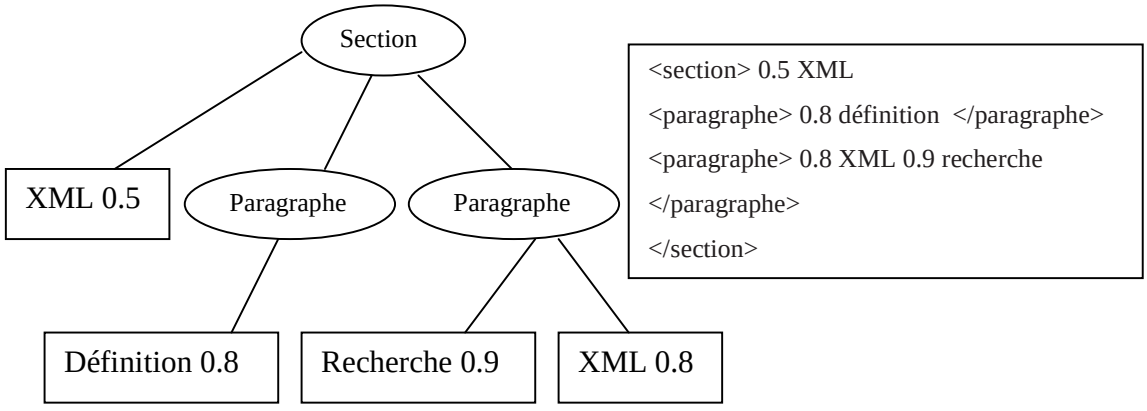


Figure II.9 *- Modèle d'augmentation*

Le poids de pertinence de l'élément section est calculé comme suit, en utilisant un facteur d'augmentation égal à 0.7 :

P([section, XML])+ P([paragraphe[2]]). P([paragraphe[2], XML)

- P([section, XML]).P([paragraphe[2]]).P([paragraphe[2], XML)

= 0.5 + 0.7*0.8 - 0.5*0.7*0.8=0.68

Le nœud paragraphe (ayant une pertinence de 0.8 par rapport à la requête) sera donc mieux classé que le nœud section.

II.6.4 Modèle XFIRM

Le modèle XFIRM [125] est basé sur un modèle de données générique permettant l'implémentation de nombreux modèles de RI et le traitement de collections hétérogènes (c'est-à-dire contenant des documents ne suivant pas la même DTD).

Le traitement des requêtes orientées contenu est effectué en deux étapes: une première étape consiste à évaluer la similarité des nœuds feuilles de l'index à la requête (on parle alors de calcul des poids des nœuds feuilles) et une seconde étape consiste à rechercher les sous arbres pertinents. La pertinence des sous arbres est évaluée en effectuant la propagation des poids des feuilles dans l'arbre du document.

II.6.4.1 Calcul des poids des nœuds feuilles

Les poids (scores), des nœuds feuilles identifiés dans l'arbre du document, sont calculés grâce à la fonction de similarité RSV (Retrieval Status Value).

Si une requête q est composée de termes et des poids associés, nous avons:

$$RSV(q, nf) = \sum_{i=1}^{T} w_i^q \cdot w_i^{nf}, \quad avec \quad w_i^q = tf_i^q \quad et \quad w_i^{nf} = tf_i^{nf} \cdot ief_i \cdot idf_i \qquad \text{(II.1)}$$

Où $w_i^q \, et \, w_i^{nf}$ sont respectivement le poids du terme i dans la requête q et dans le nœud feuille nf, $tf_i^q \, et \, tf_i^{nf}$ sont respectivement la fréquence du terme i dans la requête q et dans le nœud feuille nf, idf_i = log(|D|/|d_i|) permet d'évaluer l'importance du terme i dans la collection de documents, où |D| est le nombre total de documents de la collection et |d_i| est le nombre de documents contenant le terme i, et ief_i = log(|F_c|/|nf_i|) permet d'évaluer l'importance du terme i dans la

collection de nœuds feuilles, où $|F_c|$ est le nombre total de nœuds feuilles de la collection, et $|nf_i|$ est le nombre de nœuds feuilles contenant le terme i.

II.6.4.2 Propagation de la pertinence des nœuds feuilles

Une fois les poids des nœuds feuilles calculés, une valeur de pertinence est calculée pour chaque nœud de l'arbre de document en utilisant les poids des nœuds feuilles qu'il contient. Les termes apparaissant près de la racine d'un sous arbre paraissent plus porteurs d'information pour le nœud associé que ceux situés plus bas dans le sous arbre. Il semble ainsi intuitif que plus grande est la distance entre un nœud et son ancêtre, moins il contribue à sa pertinence. Cette intuition est modélisée par l'utilisation dans la fonction de propagation du paramètre dist(n, nf_k), qui représente la distance entre le nœud n et un de ses nœuds feuille nf_k dans l'arbre du document, c'est-à-dire le nombre d'arcs séparant les deux nœuds. Il paraît aussi intuitif que plus un nœud possède de nœuds feuilles pertinents, plus il est pertinent. Le paramètre $\left|F_n^p\right|$, qui est le nombre de nœuds feuilles descendants du nœud n ayant un score non nul est alors introduit dans la fonction de propagation. Une première évaluation de la pertinence p_n d'un nœud peut être calculée selon la formule suivante:

$$p_n = \left|F_n^p\right| \cdot \sum_{nf_k \in F_n} \alpha^{dis(n,\, nf_k)-1} \cdot (RSV_m(q, nf_k)) \qquad \text{(II.2)}$$

Où F_n est l'ensemble des nœuds feuilles nf_k descendants du nœud n, et $\alpha \in]0..1]$ un paramètre permettant de quantifier l'importance de la distance séparant les nœuds dans la formule de propagation.

Deux propositions sont également faites pour évaluer de manière plus juste l'informativité des nœuds résultats :

– l'utilisation du paramètre β pour augmenter l'importance des nœuds de petite taille durant la propagation (nœuds supposés être utilisés pour faire ressortir des informations importantes, comme par exemple les nœuds titres qui permettent de situer avec précision le sujet de leur nœud ancêtre). Le paramètre β est calculé comme suit :

$$\beta(nf_k) = \begin{cases} l_k / \Delta l & si \quad dist(n, nf_k) = 1 \quad et \quad l_k < \Delta l \\ \log(\Delta l / l_k) & si \quad dist(n, nf_k) > 1 \quad et \quad l_k < \Delta l \\ 1 & sinon \end{cases}$$

Avec Δl la taille moyenne d'un nœud feuille de la collection;

– l'utilisation du contexte des éléments (la pertinence du document qui les contient): un nœud appartenant à un document fortement pertinent doit être mieux classé qu'un nœud se trouvant dans un document de pertinence moindre.

La pertinence p_n d'un nœud n est alors définie de la façon suivante:

$$p_n = \rho \cdot \left|F_n^p\right| \cdot \sum_{nf_k \in F_n} \alpha^{dis(n,\, nf_k)-1} \cdot \beta(nf_k) \cdot (RSV_m(q, nf_k)) + (1-\rho) \cdot p_{racine}$$

Avec F_n et F respectivement l'ensemble des nœuds feuilles nf_k descendants du nœud n et l'ensemble des nœuds feuilles nf_k du document, $\left|F_n^p\right|$ et $\left|F^p\right|$ respectivement le nombre de nœuds feuilles descendant du nœud n ou du document et ayant un score non nul, et ρ ϵ [0..1] est un paramètre servant de pivot et permettant d'ajuster l'importance de la pertinence du nœud racine lors de la rétropropagation.

Les nœuds sont ensuite renvoyés à l'utilisateur par ordre décroissant de pertinence à la requête.

Les expérimentations menées sur le modèle au sein de la campagne d'évaluation INEX montrent ses bonnes performances par rapport à l'ensemble des participants [105, 126, 128].

II.6.5 Autres Approches

La plupart des modèles proposés dans la littérature ne proposent pas d'approches réellement orientées RI pour le traitement des conditions de structure des requêtes. Certaines approches proposent par exemple d'effectuer un simple filtre sur les résultats des conditions de contenu [133, 142]. D'autres approches quant à elle, cherchent à évaluer la pertinence des conditions structurelles. Une description de quelques approches est présentée dans le paragraphe suivant.

Les auteurs dans [32] proposent le langage FXpath (Fuzzy XPath), possédant les caractéristiques suivantes:

– une correspondance d'arbres floue, ce qui permet de renvoyer à l'utilisateur une liste triée d'éléments et non un ensemble non ordonné comme le fait XPath ;

– des prédicats flous, permettant à l'utilisateur de spécifier des conditions de sélection imprécises et approximatives (prédicat NEAR et prédicat CLOSE);

– une quantification floue, permettant la spécification d'opérateurs linguistiques comme opérateurs d'agrégation (tout, au moins un, la plupart, etc.).

D'autres approches cherchent elles aussi à effectuer la correspondance entre l'arbre du document et l'arbre de la requête [98, 130, 158].

YOO dans [158] définit la notion de proximité à l'aide de distance. Dans des documents structurés, la distance peut être définie en terme de nombres de mots entre des termes de nœuds feuilles ou en termes de nœuds entre les nœuds. La distance des nœuds peut être quantifiée grâce à la distance horizontale (nombre de nœuds du même niveau entre les nœuds) et à la distance verticale (nombre d'unités logiques qui peuvent être groupées pour aller d'un nœud à un autre).

Dans [128], le modèle XFIRM propose d'effectuer une correspondance vague entre l'arbre de la requête et l'arbre des documents. L'évaluation des requêtes est effectuée de la façon suivante :

1) les requêtes sont décomposées en sous requêtes élémentaires du type tg[q], où tg est une contrainte structurelle (un nom de balise) et q est une requête composée de simples mots-clés ;

2) une valeur de pertinence est calculée entre les nœuds feuilles et les conditions de contenu q des sous requêtes élémentaires, et ce en utilisant la fonction de similarité RSV (q, nf);

3) les valeurs de pertinences sont propagées dans l'arbre du document pour répondre aux conditions de structures (équation II.2);

4) les requêtes originales sont évaluées grâce à une propagation vers le haut et vers le bas des poids de pertinence des nœuds répondants aux sous requêtes élémentaires.

L'évaluation du système dans le cadre d'INEX a montré le grand intérêt de la proposition [127].

II.7 Evaluation des Systèmes de Recherche d'Information Structurée

L'évaluation d'un système de recherche se fait dans une collection de tests. Nous citons la seule compagnie, pour ce jour, qui permet de tester les systèmes de recherche d'information structurée : **INEX** (INitiative for the Evaluation of Xml retrieval), elle a été créée en 2002, son objectif est de permettre une évaluation et une comparaison aussi rigoureuse que possible des systèmes de recherche d'information structurée dans des collections XML orientées documents. Elle fournit un ensemble de documents, des requêtes proposées par les participants (experts dans le domaine) et des jugements de pertinence.

II.7.1 Collection INEX

Jusqu'en 2005, la collection INEX était composée d'articles scientifiques provenant de la IEEE Computer Society, balisés au format XML. La collection, d'environ 700 Mo, a atteint plus de 17 000 articles, publiés de 1995 à 2004, et provenant de 18 magazines ou revues différents. Un article moyen est composé d'environ 1500 éléments.

En 2006, la collection s'est renouvelée d'environ 4,5 Go, elle est composée de plus de 650 000 articles de l'encyclopédie Wikipedia [44].

II.7.2 Requêtes

Les requêtes (Topics) sont créées par les différents participants et doivent être représentatives des demandes de l'utilisateur moyen sur la collection. On distingue deux principaux types de requêtes :

- Les CO (Content Only) : ces requêtes sont exprimées en langage naturel. Les mots-clés de la requête peuvent être groupés sous forme d'expressions et précédés par les opérateurs '+' (signifiant que le terme est obligatoire) ou '-' (signifiant que le terme ne doit pas apparaître dans les éléments renvoyés à l'utilisateur).
- Les CAS (Content And Structure) : ces requêtes contiennent de plus des contraintes sur la structure des documents.

La figure II.10 suivante, donne un exemple de la composition d'une requête.

<**inex-topic** topic-id = "numéro-requête" query type = "type-requête">
<**title**> "donne la définition formelle de la requête" </**title**>
<**description**> "indiquent brièvement les intentions de l'auteur en langage naturel" </**description**>
<**narrative**>"indiquent d'une façon narrative les intentions de l'auteur en langage naturel " </**narrative**>
<**keywords**> "contient un ensemble de mots-clés qui ont permis l'exploration du corpus avant la formulation définitive de la requête " </**keywords**>
</**inex-topic**>

Figure II.10 *- Syntaxe d'une requête INEX*

Depuis 2006, les deux formes de requêtes (CO et CAS) sont regroupées et comportent deux formulations différentes :

– le titre (champ <title>) de la requête exprime le besoin en information de l'utilisateur sous forme de simples mots-clés ;

– le titre structuré (champ <castitle>) de la requête exprime le même besoin en information de l'utilisateur mais en y ajoutant des précisions sur la structure des documents et/ou des unités d'information recherchées.

– Les champs Description et Narrative, explicités en langage naturel, indiquent les intentions de l'auteur [134].

II.7.3 Tâches

La tâche principale d'INEX est la tâche de recherche ad-hoc, elle est considérée comme une simulation de l'utilisation d'une bibliothèque, où un ensemble statique de documents est interrogé avec des besoins utilisateurs, c'est à dire des requêtes. Les requêtes peuvent contenir à la fois des conditions structurelles ou de contenu, et en réponse à une requête, des éléments (et non forcément des documents) peuvent être retrouvés à partir de la bibliothèque. La tâche ad-hoc se divise en trois sous tâches:

- Tâche CO : La tâche CO (Content Only Task) a pour but de répondre avec des éléments/documents XML à des requêtes utilisateur orientées contenu (requête CO), c'est à dire des requêtes contenant des mots-clés simples. Aucune indication de structure dans la requête ne peut aider les SRI à déterminer la granularité de l'information à renvoyer.
Un exemple de requête CO est présenté dans la figure suivante :

<**inex topic** topic id="98" query type="CO">
<**title**> "Information Exchange" +"XML" "Information Integration"</**title**>
<**description**> How to use XML to solve the information exchange (information integration) problem, especially in heterogeneous data sources ? </**description**>
<**narrative**> Relevant documents/components must talk about techniques of using XML to solve information exchange (information integration) among heterogeneous data sources where the structures of participating data sources are different although they might use the same ontologies about the same content. </**narrative**>
<**keywords**> Information exchange, XML, information integration, heterogeneous data sources </**keywords**>
</**inex topic**>

Figure II.11 *- Exemple de requête CO, issue du jeu de test 2003*

- Tâche SCAS : La tâche SCAS (Strict Content And Structure Task) consiste à répondre avec des éléments/documents XML aux requêtes CAS de manière stricte, c'est à dire en respectant toutes les conditions sur la structure et le contenu énoncées dans les requêtes. Le champ Title des requêtes de la tâche SCAS est basé sur une syntaxe XPath.
Un exemple de requête CAS pour la tâche SCAS est présenté dans la figure suivante:

<**inex topic** topic id="64" query type="CAS">
<**title**> //article[about(./,'hollerith')] // sec[about(./, 'DEHOMAG')] </**title**>
<**description**> In articles discussing Herman Hollerith find sections that mention DEHOMAG </**description**>
<**narrative**> Relevant sections deal with DEHOMAG (Deutsche Hollerith Maschinen

Gesellschaft) in documents that discuss work or life of Herman Hollerith </**narrative**>
<**keywords**> Hollerith, DEHOMAG, Deutsche Hollerith Maschinen Gesellschaft </**keywords**>
</**inex topic**>

Figure II.12 *- Exemple de requête CAS, issue du jeu de test 2003*

- Tâche VCAS : La tâche VCAS (Vague Content And Structure Task), utilise elle aussi des requêtes CAS, mais pour lesquelles les participants peuvent répondre de manière vague, c'est à dire avec des éléments/documents qui satisfont globalement les requêtes. Le champ "Title" des requêtes de la tâche SCAS est basé sur le langage de requêtes NEXI [143, 144], l'extension de XPath utilisée en 2003 pour les requêtes CAS étant considérée comme trop complexe : 63% des requêtes exprimées par les participants (experts en RI) contenaient des erreurs de syntaxe.
Un exemple de requête CAS pour la tâche VCAS est présenté dans la figure suivante :

<**inex topic** topic id="149" query type="CAS">
<**title**> //article[about(.//(abs|kwd),"genetic algorithm")]// bdy//sec[about(., simulated annealing)] </**title**>
<**description**>Find sections about simulated annealing in article that mention genetic algorithms. </**description**>
<**narrative**> I have come across the Constrained Shortest Path problem in connection with a route planing program. I have become aware of a technique called Simulated Annealing known from combinatorial optimization for heuristic solutions to NP-hard problems that I wish to use in the route plaing program. I have noticed a tendency for authors that mention SA in combination with Genetic Algorithms so I expect the keyword 'genetic' to appear in relevant articles. For the section to be relevant it has to discuss usage of Simulating Annealing or refer to results relevant to the techique.</**narrative**>
<**keywords**> genetic, simulated annealing, optimization </**keywords**>
</**inex topic**>

Figure II.13 *- Exemple de requête CAS, issue du jeu de test 2004*

Depuis 2004, quatre nouvelles tâches ont été proposées aux participants :

- La tâche de "relevance feedback" : a pour but d'expérimenter l'utilisation du contenu et de la structure comme informations de base pour la formulation d'une nouvelle requête;

- La tâche de "langage naturel" : dans laquelle les utilisateurs formulent leurs requêtes en langage naturel, et donc sans avoir besoin d'apprendre un langage complexe;

- La tâche "interactive" : a pour but d'étudier le comportement des utilisateurs face à des corpus XML et donc de cerner au mieux leurs besoins;

- La tâche "hétérogène" : propose aux participants de nouvelles collections, afin de développer des approches indépendantes des DTDs.

- **La tâche multimédia**, dédiée pour le moment à la recherche de contenus images au sein de documents XML.

II.7.4 Jugements de pertinence

L'évaluation de la pertinence des SRI passe par une première phase de validation des documents renvoyés par les SRI. Chaque élément/document est jugé à la main (par les participants) pour chaque requête, en utilisant le système de jugement en ligne [112]. Les participants surlignent dans les documents les parties qui ne contiennent que de l'information pertinente (indépendamment de la structure des documents). Le jugement de pertinence (niveaux d'exhaustivité et de spécificité) de chaque élément est ensuite déduit par le système.

Le jugement de pertinence des éléments, restitués par un système de recherche, vis-à-vis des requêtes est établi par les deux dimensions suivantes : *l'exhaustivité et la spécificité*.

- **L'exhaustivité :** décrit à quel point l'élément traite le sujet de la requête, elle est évaluée par un juge humain sur une échelle à quatre niveaux:
  - Non exhaustif: aucune correspondance entre la requête et l'élément.
  - Marginalement exhaustif : uniquement certains aspects du sujet de la requête sont traités par l'élément.
  - Assez exhaustif : les aspects du sujet de la requête sont en grande partie traités par l'élément.
  - Très exhaustif : tous les aspects du sujet de la requête sont traités par l'élément.
- **La spécificité :** décrit à quel point l'élément se focalise sur le sujet de la requête, elle est considérée depuis la compagne INEX 2005 comme une dimension continue représentée par une

valeur réelle dans l'intervalle [0,1]. Elle peut être calculée comme le rapport entre la longueur du texte pertinent et la longueur totale du texte de l'élément restitué en réponse à la requête [140].

Ces deux dimensions de la pertinence reflètent la pertinence d'un élément par rapport à ses descendants. En effet, un élément peut être plus exhaustif que chacun de ses descendants pris séparément. De même, des éléments peuvent être plus spécifiques que leurs parents.
Le degré de pertinence d'un élément/document renvoyé par les systèmes de RI entrés en campagne pour chaque requête, est jugé (à la main) par les participants en utilisant un système de jugement en ligne [109, 110]. Le degré de pertinence est donné par la paire $(e, s)_{\in E.S}$, avec E.S = {(0, 0), (1, 1), (1, 2), (1, 3), (2, 1), (2, 2), (2, 3), (3, 1), (3, 2), (3, 3)} est l'ensemble des 10 valeurs possibles sur l'échelle combinant les deux dimensions d'exhaustivité et de spécificité (les autres combinaisons ne sont pas possibles, par exemple: un élément qui n'est pat exhaustif, ne peut être spécifique, et inversement).

II.7.5 Mesures d'évaluation

L'évaluation de la performance des systèmes de recherche d'information structurée a commencé par l'utilisation des méthodes basées sur les mesures de rappel et précision, en cherchant à prendre en compte la structure des documents XML [53, 55]. Ces mesures présentent cependant un inconvénient majeur: elles ne prennent pas en compte l'imbrication (overlap) des éléments, inhérente aux documents XML, et évaluent le retour d'un élément pertinent sans prendre en compte le fait qu'il soit peut-être déjà vu entièrement ou en partie par l'utilisateur. Par exemple, un système A renvoyant un élément pertinent et aussi un de ses éléments fils pertinents obtient les mêmes performances qu'un système B renvoyant deux éléments pertinents non imbriqués.
En 2003, de nouvelles mesures ont été fournies pour essayer de résoudre ce problème. Nous citons par exemple les mesures proposées dans [21], qui incorporent la taille des éléments et le concept d'imbrication dans les mesures de rappel et précision comme donné dans les équations ci-dessous :

$$rappel_{o} = \frac{\sum_{i=1}^{k} e(c_i) \cdot \frac{\left|c_i'\right|}{\left|c_i\right|}}{\sum_{i=1}^{N} e(c_i)}$$

$$precision_{o} = \frac{\sum_{i=1}^{k} s(c_i) \cdot \left|c_i'\right|}{\sum_{i=1}^{k} \left|c_i'\right|}$$

Où les éléments c_1, ..., c_k dans les équations forment une liste triée de résultats, N est le nombre total d'éléments dans la collection, $e(c_i)$, $s(c_i)$ et $|c_i|$ sont respectivement les valeurs d'exhaustivité, de spécificité et la taille de l'élément c_i, $\left|c_i'\right|$ est la taille d'un élément c' qui n'a pas été précédemment vu par l'utilisateur.

Au lieu de mesurer le rappel et la précision après qu'un certain nombre d'éléments ait été retrouvé, la taille totale de l'élément retrouvé est utilisée comme paramètre de base, alors que l'imbrication est traitée en ne considérant que les parties de l'élément qui n'ont pas déjà été vues (on considère alors que l'information pertinente est répartie uniformément au sein d'un élément).

Cependant, les mesures décrites ci-dessus ne prennent pas en compte un problème essentiel de l'évaluation : la surpopulation de la base de rappel [81]. Cette surpopulation est due aux règles d'inférence utilisées lors de l'élaboration des jugements de pertinence [109]: si un nœud est jugé pertinent, ses ancêtres doivent aussi être jugés pertinents, même si leur degré de pertinence est moindre (et ce notamment à cause de la propagation de l'exhaustivité dans l'arbre du document). Par conséquent, un taux de rappel idéal ne peut être obtenu que par les systèmes référençant tous les composants de la base de rappel, y compris les éléments imbriqués.

Afin de résoudre ce problème, les auteurs établissent dans [81] la définition d'une base de rappel idéale, qui supporterait la procédure d'évaluation suivante: les éléments de la base de rappel idéale doivent être retournés par les systèmes, les éléments proches de ceux contenus dans la base de rappel idéale peuvent être vus comme des succès partiels, mais les autres systèmes ne doivent pas être pénalisés s'ils ne les renvoient pas. Les mesures XCG (XML Cumulated Gain) sont proposées pour répondre à ces besoins. Les mesures XCG sont des extensions du «Gain

Cumulatif» proposé par Järvelin et Kekäläinen dans [75]. Les mesures de gain cumulatif ont été développées pour évaluer les systèmes selon le degré de pertinence des documents retournés. La motivation derrière XCG est d'étendre les mesures de gain cumulatif au problème des éléments imbriqués.

Dans INEX 2005, la mesure XCG a été considérée comme la mesure d'évaluation officielle. La mesure XCG inclue des mesures normalisées du gain cumulé "nXCG" qui sont orientées utilisateurs (user-oriented metric), et des mesures orientées effort-précision/gain-rappel (ep/gr) [82]. Une définition de ces mesures se trouve dans ce qui suit :

Les deux dimensions de pertinence, exhaustivité et spécificité, sont combinées en une seule valeur par une fonction dite d'agrégation, deux fonctions d'agrégations sont utilisées :

- *Fonction d'agrégation stricte :* pour évaluer si un système de recherche d'information est capable de retrouver des éléments très spécifiques et très exhaustifs.

$$f_{\text{strict}}(s,e) = \begin{cases} 1 \text{ si } e=2 \text{ et } s=1 \\ 0 \text{ sinon} \end{cases}$$

- *Fonction d'agrégation généralisée :* pour évaluer les éléments selon leur degré de pertinence.

$$f_{\text{general}}(s,e) = s \cdot e$$

Mesure de gain cumulé étendu (xCG(i)) :

Mesure de gain (score de pertinence), c'est la somme de score des éléments retournés par le système de recherche.

$$\text{xCG(i)} = \sum_{j=1}^{i} xG(j)$$

Où xG(j) est le score obtenu du j^{ème} élément retourné dans la liste des éléments retournés par le système de recherche.

Ce gain cumulé est comparé avec celui qu'aurait dû atteindre le système s'il avait produit une liste triée optimale (liste contenant tous les éléments pertinents au début par ordre décroissant de pertinence) pour obtenir le gain cumulé étendu normalisé :

$$nxCG(i) = \frac{xCG(i)}{xCI(i)}$$

Où xCI(i) est le gain cumulé idéal. Cette mesure peut être comparée au 'rappel' dans les documents plats (voir chapitre I section 7).

Effort-précision :

L'effort-précision représente l'effort (en nombre de réponses visitées) qu'un utilisateur doit fournir pour parvenir à un gain r.

$$ep(r) = \frac{\mathrm{e}_{\mathrm{ideal}}}{\mathrm{e}_{\mathrm{run}}}$$

Où $\mathrm{e}_{\mathrm{run}}$ est le rang auquel le gain r est atteint par le système, $\mathrm{e}_{\mathrm{idéal}}$ est le rang auquel le gain r est atteint par le système s'il a produit une liste triée optimale.

Gain-rappel :

Le gain-rappel est la division entre le gain obtenu en une position i de la liste retournée et le gain total auquel on peut parvenir.

$$gr(i) = \frac{xCG(i)}{xCI(n)}$$

Où xCI(n) est le gain cumulé idéal, ce n'est que xCG(n) dans lequel la liste retournée contient les éléments pertinents au début, n étant le nombre total d'éléments pertinents.

Pour évaluer et comparer les systèmes de recherche, on trace la courbe effort-précision / gain-rappel, et on étudie cette courbe comme dans le cas de la courbe *précision-rappel* dans les documents plats (chapitre **I**, section **7**).

Les mesures xCG sont loin d'être adoptées de manière définitive, une réflexion continue ayant lieu sur les mesures utilisées [83, 104, 111] et la notion de pertinence [103].

II.8 Conclusion

Les systèmes de recherche d'information structurée sont principalement conçus pour la prise en compte des documents semi structurés, ils doivent permettre de considérer les termes présents dans un document en fonction de leurs contextes d'apparition, c'est-à-dire l'endroit où ils apparaissent dans la structure du document.

La présence de la structure entraîne deux nouvelles exigences, d'une part, celle de retourner des unités d'information plus petites au lieu du document entier, et d'autre part, celle de permettre la prise en compte de la structure dans la requête pour des réponses plus précises. Par conséquent, la structure offre un apport sémantique non négligeable, puisque l'utilisateur peut préciser l'unité d'information qu'il recherche en incluant des contraintes structurelles dans sa requête.

La plupart des modèles de recherche d'information structurée proposés dans la littérature sont des adaptations des modèles traditionnels. Des extensions sont alors apportées aux modèles classiques de recherche d'information afin de prendre en compte l'impact de la présence de la structure dans les documents, et pour remédier aux différentes problématiques soulevées par cette dernière, comme l'unité d'information adéquate à retourner à une requête et la prise en compte de l'information structurelle au niveau de l'indexation d'un document. Par exemple, en ce qui concerne la pondération des termes d'indexation, les lois utilisées pour la recherche d'information classique ne sont pas forcément valides dans la recherche d'information structurée, comme la loi de Zipf. Alors de nouvelles approches sont développées, comme la loi de pondération *tf-itdf (Term Frequency – Inverse Tag and Document Frequency)* qui permet de calculer la force discriminatoire d'un terme pour une balise relative à un document donné.

L'information structurelle subit rarement un traitement spécifique par ces modèles adaptés, généralement ces derniers utilisent un filtre afin de vérifier les contraintes structurelles des requêtes. Et l'interrogation des corpus hétérogènes (c'est-à-dire comportant plusieurs DTD différentes) pose un problème ouvert jusqu'à présent, les approches proposées se base principalement sur une correspondance syntaxique entre les arbres des requêtes et ceux des documents, or dans la réalité, il faut qu'elle soit basée sur une correspondance sémantique, et ce problème bien sûr, augmente le silence lors de l'interrogation [125].

Les problèmes soulevés par l'évaluation des systèmes de RIS sont nombreux et loin d'être résolus. Ceci s'explique par le fait que l'évaluation de la RIS étant née avec la campagne d'évaluation INEX et est donc relativement récente.

Enfin, la recherche d'information structurée est un sujet d'actualité, nous assistons aujourd'hui aux déploiements de nouvelles méthodes, comme par exemple, l'intégration des outils de traitement automatique de langue pour l'interrogation, dont l'objectif est de traduire du mieux possible les requêtes exprimées en langage naturel vers un langage formel adapté au système de recherche d'information et au corpus, cette approche a été adoptée par Tannier [140]. En ce qui concerne la problématique d'interrogation des corpus hétérogènes, une des solutions proposées est de transformer l'ensemble des documents du corpus dans un format de médiation connu par l'utilisateur formulant la requête.

Dans ce chapitre, nous avons présenté les différents modèles proposés dans la littérature pour répondre aux problématiques de la recherche d'information structurée. Dans le chapitre suivant, nous proposons un nouveau modèle [18, 19] basé sur la théorie des possibilités, très indiquée pour modéliser ensemble l'imprécision et l'incertain, et plus particulièrement sur les réseaux possibilistes qui eux offrent un modèle simple et naturel pour manipuler l'information incertaine inhérente au domaine de la recherche d'information.

Deuxième partie

Un Modèle Possibiliste pour la Recherche d'Information Structurée

Chapitre III

Théorie des Possibilités

III.1 Introduction

La théorie des possibilités a été introduite par L. A. Zadeh [160] et développée par Dubois et Prade [46], en liaison avec la théorie des ensembles flous, pour permettre de raisonner sur des connaissances imprécises. L'information est représentée par une distribution de possibilité, l'incertitude sur une proposition est alors exprimée par un degré de possibilité.

Ce chapitre présente une brève introduction à la théorie des possibilités et aux réseaux possibilistes. Il définit les concepts de distribution de possibilité, nécessité, conditionnement possibiliste et enfin le concept des réseaux possibilistes.

III.2 Théorie des possibilités

III.2.1 Distribution de possibilité

La théorie des possibilités est basée sur les distributions de possibilité [14, 47]. Une distribution de possibilité est une fonction de l'ensemble Ω (Ω est considéré comme l'ensemble de toutes les interprétations possibles appelées aussi solutions ou situations. Ω est dit l'univers de discours) vers une échelle totalement ordonnée qui a une valeur maximale et une valeur minimale, généralement cette échelle est l'intervalle unitaire [0, 1] traduisant une connaissance partielle sur le monde, noté ω. L'échelle possibiliste est définie de deux manières. Dans le cadre numérique les valeurs des possibilités traduisent souvent les bornes supérieures des probabilités. Dans le cadre qualitatif, les valeurs de possibilité peuvent être considérées comme un ordre de classement des états possibles.

Les opérateurs « produit » et « minimum » peuvent être utilisés pour combiner des distributions de possibilité indépendantes dans les cadres quantitatif et qualitatif respectivement.

Une distribution de possibilité peut être représentée de deux manières : *quantitative* ou *qualitative*

Une distribution de possibilité quantitative, notée π, est définie comme suit :

$$\pi : \Omega \rightarrow [0,1]$$

$\pi(\omega)$: représente le degré de compatibilité de l'interprétation ω avec les croyances disponibles sur le monde réel si l'on représente des informations incertaines (ou le degré de satisfaction de ω si l'on représente des préférences).

$\pi(\omega) = 1$: par convention, signifie qu'il est totalement possible que ω soit le monde réel (ou que ω est totalement satisfaisante)

$\pi(\omega) = 0$: signifie que ω n'est certainement pas le monde réel (ou totalement insatisfaisante)

$1 > \pi(\omega) > 0$: signifie que ω est quelque peu possible (ou satisfaisante).

Pour la représentation qualitative [14, 77], la distribution de possibilité est représentée sous forme d'une partition bien ordonnée, notée WOP(π) (Well Ordered Partition), WOP(π)={ E_1, E_2, …, E_n }, avec les conditions suivantes :

- $E_1 \cup E_2 \ldots . \cup E_n = \Omega$
- $\forall$ ω, ω'$\in$ Ω, si ω$\in$ E_i et ω'$\in$ E_j telle que i<j alors $\pi(\omega) > \pi(\omega')$
- $E_i \cap E_j = \emptyset$ pour i≠j

Pour nos travaux, nous nous restreignons, au cadre quantitatif.

Une distribution de possibilité est dite α-normalisée, si son degré de normalisation, noté $h(\pi)$, est égal à α, par ailleurs $\alpha = h(\pi) = \max_{\omega} \pi(\omega)$

Pour α = 1, π est dite normalisée, c'est-à-dire, il existe au moins une solution qui est cohérente avec les croyances disponibles, ou bien il existe au moins une solution qui satisfait toutes les préférences.

III.2.2 Mesures de nécessité et de possibilité

Soit Φ un événement, dans la théorie des probabilités la quantité p(¬Φ) est complètement déterminée par p(Φ) où p(¬Φ) = 1- p(Φ). En effet, si Φ n'est pas probable alors ¬Φ est nécessairement probable [14].

Cependant, l'expression ''ce n'est pas possible que Φ est vrai'' n'implique pas uniquement que ¬Φ ''est possible'' mais aussi qu'il est certain.

De plus, l'expression 'il est possible que Φ est vrai' n'entraîne aucune précision sur la possibilité de Φ, ni sur son impossibilité. Ainsi, la description de l'incertain sur une occurrence de Φ nécessite deux mesures duales : la mesure de nécessité N(Φ) et la mesure de possibilité Π(Φ)

III.2.2.1 Mesure de possibilité

Pour tout sous-ensemble $\Phi \subseteq \Omega$ une mesure de possibilité peut être définie par le plus grand degré de possibilité associé aux modèles de Φ, formellement elle est définie par :

$$\Pi(\Phi) = \max_{\omega \epsilon \Phi} \pi(\omega) \text{ où } \pi \text{ est une distribution de possibilité}$$

Π(Φ) évalue la cohérence de Φ avec les informations disponibles représentées par π (Π(Φ) décrit la situation la plus normale dans laquelle Φ est vrai).

Par exemple, si on considère que nous recevons une information sur une nouvelle race d'animaux dans le pôle nord, dont les recherches ont nommé Glacyria. Ainsi, pour la question : *quelle est la possibilité que Glacyria aie deux pattes ?* On peut dire qu'il est *fortement possible* (i.e : son degré de possibilité est 1) puisque nous ne connaissions pas Glacyria et nous n'avons aucune information qui contredit le fait que cet animal possède deux pattes. Cette réponse est cohérente avec notre connaissance.

Le tableau III.1 suivant donne les propriétés principales de la mesure de possibilité [12].

$\Pi(\Phi)=1$ et $\Pi(\neg\Phi)=0$	Φ est certainement vrai.
$\Pi(\Phi)=1$ et $\Pi(\neg\Phi)\in]0,1[$	Φ est quelque peu certain.
$\Pi(\Phi)=1$ et $\Pi(\neg\Phi)=1$	Ignorance totale (Φ inconnu).
$\Pi(\Phi)>\Pi(\psi)$	Φ est à priori plus plausible que ψ
$\max(\Pi(\Phi), \Pi(\neg\Phi))=1$	Φ et $\neg\Phi$ ne sont pas impossibles simultanément (c'est le seul lien qui existe entre $\Pi(\Phi)$ et $\Pi(\neg\Phi)$).
$\Pi(\Phi \vee \psi)= \max(\Pi(\Phi), \Pi(\psi))$	axiome de décomposition (ou de disjonction).
$\Pi(\Phi \wedge \psi)\leq \min(\Pi(\Phi), \Pi(\psi))$	axiome de conjonction.

Tableau III.1 - *Mesure de possibilité Π (cas des distributions normalisées)*

III.2.2.2 Mesure de nécessité

La mesure de nécessité est définie par $N(\Phi) = 1- \Pi(\neg\Phi) = \min_{\omega\notin\Phi}(1-\pi(\omega))$

$N(\Phi)$ évalue à quel point Φ est inféré à partir des informations disponibles exprimées par π (reflète la situation la plus normale dans laquelle Φ est fausse).

L'équation de dualité $N(\Phi) = 1- \Pi(\neg\Phi)$ étend celle de la logique propositionnelle classique où une formule est déduite à partir d'un ensemble de formules propositionnelles si et seulement si sa négation est incohérente avec cet ensemble.

Notons que $N(\Phi) >0 \Rightarrow \Pi(\Phi)=1$, ce qui signifie qu'un événement est complètement possible avant qu'il soit quelque peu certain. Cette propriété affirme l'inégalité $N(\Phi) \leq \Pi(\Phi)$.

Le tableau III.2 suivant donne les propriétés principales de la mesure de nécessité [12].

$N(\Phi)=1$ et $N(\neg\Phi)=0$	Φ est certain.
$N(\neg\Phi)=0$ et $N(\Phi)\in]0,1[$	Φ est quelque peu certain.
$N(\Phi)=0$ et $N(\neg\Phi)=0$	Ignorance totale (Φ est inconnu).
$\min(N(\Phi), N(\neg\Phi))=0$	L'unique lien entre $N(\Phi)$ et $N(\neg\Phi)$.
$N(\Phi \wedge \psi)= \min(N(\Phi), N(\psi))$	Axiome de conjonction.

Tableau III.2 - *Mesure de nécessité N (cas des distributions normalisées)*

La distance entre $N(\Phi)$ et $\Pi(\Phi)$ évalue le niveau d'ignorance sur Φ.

III.2.3 Conditionnement possibiliste

En logique possibiliste, le conditionnement possibiliste consiste à modifier la distribution de possibilité initiale π à l'arrivée d'une nouvelle information 'e'. Soit Φ, un sous ensemble de Ω, tel que Φ=[e] qui est l'ensemble des modèles de 'e'. La distribution initiale π est remplacée par une nouvelle distribution $\pi' = \pi(\bullet/\Phi)$.

Dans le cadre qualitatif, le degré de possibilité maximal est affecté aux meilleurs éléments de Φ.

$$\pi(\omega/_m\Phi) = \begin{cases} 1 & \text{si } \pi(\omega) = \Pi(\omega) \text{ et } \omega \in \Phi \\ \pi(\omega) & \text{si } \pi(\omega) < \Pi(\omega) \text{ et } \omega \in \Phi \\ 0 & \text{sinon} \end{cases}$$

$/_m$: conditionnement basée sur le minimum.

Dans le cadre quantitatif, les éléments de Φ sont proportionnellement modifiés

$$\pi(\omega/_p\Phi) = \begin{cases} \dfrac{\pi(\omega)}{\Pi(\Phi)} & si \ \omega \in \Phi \\ 0 & \text{sinon} \end{cases} \qquad \text{(III.1)}$$

$/_p$: conditionnement basé sur le produit. Notons que c'est exactement la même définition qu'en théorie des probabilités : elle préserve la valeur relative des degrés de possibilités des éléments de Φ. La seule différence est que $\Pi(\Phi)$ est calculée avec la règle du maximum et non la somme.

III.3 Les réseaux possibilistes

III.3.1 Définition

Un réseau possibiliste orienté sur un ensemble de variables $V = \{V_1, V_2, \ldots, V_n\}$ est caractérisé par [14, 22]:

- Une composante qualitative (graphique) composée d'un graphe acyclique orienté (DAG : Directed Acyclic Graph). La structure du DAG représente l'ensemble des variables ainsi que l'ensemble des relations d'indépendances entre ces variables.
- Une composante quantitative (numérique), qui consiste à quantifier les différents liens du DAG en utilisant des distributions de possibilités conditionnelles de chaque nœud dans le

contexte de ses parents. Toute distribution de possibilité conditionnelle doit vérifier la contrainte de normalisation, pour chaque variable V_i de l'ensemble V, suivante :

Soit U_{Vi} l'ensemble des parents de V_i

Si $U_{Vi} = Ø$ (c'est-à-dire V_i est un nœud racine), alors la possibilité à priori relative à Vi doit satisfaire :

$$\max_{ai} \Pi(a_i) = 1, \ \forall \ a_i \in \ dom(V_i)$$

Si $U_{Vi} \neq Ø$ (c'est-à-dire V_i n'est pas un nœud racine), alors la distribution conditionnelle de V_i dans le contexte de ses parents doit satisfaire :

$$\max_{ai} \Pi(a_i / Par_{Vi}) = 1, \ \forall \ a_i \in \ dom(V_i)$$

Avec :

$dom(V_i)$: le domaine de V_i

Par_{Vi} : l'ensemble des configurations possibles des parents de V_i

La théorie des possibilités offre deux définitions du conditionnement, ce qui conduit à deux définitions des réseaux possibilistes. Les réseaux possibilistes basés sur le produit sont très similaires aux réseaux probabilistes.

III.3.2 Réseaux possibilistes basés sur le produit

Un réseau possibiliste basé sur le produit et orienté sur un ensemble de variables $V = \{V_1,V_2,\ldots,V_n\}$, noté par GP_p, est un graphe possibiliste [47] où les possibilités conditionnelles sont définies en utilisant le conditionnement basé sur le produit (voir section III.2.3).

Les réseaux possibilistes basés sur le produit [59] sont appropriés pour une interprétation numérique de l'échelle possibiliste.

La distribution de possibilité des réseaux possibilistes basés sur le produit, notée π_p, est obtenue par la règle de chaînage suivante:

$$\pi_{p(V1,\ldots,Vn)} = PROD_{i=1,\ldots,n}\ \pi\ (V_i/Par_{Vi}) \qquad \text{(III.2)}$$

Avec :

PROD : l'opérateur produit

III.3.3 Réseaux possibilistes basés sur le minimum

Un réseau possibiliste basé sur le minimum et orienté sur un ensemble de variables $V = \{V_1,V_2,\ldots,V_n\}$, noté par GP_m, est un graphe possibiliste [47] où les possibilités conditionnelles sont définies en utilisant le conditionnement basé sur le minimum (voir section III.2.3).

Les réseaux possibilistes basés sur le minimum sont appropriés pour une interprétation ordinale de l'échelle possibiliste.

La distribution de possibilité des réseaux possibilistes basés sur le minimum, notée π_m, est obtenue par la règle de chainage suivante:

$$\pi_{m(V1,\ldots,Vn)} = MIN_{i=1,\ldots,n}\ \pi\ (Vi/Par_{Vi})$$

Avec :

MIN : l'opérateur minimum

III.3.4 Logique possibiliste

La logique possibiliste [46, 48] est le résultat de l'application d'un formalisme logique à la théorie des possibilités .Une formule logique possibiliste est une paire (Φ,a) telle que Φ est une formule propositionnelle et a est un réel dans l'intervalle [0,1] qui représente le degré de nécessité minimal associé à Φ, c'est-à-dire $N(\Phi) \geq a$. On note que $N(\Phi)>N(\psi)$ signifie que la formule Φ est plus certaine que la formule ψ.

Les informations sont représentées en logique possibiliste au moyen d'une base possibiliste qui est un ensemble de formules quantifiées par des mesures de nécessité de la forme $B = \{(\Phi_i,a_i), i=1..n\}$ où a_i est le degré de nécessité associé à Φ_i .

Les formules avec un poids nul ne sont pas représentées dans la base, car on ne représente que les informations explicites.

La représentation sémantique des informations exprimée par B est donnée au moyen d'une distribution de possibilité qui associe à chaque interprétation possible son degré de possibilité par

rapport aux informations de B. Etant donnée une base possibiliste B, on peut générer une distribution de possibilité unique, notée π_B, où les interprétations qui satisfont toutes les formules de B auront le plus grand degré de possibilité à savoir 1, et les autres interprétations seront ordonnées par rapport à la formule du plus grand poids qu'elle falsifient [12, 77]. Ainsi on obtient

$$\forall \omega \in \Omega, \pi_B(\omega) = \begin{cases} 1 & \text{si } \forall (\Phi_i, a_i) \in B,\ \omega \in [\Phi_i] \\ 1 - \max \{a_i : (\Phi_i, a_i) \in B \text{ et } \omega \notin [\Phi_i]\} & \text{sinon} \end{cases}$$

Dans la logique possibiliste, les règles sont modélisées par des clauses logiques:

$$p \rightarrow q = \neg p \vee q$$

Des valeurs sont attachées aux bornes inférieures des degrés de nécessité et de possibilité de p et q qui sont considérées comme des propositions booléennes.

Les axiomes de la théorie des possibilité permettent de modéliser p implique q avec un poids $\alpha > 0$ par l'inégalité :

$$N(p \rightarrow q) \geq \alpha$$

ou bien d'une manière équivalente par :

$$\Pi(p \wedge \neg q) \leq 1 - \alpha$$

pour signifier que $p \wedge \neg q$ est quelque peu impossible.

La distribution de possibilité exprimant cette information (connaissance) est π telle que :

$\pi(x) = 1 - \alpha$ si $(p \wedge \neg q)$ vraie à l'état x

$= 1$ sinon

La distribution de possibilité induite par plusieurs propositions, mesurée par des nécessités, est obtenue par une intersection floue (utilisant le minimum) des distributions de possibilité induites par chaque proposition.

III.4 Conclusion

Les concepts de bases de la théorie des possibilités et des réseaux possibilistes introduits dans ce chapitre seront utilisés pour la proposition d'un nouveau modèle de recherche d'information structurée qui sera présenté dans le prochain chapitre.

Chapitre IV

MPRIX : modèle possibiliste pour la recherche d'informations XML agrégée

IV.1 Introduction

Le problème auquel sont confrontés les systèmes d'accès aux documents structurés est celui de l'unité d'information à sélectionner dans le cas où la requête de l'utilisateur ne comporte que des mots clés [78]. En effet, les techniques classiques de RI (plein texte) considèrent le document entier comme un granule d'information indivisible, or dans le cas des documents XML tout élément (sous arbre d'un document XML) peut être une réponse potentielle à la requête de l'utilisateur. Le défi à relever est alors d'arriver à identifier automatiquement l'unité d'information, en l'occurrence les parties du document XML, pertinente par rapport à la requête de l'utilisateur. Nos travaux s'inscrivent dans ce cadre et proposent MPRIX (Modèle Possibiliste pour la Recherche d'Informations XML agrégée), un modèle permettant de sélectionner automatiquement un ensemble d'éléments indépendants (agrégat) qui répond le mieux au besoin de l'utilisateur formulé à travers une liste de mots clés. De plus, afin de mieux prendre en compte la structure hiérarchique d'un document XML, les dépendances entre des éléments, ainsi que l'incertitude liée à la notion de pertinence; le modèle MPRIX, que nous proposons, trouve ses fondements théoriques dans les réseaux possibilistes. La structure réseau fournit une manière naturelle de représenter les liens entre, un document, ses éléments (balises) et son contenu. Quant à la théorie des possibilités, elle permet de quantifier de manière qualitative et quantitative les différents liens sous jacents. Elle permet notamment d'exprimer le fait qu'un terme est certainement ou possiblement pertinent vis-à-vis d'un élément et/ou d'un document et de mesurer à quel point un élément (ou un ensemble d'éléments) peut nécessairement ou possiblement répondre à la requête de l'utilisateur.

Ce chapitre est structuré de la manière suivante. La section 2 présente un état de l'art succinct sur la recherche d'information agrégée dans les documents XML. La section 3 décrit, de manière détaillée, le modèle MPRIX et présente un exemple illustrant ce modèle.

IV.2 Motivations

Des approches proches du modèle MPRIX que nous proposons, ont été proposées dans la littérature. Nous citons par exemple celle basée sur la théorie de l'évidence de Dempster-Schafer [91, 92]. Dans cette approche, pour tout document et toute requête, il est possible de définir, pour chaque sous arbre, deux mesures : la croyance et l'incertitude. Un opérateur d'agrégation permet de combiner la croyance de ces sous structures pour calculer la pertinence d'un document. Nous citons aussi l'approche de Piwowarski [106] qui s'appuie sur le formalisme des Réseaux Bayésiens (RB) et qui calcule des probabilités conditionnelles qui sont l'entrée du RB, en fonction de la requête. La recherche correspond à l'inférence dans le RB. Ces deux modèles retournent une liste d'éléments triés par ordre décroissant de leurs pertinences et ne traitent donc pas l'agrégation des éléments XML. Notre modèle est le premier travail qui traite le problème de la recherche agrégée dans les documents XML en utilisant la théorie des possibilités.

L'objectif de la recherche agrégée est de rassembler des informations provenant de diverses sources pour construire des réponses incluant toute l'information pertinente à la requête. Le principal défi de la recherche agrégée est comment identifier et intégrer des résultats pertinents hétérogènes dans un même résultat pour une requête donnée.

A notre connaissance le problème d'agrégation des éléments d'une collection de documents XML n'est pas abordée dans la littérature. Comme il a été mentionné ci-dessus, la plupart des approches de recherche d'information XML [91, 92, 102, 106, 133] renvoient une liste d'éléments disjoints en réponse à une requête. Ces approches considèrent un documents XML comme un ensemble d'éléments, chaque élément lui est attribué un score de pertinence en utilisant soit un modèle de langage [102], soit un modèle probabiliste [3, 133] soit un modèle vectoriel [61, 128].

La recherche agrégée représente un domaine en pleine évolution. La problématique d'agrégation d'éléments issus d'une collection de documents XML n'est pas encore traitée dans la littérature. En effet, les approches proposées et qui traitent cette problématique sont limitées aux documents Web [4, 9, 85, 87, 116, 138, 139]. Cependant, quelques systèmes de recherche d'information XML commencent à agréger des résultats sous forme de résumés. Par exemple, eXtract [71] est un système de RI qui génère des résultats sous forme de fragments XML. Un fragment XML est qualifié comme résultat s'il répond à quatre caractéristiques: autonome

(compréhensif par l'utilisateur), distinct (différent des autres fragments), représentatif (des thèmes de la requête) et succinct. XCLUSTERs [113] est un modèle de représentation de résumés XML. Il regroupe quelques éléments XML et utilise un petit espace pour stocker les données. L'objectif est de fournir des extraits significatifs afin que les utilisateurs puissent facilement évaluer la pertinence des résultats de recherche.

Notre modèle se situe à la jonction de la recherche des éléments les plus pertinents à partir de documents XML et leur agrégation dans un même résultat. Notre objectif est d'assembler automatiquement des éléments pertinents, non redondants et complémentaires d'un corpus de documents XML qui répondent le mieux au besoin de l'utilisateur formulé à travers une liste de mots clés. Le modèle que nous proposons trouve ses fondements théoriques dans la théorie des possibilités [25, 45, 160], et plus particulièrement dans les réseaux possibilistes. Ces réseaux offrent un modèle simple et naturel pour représenter la structure hiérarchique des documents XML et la logique possibiliste offre un cadre théorique intéressant pour mieux prendre en compte l'ignorance et l'incertitude inhérentes à la recherche d'information. On trouve cette incertitude dans, la notion de pertinence d'un document vis-à-vis d'une requête, le degré de représentativité d'un terme dans un document ou une partie de document et l'identification de la partie pertinente répondant à la requête. Le modèle MPRIX permet d'interpréter la pertinence dans un cadre possibiliste [20]. Il sépare les raisons de sélectionner un élément pertinent de celles de le rejeter, en utilisant deux mesures : la nécessité et la possibilité. La possibilité de pertinence tente d'éliminer les éléments non pertinents. La nécessité de pertinence met l'accent (le "focus") sur les éléments qui semblent très pertinents.

Les mesures de possibilité et de nécessité sont utilisées pour quantifier les relations de dépendance entre les termes et les éléments du document qu'ils indexent et entre éléments et document; et permettent de restituer une agrégation d'éléments (regroupement d'éléments) nécessairement ou possiblement pertinents étant donné une requête.

Grâce aux réseaux possibilistes, notre modèle de recherche agrégée dans les documents XML fournit le premier cadre unifié pour la recherche d'éléments pertinents et leur agrégation dans un même résultat.

IV.3 Architecture du modèle MPRIX

Afin de mieux prendre en compte la structure hiérarchique d'un document XML, les dépendances entre éléments, ainsi que l'incertitude liée à la notion de pertinence, le modèle MPRIX [19, 22, 23, 24] trouve ses fondements théoriques dans la théorie des possibilités et plus précisément dans les réseaux possibilistes. La structure réseau fournit une manière naturelle de représenter les liens entre, un document, ses éléments (balises) et son contenu. Elle permet surtout une sélection automatique d'un ensemble d'éléments pertinents provenant de différentes parties du document XML. Quant à la théorie des possibilités, elle permet de quantifier de manière qualitative et quantitative les différents liens sous jacents. Elle permet notamment d'exprimer le fait qu'un terme est certainement ou possiblement pertinent vis-à-vis d'un élément et/ou d'un document et de mesurer à quel point un élément (ou un ensemble d'éléments) peut nécessairement ou possiblement répondre à la requête de l'utilisateur.

Le modèle MPRIX est représenté par un réseau possibiliste d'architecture illustrée par la figure IV.1. D'un point de vue qualitatif le graphe permet de représenter les nœuds documents, termes d'indexation, nœuds (balises d'un document XML). Les liens entre les nœuds permettent de représenter les relations de dépendances entre les différents nœuds.

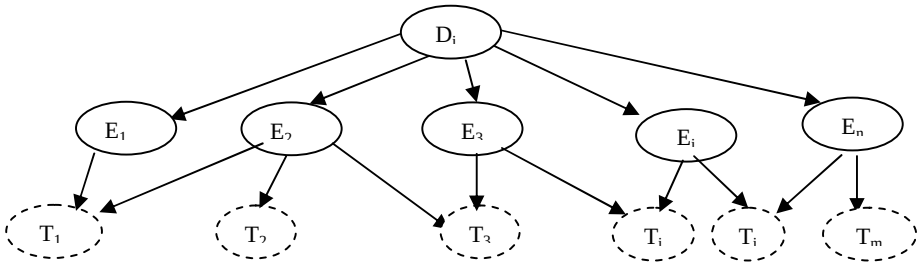


Figure IV.1*- Architecture du modèle MPRIX*

Dans ce modèle, les relations document-éléments et éléments-termes sont modélisées par des mesures de possibilité et de nécessité. La requête de l'utilisateur déclenche un processus de propagation pour retrouver des documents ou des portions de documents nécessairement ou au moins possiblement pertinents par rapport à la requête.

IV.3.1 Description du modèle MPRIX

Les nœuds documents représentent les documents de la collection. Chaque nœud document D_i, représente une variable aléatoire binaire prenant des valeurs dans l'ensemble $dom(D_i) = \{d_i, \neg d_i\}$, où d_i représente 'le document D_i est pertinent pour une requête donnée', et $\neg d_i$ représente 'le document D_i n'est pas pertinent pour la requête donnée'.

Les nœuds E_1, E_2, …E_n, représentent les balises du document D_i. Chaque nœud E_i, représente une variable aléatoire binaire prenant des valeurs dans l'ensemble $dom(E_i) = \{e_i, \neg e_i\}$. L'instanciation $E_i = e_i$ signifie que l'élément 'e_i' est pertinent pour la requête; $E_i = \neg e_i$ signifie que l'élément 'e_i' est non pertinent pour la requête.

Les nœuds $T_1, T_2, \ldots, T_m$ sont les nœuds termes. Chaque nœud terme Ti représente une variable aléatoire binaire prenant des valeurs dans l'ensemble $dom(T_i) = \{t_i, \neg t_i,\}$ où L'instanciation $T_i = t_i$ signifie que le terme 'T_i' est représentatif du nœud père auquel il est rattaché, $T_i = \neg t_i$ signifie que le terme 'T_i' est non représentatif du nœud père auquel il est relié. Il faut noter qu'un terme est relié aussi bien au nœud qui le comporte ainsi qu'à tous les ascendants de ce dernier

Le passage du document vers la représentation sous forme de réseau possibiliste se fait de manière assez simple. Il consiste à ramener tous les nœuds (balises du document) au niveau des variables E_i. Les valeurs qui seront assignées aux arcs de dépendances entre nœuds termes - nœuds balises et nœuds balises - nœud document dépendent du sens que l'on donne à ces liens.

Chaque variable structurelle E_i, $E_i \in E = \{E_1, E_2\ldots, E_m\}$, dépend directement de son nœud parent qui est le nœud racine D, dans le réseau possibiliste du document.

Chaque variable de contenu T_i, $T_i \in T = \{T_i, \ldots, T_k\}$ dépend uniquement de sa variable structurelle (élément structurel ou balise). Il faut également noter que la représentation fait apparaître un seul document. En fait on considère que les documents sont indépendants les uns des autres donc on peut raisonner en considérant le sous réseau qui représente le document que l'on traite.

Nous notons par T(E) (resp. T (Q)) l'ensemble des termes d'indexation des éléments du document (resp. de la requête).

Les arcs (ou liens) sont orientés et on en distingue deux types :

– Les liens termes-balises. Ces liens relient chaque nœud terme Ti ∈ T(E) à chaque nœud Ei où il apparaît.

– Les liens balises-document. Ces liens relient chaque nœud balise de E au document qui le comporte, en l'occurrence D dans notre cas. Nous discuterons dans la section IV.3.3, l'interprétation que nous donnons à ces différents liens et la manière de les quantifier.

IV.3.2 Evaluation d'une requête par propagation

L'évaluation d'une requête est effectuée par à un processus de propagation [18, 19, 22]. Elle consiste à injecter une nouvelle évidence à travers les arcs activés du réseau pour rechercher les documents et les éléments pertinents par rapport à la requête. Comme nous l'avons souligné précédemment, nous modélisons la pertinence selon deux dimensions : la nécessité et la possibilité de pertinence. Notre modèle doit être capable d'inférer des propositions du type :

– « le document d_i est pertinent pour la requête Q » est possible à un certain degré ou non, quantifiée par $\Pi(d_i/Q)$.

– « le document d_i est pertinent pour la requête Q » est certain ou non, quantifiée par $N(d_i/Q)$.

Le premier type de proposition permet d'éliminer les documents (et les éléments) non pertinents, c'est-à-dire ceux qui ont une faible possibilité. La deuxième proposition concentre l'attention sur ceux qui semblent très pertinents.

Pour le modèle de base présenté ici, nous prendrons les hypothèses suivantes :

Hypothèse1 : Un document a autant de possibilité d'être pertinent que non pertinent pour un utilisateur donné, soit $\Pi(d_i) = \Pi(\neg d_i) = 1$, quelque soit i.

Hypothèse2 : La requête est composée d'une simple liste de mots-clés $Q= \{t_1, t_2, ..., t_k\}$. L'importance relative des termes entre eux dans la requête est ignorée.

Afin d'évaluer les quantités, $\Pi(d_i/Q)$ *et* $N(d_i/Q)$, nous procédons de la manière suivante (NB : nous nous sommes fortement inspirés du modèle développé par A. Brini et al. [33] et utilisé pour la recherche plein texte):

$$\Pi(d_i / Q) = \frac{\Pi(Q \wedge d_i)}{\Pi(Q)} \qquad \text{(IV.1)}$$

$$N(d_i / Q) = 1 - \Pi(\neg d_i / Q) \quad = \quad 1 - \frac{\Pi(Q \wedge \neg d_i)}{\Pi(Q)} \qquad \text{(IV.2)}$$

La possibilité de Q est donnée par :

$$\Pi(Q) = \max(\Pi(Q \wedge d_i), \Pi(Q \wedge \neg d_i)) \qquad \text{(IV.3)}$$

L'équation (IV.1) applique la définition de la possibilité conditionnelle (équation III.1), chapitre III, section III.2.3), et l'équation (IV.2) résulte de la dualité entre possibilité et nécessité. L'équation (IV.3) applique la propriété caractéristique de la mesure de possibilité [14]. On en déduit:

$$\Pi(d_i/Q) = \min(1, \frac{\Pi(Q \wedge d_i)}{\Pi(Q \wedge \neg d_i)}) \quad ; \Pi(\neg d_i/Q) = \min(1, \frac{\Pi(Q \wedge \neg d_i)}{\Pi(Q \wedge d_i)}) \qquad \text{(IV.4)}$$

Nous cherchons donc à calculer $\Pi(Q \wedge D_i)$ en prenant les deux valeurs possibles de $D_i = \{d_i, \neg d_i\}$. Etant donné la topologie du graphe et de la distribution de possibilité des réseaux possibilistes basés sur le produit et obtenue par la règle de chaînage (équation III.2, chapitre III, section III.3.2), on trouve:

$$\Pi(Q \wedge D_i) = \max_{\theta^e \in \theta^E} (\Pi(Q/\theta^e) * \Pi(\theta^e/D_i) * \Pi(D_i)) \qquad \text{(IV.5)}$$

Où :

- θ^E : ensemble de toutes les agrégations, d'éléments, possibles du document XML.
- θ^e : sous ensemble de θ^E qui contient uniquement les nœuds E_i (du réseau possibiliste associé au document XML) activés par les termes de la requête. A titre d'exemple, considérons que les termes de la requête sont reliés à deux balises (e_1 et e_2), les agrégations possibles forment le sous ensemble $\theta^e = \{(e_1, e_2), (\neg e_1, e_2), (e_1, \neg e_2), (\neg e_1, \neg e_2)\}$.

De plus en supposant que les variables termes et les variables balises sont indépendantes. L'équation (IV.5) devient alors :

$$\Pi(Q \wedge D_i) = \max_{\forall \theta \in \theta^e} \left(\prod_{E_j \in \theta} \left(\prod_{T_i \in (T(E) \wedge T(Q))} (\Pi(t_i / \theta_{E_j})) \right) * \prod_{E_j \in \theta} (\Pi(\theta_{E_j} / D_i)) * \Pi(D_i) \right) \quad \text{(IV.6)}$$

Où :

- $\prod$ est l'opérateur produit et Π est l'opérateur possibiliste.

- $T_i \in (T(E) \wedge T(Q))$: représente les termes de la requêtes qui indexent les balises (éléments) du document XML.

- θ^e : ensemble des agrégats possibles des éléments (nœuds E_i) activés par les termes de la requête.

- θ : est un agrégat appartenant à l'ensemble θ^e.

- θ_{E_j} : représente l'instance de l'élément E_j dans l'agrégat θ (par exemple, la valeur de E_1 dans l'agrégat (e_1, e_2) est e_1).

La sélection des unités d'informations (agrégats) pertinentes est inhérente au modèle. En effet, l'équation (IV.6) calcule la pertinence en considérant toutes les combinaisons possibles des éléments activés par les termes de la requête (i.e., θ^e). L'agrégation des éléments qui sera sélectionnée est celle qui inclut les terme de la requête et qui présente la meilleure pertinence (la pertinence maximale) en termes de nécessité et/ou de possibilité.

Pour chaque document D_i de la collection, notre modèle est capable de sélectionner la meilleure agrégation d'éléments susceptible d'être pertinente par rapport à la requête de l'utilisateur. Cette agrégation, notée $\theta^*_{D_i}$, est obtenue par [24]:

$$\theta^*_{D_i} = \arg\max_{\forall \theta \in \theta^e} \Pi(Q \wedge D_i) \quad \text{(IV.7)}$$

Par conséquent, pour chaque document, nous sélectionnons un agrégat. Les agrégats sont alors ordonnés selon la valeur de nécessité (équation IV.2) puis selon la valeur de possibilité (équation IV.4) de leurs documents correspondants.

Pour mieux comprendre nos propos, considérons deux documents D_1 et D_2 avec leurs pertinents agrégats $\theta^*_{D_1}$ et $\theta^*_{D_2}$ respectifs (sélectionnés selon l'équation IV.7).

- Si $N(D_1/Q) > N(D_2/Q)$ alors $\theta^*_{D_1}$ sera ordonné (ou classé) avant $\theta^*_{D_2}$.

- Si $N(D_1/Q) = N(D_2/Q)$ alors nous considérons, $\Pi(D_1/Q)) > \Pi(D_2/Q)$ et par conséquent, l'agrégat $\theta^*_{D_1}$ sera classé avant l'agrégat $\theta^*_{D_2}$.

- Ainsi de suite… (si des valeurs sont égales alors on classe en premier n'importe quel agrégat d'entre eux).

De plus, nous devons mentionner que pour ce premier travail, nous supposons que les documents sont indépendants et par conséquent, nous avons un réseau possibiliste par document. L'approche vise à sélectionner la meilleure agrégation d'éléments du même document. Pour une agrégation d'éléments de différents documents, nous devons construire un réseau possibiliste unique pour toute la collection de documents ce qui va être pris en charge dans nos travaux futurs.

Une étape primordiale de notre modèle concerne la quantification et le sens que nous donnons aux différents arcs. Afin d'évaluer les différents degrés Π et N entre les nœuds du réseau, nous nous sommes inspirés du modèle [33] utilisé pour la recherche d'information plein texte.

IV.3.3 Détermination de la valeur des arcs

IV.3.3.1 Valeur de l'arc nœud balise- nœud terme

Dans le domaine de la recherche d'information, les termes utilisés pour représenter le contenu d'un document, sont pondérés de manière à mieux caractériser le contenu de ce document [10]. Le même principe est utilisé en recherche d'information structurée. Les poids sont généralement calculés en utilisant l'information de fréquence d'apparition des termes et la fréquence inverse du document.

En recherche d'information, il a été montré [13, 120, 125] que les performances du système peuvent être améliorées si l'on représente un élément par son propre contenu et par celui de ses éléments fils. Dans notre modèle, nous distinguons les termes possiblement représentatifs des éléments du document et ceux nécessairement représentatifs de ces éléments (termes qui suffisent pour caractériser les éléments). Pour ce faire, nous considérons les hypothèses suivantes :

- **Hypothèse3 :** Un terme est représentatif d'un élément (balise) s'il apparaît dans cet élément.

- **Hypothèse4 :** Un terme est nécessairement représentatif de l'élément s'il apparaît fréquemment dans cet élément et peu fréquemment dans les autres éléments du document.

D'après l'hypothèse3, la possibilité de pertinence d'un terme (t_i) pour représenter un élément (e_j), notée $\Pi(t_i / e_j)$, est calculée comme suit [24]:

$$\Pi(t_i / e_j) = tf_{ij} / \max_{\forall t_k \in e_j} (tf_{kj}) \qquad \text{(IV.8)}$$

Où: tf_{ij} représente la fréquence du terme 't_i' dans l'élément 'e_j'.

Un terme ayant un degré de possibilité 0 signifie que le terme n'est pas représentatif de l'élément. Si le degré de possibilité est strictement supérieur à 0, alors le terme est possiblement représentatif de la balise. S'il apparaît avec un degré de possibilité maximum, alors il est considéré comme le meilleur candidat potentiel pour la représentation et donc la restitution de la balise.

Notons que:

$$\max(\Pi(t_i / e_j)) = 1, \ \exists t_i \in e_j \qquad \text{(IV.9)}$$

Dans un document structuré, un terme nécessairement représentatif d'un élément est un terme qui contribue à sa restitution en réponse à une requête. Ce terme est appelé terme discriminant et c'est un terme qui apparaît fréquemment dans peu d'éléments du document structuré [33]. Le facteur communément utilisé en RI pour quantifier le pouvoir discriminant d'un terme est *idf* (*ief* en RI structurée). Par conséquent, un degré de nécessaire pertinence, β_{ij}, du terme t_i pour représenter l'élément e_j, sera défini par (voir logique possibiliste section III.3.4) :

$$N(t_i \rightarrow e_j) \geq \beta_{ij} = \mu(tf_{ij} * ief_{ij}) * idf = \mu(tf_{ij} * \log(\frac{Ne}{ne_i + 1}) * \log(\frac{N}{n_i + 1})) \qquad \text{(IV.10)}$$

Avec :

- ief : Inverse Element Frequency.
- idf : Inverse Document Frequency.
- tf_{ij}: fréquence du terme 't_i' dans l'élément 'e_j'.

- N et Ne : respectivement le nombre de documents et d'éléments dans la collection.
- n_i et ne_i : respectivement le nombre de documents et le nombre d'éléments contenant le terme t_i.
- μ : est une fonction de normalisation. Une manière simple de normaliser est de diviser par la valeur maximale du facteur.

Il faut noter que cette formule a été choisie compte tenu des performances qu'elle a obtenues dans le cadre des travaux de Sauvagnat [125].

Ce degré de nécessaire pertinence va permettre de borner la possibilité que le terme est compatible avec le rejet de l'élément par :

$$\Pi (t_i \ / \neg e_j) \leq 1 - \beta_{ij} \qquad \text{(IV.11)}$$

Remarque: cette équation est déduite par définition dans la logique possibiliste (voir section III.3.4).

Nous récapitulons la distribution de possibilité définie sur le produit cartésien $\{e_j, \neg e_j\}$ x $\{t_i, \neg t_i\}$ par le tableau suivant [19, 20, 23, 24]:

Π	e_j	$\neg e_j$
t_i	$tf_{ij} / \max(tf_{kj}), (\forall t_k \in e_j)$	$1 - \beta_{ij}$
$\neg t_i$	1	1

Tableau IV.1*- Distribution de possibilité définie sur l'ensemble des termes T*

IV.3.3.2 Valeur de l'arc nœud document – nœud balise

L'arc nœud document – nœud balise (ou arc racine - élément) indique l'intérêt de propager une information d'un élément vers le nœud racine document. Les nœuds apparaissant près de la racine d'un arbre paraissent plus porteurs d'information pour le nœud racine que ceux situés plus bas dans l'arbre [125]. Il semble ainsi intuitif que plus grande est la distance entre un nœud et la racine de l'arbre, moins il contribue à sa pertinence. Nous modélisons cette intuition par l'utilisation dans la fonction de propagation du paramètre *dist(racine, n)*, qui représente la distance entre le nœud racine et un de ses nœuds descendants (balises) n dans l'arbre hiérarchique du document, c'est à dire le nombre d'arcs séparant les deux nœuds.

Le degré de possibilité de propagation d'une balise (e_i) vers le nœud document d_i est défini par $\prod (e_j / d_i)$ et est quantifié comme suit [19, 20, 23, 24]:

$$\prod (e_j / d_i) = \alpha^{\,dist(di,\, ej)-1} \qquad \text{(IV.12)}$$

Avec:

- $dist(d_i, e_j)$ la distance de la balise e_j à la racine d_i conformément à la structure hiérarchique du document.
- $\alpha \in]0..1]$ est un paramètre permettant de quantifier l'importance de la distance séparant les nœuds balises (éléments structurels du document) à la racine du document.

Concernant la nécessité de propager, de manière intuitive, on peut penser que le concepteur d'un document utilise les nœuds de petite taille pour faire ressortir des informations importantes. Ces nœuds peuvent ainsi donner des indications précieuses sur la pertinence de leurs nœuds ancêtres. Un nœud titre dans une section par exemple permet de situer avec précision le sujet de son nœud ancêtre section. Il est donc nécessaire de propager le signal calculé au niveau du nœud vers le nœud racine. Pour répondre à cette intuition, nous proposons de calculer la nécessité de propagation de pertinence d'un élément e_j vers le nœud racine d_i, notée $N(e_j \rightarrow d_i)$ comme suit :

$$N(e_j \rightarrow d_i) = 1 - \frac{le_j}{dl} \qquad \text{(IV.13)}$$

Où: le_j est la taille du nœud balise (élément structurel) e_j et dl la taille d'un document (en nombre de termes).

D'après l'équation (IV.13), plus un élément est de taille petite plus la nécessité de le propager est grande. Par conséquent, nous déduisons l'équation suivante:

$$\prod (e_j / \neg d_i) = le_j/dl \qquad \text{(IV.14)}$$

Nous récapitulons la distribution de possibilité définie sur le produit cartésien $\{d_i, \neg d_i\} \times \{e_j, \neg e_j\}$ par le tableau suivant [19, 20, 23, 24]:

Π	d_i	$\neg d_i$
e_j	$\alpha^{dist(d_i, e_j)-1}$	le_j/dl
$\neg e_j$	1	1

Tableau IV.2*- Distribution de possibilité définie sur l'ensemble des éléments E*

IV.3.4 Exemple illustratif

Un exemple de document XML (un extrait d'un document) relatif à un ouvrage va être utilisé pour illustrer notre approche. Le document XML exemple ainsi que le réseau possibiliste qui lui est associé sont présentés dans ce qui suit :

<Ouvrage>
<Titre> Recherche d'Information </Titre>
<Résumé>Devant la masse croissante d'information … </Résumé>
….
<Chapitre>
<Titre chapitre> Indexation </titre chapitre>
<Paragraphe> L'indexation est le processus destiné à représenter par les
éléments d'un langage documentaire ou naturel des… </Paragraphe>
</Chapitre>
</Ouvrage>

La structure hiérarchique du document XML 'ouvrage' est comme suit :

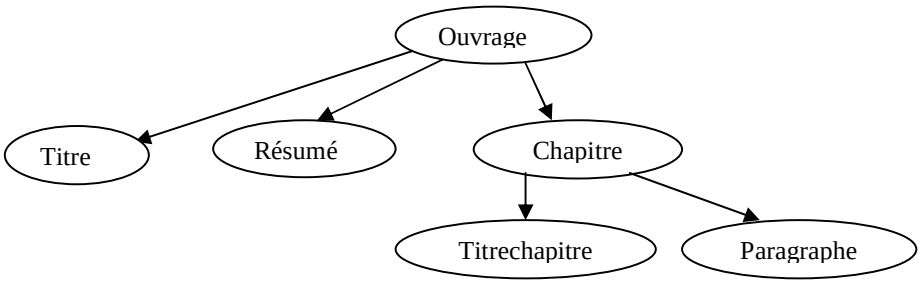


Figure IV.2*- Structure hiérarchique du document XML 'ouvrage'*

Le réseau possibiliste associé au document XML 'ouvrage' est comme suit :

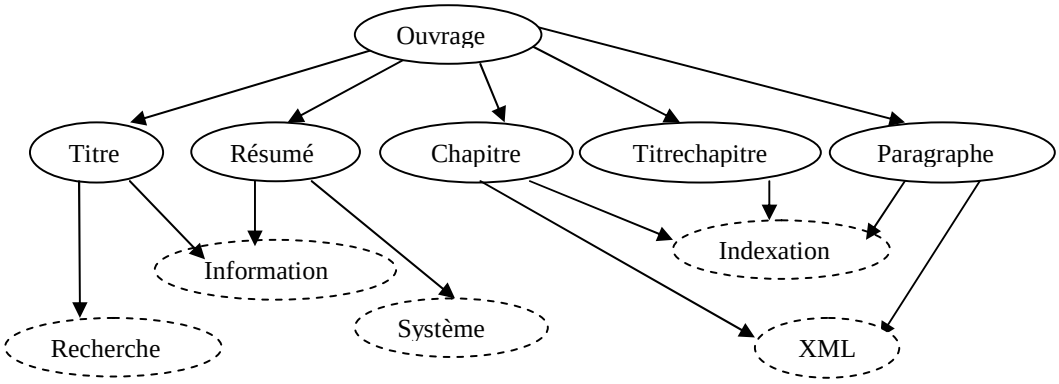


Figure IV.3- *Réseau possibiliste du document XML 'ouvrage'*

Pour cet exemple, l'ensemble des éléments est E= {e_1=Titre, e_2=Résumé, e_3=Chapitre, e_4=Titrechapitre, e_5=Paragraphe}. L'ensemble des termes d'indexation des éléments, calculé en utilisant le contenu de chaque élément ainsi que celui de ses éléments fils dans le document, est tel que T(E) = {t_1=Recherche, t_2=Information, t_3=Système, t_4=Indexation, t_5=XML}. On ne considère que quelques termes pour ne pas encombrer l'exemple.

La matrice contenant les valeurs des arcs nœud balise - nœud terme du réseau possibiliste du document 'ouvrage' est donnée dans le tableau IV.3. Nous rappelons que ces valeurs sont calculées à l'aide de la distribution de possibilité définie sur l'ensemble des termes T (voir tableau IV.1).

Π (t_i/e_j)	t_1	t_2	t_3	t_4	t_5
Titre	1	1	0	0	0
¬Titre	0	0	1	1	1
Résumé	0.50	1	1	0.25	0
¬Résumé	0.50	0	0	0.88	1
Chapitre	0	0	0	0.70	0.50
¬Chapitre	1	1	1	0.10	0.20
Titrechapitre	0	0	0	1	0
¬Titrechapitre	1	1	1	0	1
Paragraphe	0	0	0	0.88	1
¬Paragraphe	1	1	1	0.05	0

Tableau IV.3- *Distribution de possibilité Π (t_i/e_j)*

La matrice contenant les valeurs des arcs racine-nœud balise du réseau possibiliste du document 'ouvrage' est donnée dans le tableau IV.4 (nous prenons α = 0,6 et dl=100). Nous rappelons que ces valeurs sont calculées à l'aide de la distribution de possibilité définie sur l'ensemble des éléments E (voir tableau IV.2).

$\Pi(e_j/d_i)$	e_1	$\neg e_1$	e_2	$\neg e_2$	e_3	$\neg e_3$	e_4	$\neg e_4$	e_5	$\neg e_5$
ouvrage	1	1	1	1	1	1	0,6	1	0,6	1
¬ouvrage	0,02	1	0,1	1	1	1	0,01	1	1	1

Tableau IV.4- *Distribution de possibilité $\Pi(e_j/d_i)$*

Lorsque la requête est posée, un processus de propagation est déclenché à travers le réseau modifiant les valeurs de possibilités a priori. Dans ce modèle la formule de propagation utilisée est l'équation (IV.6).

Soit une requête Q composée des mots clés 'Recherche' et 'Indexation', Q = {Recherche, Indexation}.

D'après l'hypothèse 1, $\Pi(d) = \Pi(\neg d) = 1, \ \forall i$.

Etant donnée la requête Q, le processus de propagation (équation IV.6) considère uniquement les configurations (agrégats) de E qui comportent les termes de la requête 'Recherche' et 'Indexation' ; en l'occurrence seules les balises (éléments) e_1=Titre, e_3=Chapitre, e_4=Titrechapitre et e_5=Paragraphe seront considérées.

Les configurations qu'il faut donc considérées sont : $\theta^e = \{\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6, \theta_7, \theta_8, \theta_9, \theta_{10}, \theta_{11}, \theta_{12}, \theta_{13}, \theta_{14}, \theta_{15}, \theta_{16}\}$ avec $\theta_1 = (e_1, e_3, e_4, e_5)$, $\theta_2 = (e_1, e_3, e_4, \neg e_5)$, $\theta_3 = (e_1, e_3, \neg e_4, e_5)$, $\theta_4 = (e_1, \neg e_3, e_4, e_5)$, $\theta_5 = (\neg e_1, e_3, e_4, e_5)$, $\theta_6 = (e_1, e_3, \neg e_4, \neg e_5)$, $\theta_7 = (e_1, \neg e_3, e_4, \neg e_5)$, $\theta_8 = (\neg e_1, e_3, e_4, \neg e_5)$, $\theta_9 = (e_1, \neg e_3, \neg e_4, e_5)$, $\theta_{10} = (\neg e_1, e_3, \neg e_4, e_5)$, $\theta_{11} = (\neg e_1, \neg e_3, e_4, e_5)$, $\theta_{12} = (e_1, \neg e_3, \neg e_4, \neg e_5)$, $\theta_{13} = (\neg e_1, e_3, \neg e_4, \neg e_5)$, $\theta_{14} = (\neg e_1, \neg e_3, e_4, \neg e_5)$, $\theta_{15} = (\neg e_1, \neg e_3, \neg e_4, e_5)$ et $\theta_{16} = (\neg e_1, \neg e_3, \neg e_4, \neg e_5)\}$.

Nous utilisons l'équation (IV.6) pour calculer la pertinence de ces différents agrégats ($\theta_1, \ldots, \theta_{16}$):

Pour d_i = ouvrage:

$\Pi(Q \wedge D_i)_{\theta_1}$ = Π (Recherche/e_1). Π (Indexation/e_3). Π (Indexation/e_4). Π (Indexation/e_5).

Π (e_1/ouvrage). Π (e_3/ouvrage). Π (e_4/ouvrage). Π (e_5/ouvrage)

= 1* 0.70 * 1 * 0.88 * 1 * 1 * 0.6 * 0.6 = 0.222

$\Pi(Q \wedge D_i)_{\theta_2}$ = Π (Recherche/e_1). Π (Indexation/e_3). Π (Indexation/e_4). Π (Indexation/¬e_5).

Π (e_1/ouvrage). Π (e_3/ouvrage). Π (e_4/ouvrage). Π (¬e_5/ouvrage)

= 1* 0.70* 1* 0.05* 1* 1* 0.6*1 = 0.021

$\Pi(Q \wedge D_i)_{\theta_3}$ = Π (Recherche/e_1). Π (Indexation/e_3). Π (Indexation/¬e_4). Π (Indexation/e_5).

Π (e_1/ouvrage). Π (e_3/ouvrage). Π (¬e_4/ouvrage). Π (e_5/ouvrage)

= 1* 0.70* 0*0.88* 1* 1* 1* 1 = 0

$\Pi(Q \wedge D_i)_{\theta_4}$ = Π (Recherche/e_1). Π (Indexation/¬e_3). Π (Indexation/e_4). Π (Indexation/e_5).

Π (e_1/ouvrage). Π (¬e_3/ouvrage). Π (e_4/ouvrage). Π (e_5/ouvrage)

= 1* 0.10* 1* 0.88*1* 1* 0.6* 1 = 0.053

$\Pi(Q \wedge D_i)_{\theta_5}$ = Π (Recherche/¬e_1). Π (Indexation/e_3). Π (Indexation/e_4). Π (Indexation/e_5).

Π (¬e_1/ouvrage). Π (e_3/ouvrage). Π (e_4/ouvrage). Π (e_5/ouvrage)

= 0 * 0.70 * 1* 0.88* 1* 1* 0.6* 0.6 = 0

$\Pi(Q \wedge D_i)_{\theta_6}$ = Π (Recherche/e_1). Π (Indexation/e_3). Π (Indexation/¬e_4). Π (Indexation/¬e_5).

Π (e_1/ouvrage). Π (e_3/ouvrage). Π (¬e_4/ouvrage). Π (¬e_5/ouvrage)

= 1 * 0.70 * 0 * 0.05 * 1 * 1 * 1 * 1 = 0

$\Pi(Q \wedge D_i)_{\theta_7}$ = Π (Recherche/e_1). Π(Indexation/¬e_3). Π (Indexation/e_4). Π (Indexation/¬e_5).

Π (e_1/ouvrage). Π (¬e_3/ouvrage). Π (e_4/ouvrage). Π (¬e_5/ouvrage)

= 1* 0.10* 1* 0.05* 1* 1* 0.6* 1 = 0.003

$\Pi(Q \wedge D_i)_{\theta_8}$ = Π (Recherche/¬e_1). Π(Indexation/e_3). Π(Indexation/e_4). Π(Indexation/¬e_5).

Π (¬e_1/ouvrage). Π (e_3/ouvrage). Π (e_4/ouvrage). Π (¬e_5/ouvrage)

= 0* 0.70* 1* 0.05* 1* 1* 0.6* 1 = 0

$\Pi(Q \wedge D_i)_{\theta_9}$ = Π (Recherche/e_1). Π (Indexation/¬e_3). Π(Indexation/¬e_4). Π (Indexation/e_5).

Π (e_1/ouvrage). Π (¬e_3/ouvrage). Π (¬e_4/ouvrage). Π (e_5/ouvrage)

= 1* 0.10* 0* 0.88* 1* 1* 1* 0.6 = 0

$\Pi(Q \wedge D_i)_{\theta_{10}}$ = Π(Recherche/¬e_1). Π(Indexation/e_3). Π(Indexation/¬e_4). Π(Indexation/e_5).

Π (¬e_1/ouvrage). Π (e_3/ouvrage). Π (¬e_4/ouvrage). Π (e_5/ouvrage)

= 0* 0.70* 0* 0.88* 1* 1* 1* 0.6 = 0

$\Pi(Q \wedge D_i)_{\theta_{11}}$ = Π(Recherche/¬e_1). Π(Indexation/e_3). Π(Indexation/¬e_4). Π(Indexation/e_5).

Π (¬e_1/ouvrage). Π (e_3/ouvrage). Π (¬e_4/ouvrage). Π (e_5/ouvrage)

= 0* 0.70* 0* 0.88* 1* 1* 1* 0.6 = 0

$\Pi(Q \wedge D_i)_{\theta_{12}}$ = Π(Recherche/e_1). Π(Indexation/¬e_3). Π(Indexation/¬e_4). Π(Indexation/¬e_5).

Π (e_1/ouvrage). Π (¬e_3/ouvrage). Π (¬e_4/ouvrage). Π (¬e_5/ouvrage)

= 1* 0.10* 0* 0.05* 1* 1* 1* 1 = 0

$\Pi(Q \wedge D_i)_{\theta_{13}}$ = Π(Recherche/¬e_1). Π(Indexation/e_3). Π(Indexation/¬e_4).Π(Indexation/¬e_5).

Π (¬e_1/ouvrage). Π (e_3/ouvrage). Π (¬e_4/ouvrage). Π (¬e_5/ouvrage) = 0

$\Pi(Q \wedge D_i)_{\theta_{14}}$ = Π(Recherche/¬e_1).Π(Indexation/¬e_3). Π(Indexation/e_4). Π(Indexation/¬e_5).

Π (¬e_1/ouvrage). Π (¬e_3/ouvrage). Π (e_4/ouvrage). Π (¬e_5/ouvrage) = 0

$\Pi(Q \wedge D_i)_{\theta_{15}}$ = Π(Recherche/¬e_1).Π(Indexation/¬e_3). Π(Indexation/¬e_4).Π(Indexation/e_5).

Π (¬e_1/ouvrage). Π (¬e_3/ouvrage). Π (¬e_4/ouvrage). Π (e_5/ouvrage) = 0

$\Pi(Q \wedge D_i)_{\theta_{16}}$ = Π(Recherche/¬e_1).Π(Indexation/¬e_3).Π(Indexation/¬e_4).Π(Indexation/¬e_5).

Π (¬e_1/ouvrage). Π (¬e_3/ouvrage). Π (¬e_4/ouvrage). Π (¬e_5/ouvrage) = 0

Pour ¬d_i = ¬ ouvrage:

$\Pi(Q \wedge D_i)_{\theta_1}$ = Π (Recherche/e_1). Π (Indexation/e_3). Π(Indexation/e_4). Π(Indexation/e_5).

Π (e_1/¬ouvrage). Π (e_3/¬ouvrage). Π (e_4/¬ouvrage). Π (e_5/¬ouvrage)

= 1* 0.70* 1* 0.88* 0.02* 1* 0.01* 1 = 0.000123

$\Pi(Q \wedge D_i)_{\theta_2}$ = Π(Recherche/e_1). Π(Indexation/e_3). Π(Indexation/e_4). Π(Indexation/¬e_5).

Π(e_1/¬ouvrage). Π(e_3/¬ouvrage). Π(e_4/¬ouvrage). Π(¬e_5/¬ouvrage)

= 1* 0.70* 1* 0.05* 0.02* 1* 0.01* 1= 0.000007

$\Pi(Q \wedge D_i)_{\theta_3}$ = Π(Recherche/e_1). Π(Indexation/e_3). Π(Indexation/¬e_4). Π(Indexation/e_5).

Π(e_1/¬ouvrage). Π(e_3/¬ouvrage). Π(¬e_4/¬ouvrage). Π(e_5/¬ouvrage)

=1* 0.70* 0* 0.88* 0.02* 1* 1* 1= 0

$\Pi(Q \wedge D_i)_{\theta_4}$ = Π (Recherche/e_1). Π(Indexation/¬e_3). Π(Indexation/e_4). Π(Indexation/e_5).

Π(e_1/¬ouvrage). Π(¬e_3/¬ouvrage). Π(e_4/¬ouvrage). Π(e_5/¬ouvrage)

= 1* 0.10* 1* 0.88* 0.02* 1* 0.01* 1= 0.00017

$\Pi(Q \wedge D_i)_{\theta_5}$ = Π(Recherche/¬e_1). Π(Indexation/e_3). Π(Indexation/e_4). Π(Indexation/e_5).

Π(¬e_1/¬ouvrage). Π(e_3/¬ouvrage). Π(e_4/¬ouvrage). Π(e_5/¬ouvrage) = 0

$\Pi(Q \wedge D_i)_{\theta_6}$ = Π(Recherche/e_1). Π(Indexation/e_3). Π(Indexation/¬e_4). Π(Indexation/¬e_5).

Π(e_1/¬ouvrage). Π(e_3/¬ouvrage). Π(¬e_4/¬ouvrage). Π(¬e_5/¬ouvrage)

= 1* 0.70* 0* 0.05* 0.02* 1* 1* 1 = 0

$\Pi(Q \wedge D_i)_{\theta_7}$ = Π(Recherche/e_1). Π(Indexation/¬e_3). Π(Indexation/e_4). Π(Indexation/¬e_5).

Π (e_1/¬ouvrage). Π (¬e_3/¬ouvrage). Π (e_4/¬ouvrage). Π (¬e_5/¬ouvrage)

= 1* 0.10* 1* 0.05* 0.02* 1* 0.01* 1 = 0.000001

$\Pi(Q \wedge D_i)_{\theta_8}$ = Π(Recherche/¬e_1). Π(Indexation/e_3). Π(Indexation/e_4). Π(Indexation/¬e_5).

Π (¬e_1/¬ouvrage). Π (e_3/¬ouvrage). Π (e_4/¬ouvrage). Π (¬e_5/¬ouvrage) = 0

$\Pi(Q \wedge D_i)_{\theta_9}$ = Π(Recherche/e_1). Π(Indexation/¬e_3). Π(Indexation/¬e_4). Π(Indexation/e_5).

Π(e_1/¬ouvrage). Π(¬e_3/¬ouvrage). Π(¬e_4/¬ouvrage). Π(e_5/¬ouvrage)

= 1* 0.10* 0* 0.05* 0.02* 1* 1* 1 = 0

$\Pi(Q \wedge D_i)_{\theta_{10}}$ = Π(Recherche/¬e_1).Π(Indexation/e_3).Π(Indexation/¬e_4). Π(Indexation/e_5).

Π(¬e_1/¬ouvrage). Π(e_3/¬ouvrage). Π(¬e_4/¬ouvrage). Π(e_5/¬ouvrage) = 0

$\Pi(Q \wedge D_i)_{\theta_{11}}$ = Π(Recherche/¬e_1). Π(Indexation/¬e_3). Π(Indexation/e_4).Π(Indexation/e_5).

Π(¬e_1/¬ouvrage). Π(¬e_3/¬ouvrage). Π(e_4/¬ouvrage). Π(e_5/¬ouvrage) = 0

$\Pi(Q \wedge D_i)_{\theta_{12}}$ = Π(Recherche/e_1).Π(Indexation/¬e_3). Π(Indexation/¬e_4).Π(Indexation/¬e_5).

Π(e_1/¬ouvrage). Π(¬e_3/¬ouvrage). Π(¬e_4/¬ouvrage). Π(¬e_5/¬ouvrage)

= 1* 0.10* 0* 0.05* 0.02* 1* 1* 1 = 0

$\Pi(Q \wedge D_i)_{\theta_{13}}$ = Π(Recherche/¬e_1).Π(Indexation/e_3).Π(Indexation/¬e_4).Π(Indexation/¬e_5).

Π(¬e_1/¬ouvrage). Π(e_3/¬ouvrage). Π(¬e_4/¬ouvrage). Π(¬e_5/¬ouvrage) = 0

$\Pi(Q \wedge D_i)_{\theta_{14}}$ = Π(Recherche/¬e_1).Π(Indexation/¬e_3).Π(Indexation/e_4).Π(Indexation/¬e_5).

$$\Pi(\neg e_1/\neg ouvrage).\ \Pi(\neg e_3/\neg ouvrage).\ \Pi(e_4/\neg ouvrage).\ \Pi(\neg e_5/\neg ouvrage) = 0$$

$$\Pi(Q \wedge D_i)_{\theta_{15}} = \Pi(Recherche/\neg e_1).\Pi(Indexation/\neg e_3).\Pi(Indexation/\neg e_4).\Pi(Indexation/e_5).$$

$$\Pi(\neg e_1/\neg ouvrage).\ \Pi(\neg e_3/\neg ouvrage).\ \Pi(\neg e_4/\neg ouvrage).\ \Pi(e_5/\neg ouvrage) = 0$$

$$\Pi(Q \wedge D_i)_{\theta_{16}} = \Pi(Recherche/\neg e_1).\Pi(Indexation/\neg e_3).\Pi(Indexation/\neg e_4).\Pi(Indexation/\neg e_5).$$

$$\Pi(\neg e_1/\neg ouvrage).\ \Pi(\neg e_3/\neg ouvrage).\ \Pi(\neg e_4/\neg ouvrage).\ \Pi(\neg e_5/\neg ouvrage) = 0$$

D'après l'équation (IV.7) l'agrégat sélectionné est celui qui maximise la valeur $\Pi(Q \wedge D_i)$. Ainsi, pour la requête Q = {Recherche, Indexation}, c'est l'agrégat θ_1 = (e_1, e_3, e_4, e_5) = (titre, chapitre, titrechapitre, paragraphe), avec la valeur maximale de $\Pi(Q \wedge D_i)$, qui sera retournée à l'utilisateur comme réponse à sa requête.

Cet exemple utilise un seul document, or notre objectif est de montrer comment les différentes valeurs (N et Π) sont utilisées. Comme il a été mentionné dans la partie évaluation de la requête (section IV.3.2), dans le cas de plusieurs et comme nous supposons qu'ils sont indépendants, nous utilisons exactement la même procédure pour chaque document. Lorsque les agrégats pertinents sont sélectionnés, nous calculons la nécessité et la possibilité de leurs documents correspondants, puis nous les classons en fonction de ces valeurs.

IV.4 Conclusion

Aujourd'hui, les systèmes de recherche d'information structurée de type XML sont capables de retrouver différents types d'informations avec des granularités différentes, mais peu de travaux de recherche les combinent et les assemblent afin de construire des réponses comportant toute l'information pertinente pour la requête de l'utilisateur. Le modèle MPRIX proposé, contrairement à tous ces modèles, vient pour répondre à ce type d'attente. En effet, le modèle MPRIX permet, grâce à ses fondements théoriques (théorie des possibilité et réseau possibiliste); de fournir à l'utilisateur un ensemble d'éléments pertinents regroupés dans un même résultat appelé Agrégat. Le modèle MPRIX fournit un nouveau cadre formel et unifié pour la recherche d'éléments pertinents et leur agrégation dans un même résultat. L'implémentation de notre modèle ainsi que son évaluation seront présentées dans le prochain chapitre.

Chapitre V
Expérimentations et Résultats

V.1 Introduction

Notre modèle est conçu principalement pour répondre à la problématique de la recherche d'information par contenu où aucune indication de structure n'est donnée dans la requête de l'utilisateur et qui peut aider le SRI à déterminer la granularité de l'information à retourner. Par conséquent, notre évaluation porte sur la recherche par contenu qui sert de référence pour mesurer l'impact de la structure dans la recherche d'information.

L'objectif de nos expérimentations est de mesurer les performances et la viabilité de notre modèle. Pour cela, un prototype de recherche d'informations structurées basé sur le modèle possibiliste MPRIX a été conçu et réalisé.

Dans ce chapitre nous décrivons tout d'abord l'architecture du prototype que nous avons développé pour valider la faisabilité de notre approche, puis nous détaillons le protocole d'évaluation ainsi que les expérimentations que nous avons effectuées pour mettre à l'épreuve notre approche.

V.2 Prototype

V.2.1 Architecture générale du prototype

Le modèle possibiliste, de recherche d'informations structurées, proposé a donné lieu au développement d'un prototype permettant l'indexation et l'interrogation de collections de documents XML. Le prototype est réalisé entièrement en langage Java (1.5) en utilisant des API (Application Programming Interface) telles que l'API DOM (Document Object Model) pour analyser les documents XML et JDBC (Java DataBase Connectivity) pour l'accès à l'index implémenté sous le SGBD relationnel PostgreSQL. La figure IV.4 ci-dessous synthétise l'architecture générale du prototype.

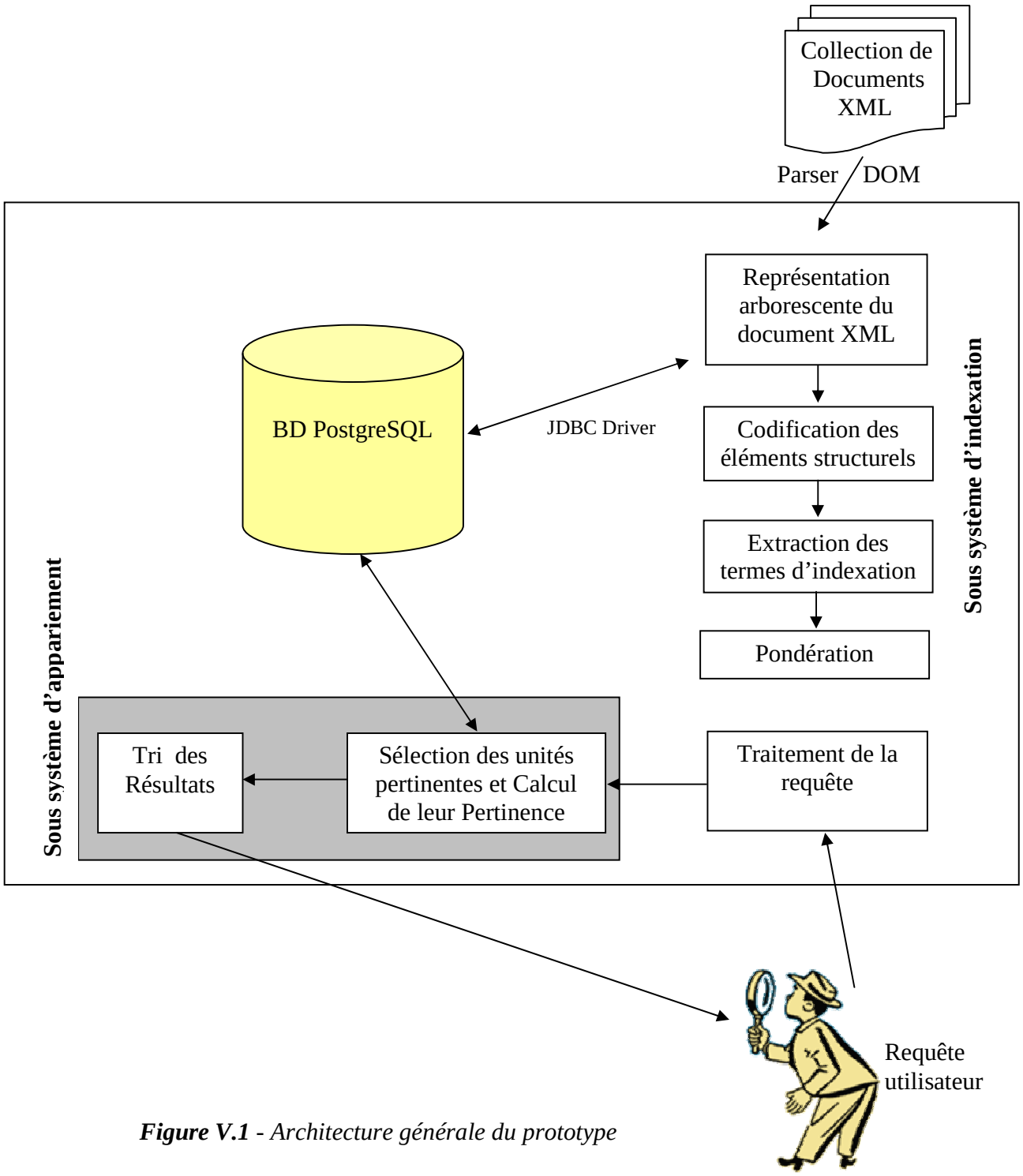


Figure V.1 *- Architecture générale du prototype*

La base de données PostgreSQL stockant les index est l'élément central de cette architecture. En complément de cette base, l'architecture comprend deux principaux modules :

1- Module d'indexation : qui analyse un document XML afin de vérifier sa validité, retourne sa structure arborescente grâce au Parser DOM, extrait les descripteurs (termes) d'index lui correspondant, puis sauvegarde ses derniers dans une base de données PostgreSQL. Ce module permet en outre une indexation incrémentale, ce qui permet de mettre à jour l'index lors de la suppression d'un document ou de l'ajout d'un nouveau document dans la collection de documents XML.

L'indexation peut se faire sur n'importe quelle structure de document XML. Elle conserve toute l'information structurelle du document XML. En effet, l'indexation se fait sur tous les nœuds du document.

2- Module d'appariement : assure le processus de sélection de l'information pertinente à renvoyer comme réponse à la requête utilisateur. Pour cela, il utilise la base de données générée par le module d'indexation.

V.2.2 Schéma de stockage

V.2.2.1 Modèle de représentation des documents

Un document XML est vu sous sa forme hiérarchique et il est composé des quatre ensembles suivants :

- un ensemble d'éléments structurels (non feuilles), représentant uniquement l'information structurelle du document, ce sont les nœuds internes de l'arbre.
- un ensemble d'éléments feuilles, représentant l'information textuelle du document, ce sont les nœuds feuilles de l'arbre.
- un ensemble d'attributs.
- un ensemble d'arc représentant les différents liens de descendance entre les éléments.

Pour la codification des éléments, nous considérons que chaque élément possède une étiquette (identifiant) unique, les étiquettes sont calculées à partir de la représentation arborescente du document de façon à pouvoir reconstruire la structure arborescente du document à partir de ses descripteurs d'index. Tout terme apparaissant dans un élément peut être repéré à l'aide de cette représentation.

Ces étiquettes sont le résultat de la concaténation des positions de chaque nœud par rapport à ces frères qui le précédent de gauche à droite (l'identifiant de la racine du graphe est l'identifiant du

document où il figure). Notons que cette identification nous est utile lors de la remontée des termes d'indexation des éléments feuilles vers leurs éléments parents. En effet, à partir de l'identifiant d'un élément nous pourrons retrouver tous les identifiants de ses ascendants. Cette représentation est illustrée par la figure suivante :

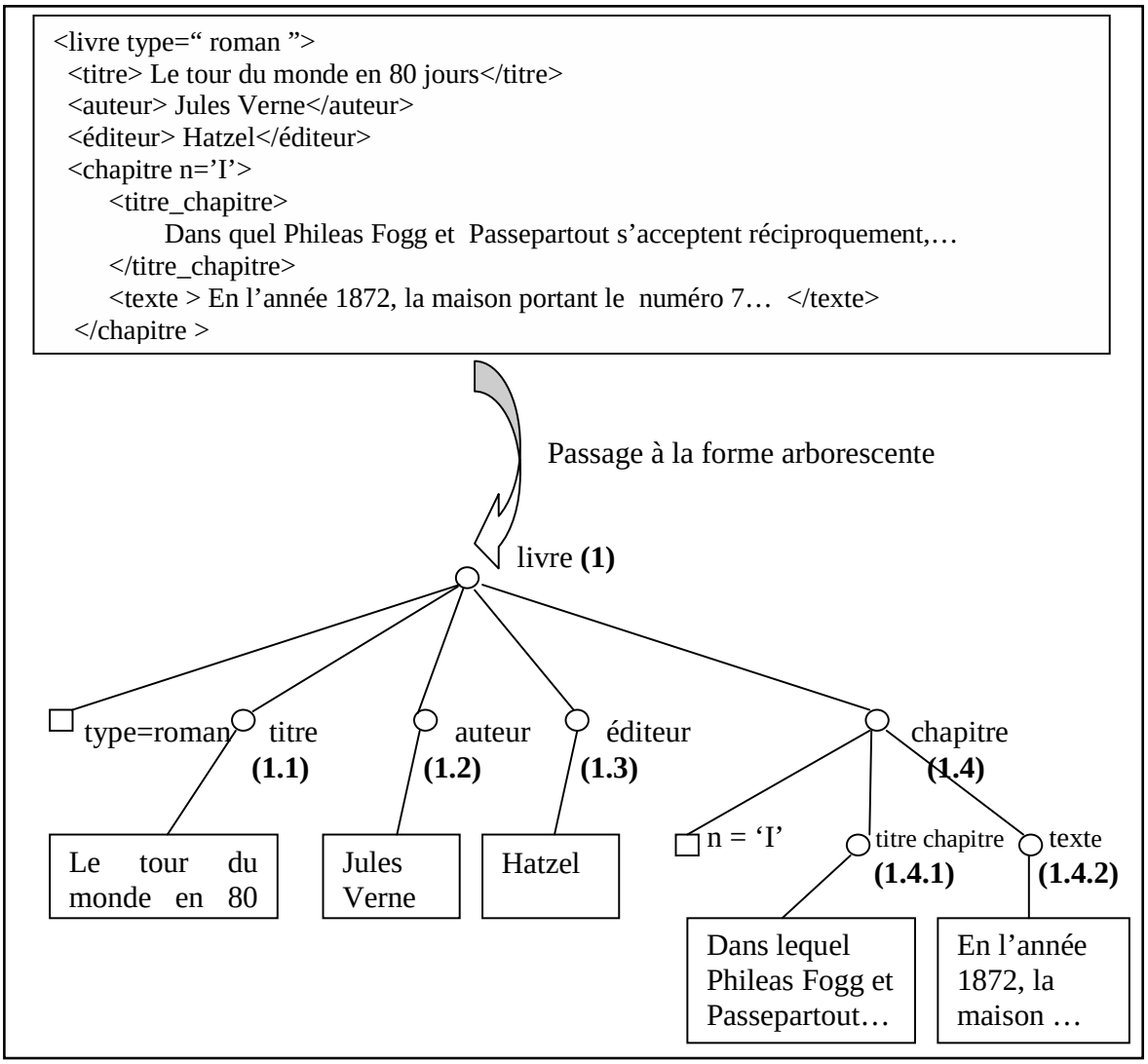


Figure V.2 *- Codification des éléments structurels d'un document XML*

V.2.2.2 Indexation

La méthode d'indexation que nous avons implémentée est la méthode basée sur les arbres. Elle permet de traiter et d'indexer des collections de documents hétérogènes, c'est à dire des documents XML possédant des DTD (Document Type Definition) (voir Annexe) différentes. De plus, elle permet de traiter le contenu mixte c'est-à-dire elle prend en charge des documents XML

où figurent des éléments possédant, en même temps, des éléments non feuille et des éléments feuilles comme éléments fils. Le processus d'indexation est composé de quatre étapes principales illustrées dans la figure VI.6 suivante:

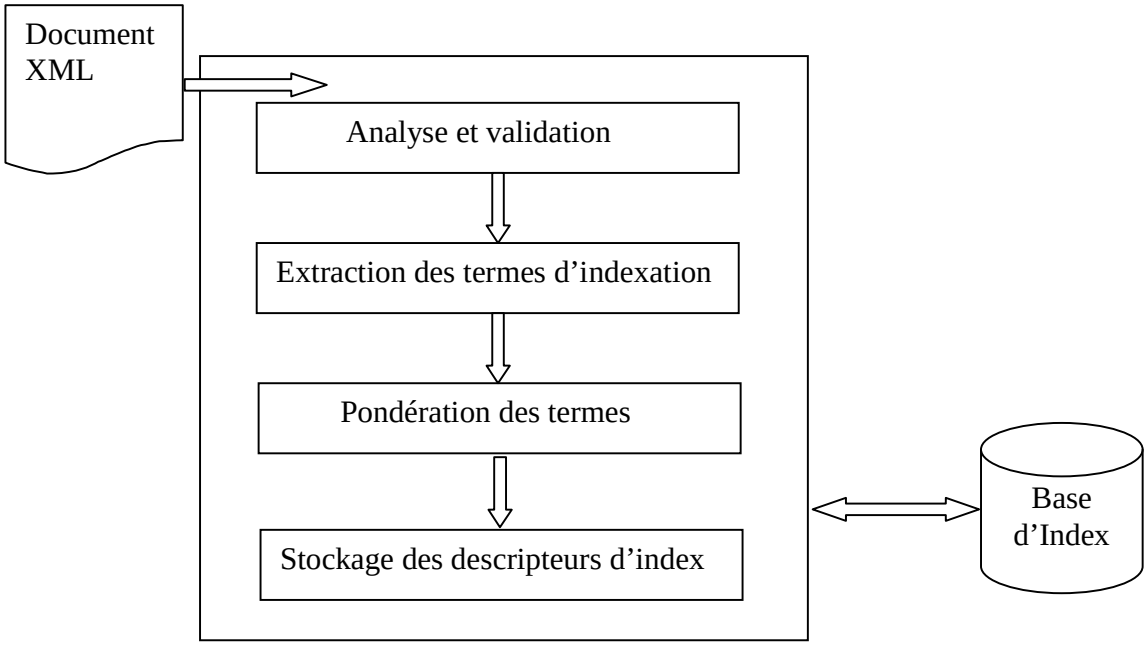


Figure V.3 *- Etapes principales du module d'indexation*

a- ***Analyse et Validation :*** le système n'indexe que des documents XML bien formés ou valides, cette étape est réalisée par le parseur DOM qui répond le mieux à notre modèle de représentation en nous fournissant une représentation arborescente du document XML donné en entrée.

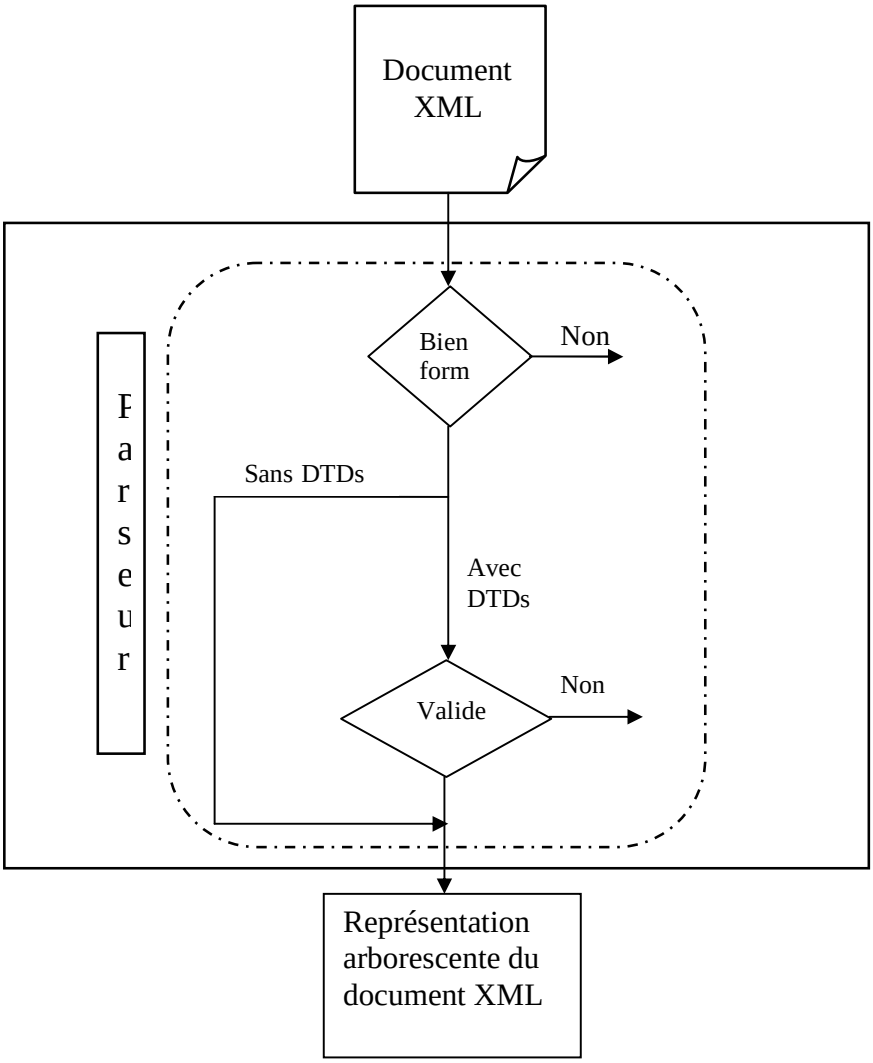


Figure V.4 *- Schéma de l'analyse et de la validation*

b- ***Extraction des termes d'indexation :*** dans cette étape, le contenu textuel brut de chaque élément feuille est extrait. Pour définir les termes de l'index, chaque contenu textuel subira les traitements lexicaux suivants :

- ***Tokenisation :*** Cette étape permet de reconnaître les différents séparateurs entre les mots, elle permet alors de convertir le texte d'un élément en un ensemble de mots (ou tokens).
- ***Elimination des mots vides :*** Les mots vides sont des mots qui portent peu de sens au texte, bien qu'ils soient indispensables à sa compréhension (comme les conjonctions de

coordination, les adverbes, …). Ces derniers doivent être éliminés pour éviter d'encombrer l'indexation, cela est réalisé par l'utilisation d'une liste de mots vides (appelée aussi anti-dictionnaire ou *stop list*).

- ***Lemmatisation :*** Un mot donné peut avoir plusieurs formes dans un texte, mais leurs sens reste le même, donc il suffit d'utiliser un seul pour représenter le concept. Cette étape, consiste à substituer les mots par leur racine (lemme). La lemmatisation est réalisée par l'algorithme de Porter [115].

Ces traitements sont résumés dans la figure ci-dessous :

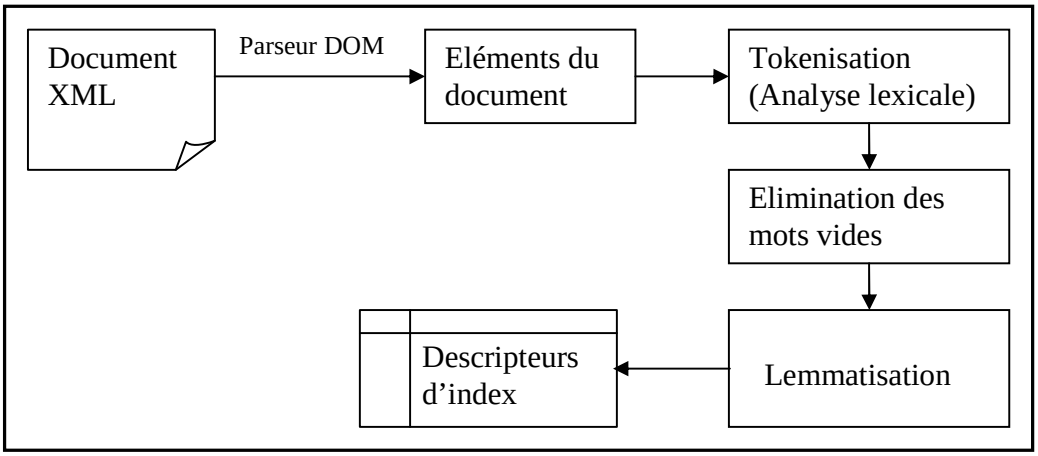


Figure V.5 *- Principales étapes de l'extraction des unités d'indexation*

Notons qu'à ce niveau, la fréquence d'apparition de chaque terme d'indexation dans l'élément feuille est calculée. Cette valeur servira dans l'étape de pondération pour calculer le poids de chaque terme au niveau de l'élément auquel il appartient.

c- ***Pondération et propagation des termes :*** La pondération détermine le poids des termes au niveau des éléments. Le poids doit modéliser l'importance d'un terme dans le nœud, mais aussi au sein du document et de la collection.

La tâche de la pondération se résume à calculer pour tout terme d'un élément donné les mesures '*tf*', '*ief*' *et* '*idf*' ; puis de les agréger en une seule unité pour avoir le poids **w** du terme dans l'élément. Pour cela, nous devons d'abord propager récursivement les termes d'indexation des

éléments feuilles vers les éléments parents en calculant le nombre d'occurrence dans chaque élément parent en appliquant la règle des cumuls suivante:

$$\text{nbr\_occ}(t_i, e_p) = \sum_{j=1}^{nbr\_element\_fils} nbr\_occ(ti, ej)$$

Avec :

- $\text{nbr\_occ}(t_i, e_p)$ est le nombre d'occurrences du terme t_i dans l'élément parent e_p
- $\text{nbr\_occ}(t_i, e_j)$ est le nombre d'occurrences du terme t_i dans l'élément fils e_j

La mesure $\boldsymbol{tf_{ij}}$ représente la fréquence d'apparition du terme 't_i' dans l'élément 'e_j' et se calcule comme suit:

$$\text{tf}_{ij} = \frac{\text{nbr\_occ}(t_i, e_j)}{\text{nbr\_terme}(e_j)}$$

Avec :

- $\text{nbr\_occ}(t_i, e_j)$ est le nombre d'occurrences du terme t_i dans l'élément e_j
- $\text{nbr\_terme}(e_j)$ est le nombre de termes contenus dans e_j

La mesure $\mathbf{ief_i}$ [145,150] représente la fréquence inverse d'élément dans le document pour le terme 't_i', elle est calculée à l'aide de la formule suivante:

$$\text{ief}_i = \log \frac{Ne}{ne_i}$$

Où Ne est le nombre total d'éléments dans le document et ne_i le nombre d'éléments du document contenant le terme 't_i'.

La mesure **idf_i** [161] représente l'importance d'un terme 't_i' au sein de toute la collection, il est calculé à l'aide de la formule suivante:

$$\mathrm{idf}_i = \log \frac{N}{n_i}$$

Où N est le nombre de documents de la collection et n_i le nombre de documents de la collection contenant le terme 't_i'.

Ainsi nous calculons le poids d'un terme 't_i' dans un élément et dans un document donnés par la formule suivante :

$$w_{ij} = tf_{ij} \,.\, ief_i \,.idf_i$$

A la fin de cette étape nous aurons pour chaque élément du document l'ensemble des termes d'indexation qui lui sont associés avec leur poids $\mathbf{W}_{ij}$ respectifs.

d- ***Stockage des descripteurs d'index:*** Dans cette étape, nous allons implémenter l'index dans une base de données relationnelle pour un traitement plus explicite des données. Nous détaillerons les étapes principales de la construction du schéma relationnel correspondant à la base d'index dans la section suivante.

V.2.2.3 Structure de la base d'index

Dans notre prototype, les index sont stockés sous forme de tables dans une base de données relationnelle PostgrSQL. Le schéma entité-association de notre base d'index est illustré dans la figure V.6 suivante:

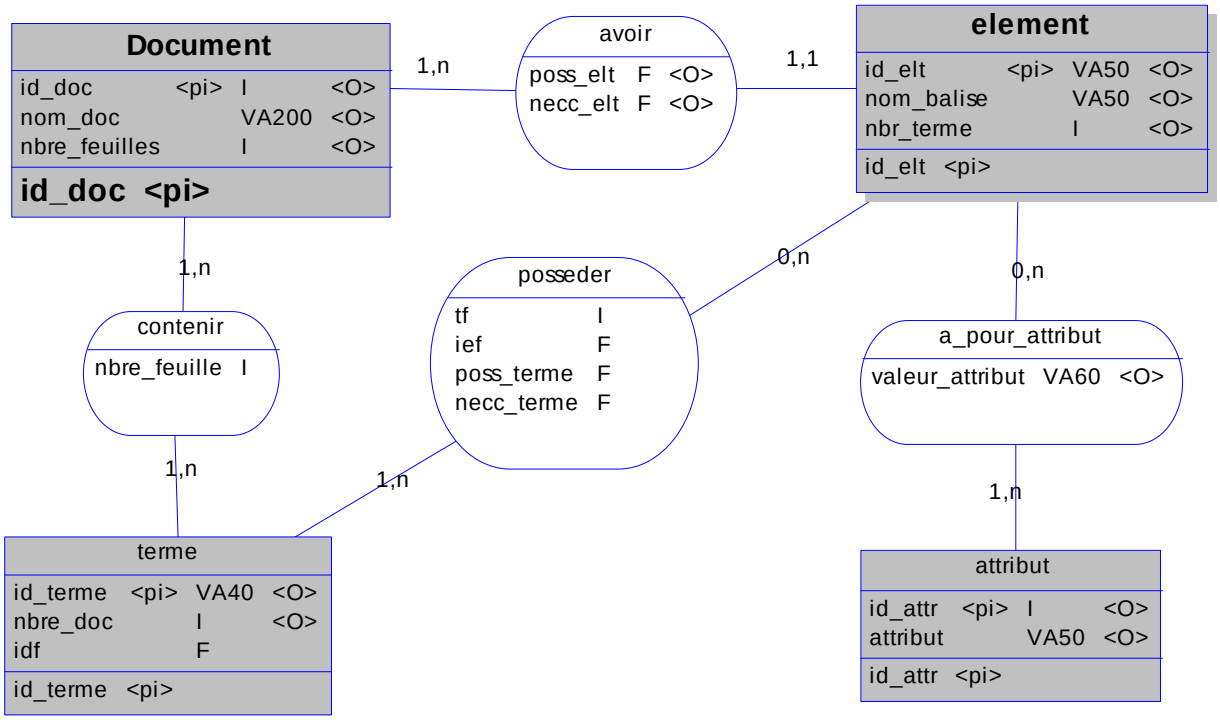


Figure V.6 *- Schéma entité association de la base Index*

À partir du schéma de la figure V.6, et en appliquant les règles de passage du modèle entité-association au modèle relationnel, nous aurons le schéma générique de la base d'index illustré par la figure V.7 suivante:

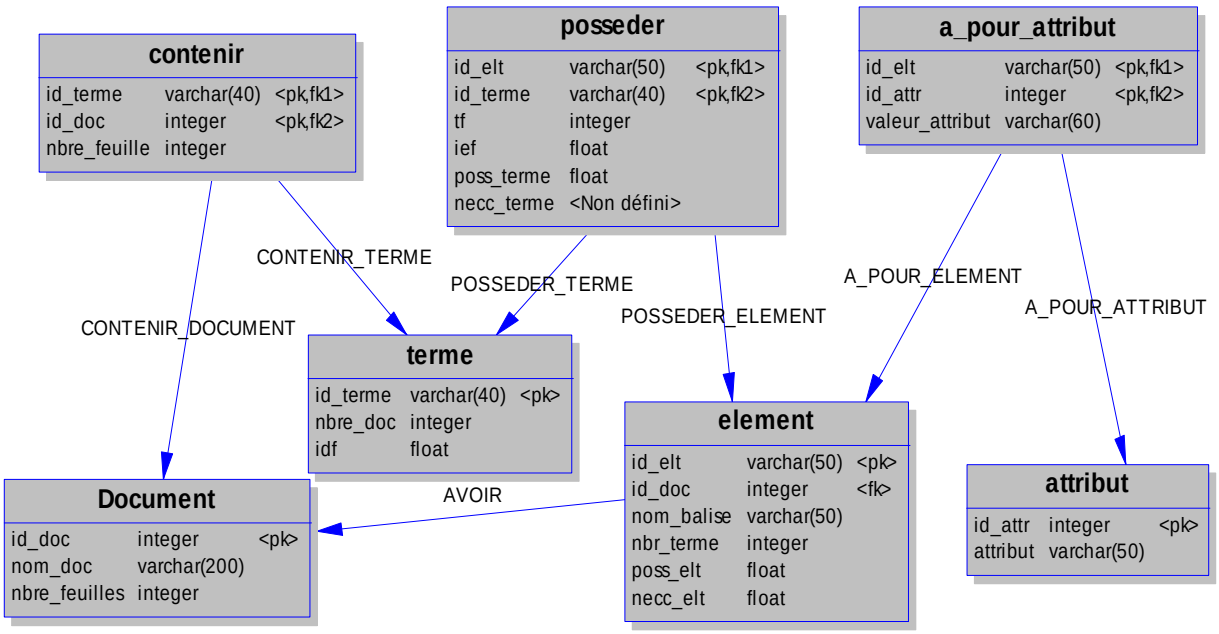


Figure V.7 *- Schéma relationnel de la base Index*

Le schéma de ses tables est détaillé dans le tableau V.1 suivant :

Table	**Description**
Document	**Document** (Id_doc, Nom_doc, Nbre_feuilles) ***Id_doc*** est l'identifiant unique de chaque document, ***Nom_doc*** est le chemin du fichier du document XML, et ***Nbre_feuilles*** est le nombre total d'éléments feuilles du document XML.
Element	**Element**(Id_elt, Id_doc, Nom_balise) ***Id_elt*** est l'identifiant unique de chaque élément, ***nom_balise*** est le nom de la balise, ***nbr_terme*** est le nombre total de termes de l'élément, ***poss_elt*** et ***necc_elt*** sont respectivement le degré de possibilité et de nécessité de l'élément dans le document.
Terme	**Terme** (Id_terme, Nbre_doc) ***Id_terme*** est l'identifiant unique du terme (c'est le terme lui-même) et ***Nbre_doc*** est le nombre de document dans lesquels le terme apparaît. ***Idf*** est la fréquence inverse du terme dans le document.

Attribut	**Attribut** (Id_attr, Attribut) ***Id_attr*** est l'identifiant unique de l'attribut et ***Attribut*** est le nom de l'attribut.
Contenir	**Contenir** (Id_terme, Id_doc, Nbre_feuille) **Nbre_feuille** est le nombre de feuille du document ***id_doc*** où le terme ***Id_terme*** apparaît.
Posseder	**Posseder**(Id_elt, Id_terme, Tf, Ief, Poss_terme, Necc_terme) ***Tf*** et ***Ief*** sont les deux mesures vues dans la ***section 2,*** elles servent pour calculer le poids du terme ***Id_terme*** dans l'élément ***Id_elt*** ***Poss_terme*** et ***Necc_terme*** sont respectivement le degré de possibilité et de nécessité du terme ***Id_terme*** dans l'élément ***Id_elt***
A_Pour_Attribut	**A_pour_attribut** (Id_elt, Id_attr, Valeur_attribut) **Valeur_attribut** est la valeur de l'attribut ayant l'identifiant ***Id_attr*** dans l'élément ***Id_elt***

Tableau V.1 - *Description des tables de la base d'index de MPRIX*

V.3 Expérimentations

V.3.1 Introduction

Avec l'avènement des documents structurés, les systèmes de recherche d'information sont capables de retrouver des informations de granularités différentes et appartenant à différentes parties du document XML. Cependant, ces informations sont retournées sous forme d'une liste d'éléments ordonnés par score de pertinence par rapport à la requête de l'utilisateur. Or, l'information recherchée peut être composée de plusieurs fragments du document. La recherche d'informations agrégée cherche à assembler des informations issues de parties différentes du document XML afin de fournir des réponses comportant toute l'information pertinente pour la requête.

Notre modèle [19, 23] est conçu principalement pour répondre à la problématique de la recherche d'information par contenu où aucune indication de structure n'est donnée dans la requête de l'utilisateur concernant la granularité de l'information à retourner. Par conséquent, notre

évaluation concerne l'impact et l'apport des nouvelles propositions de notre modèle (la recherche agrégée dans les documents XML et la modélisation possibiliste) sur le degré de satisfaction du besoin de l'utilisateur.

Le domaine de la recherche agrégée est relativement récent ce qui explique l'absence de cadre formel et de mesures d'évaluation pour évaluer les systèmes de recherche agrégée. Par conséquent, la seule solution envisageable pour évaluer notre modèle est de mener une étude d'usage [86, 88, 149] afin d'évaluer et d'analyser certaines caractéristiques de notre approche.

L'objectif principal de cette étude est de montrer l'intérêt de la recherche agrégée dans des corpus de documents XML. Cette étude permet d'évaluer si le résultat sous forme d'agrégat, fournit par le modèle possibiliste MPRIX, permet de construire des réponses comportant toute l'information pertinente pour la requête. Autrement dit, si les éléments de l'agrégat sont non redondants, indépendants et complémentaires et s'ils permettent ensemble de donner une vue plus complète et rapide sur les parties pertinentes du document XML.

V.3.2 Méthodologie

V.3.2.1 Système MPRIX

Pour valider notre modèle de recherche d'information structurée, le prototype MPRIX a été implémenté. Ce prototype présente une interface conviviale pour l'interrogation et la présentation des résultats à l'utilisateur. Cette interface permet une formulation simple de requête par contenu (mots clés) et présente les résultats sous forme d'une liste triée, par ordre de pertinence, de configurations ou d'agrégats (ensemble d'éléments regroupés dans un même résultat) répondants au mieux au besoin de l'utilisateur.

V.3.2.2 Protocole d'évaluation

Pour l'évaluation [24] nous avons pris un échantillon, d'environ 1000 documents XML, tiré de la collection INEX 2005. Cette collection est composée d'articles scientifiques provenant de la IEEE Computer Society, balisés au format XML. La collection est composée d'articles provenant de plusieurs magazines ou revues différentes et de plusieurs milliers d'articles de l'encyclopédie Wikipedia. Un article moyen est composé d'environ 1500 éléments.

Un document XML de la collection INEX est composé d'un seul article lui même composé de plusieurs sections (articles IEEE) pouvant appartenir à des volumes différents.

Nous avons pris aussi un ensemble de 20 requêtes par contenu (composées de mots clés) tirées de la même collection. Pour les participants, nous avons pris 30 utilisateurs appartenant à trois disciplines différentes et qui sont l'informatique, la bibliothéconomie et l'électronique. Chaque utilisateur dispose de six (6) requêtes (parmi les 20 requêtes retenues) qu'il soumettra une à une au système MPRIX. Une fois la requête exécutée par le système et le résultat de la recherche affiché, l'utilisateur répondra à sept (7) questions qui ont été soigneusement choisies pour arriver à notre objectif et en même temps pour éviter qu'il soit trop complexe d'y répondre. Le questionnaire est fourni en annexe B. Ce questionnaire permet en fait d'évaluer notre système sur 4 critères décrits en section V.3.2.4.

Pour chaque critère d'évaluation nous aurons 9 jugements utilisateurs par requête, soit 180 jugements utilisateurs à analyser pour l'ensemble des 20 requêtes. L'analyse de toutes les réponses aux questions de tous les utilisateurs, pour les différents critères d'évaluation, nous permettra d'évaluer le modèle MPRIX. En d'autres termes, évaluer l'intérêt d'utiliser plusieurs éléments (parties du document XML) pour répondre à des besoins en information.

V.3.2.3 Requêtes

Les requêtes utilisées dans notre évaluation sont tirées de la collection INEX 2005 et sont en nombre de 20. Pour faciliter notre évaluation, nous avons attribué à chaque requête un identifiant parmi la liste des identifiants suivants : Q1, Q2, …, Q20.

Ces requêtes sont les suivantes :

Q1- Software engineering

Q2- Test and verification

Q3- analysis and methodology

Q4- Ontologies case study

Q5- Markov model equation

Q6- American university libraries

Q7- user-centered design of web sites

Q8- computer assisted composing music notes

Q9- learning object granularity

Q10- capabilities limitations commercial speech recognition software

Q11- electronic commerce business strategies

Q12- wireless multimedia

Q13- latent semantic analysis

Q14- brain research differential geometry

Q15- theory Database experiment

Q16- Synthesizers for music creation

Q17- call for papers conference workshop multimedia

Q18- Central Intelligence Agency Federal Bureau of Investigation personal privacy

Q19- machine translation approaches

Q20- quantum computation

Dans ce qui suit nous explicitons les questions auxquelles les utilisateurs devraient répondre pour chaque requête.

V.3.2.4 Critères d'évaluation

Nos expérimentations sont basées sur les critères d'évaluation suivants [24]: redondance, indépendance, complémentarité et pertinence.

*V.3.*2.4.1 Redondance

La question posée est : 'Les éléments de l'agrégat sont-ils redondants ? '

Cette question permet de savoir si on trouve un même élément plus d'une fois dans un agrégat résultat. Si c'est le cas, alors on peut déduire que notre modèle fournit des informations en plus de ce qu'il faut. Ce qui est communément appelé, dans le domaine de la recherche d'information, par le bruit.

Pour évaluer ce critère, pour chaque requête, une fois les agrégats affichés, l'utilisateur regarde dans chaque agrégat s'il n'y a pas des éléments XML qui sont répétés puis répond à la question 'Les éléments de l'agrégat sont-ils redondants ? ' par 'oui' (si le éléments ne sont pas redondants) ou par 'non' (si il existe au moins un élément qui est redondant).

Pour ce critère, nous aurons 9 jugements utilisateur par requête, donc 180 jugements utilisateur à analyser pour l'ensemble des 20 requêtes [24].

*V.3.*2.4.2 Indépendance

La question posée est : ‘Les éléments de l’agrégat sont-ils indépendants. Autrement dit, sont-ils des éléments qui proviennent généralement de différentes parties (éléments frères et/ou éléments de parents différents) du document XML? ’

L’objectif de cette question est de montrer que notre modèle a la capacité de générer automatiquement des agrégats contenant des informations qui peuvent provenir de parties différentes du document XML, ce qui permet d’orienter de manière rapide l’utilisateur vers les parties pertinentes du document et parfois le renseigner sur la diversité des informations du corpus en rapport avec son besoin en information [24].

Pour ce critère, l'utilisateur soumet une requête au système. Dans le résultat affiché, l'utilisateur identifie le lien entre les éléments de l'agrégat et regarde si les éléments viennent de différentes parties du document XML (éléments de même parents ou de parents différents), puis répond à la question ‘Les éléments de l’agrégat sont-ils indépendants?’ par ‘oui’ (si les éléments sont frères et/ou sont de parents différents, i.e., pas d’imbrication) ou bien ‘non’ (s’il existe au moins un élément et son ancêtre, i.e., il y a imbrication).

*V.3.*2.4.3 Complémentarité

Les deux questions posées sont :

- ‘Trouvez-vous que le résultat sous forme d’agrégat contient des informations complémentaires qui permettent d’avoir une vue plus complète sur le résultat de la recherche ? ‘
- ‘Trouvez-vous que le résultat sous forme d’agrégat a aidé à améliorer le résultat de la recherche ?’

L’objectif de ces deux questions est de savoir si les éléments qui composent l’agrégat sont complémentaires et si cette complémentarité permet une meilleure satisfaction par rapport au besoin en information de l’utilisateur [24].

Pour évaluer le critère de complémentarité, chaque utilisateur soumet ses six requêtes, une à une, au système. Pour chaque résultat affiché, l'utilisateur doit lire le contenu de chaque élément de l'agrégat pour être en mesure de répondre aux deux questions liées à ce critère d'évaluation par ‘oui’ ou ‘non’.

*V.3.*2.4.4 Pertinence

Un des critères les plus importants à évaluer est la pertinence de l'agrégat [24]. Pour cela, la question posée est: ' Par rapport à la requête, l'agrégat est-il pertinent, partiellement pertinent ou non pertinent ? '

La deuxième question posée concerne la pertinence des parties (sous ensemble d'éléments) de l'agrégat. Pour cela, la question posée est: 'Si on prend qu'une partie de l'agrégat résultat. Par rapport à ce dernier, cette partie est-elle pertinente, partiellement pertinente ou non pertinente ?'

L'objectif de ces deux questions est de savoir si, par rapport à ses parties, l'agrégat donne toute l'information utile et nécessaire pour la satisfaction du besoin en information exprimé par la requête de l'utilisateur.

Pour évaluer le critère de pertinence, chaque utilisateur soumet ses six requêtes, une à une, au système. Pour chaque résultat affiché, l'utilisateur doit lire le contenu de chaque élément de l'agrégat pour être en mesure de répondre aux deux questions liées à ce critère d'évaluation par 'pertinent', 'partiellement pertinent' ou 'non pertinent'.

V.3.3 Résultats et analyse

Cette section présente l'analyse des résultats de nos expérimentations [24] basées sur les critères d'évaluation : redondance, indépendance, complémentarité et pertinence.

Pour chaque critère d'évaluation et chaque requête, nous avons récolté 9 jugements utilisateurs à analyser.

V.3.3.1 Redondance

Nous avons constaté, à partir des questionnaires, que tous les utilisateurs ont déclaré que, pour toutes les requêtes soumises au système, les agrégats retournés contiennent des éléments (parties du document) qui ne se répètent jamais, c'est-à-dire non redondants. Les utilisateurs n'ont détecté aucune redondance.

V.3.3.2 Indépendance

Notre objectif ici est de montrer que les éléments qui constituent l'agrégat, retourné par le système MPRIX, peuvent être issus de parties différentes (éléments frères et/ou éléments de parents différents) du document XML.

L'histogramme de la figure V.8 suivante montre la répartition des résultats (réponse 'oui') récoltés par requête.

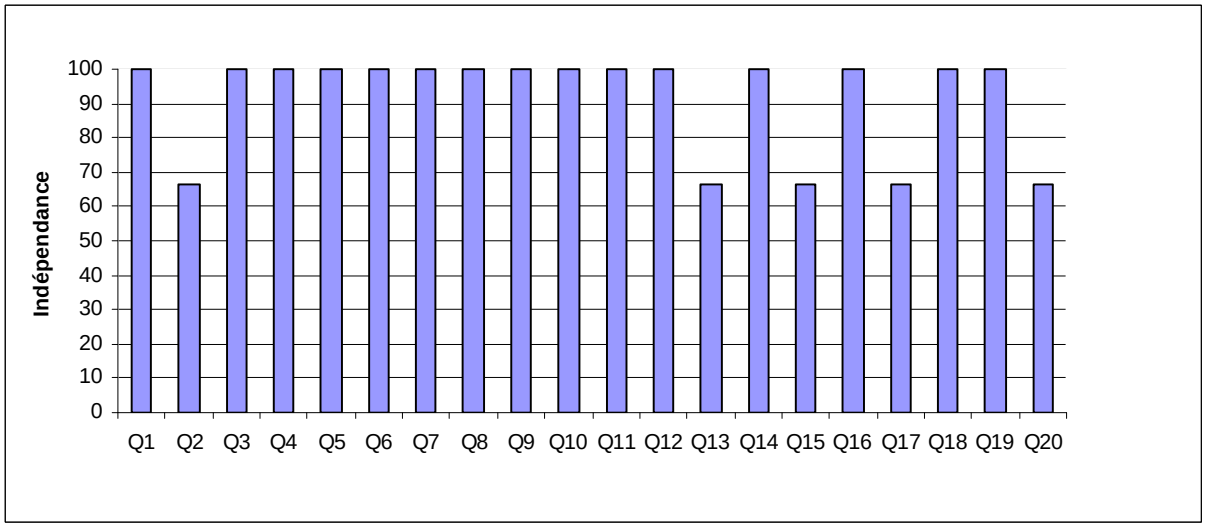


Figure V.8 - *Répartition des résultats par requête par rapport au critère d'Indépendance*

On peut voir sur l'histogramme de la figure V.8, que pour chaque requête, la majorité des utilisateurs ont répondu par 'oui' à la question:'Les éléments de l'agrégat sont-ils indépendants ? On peut voir aussi sur le camembert de la figure V.9 que pour le critère Indépendance, environ 92% des réponses des participants étaient positives.

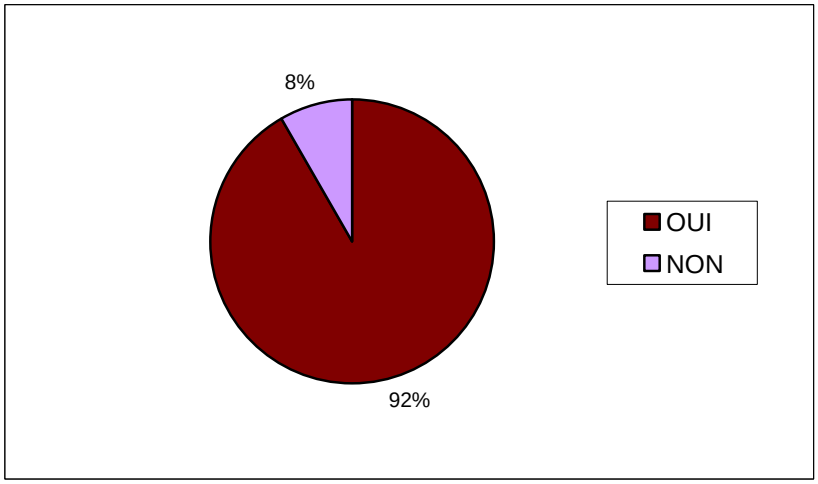


Figure V.9 - *Résultats d'évaluation du critère d'indépendance pour la totalité des réponses*

Ces résultats prouvent que notre modèle génère des agrégats composés d'éléments indépendants qui, généralement, contiennent des informations pouvant provenir de parties différentes du document XML, ce qui permet d'orienter de manière rapide l'utilisateur vers les parties pertinentes du document et parfois le renseigner sur la diversité des informations du corpus en rapport avec son besoin en information ce qui pourra l'aider, si nécessaire, à reformuler sa requête.

V.3.3.3 Complémentarité

L'objectif ici est double. En effet, nous voulons savoir si les éléments de l'agrégat résultat sont complémentaires et dans le cas où c'est vrai, si cette complémentarité est utile.
L'histogramme de la figure V.10 montre la répartition des résultats récoltés par requête.

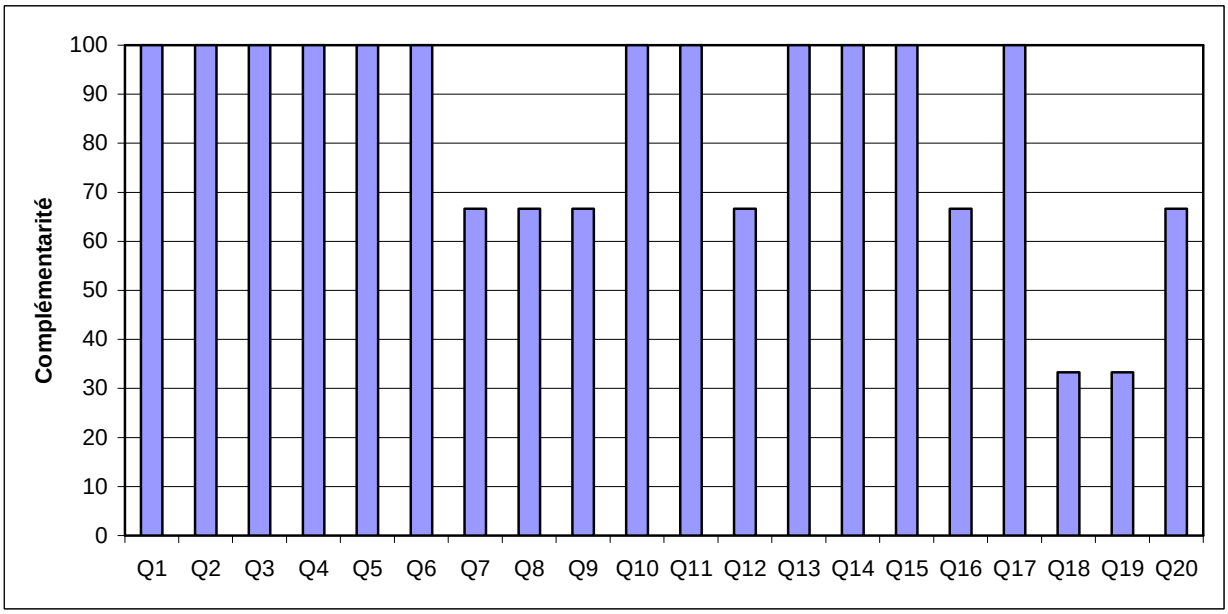


Figure V.10 *- Répartition des résultats par requête par rapport au critère de Complémentarité*

On constate que pour la majorité des requêtes, les réponses concernant la complémentarité des éléments de l'agrégat sont 100% positives. Ceci prouve l'aptitude de notre modèle de recherche agrégée à regrouper dans un même résultat de recherche des informations similaires par rapport à la requête de l'utilisateur. Pour quelques autres requêtes (Q7, Q8, Q9, Q12, Q16, Q18, Q19 et Q20), les réponses ne sont pas 100% positives ce qui laisse déduire que les éléments de l'agrégat

peuvent être non complémentaires par rapport à la requête. En effet, pour la requête Q18 (‘Central Intelligence Agency Federal Bureau of Investigation personal privacy’) nous constatons que différentes thématiques sont exprimées dans cette requête tel que intelligence, investigation et privacy qui est un thème très générique en informatique ce qui explique que les éléments de l’agrégat résultat peuvent être sémantiquement non similaires par rapport au besoin de l’utilisateur et par conséquent non complémentaires. De même pour les requêtes Q7, Q8, Q9, Q12, Q16, Q19 et Q20 qui sont composées de thèmes très génériques.

L’histogramme suivant (figure V.11) donne, par requête, les jugements des utilisateurs concernant l’intérêt de l’agrégation des résultats dans la recherche d’information XML.

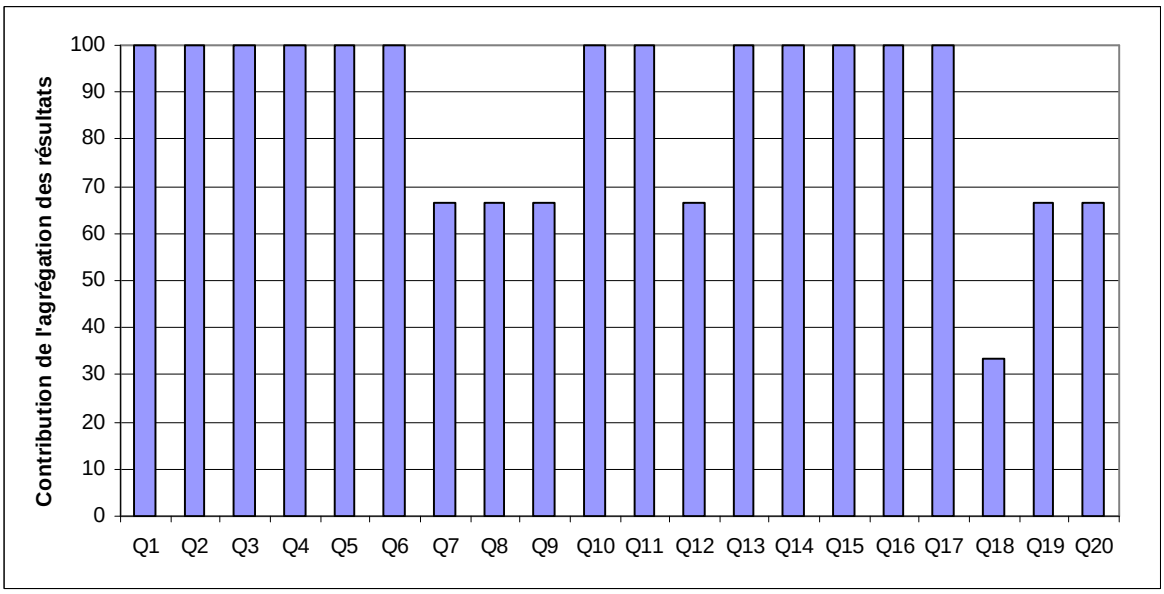


Figure V.11 *- Répartition des jugements concernant l’intérêt de l’agrégation des résultats dans la recherche d’information XML*

Les résultats récoltés nous montrent clairement l’intérêt d’agréger dans un même résultat des éléments non redondants. En effet, pour les requêtes (Q1, Q2, Q3, Q4, Q5, Q6, Q10, Q11, Q13, Q14, 15, Q17) dont les éléments de l’agrégat résultat étaient jugés complémentaires, l’amélioration est estimée à 100%. Cette amélioration est exprimée par l’amélioration de l’interprétation des résultats ainsi que par la réduction des efforts que l’utilisateur doit fournir pour localiser les informations qu’il recherche. Tandis que pour les requêtes (Q7, Q8, Q9, Q12, Q16, Q19, Q20) dont les éléments de l’agrégat résultat étaient jugés non complémentaires à

100%, l'amélioration n'est pas très visible du fait de la non similarité sémantique des éléments de l'agrégat par rapport aux thèmes de la requête. Néanmoins comme l'ont constaté certains utilisateurs, ce type d'agrégation est très intéressant, et même utile, lorsque le besoin en information de l'utilisateur est générique car il permet une distinction très fine des différentes thématiques exprimées dans la requête et par conséquent, peut aider l'utilisateur, si nécessaire, à reformuler son besoin en information.

La première conclusion qu'on peut tirer des résultats ci-dessus (redondance et complémentarité), est que nos agrégats sont composés de différentes parties (éléments) du document XML. Ces éléments sont non redondants, indépendants (pas d'imbrication) et complémentaires. Ce résultat, bien sûr, est une prémisse pour s'intéresser à la recherche agrégée. Si des éléments au sein d'un agrégat ne sont pas complémentaires, redondants ou pas indépendants (overlap), ceci signifie qu'il n'est pas nécessaire d'aller plus loin pour évaluer leur pertinence.

V.3.3.4 Pertinence

Nous voulons savoir, à travers l'analyse du critère de pertinence, si les résultats retournés par le système sont pertinents et si par rapport à ses parties (sous ensemble d'éléments), l'agrégat donne toute l'information utile et nécessaire pour la satisfaction du besoin en information exprimé par la requête de l'utilisateur.

Pour évaluer la pertinence des résultats du système MPRIX, la question posée est: ‘ Par rapport à la requête, l'agrégat est-il pertinent, partiellement pertinent ou non pertinent ? ‘

L'histogramme suivant (figure V.12) donne, par requête, les jugements des utilisateurs concernant la pertinence de l'agrégat par rapport à la requête.

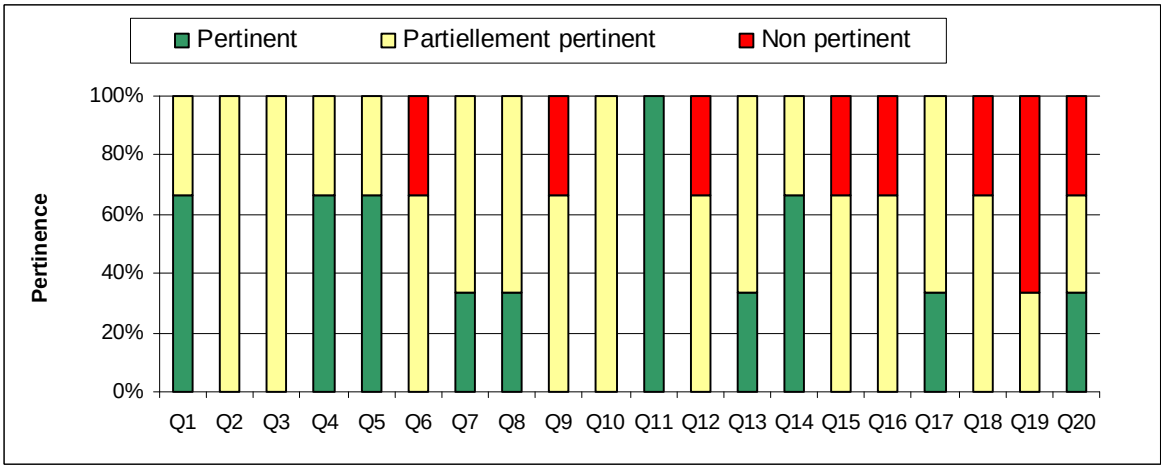


Figure V.12 - *Répartition des résultats de pertinence de l'agrégat par rapport à la requête*

Un document XML de la collection INEX est composé d'un seul article lui même composé de plusieurs sections (articles IEEE) pouvant appartenir à des volumes différents et traitant par fois des thématiques différentes non sémantiquement liées, ce qui a rendu difficile la tâche d'évaluation de la pertinence pour les utilisateurs. Ce qui a aussi compliqué la tâche d'évaluation des utilisateurs est que certaines requêtes prédéfinies (tirées de la collection INEX 2005) traitent plusieurs thématiques à la fois (c'est le cas par exemple des requêtes Q1O, Q12, Q15 et Q18) ou des thèmes trop génériques telles que, par exemple, les requêtes Q2, Q3, Q6, Q19, Q20, Q16 et Q9.

Le camembert de la figure V.13 présente les résultats du jugement de pertinence de l'agrégat pour l'ensemble des vingt requêtes.

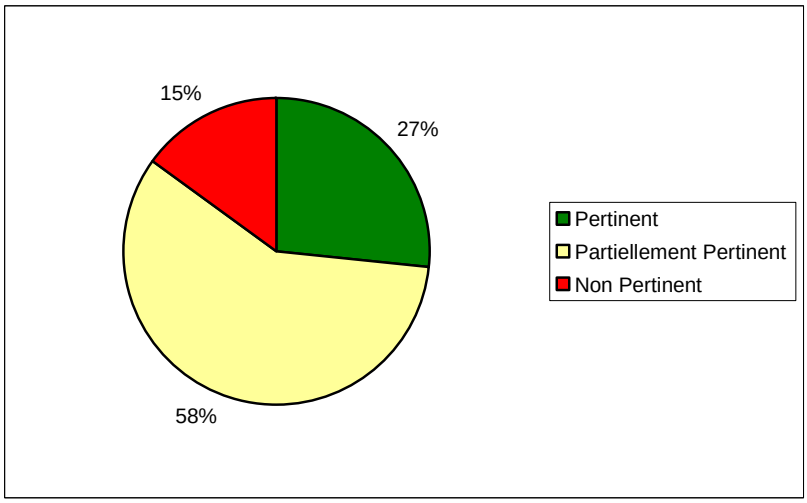


***Figure V.13** - Résultats du jugement de pertinence de l'agrégat pour l'ensemble des réponses des utilisateurs*

Environ 58% des réponses étaient pour une agrégation partiellement pertinente, 27% des réponses étaient pour une agrégation pertinente contre seulement 15% non pertinentes.

La plupart des utilisateurs, dans la réponse à la question sept ('Avez-vous des commentaires à faire ?') du questionnaire, ont constaté que le résultat sous forme d'agrégat les a beaucoup aidé à être plus objectifs dans le jugement de la pertinence de l'agrégat et que certaines requêtes génériques nécessitent une reformulation en des requêtes plus spécifiques ou précises en s'aidant des résultats obtenus par le système MPRIX.

La deuxième évaluation concerne la pertinence des parties de l'agrégat. Pour cela, la question posée est: 'existe-t-il une partie de l'agrégat qui est pertinente, partiellement pertinente ou non pertinente?'

L'histogramme suivant (figure V.14) donne, par requête, les jugements des utilisateurs concernant la pertinence des parties de l'agrégat par rapport à la requête.

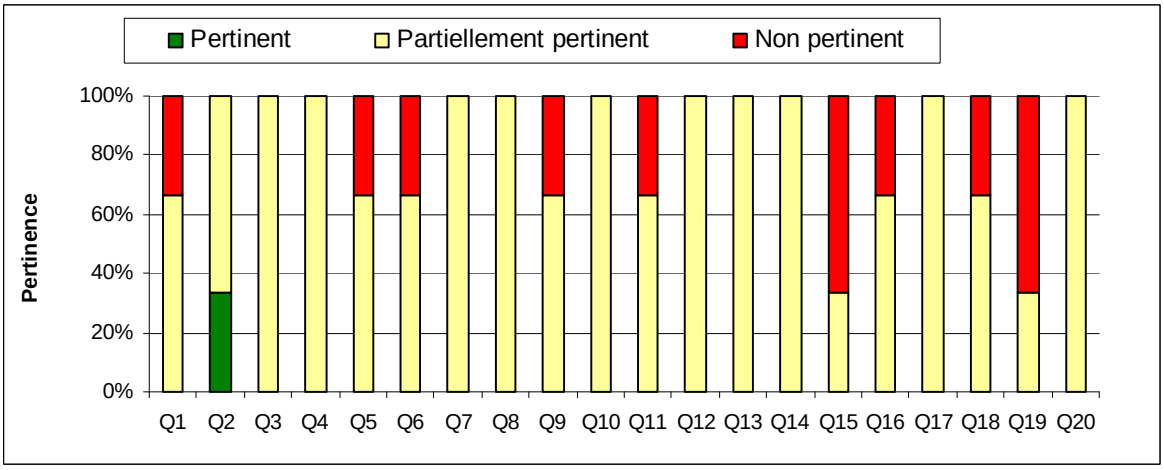


***Figure V.14** - Répartition des résultats de pertinence des parties de l'agrégat par rapport à la requête*

Les résultats récoltés et affichés dans l'histogramme de la figure V.14, concernant la pertinence des parties de l'agrégat par rapport à la requête, montrent que pratiquement pour toutes les requêtes les utilisateurs ont trouvé que les différents sous ensembles d'éléments (parties) des différents agrégats résultats étaient partiellement pertinents.

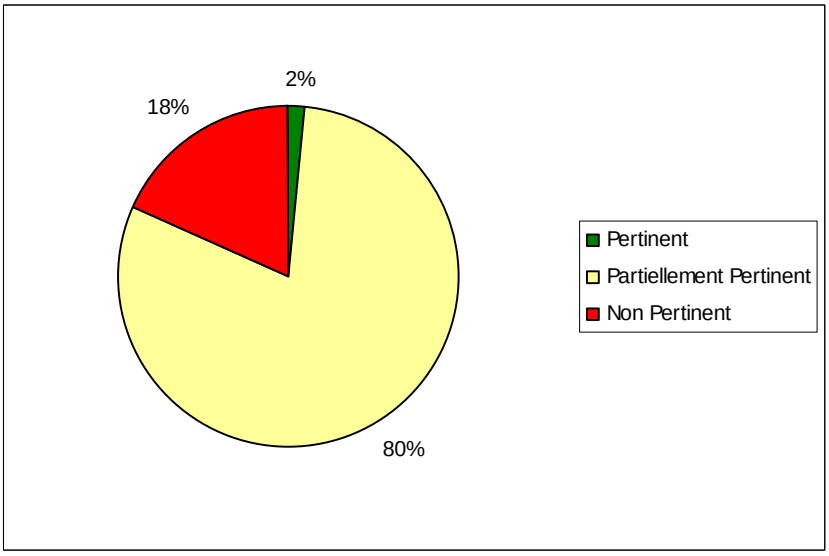


***Figure V.15** - Résultats du jugement de pertinence des parties de l'agrégat, par rapport à la requête, pour l'ensemble des vingt requêtes*

Les résultats affichés par la figure V.15 montrent qu'environ 80% des réponses étaient pour une agrégation partiellement pertinente, 18% des réponses étaient pour une agrégation non pertinente contre seulement 2% pertinentes. Ces résultats montrent clairement que l'agrégat résultat est nettement mieux que ses parties (sous ensemble d'éléments). Ceci est expliqué par le fait que l'agrégat rassemble un ensemble d'éléments non redondants et complémentaires formant une réponse plus complète qui permet d'orienter l'utilisateur vers les éléments pertinents plus rapidement.

Nous pouvons dire que les résultats de cette expérimentation sont cohérents et qu'il existe un lien étroit entre les critères d'évaluation choisis. En effet, si nous prenons les critères complémentarité et pertinence; pour une requête donnée quand les éléments de l'agrégat résultat sont complémentaires ce dernier est jugé comme pertinent ou partiellement pertinent (exemple des requêtes Q1, Q4, Q5, Q11, Q14). Par contre quand les éléments de l'agrégat sont non complémentaires le résultat est jugé comme partiellement pertinent ou non pertinent (exemple des requêtes Q9, Q12, Q16, Q18, Q19).

Le critère d'indépendance a un rôle important dans l'évaluation. Il montre que notre approche génère des agrégats, composés d'éléments disjoints, contenant généralement des informations qui peuvent venir de différentes parties du document XML. Grâce à cette diversité en information, l'évaluateur réussit à porter un jugement sur la complémentarité et la pertinence du résultat.

V.3.4 Discussion

Cette évaluation expérimentale montre que le résultat sous forme d'agrégat donne une autre dimension à la recherche d'information structurée [24]. En effet, l'agrégat rassemble des éléments (parties) non redondants du document XML. Ces éléments peuvent être sémantiquement complémentaires et dans ce cas l'agrégat permet une amélioration de l'interprétation des résultats, oriente l'utilisateur vers les éléments pertinents, du document XML, plus rapidement et permet aussi de réduire les efforts que l'utilisateur doit fournir pour localiser les informations qu'il recherche. Cependant, les éléments de l'agrégat peuvent, dans certains cas, être non complémentaires c'est-à-dire non sémantiquement liés par rapport au besoin en information exprimé par la requête de l'utilisateur. Ce type d'agrégation est très utile car il permet une distinction très fine des différentes thématiques exprimées dans la requête de

l'utilisateur en cas où le besoin en information de l'utilisateur est générique. Il permet aussi de renseigner l'utilisateur sur la diversité des informations du corpus en rapport avec son besoin en information et par conséquent, l'aider, si nécessaire, à reformuler sa requête.

V.3.5 Conclusion

Cette évaluation par expérimentation a montré que le modèle MPRIX, grâce à la théorie des possibilités et plus particulièrement aux réseaux possibilistes, fournit un cadre formel pour agréger dans un même résultat des éléments non redondants, indépendants et complémentaires. Cette complémentarité a l'avantage de réduire la redondance d'information et d'offrir une vision plus riche de l'information existante.

Le modèle MPRIX est un modèle, de recherche d'information XML, capable de retrouver différents types d'informations (texte, image, vidéo…) avec des granularités différentes. Il permet, de manière simple et automatique, d'assembler des éléments issues de parties différentes du document XML, afin de construire des réponses comportant toute l'information pertinente pour la requête. C'est un modèle qui permet une recherche d'informations agrégée.

Grâce à ces résultats sous forme d'agrégat, le modèle MPRIX se caractérise par les points suivants :

- Permet une distinction très fine des différentes thématiques exprimées dans la requête de l'utilisateur en cas où le besoin en information de l'utilisateur est générique.
- Améliore l'interprétation des résultats de recherche.
- Réduit les efforts que l'utilisateur doit fournir pour localiser les informations qu'il recherche.
- Oriente l'utilisateur vers les éléments pertinents, du document XML, plus rapidement.
- Renseigne l'utilisateur sur la diversité des informations du corpus en rapport avec son besoin en information.
- Aide à la reformulation de la requête.

Toutes les études effectuées pour évaluer la recherche agrégée se sont basées sur des études d'usage. Toutefois, même si elles permettent d'évaluer une partie des fonctionnalités de la recherche agrégée, il est nécessaire de trouver des méthodes standard qui permettent d'évaluer les systèmes de recherche d'informations agrégée afin de pouvoir comparer les différentes techniques d'agrégation.

Conclusion Générale

Synthèse

Notre travail se situe dans le contexte de la recherche d'information et plus particulièrement la recherche d'information dans des documents semi structurés de type XML. La spécificité de la recherche d'information dans les documents XML concerne la manière de manipuler efficacement la structure (balises) et le contenu du document pour mieux répondre aux besoins de l'utilisateur.

Dans ce travail, nous nous sommes intéressés à proposer un modèle possibiliste de recherche d'informations qui exploite l'information textuelle et structurelle contenue dans les documents semi structurés de type XML. Le modèle que nous proposons se caractérise par les points suivants:

1. Il propose un modèle de représentation des données générique qui conserve la structure arborescente du document. Ce modèle permet d'indexer des documents semi structurés possédant des structures hétérogènes, sans avoir de connaissance a priori sur cette structure. Nous nous sommes également intéressés au problème de la pondération des termes d'indexation, et nous proposons pour ce faire de tenir compte de l'importance locale (au niveau de l'élément) et globale (au niveau de la collection) des termes.

2. Notre modèle définit la pertinence en essayant de modéliser différents aspects auxquels elle est reliée. Les travaux que nous proposons s'inscrivent dans la définition d'un nouveau modèle de RIS permettant notamment une nouvelle modélisation de la pertinence. Par conséquent, nous proposons de modéliser la pertinence dans un cadre possibiliste. La théorie des possibilités et la logique possibiliste proposent des mesures duales traitant l'incertitude liée à l'information de manière flexible. Ainsi, nous donnons deux sens différents mais complémentaires à la notion de pertinence. Nous proposons une pertinence certaine et une pertinence plausible d'un document (ou élément) étant donnée une requête. Plus précisément, notre modèle sépare les raisons de sélectionner un élément pertinent de celles de le rejeter, en utilisant deux mesures : la nécessité et la possibilité. La possibilité de pertinence tente d'éliminer

les éléments non pertinents. La nécessité de pertinence met l'accent sur les éléments qui semblent très pertinents.

3. Nous traitons le contenu des éléments XML en faisant appel aux fondements de la recherche d'information traditionnelle par adaptation à la nature des documents semi structurés. Or, dans les documents semi structurés le contenu textuel n'apparaît souvent que dans les nœuds feuilles. Cependant, un nœud interne peut être pertinent même s'il ne contient aucun terme d'indexation, la pertinence ne provient pas seulement de la structure. Afin de rétablir l'ordre entre les nœuds pour que les feuilles ne soient pas privilégiées par rapport aux nœuds internes, ceux-ci doivent avoir un score relatif au contenu. Pour propager le texte se situant dans les nœuds feuilles vers leurs ancêtres, permettant de pondérer chaque terme dans ces derniers et contrairement aux modèles de RI utilisant la propagation des scores notre modèle se base sur la propagation du contenu et donc au lieu de la propagation du score, on propage le contenu d'un nœud à un autre via les liens de descendance et on calcule son score de manière indépendante.

4. Au niveau de la recherche des unités d'information pertinentes, nous proposons un modèle de recherche d'informations agrégée permettant de retrouver les unités d'information les plus pertinente par rapport à une requête. Ce modèle repose sur les réseaux possibilistes. Il permet d'identifier, de sélectionner et d'assembler, de manière automatique, des éléments issues de parties différentes du document XML, afin de construire une réponse comportant toute l'information pertinente pour la requête.

Notre modèle se situe à la jonction de la recherche des éléments les plus pertinents à partir de documents XML et leur agrégation dans un même résultat. Il fournit le premier cadre unifié pour la recherche d'éléments pertinents et leur agrégation.

Pour valider ces propositions, un prototype a été implémenté. Ce prototype fournit une interface permettant une formulation simple de requête par contenu (mots clés) et présente les résultats sous forme d'une liste triée, par ordre de pertinence, d'agrégats (groupe d'éléments non redondants) répondants au mieux au besoin de l'utilisateur. Grâce à ce prototype, nous avons effectué une série d'expérimentations sur un échantillon d'environ 3000 documents de la collection issue de la campagne d'évaluation INEX. Cette évaluation par expérimentation a montré que le modèle MPRIX, grâce à la théorie des possibilités et plus particulièrement aux réseaux possibilistes, fournit un cadre formel pour agréger dans un même résultat des éléments

non redondants, indépendants et complémentaires. Cette complémentarité a l'avantage de réduire la redondance d'information et d'offrir une vision plus riche de l'information existante.

Grâce à ses résultats sous forme d'agrégat, le modèle MPRIX se caractérise par les points suivants :

- Permet une distinction très fine des différentes thématiques exprimées dans la requête de l'utilisateur en cas où le besoin en information de l'utilisateur est générique.
- Améliore l'interprétation des résultats de recherche.
- Réduit les efforts que l'utilisateur doit fournir pour localiser les informations qu'il recherche.
- Oriente l'utilisateur vers les éléments pertinents, du document XML, plus rapidement.
- Renseigne l'utilisateur sur la diversité des informations du corpus en rapport avec son besoin en information.
- Aide à la reformulation de la requête.

Perspectives

Les perspectives de ce travail sont nombreuses:

- A très court terme, la question que nous souhaitons traiter est l'évaluation des systèmes de recherche d'information renvoyant des agrégats. A l'heure actuelle, il n'existe aucun protocole, ni collections, ni métriques pour effectuer ce type d'évaluation. Nos premières investigations sur le sujet serviront de socles pour aborder cette importante question.
- Nous souhaitons également étendre notre modèle pour prendre en compte les requêtes portant sur la structure et le contenu des unités d'information, dans lesquelles l'utilisateur spécifie des besoins précis sur certains éléments de structure. Dans ce type de requêtes, l'utilisateur peut utiliser les conditions de structure pour indiquer le type des éléments qu'il désire voir renvoyer, mais aussi plus simplement pour préciser son besoin. Nous pourrons également étendre le modèle pour prendre en comte des requêtes flexibles intégrant des préférences utilisateur indiquant l'importance accordée à chaque terme de la requête.
- Gestion de corpus de documents hétérogènes, c'est-à-dire des documents ayant des DTD différentes. Pour les requêtes par contenu (CO), notre approche pourra être employée pour

interroger de tels corpus. Mais pour les requêtes par structure et contenu (CAS), les conditions de structures exprimées par les utilisateurs dans la requête ne correspondent pas forcément exactement aux DTD des documents présents dans le corpus, mais ces derniers pourraient pourtant être pertinents pour l'utilisateur. Les approches pour les corpus hétérogènes cherchent à vérifier des *correspondances sémantiques*. Une première solution est d'utiliser un lexique, un *thesaurus* ou une ontologie pour faire correspondre les conditions de structures exprimées dans la requête avec les types d'éléments effectivement présents dans la collection [142]. D'autres approches, comme celle proposée par Denoyer *et al.* dans [43] ou Abiteboul *et al.* dans [2] visent à proposer un format médian dans lequel tous les documents du corpus (et éventuellement les requêtes) peuvent être transformés pour ensuite appliquer des techniques traditionnelles de traitement des requêtes structurées.

- Intégration de la notion de réinjection de la pertinence (relevance feedback) à notre modèle. Dans le cadre de la recherche d'information structurée, la notion de réinjection de la pertinence intègre à la fois les notions de structure et de contenu. Toute la question est alors de savoir comment intégrer l'information structurelle dans la formulation de la nouvelle requête. Comment interpréter les unités d'information jugées pertinentes et non pertinentes par l'utilisateur? Doit-on considérer la structure des réponses comme une contrainte forte qu'il vaut mieux respecter, ou au contraire comme une indication des nœuds les plus probablement pertinents?

Bibliographie

[1] S. ABITEBOUL, D. QUASS, J. MCHUGH, J. WIDOM, J.L. WIENER, “The Lorel query language for semi structured data”, International Journal on Digital Libraries, vol. 1, n°1, p. 68-88, 1997.

[2] S. ABITEBOUL, I. MANOLESCU, B. NGUYEN, N. PRADA, “A test platform for the INEX heterogeneous track”, Pre-proceedings of INEX 2004, Dagstuhl, Allemagne, p.177-182, 2004.

[3] M. ABOLHASSANI and N. FUHR, “Applying the divergence from randomness approach for content-only search in XML documents”, In Proceedings of ECIR 2004, Sunderland, p. 409-419, 2004.

[4] R. AGRAWAL, S. GOLLAPUDI, A. HALVERSON, “Diversifying search results”, ACM Int. Conference on WSDM, 2009.

[5] J. ALLAN, J. CALLAN, M. SANDERSON, J. XU and S. WEGMANN, “INQERY at TREC-7”, in Proceedings of TREC-7, p. 201–216, 1998.

[6] S. ALDER, “eXtensible Stylesheet Language (XSL), version 1.0.”, Technical report, World Wide Web Consortium (W3C), W3C Recommendation, 2001.

[7] S. AMER-YAHIA, C. BOTEV, and J. SHANMUGASUNDARAM, “TexQuery: A fulltext search extension to Xquery”, in Proceedings of WWW 2004, 2004.

[8] V. N. ANH and A.MOFFAT, “Compression and an IR approach to XML Retrieval”, proceedings of INEX 2002 Workshop, Dagstuhl, Allemagne, 2002.

[9] J. ARGUELLO, F. DIAZ and J. CALLAN, “Learning to aggregate vertical results into web search results”, in Proceedings of the 20th ACM Conference on Information and Knowledge Management,CIKM'11, Glasgow, United Kingdom, 2011.

[10] R. BAEZA-YATES, B. RIBEIRO-NETO, “Modern information retrieval”, ACM Press Books, Addison-Wesley, 1999.

[11] M. BAZIZ, “Indexation conceptuelle guidée par ontologie pour la recherche d’information”, thèse de Doctorat, université de Paul Sabatier, Toulouse (France), 2005.

[12] N. BEN AMOR, “Qualitative Possibilistic Models : From independence to propagation Algorithms”, thèse de Doctorat, université de Tunis, 2002.

[13] M. BEN AOUICHA, "Une approche algébrique pour la recherche d'information structurée", thèse de Doctorat, Université Paul Sabatier, Toulouse (France), 2009.

[14] S. BENFERHAT, D. DUBOIS, L. GARCIA and H. PRADE, "Possibilistic logic bases and possibilistic graphs", In Proc. of the 15th Conference on Uncertainty in Artificial Intelligence, p.57-64, 1999.

[15] F.Z. BESSAI, "La recherche d'information dans une base de documents structurés en XML", EdiCulture, séminaire sur XML et l'Echange de Données Informatiques, Ecole Nationale d'Administration, ENA, Hydra Alger, Algérie, Avril 2003.

[16] F.Z. BESSAI, Z. ALIMAZIGHI**, "**Storage and retrieval document structured in XML**",** The 4th International Multi-Conference on Computer Science and Information Technology, CSIT 2006, Amman, Jordanie, p. 276-284, 2006.

[17] F.Z. BESSAI, Z. ALIMAZIGHI, "Xsearcher: Un système de stockage et d'interrogation de document structurés en XML**",** Revue de l'Information Scientifique et Technique**,** Vol.16, N° 01, 2006, p.11-25, ISSN 1111-0015.

[18] F.Z. BESSAI, M. BOUGHANEM, **"**Recherche d'Information Structurée : Vers un modèle possibiliste pour la Recherche d'Information dans des Documents Structurés**",** XXIV ème Congrès Informatique des Organisations et Systèmes d'information et de décision, INFORSID'06, Vol. I, p. 275-290, 2006,

[19] F.Z. BESSAI, M. BOUGHANEM, Z. ALIMAZIGHI, **"**Vers un modèle possibiliste pour la recherche d'information dans des documents structurés**"**, Document numérique, Hermès Science, Lavoisier, volume 10, n°1/2007, p.109-130, ISBN 978-2-7462-1969-4.

[20] F.Z. BESSAI, M. BOUGHANEM, Z. ALIMAZIGHI, "Possibility and Necessity Measures for Relevance Assessment", in proceeding of the ACM first Ph. D. Workshop in CIKM (ACM Sixteenth Conference on Information and Knowledge Management), Novembre 2007, Lisbon, Portugal, p.155-162, ISBN 978-1-59593-961-6

[21] F.Z. BESSAI, Z. ALIMAZIGHI**,** "Stockage et Interrogation dans une base de documents XML Hétérogènes**"**, 1ere Conférence Internationale Systèmes d'Information et Intelligence Economique SIIE'2008, p.407-424, Tome II, ISBN 9978-9973-868-20-6, Hammamet, Tunisie, Février 2008.

[22] F.Z. BESSAI-MECHMACHE and Z. ALIMAZIGHI, "Possibilistic Networks for Aggregated Search in XML Documents", in proceedings of International Conference on Information & Communication Systems, ICICS'2011, p.67-72, ISBN 978-1-4507-8208-1, Irbid, Jordan, May 2011.

[23] F.Z. BESSAI-MECHMACHE and Z. ALIMAZIGHI, "Aggregated Search in XML Documents", International Journal of Emerging Technologies in Web Intelligence, JETWI, Vol 4, No 2, p.181-188, ISSN 1798-0461, May 2012.

[24] F. Z. BESSAI-MECHMACHE and Z. ALIMAZIGHI, " Possibilistic Model for Aggregated Search in XML Documents", International Journal of Intelligent Information and Database Systems, IJIIDS, Inderscience Publishers, Vol 6, No 4, p.381-404, ISSN (Online)1751-5866, ISSN (Print)1751-5858, 2012.

[25] C. BORGELT,J. GEBHARDT and R. KRUSE, "Possibilistic graphical models", Computational Intelligence in Data Mining, Courses and Lectures 408, Springer, p.1-68, Wien, 2000.

[26] P. BORLUND and P. INGWERSEN, "Measures of relative relevance and ranked half-life: performance indicators for interactive IR", In Proc. of the International ACM-SIGIR conference, p. 24-28, 1998.

[27] M. BOUGHANEM, "Les Systèmes de Recherche d'Information:d'un modèle classique à un modèle connexionniste", thèse de Doctorat, université de Paul Sabatier, Toulouse (France), 1992.

[28] M. BOUGHANEM, C. Soulé-Dupuy, "A Connexionist Model for Information Retrieval ", p.260-265, DEXA 1992.

[29] M. BOUGHANEM and M.TMAR, "Modèle auto adaptatif de filtrage d'information : apprentissage Incrémental du profil et de la fonction de décision", VSST'2001, Veille Stratégique Scientifique et Technologique, Vol1, p.313-319, Barcelone, 2001.

[30] M. BOUGHANEM and A. BRINI, "Introduction de la gradualité dans le jugement utilisateur", In Actes de la conférence Extraction et Gestion des Connaissances, vol. 17, p.343-348, 2003.

[31] M. BOUGHANEM, K. SAUVAGNAT, "Propositions pour la pondération des termes et l'évaluation de la pertinence des éléments en recherche d'information structurée", CORIA, Organisé par IRIT, Toulouse (France), 2006.

[32] N. BRADLEY, "The XML Companion", Addison Wesley, 3rd édition, 2002.

[33] A. BRINI, M. BOUGHANEM, D. DUBOIS, "A Model for Information Retrieval Based on Possibilistic Networks", String Processing and Information Retrieval (SPIRE 2005), Buenos Aires, LNCS, Springer Verlag, p. 271-282, 2005.

[34] P. BUNEMAN, S. DAVIDSON, G. HILLEBRAND and D. SUCIU, "A query language and optimization techniques for unstructured data", In Proceedings of ACM SIGMOD International Conference on Management of Data, Montréal, p. 505-516, 1996.

[35] D. CARMEL, D. EFRATY, G. LANDAU, Y. MAAREK and Y. MASS, "An extension of the vector space model for querying XML documents via XML fragments", in the

Workshop Notes of the ACM SIGIR'2002 Workshop on XML and IR, Tampere, Finland, Aug 2002.

[36] S. CERI, S. COMAI, E. DAMIANI, P. FRATERNALI, S. PARABOSCHI and L. TANCA, "XML-GL: A graphical language for querying and restructuring WWW Data", in Proceedings Of the 8th Int. WWW Conference, WWW8, p.151-165, Canada, May 1999.

[37] E.CHARNIAK, "Bayesian Networks without Tears", in AI Magazine, p. 51-61, 1991.

[38] D. CHAMBERLIN, J. ROBIE, D. FLORESCU, "Quilt : An XML query language for heterogeneous data sources", in Proceedings of the 3rd International Workshop on World Wide Web and databases, Dallas, USA, p.1-25, 2000.

[39] J. CLARK, S. DEROSE, "XML Path Language(XPath), Version 1.0", Technical report, World Wide Web Consortium (W3C), W3C Recommendation, November 1999.

[40] C. CLEVERDON, "Progress in documentation. Evaluation of information retrieval systems", Journal of Documentation 26, p.55-67, 1970.

[41] S. DEERWESTER, S. DUMAIS, G. FURNAS, T. LANDAUER, and R. HARSHMAN, "Indexing by latex semantic analysis", Journal of the American Society for Information Science, 41(6), p.391-407,1990.

[42] L.DENOYER, "Apprentissage et inférence statistique dans les bases de documents structurés: Application aux corpus de documents textuels", thèse de Doctorat, université de Paris 06, France, 2004.

[43] L. DENOYER, G. WISNIEWSKI, P. GALLINARI, "Document Structure matching for heterogeneous corpora", Proceedings of XML and IR workshop, SIGIR 2004, Sheffield,G.B., 2004.

[44] L. DENOYER, P. GALLINARI, "The Wikipedia XML Corpus", Pre-proceedings of INEX 2006, Dagstuhl, Allemagne, p. 379-384, 2006.

[45] S. DEROSE, E. MALER, and D. ORCHARD, "XML linking language (Xlink), version 1.0.", Technical report, World Wide Web Consortium (W3C), W3C Recommendation, 2001.

[46] D. DUBOIS and H. PRADE, (with the collaboration of H.FARRENY, R.MARTIN-LOUAIRE and C.TESTEMALE), "Possibility theory: an approach to computerized processing of uncertainty", Plenium Press, New York, 1988.

[47] D. DUBOIS, J. LANG, and H. PRADE, "possobilistic logic", In D. GABBAY, C.J. HOGGER and J.A. ROBINSON editors, Handbook of logic in Artificial Intelligence and Logic Programming, Volume 3, p. 439-513, Clarendon Press-Oxford, 1994.

[48] D. DUBOIS and H. PRADE, "Possibility Theory: qualitative and quantitative aspect", In D. Gabbay and P. Smets, editors, Handbook on Defeasible Reasoning and Uncertainty Management Systems, vol.1, p.21-42, Kluwer Academic Publishers, The Netherlands, 1998.

[49] S. DUMAIS. "Latent Semantic Indexing (LSI)", Proceeding of TREC-3, 1995.

[50] D. C. FALLSIDE (IBM), "XML Schema", W3C Recommendation, 2001.

[51] M.FULLER, E.MACKIE, R.SACKS-DAVIS, and R.WILKINSON, "Structural answers for a large structured document collection", In Proceedings of ACM SIGIR'93, Pittsburgh, p.204-213, 1993.

[52] N.FUHR, "Language Models and Uncertain Inference in Information Retrieval", In Proceeding of the Language Modelling and IR workshop, p. 6-11, 2001.

[53] N. FUHR, N. GOVERT, G. KAZAI, M. LALMAS, "Proceedings of the first workshop of the INitiative for the Evaluation of XML Retrieval (INEX 2002)", Dagstuhl, Germany, 2002.

[54] N. FUHR, K.GROSSJOHANN, "XIRQL: a query language for Information retrieval in XML documents", In Proceedings of SIGIR 2001, New Orleans, USA, 2001.

[55] N. FUHR, M. LALMAS, S. MALIK, "INEX 2003 Workshop proceedings", Dagstuhl, Germany, 2003.

[56] G. GARDARIN, "XML : Des bases de données aux services web", édition DUNOD, Paris, 2002.

[57] E. GAUSSIER et M. STEFANI, "Assistance intelligence à la recherche d'information", Hermes Sciences Publisher, 2003.

[58] J.GEBHARDT and R. KRUSE, "Background and perspectives of possibilistic graphical models", in proceeding of European Conference of Symbolic and Quantitative Approaches to Reasoning and Uncertainty, p.108-121, Germany, 1997.

[59] M.GERY, "Indexation et interrogation de chemins de lecture en contexte pour la recherche d'information structurée sur le web", thèse de Doctorat, université de Joseph Fourrier, Grenoble I, France, 2002.

[60] C. GOLDFARB, "The SGML Handbook", Oxford University Press, Oxford, 1990.

[61] N. GOVERT, M. ABOLHASSANI, N. FUHR, K. GROSSJOHANN, "Content-oriented XML retrieval with HyReX", In Proceedings of the first INEX Workshop, p. 26-32, Dagstuhl, Germany, 2002.

[62] N. GOVERT, G. KAZAI, N. FUHR, M. LALMAS, "Evaluating the effectiveness of content-oriented XML retrieval", Technical Report Computer Science 6, University of Dortmund, 2003.

[63] T. GRABS, H.J. SCHECK, "Flexible information retrieval from XML with PowerDB XML", in Proceedings in the First INEX Workshop, p. 26-32, 2002.

[64] T.GRABS, "Storage and Retrieval of XML Documents within a Cluster of Database Systems", PhD thesis, Ecole Polytechnique Fédérale de Zürich, 2003.

[65] P.GROSSO, E.MALER, J. MARSH, and N. WALSH, "XML pointer language (Xpointer)", Technical report, World Wide Web Consortium, W3C Recommendation, 2003.

[66] S.P. HARTER, "A probabilistic approach to automatic keyword indexing. part ii. An algorithm for probabilistic indexing", Journal of the American Society for Information Science (JASIS), 35(3) :280–289, 1975.

[67] E.R.HAROLD & W.SCOTT MEANS, "XML in a nutshell, manuel de référence", 2e édition en français, Editions O'REILLY, ISBN = 2-84177-223-3, 2002.

[68] K. HATANO, H. KINUTANI, M. YOSHIKAWA and S. UEMURA, "Information Retrieval System for XML Documents", Database and Expert Systems Applications, ISBN 978-3-540-44126-7, p. 5-33, 2002.

[69] Y.HAYASHI, J. TOMITA, G. KIKOI, "Searching text-rich XML documents with relevance ranking", In Proceeding ACM SIGIR 2000 Workshop on XML and IR, p. 27-35, Athens, 2000.

[70] D.HIEMSTRA, "Term specific smoothing for the language modelling approach to information retrieval: The importance of a query term", In Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval, Finland, p.35-41, 2002.

[71] Y. HUANG, Z. LIU and Y. CHEN, "Query biased snippet generation in XML search", ACM SIGMOD, p. 315-326, 2008.

[72] D. HUNTER, C. CAGLE, D. GIBBONS, N. OZU, J. PINNOCK et P. SPENCE, "Initiation à XML avec 3 études de cas détaillées", Editions Eyrolles, ISBN = 2-212-09248-2, 2001.

[73] M. IHADJADENE, "Méthodes avancées pour les systèmes de recherche d'information", Hermes Sciences Publisher, 2004.

[74] INEX, Initiative for the Evaluation of XML Retrieval: http://inex.is.informatik.uni-duisburg.de

[75] K.JÄRVELIN and J. KEKÄLÄINEN, "Cumulated gain-based evaluation of IR Techniques", ACM Transactions on Information Systems, 20(4): p. 422–446, 2002.

[76] F.V. JENSEN, "An Introduction to Bayesian Netwroks", UCL Press, 1996.

[77] S. KACI, "Connaissances et Préférences : Représentation et Fusion en Loqique possibiliste", thèse de Doctorat, université de Paul Sabatier, Toulouse III, France, 2002.

[78] J. KAMPS, M. MARX, M. De RIJKE, B. SIGURBJÖRNSSON, "XML Retrieval: What to retrieve?", ACM SIGIR Conference on Research and Development in Information Retrieval, p. 409-410, 2003.

[79] J. KAMPS, M.RIJKE and B. SIGURBJORNSSON, "Length normalization in XML retrieval", In Proceedings of SIGIR 2004, Sheffield (England), p. 80–87, 2004.

[80] G. KAZAI, M. LALMAS, T. ROELLEKE, "Focused document retrieval", 9th International Symposium on string processing and information retrieval, Lisbonne, Portugal, Septembre 2002.

[81] G. KAZAI, M. LALMAS, A.P. DE VRIES, "The overlap problem in Content-oriented XML retrieval evaluation", Proceedings of SIGIR 2004, Sheffield, Angleterre, p.72-79, 2004.

[82] G. KAZAI and M. LALMAS, "INEX 2005 Evaluation Metrics", In Proceedings of INEX 2005 Workshop, p. 401-406, November 2005.

[83] G. KAZAI, "Choice of Parameter Values for the INEX Evaluation Metrics: Sensitivity Analysis", In Proceedings of INEX 2006 Workshop, 2006.

[84] J. KEKALAINEN and K. JARVELIN, "Evaluating information retrieval systems under the challenges of interaction and multidimensional dynamic relevance", In Bruce, H., Fidel, R., P. Ingwersen, P. Vakkari, eds. Emerging Frameworks and Methods, Seattle, Colorado: Libraries Unlimited, p. 253-270, 2002.

[85] A. KOPLIKU, "Aggregated Search: From information nuggets to aggregated documents", CORIA, p. 507-514, 2009.

[86] A. KOPLIKU, F. DAMAK, K. PINEL-SAUVAGNAT and M. BOUGHANEM, "A user study to evaluate aggregated search", In IEEE/WIC/ACM International Conference on Web Intelligence (WIC 2011), France, 2011.

[87] A. KOPLIKU, M. BOUGHANEM and K. PINEL-SAUVAGNAT, "Searching within Web pages: Retrieving attributes from HTML tables in the Web" in *Conference on Information and Knowledge Management (CIKM 2011)*, Glasgow, United Kingdom, 2011.

[88] R.KOSOKA, C.G. HEALEY, V. INTERRANTE, D.H. LAIDLAW and C. WARE "User studies: Why, how and when?", IEEE Computer Graphics and Applications, 23(4):20-25, 2003.

[89] K. L. KWOK, "A neural network for probabilistic information retrieval", 12th International ACM SIGIR Conference on Research and Development in Information Retrieval, p. 21-30, 1989.

[90] K.L. KWOK, "A network approach to probabilistic information retrieval", ACM transactions on information systems, p. 324-353, 1995.

[91] M. LALMAS, "Dempster-Shafer's Theory of Evidence Applied to Structured Documents: Modelling Uncertainty", In Proceedings of the 20th Annual International ACM SIGIR, p. 110-118, Philadelphia, PA, USA, 1997.

[92] M. LALMAS, P. VANNOORENBERGHE, "Indexation et recherche de documents XML par les fonctions de croyance", Première Conférence en Recherche d'Information et Applications, CORIA'2004, p. 143-160, 2004.

[93] C. LELOUP, "Moteurs d'indexation et de recherche", Edition Eyrolles, 1998.

[94] A. LEVY, M. FERNANDEZ, D. SUCIU, D. FLORESCU, A. DEUTSCH, "XML-QL: A query language for XML", World Wide Web Consortium technical report, Number NOTE-xml-ql-19980819, 1998.

[95] H. LUHN, "The Automatic creation of literature abstracts", IBM Journal of Research and development, 2(2), p.159-165, 1958.

[96] R.W.LUK, H.LEONG, T.S.DILLON, A.T.SHAN, W.B.CROFT, J.ALLAN, "A survey in indexing and searching XML documents", Journal of the American Society for Information Science and Technology, vol. 53, n°3, p. 415-435, 2002.

[97] M.MARON and J.KUHNS, "On relevance, probabilistic indexing and information retrieval", Journal of the Association for Computing Machinery, p. 216–244, 1960.

[98] Y. MASS, M. MANDELBROD, "Retrieving the most relevant XML components", Proceedings of INEX 2003, Dagstuhl, Allemagne, 2003.

[99] L. MIGNET, D. BARBOSA and P.VELTRI, "The XML web: a first study", In Proceedings of WWW2003, Budapest, Hungary, 2003.

[100] S. MIZZARO, "Relevance, the whole history", Journal of the American society for information science, 48(9), p. 810–832, 1997.

[101] J. MOTHE, “Modèle Connexionniste pour la Recherche d’Information, Expansion dirigée de requêtes et apprentissage”, thèse de Doctorat, université de Paul Sabatier, Toulouse (France), 1994.

[102] P. OGILVIE and J. CALLAN, “Using Language Models for Flat Text Queries in XML Retrieval”, In Proceedings of INEX 2003 Workshop, Allemagne, p. 12-18, 2003.

[103] J. PEHCEVSKI, “Relevance in XML retrieval: the user perspective”, Proceedings of the SIGIR 2006 Wokshop on XML Element Retrieval Methodology, Seattle, Etats-Unis, 2006.

[104] J. PEHCEVSKI, J.A.THOM, A.M. VERCOUSTRE, “XML Retrieval Evaluation Revisited: Comparison of Metrics”, Pre-proceedings of INEX ‘06, Germany, 2006.

[105] K. PINEL-SAUVAGNAT, M. BOUGHANEM, “Propositions pour la pondération des termes et l’évaluation de la pertinence des éléments en recherche d’information structurée”, Information - Interaction - Intelligence, vol. 6, n°2, Cépaduès Editions, 2006.

[106] B. PIWOWARSKI, L. DENOYER, P. GALLINARI, “Un modèle pour la recherche d’information sur des documents structurés”, JADT 2002: 6es Journées internationales d’Analyse statistique des Données Textuelles, Saint-Malo (France), Mars 2002.

[107] B.PIWOWARSKI, G.E.FAURE, P.GALLINARI, “Bayesian networks and INEX”, In INEX 2002 Workshop Proceedings, p. 149-153, Germany, 2002.

[108] B. PIWOWARSKI, “Techniques d’apprentissage pour le traitement d’informations structurées : application à la recherche d’information”, thèse de Doctorat, Université de Paris 6 (France), 2003.

[109] B. PIWOWARSKI, “Working group report: the assessment tool”, In Proceedings of INEX’03, Germany, p. 181-183, 2003.

[110] B. PIWOWARSKI and M. LALMAS, “Interface pour l’évaluation de systèmes de recherche sur des documents XML”, Actes de CORIA 2004, Toulouse, France, p.109-121, 2004.

[111] B. PIWOWARSKI, “EPRUM metrics and INEX 2005”, In Proceedings of INEX 2005 Workshop, p. 30-42, November 2005.

[112] B. PIWOWARSKI, “INEX 2006 Relevance Assessment Guide”, Pre-proceedings of INEX 2006, Dagstuhl, Germany, p. 401- 408, 2006.

[113] N. POLYZOTIS and M. N. GAROFALAKIS, “XCluster synopses for structured XML content”, ICDE, p. 63, 2006

[114] J. PONTE and W.CROFT, “A language modelling approach to information retrieval”, In Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, p. 40–48, 1998.

[115] M.F. PORTER, “An algorithm for suffix stripping”, Program 14, 1980.

[116] D.RADEV, J.OTTERBACHER, A.WINKEL and S.BLAIR-GOLDENSOHN, “NewsInEssence: summarizing online news topics”, Communications of the Association of Computing Machinery, p. 95-98, 2005.

[117] C.J.V. RIJSBERGEN, “Information retrieval”, Butterworth, ISBN 0-408-70929-4, 1979.

[118] J. ROBIE, J. LAPP, D. SCHACH, “XML Query Language (XQL)”, Proceedings of W3C QL’98 (Query Languages 98), Massachusetts, 1998.

[119] J.ROCCHIO, “Relevance feedback in information retrieval”, Prentice Hall Inc, Englewood Cliffs, NJ, 1971.

[120] T.ROELLEKE, M. LALMAS, G. KAZAI, J. RUTHVEN, S. QUICKER, “The accessibility dimension for structured document retrieval”, BCS-IRSG European Conference on Information Retrieval (ECIR), Glasgow, Mars 2002.

[121] C. ROUSSEY, “Une méthode d’indexation sémantique adaptée aux corpus multilingues”, thèse de Doctorat, Institut des Sciences Appliquées de Lyon (France), 2001.

[122] G. SALTON, “The SMART retrieval system: Experiments in automatic document processing”, Prentice Hall, 1970.

[123] G. SALTON and M. McGill, “Introduction to modern information retrieval”, Mc Graw-Hill Int. Book Co, 1983.

[124] T. SARACEVIC, “Introduction to information science, the concept of relevance”, p. 111-151, R.R.Bowker Company, New York, 1970.

[125] K. SAUVAGNAT, “Modèle flexible pour la recherche d’information dans des corpus semi structurés”, thèse de Doctorat, Université Paul Sabatier, Toulouse (France), 2005.

[126] K. SAUVAGNAT, M. BOUGHANEM, “A la recherche de nœuds informatifs dans des corpus de documents XML - Ou pourquoi on a toujours besoin de plus petit que soi... ”, Actes de CORIA 05, Grenoble, France, 2005.

[127] K. SAUVAGNAT, L. HLAOUA, M. BOUGHANEM, “XFIRM at INEX 2005: ad-hoc and relevance feedback tracks”, in proceedings of INEX 2005 Workshop, pp: 72-83, 2005.

[128] K. SAUVAGNAT, M. BOUGHANEM, C. CHRISMENT, "Answering content-and-structure-based queries on XML documents using relevance propagation", Information Systems, Special Issue SPIRE 2004, vol. 31, p. 621–635, Elsevier, 2006.

[129] K. SAUVAGNAT, L. HLAOUA, M. BOUGHANEM, "XML retrieval: what about using contextual relevance?", ACM Symposium on Applied Computing (SAC)-IAR (Information Access and Retrieval) , Dijon, 2006.

[130] T.SCHLIEDER, H. MEUSS, "Querying and Ranking XML Documents", Journal of the American Society for Information Science and Technology, 53(6): 489-503, 2002.

[131] W. SHAW, R.BURGIN, and P. HOWELL, "Performance standards and evaluations in IR test collections: Cluster-based retrieval models", Information Processing and Management, 33(1):1–14, 1997.

[132] SIGIR: Special Interest Group of Information Retrieval: http://www.acm.org/sigs/sigir/

[133] B. SIGURBJORNSSON, J. KAMPS, and M. RIJKE, "An element-based approach to XML retrieval", In Proceedings of INEX 2003 workshop, p. 19-26, Dagstuhl, Germany, 2003.

[134] B. SIGURBJÖRNSSON, B. LARSEN, M. LALMAS, S. MAALIK, "INEX04 Guidelines for topic development", Pre-proceedings of INEX 2005, Dagstuhl, Allemagne, p. 212-218, 2004.

[135] A.F.SMEATON, "Information retrieval and natural language processing", In proceedings of a conference jointly sponsored by ASLIB, University of York, p. 2, Mars 1989.

[136] F. SONG and W.B. CROFT, "A general language model for information retrieval", In Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, p. 279–280, 1999.

[137] T. STRZALKOWSKI, "Natural language processing in large-scale text retrieval tasks", In Text REtrieval Conference (TREC-1), p.173, 1993.

[138] S. SUSHMITA and M. LALMAS, "Using digest pages to increase user result space: preliminary designs", Special Interest Group on Information Retrieval (SIGIR) 2008 Workshop on Aggregated Search, 2008.

[139] S. SUSHMITA, H. JOHO and M. LALMAS, "A task-based evaluation of an aggregated search interface", In SPIRE, p. 322-333, 2009.

[140] X. TANNIER, "Extraction et recherche d'information en langage naturel dans les documents semi structurés", thèse de Doctorat, Ecole Nationale Supérieure des Mines de Saint-Etienne et de l'Université Jean Monnet (France), 2006.

[141] H.TEBRI, “Formalisation et spécification d’un système de filtrage incrémental d’information”, thèse de Doctorat, université de Paul Sabatier, Toulouse (France), 2004.

[142] A.THEOBALD, G.WEIKUM, “The Index-Based XXL Search Engine for Querying XML Data with Relevance Ranking”, EDBT 2002, 8th International Conference on Extending Database Technology, Prague, p. 477-495, 2002.

[143] A. TROTMAN and B. SIGURBJÖRNSSON, “Narrowed Extended XPath I (NEXI)”, In INEX 2003 proceedings, Dagstuhl, Allemagne, p. 219-237, December 2004.

[144] A.TROTMAN and B. SIGURBJÖRNSSON, “NEXI, now and next “, In INEX 2003 proceedings, Dagstuhl, Allemagne, p. 10-15, December 2004.

[145] A.TROTMAN, “Choosing document structure weights”, Information Processing and Management, vol. 41, n°2, p. 243-264, 2005.

[146] H. TURTLE and W. CROFT, “Inference networks for document retrieval”, In Proceedings of ACM SIGIR 90, p.1-24, 1990.

[147] H. TURTLE, “Inference Networks for Document Retrieval”, thèse de Doctorat, université de Massachussetts, Amhesrt, 1991.

[148] J.N. VITTAUT, B. PIWOWARSKI, P. GALLINARI, “An algebra for Structured Queries in Bayesian Networks”, INEX 2004 Pre-proceedings, Dagstuhl, Allemagne, p. 58-65, 2004.

[149] B.M. WILDEMUTH, “Why conduct user studies? The role of empirical evidence in improving the practice of librarianship”, In INFORUM ’03*:* 9th Conference on Professional Resources, Prague, 2003.

[150] J. WOLFF, H. FLÖRKE, A. CREMERS, “Searching and browsing collections of structural information”, Proceedings of IEEE advances in digital libraries, Washington, p. 141-150, 2000.

[151] W3C, “eXtensible Markup Language (XML)1.0.”, Technical report, World Wide Web Consortium (W3C), Technical report, February 1998.

[152] W3C, “DOM Level1 (Document Object Model)”, Technical report, World Wide Web Consortium, W3C standard, October 1998.

[153] W3C, “XML Schema”, Technical report, World Wide Web Consortium, W3C Recommendation, 2001.

[154] W3C, “eXtensible Stylesheet Language (XSL), version 1.0”, Technical report, World Wide Web Consortium,W3C Recommendation, October 2001.

[155] W3C, “XQuery: A Query Language for XML”, World Wide Web Consortium, W3C Working Draft, February 2001.

[156] W3C, “XQuery and XPath Full-Text Use Cases”, Report, World Wide Web Consortium, W3C working draft, February 2003.

[157] W3C, “Namespaces in XML 1.0 (Second Edition)”, World Wide Web Consortium, W3C Recommendation, August 2006.

[158] S.YOO, “An XML Retrieval Model based on structural proximities”, INEX 2002 Workshop Proceedings, Dagstuhl, Allemagne, p. 60-64, 2002.

[159] L. A. ZADEH, “Fuzzy sets”, Information and control, 8: 338–353, 1965.

[160] L.A.ZADEH, “Fuzzy sets as a basis for a theory of possibility”, Fuzzy Sets and Systems 1:3–28, 1978.

[161] H. ZARGAYOUNA, “Contexte et sémantique pour une indexation de documents semi structurés”, CORIA, IRIT, Toulouse (France), 2004.

[162] G. ZIPF, “Human Behaviour and the Principle of Least Effort”, Addison-Wesley, 1949.

Annexe A

Réalisation et mise en œuvre du prototype MPRIX

1 Introduction

Le prototype MPRIX est composé de deux principaux modules :

- Module indexation : analyse un document XML afin de vérifier sa validité, puis extrait les descripteurs d'index lui correspondant, ces derniers sont sauvegardés dans la base d'index.
- Module appariement : assure le processus de sélection de l'information pertinente à renvoyer comme réponse à la requête utilisateur, il utilise les résultats du module indexation.

Nous présenterons alors, dans la première partie de ce chapitre, l'ensemble des outils de développement utilisés. Dans la seconde partie, nous détaillerons le processus d'implémentation du prototype MPRIX, ainsi que les principales interfaces qui le composent.

2 Environnement de développement

Dans cette partie, nous présentons les outils utilisés comme base pour notre développement

2.1 Système d'exploitation

Notre prototype a été développé sous le système d'exploitation Windows XP, mais comme il est développé en langage java, il peut être intégré dans n'importe quel autre système d'exploitation supportant la machine virtuelle java (Windows 98/00, Linux, …).

2.2 Langage de programmation

Le langage java est un langage de programmation orienté objet mis au point par Sun Microsystems. Sa caractéristique principale est qu'il est indépendant de toute plate forme, il est possible d'exécuter des programmes java sur tous les environnements qui possèdent une Java Virtual Machine (JVM), ce concept est à la base du slogan de Sun pour java : WORA (Write Once, Run Anywhere : écrire une fois, exécuter partout). Sun fournit aussi gratuitement un ensemble d'outils et d'API pour permettre le développement de programmes avec ce langage, ce kit est nommé JDK (Java Development Kit).

Java est caractérisé aussi par la réutilisabilité de son code ainsi que la simplicité de sa mise en œuvre.

Pour le développement de notre prototype, nous avons choisi le langage Java. De part ces qualités en tant que langage évolué, java permet un traitement simple et efficace des documents XML en fournissant des API spécialisées qui sont DOM et SAX. Elles sont fournies dans un ensemble de package connu sous le nom JAXP (Java API for XML Parsing).

Le prototype MPRIX utilise L'API DOM, qui est basée sur un traitement hiérarchique, l'analyseur construit une structure hiérarchique contenant des objets représentant les éléments du document, et dont les méthodes permettent d'accéder aux propriétés (parcours, mise à jour).

- L'interface Java/JDBC :

Pour se connecter à la base Index, nous avons utilisé l'interface JDBC. JDBC (acronyme qui signifie "Java DataBase Connectivity"), est un ensemble de classes Java qui permettent de se connecter à une base de données, généralement distante sur le réseau, et d'interroger cette base afin d'en extraire des données. La principale caractéristique de JDBC est le fait qu'elle est complètement indépendante de tout SGBD, la même application peut être utilisée pour accéder à une base ORACLE, PostgreSQL, MySQL, etc. Par Conséquent, on n'a pas besoin d'apprendre une nouvelle API quand on change de système, d'où la réutilisation totale du code.

2.3 PostgreSQL

PostgreSQL est un SGBDR (Système de Gestion de Base de Données Relationnelles) fonctionnant sur diverses plates-formes matérielles sous différents systèmes d'exploitation.

L'une des principales qualités de PostgreSQL est d'être un logiciel libre, c'est-à-dire gratuit et open source (les sources du logiciel sont disponibles dans http://www.postgresl.org).

Il possède de nombreuses caractéristiques qui font de lui un SGBDR robuste et puissant et qui sont comme suit :

- Il reconnaît la plupart des spécifications SQL.
- Il intègre des bibliothèques pour de nombreux langages, afin de permettre d'accéder aux enregistrements à partir de programmes écrits en : Java (JDBC), langage C/C++, Perl…etc.
- Il inclut l'API ODBC permettant à n'importe quelle application supportant ce type d'interface d'accéder à des bases de données de type PostgreSQL.

▪ Il peut être étendu de plusieurs façons par l'utilisateur en ajoutant par exemple de nouveaux types de données ou de nouvelles fonctions d'indexation (grâce à sa particularité d'être open source).

▪ Permet la gestion des langages de procédure en interne, il intègre d'ailleurs un langage natif, appelé PL/pgSQL, comparable au langage de procédure d'Oracle PL/SQL.

Pour tirer parti des avantages offerts par ce SGBDR, la base Index de notre système de recherche est implémentée en PosgreSQL version 8.0.

3 Implémentation du prototype MPRIX

Les principales classes réalisées, ainsi que les principales méthodes qui les définissent, pour l'implémentation du prototype MPRIX sont résumées dans les deux tableaux suivants :

Module indexation	
Classes	**Méthodes Principales**
AnalyseXML	**VerifierValidite() :** vérifier la validité et construit l'arbre DOM associé au document XML donné en entrée.
	CodifierElement() : ajoute l'attribut id_elt à l'arbre DOM du document XML, cet attribut constitut l'identifiant d'un élément, il est calculé de façon à sauvegarder le chemin menant de la racine vers cet élément.
Indexation	**ExtractionContenu() :** Extrait le contenu d'un document donné en entrée sous sa forme d'arbre DOM, les résultats sont sauvegardés dans la base Index.
	EliminerSeparateur() : transforme une chaîne de caractères en une chaîne composée de mots séparés uniquement par des blancs.
	EliminerMotvide() : élimine les mots considérés vides, ces mots apparaissent dans la liste des mots vides déjà définie.
	Extraire_attribut() : extrait les attributs d'un document, donné en entrée sous forme d'arbre DOM.
PorterStemmer	**Stem() :** retourne le lemme d'un mot donné.

Stopliste	**Stopliste() :** crée une nouvelle instance de la classe Stopliste qui contient la liste des mots vides définis.
Pondération	**Terme_Document() :** retourne la liste des termes du document qui seront pondérés.
	Remonter_Terme() : calcule les différentes mesures, qui servent pour le calcul des poids des termes, et remonte les termes au niveau des éléments du document.

Tableau A.1 - *Classes et méthodes principales correspondant au module indexation*

Module appariement	
Classes	**Méthodes Principales**
Document	**charger_elts() :** construit le réseau possibiliste associé au document donné en entrée.
Enreg_elt	**enreg_elt() :** construit la liste des mots appartenant à un élément
Recherche	**retourner_elts() :** retourne la liste des éléments du document contenant les termes de la requête.
Resultat_doc	**Calcul_Poss() :** retourne la liste des configurations des éléments répondant à la requête. à toute configuration, il lui est associé deux mesures : possibilité et nécessité de pertinence.
Rech_Attribut	**recuperer_requete () :** analyse la requête de l'utilisateur, qui se porte uniquement sur les attributs, afin qu'elle soit prise en compte dans le processus de recherche sur les attributs.
	Ens_elt() : retourne l'ensemble des éléments qui contiennent les attributs spécifiés (avec leurs valeurs) dans la requête se portant sur les attributs.

Tableau A.2 - *Classes et méthodes principales correspondant au module appariement*

4 Description

L'interface homme/machine représente l'élément clé dans l'utilisation de tout système informatique. Les interfaces du prototype MPRIX sont simples, conviviales et d'utilisation

facile.

4.1 Interface principale

L'interface illustrée par la figure A.1 ci-dessous représente l'interface de la tâche principale du système MPRIX, qui est la recherche d'information structurée de type XML.

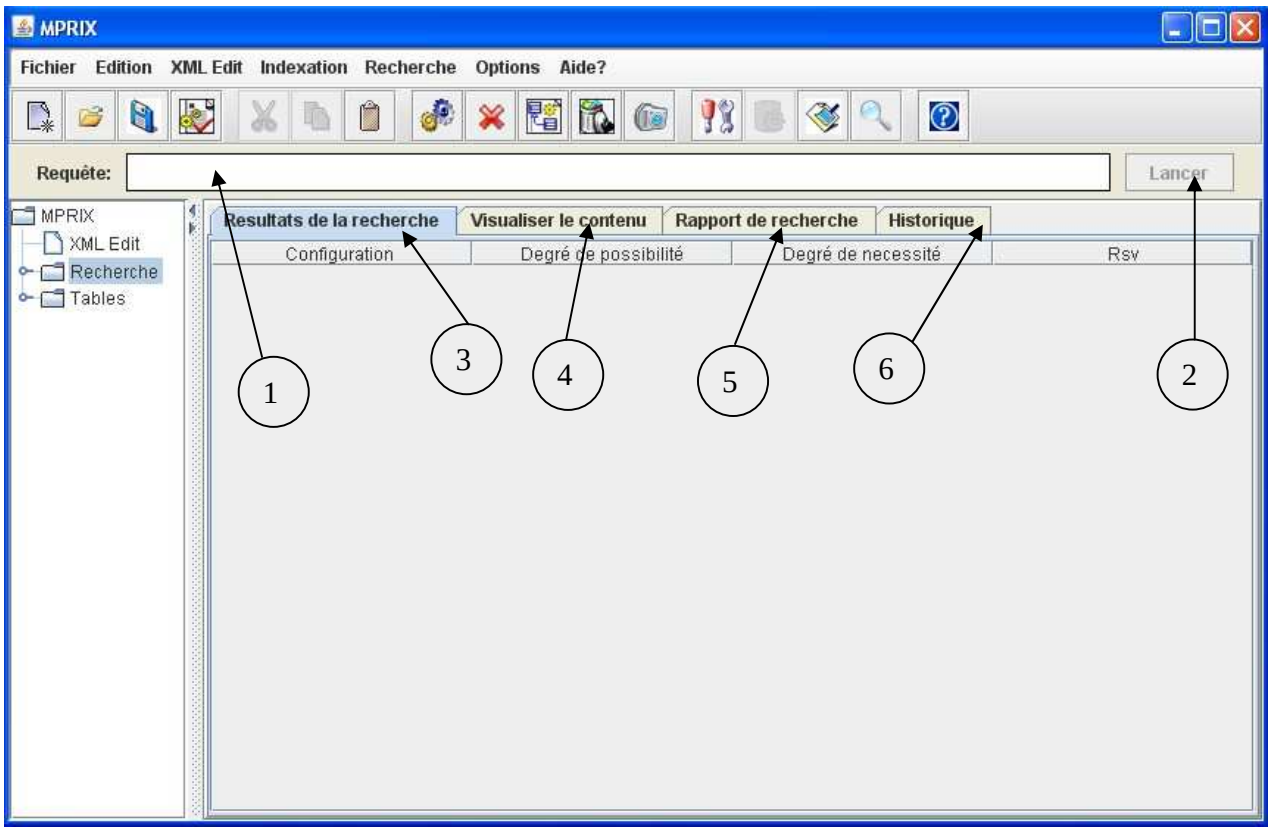


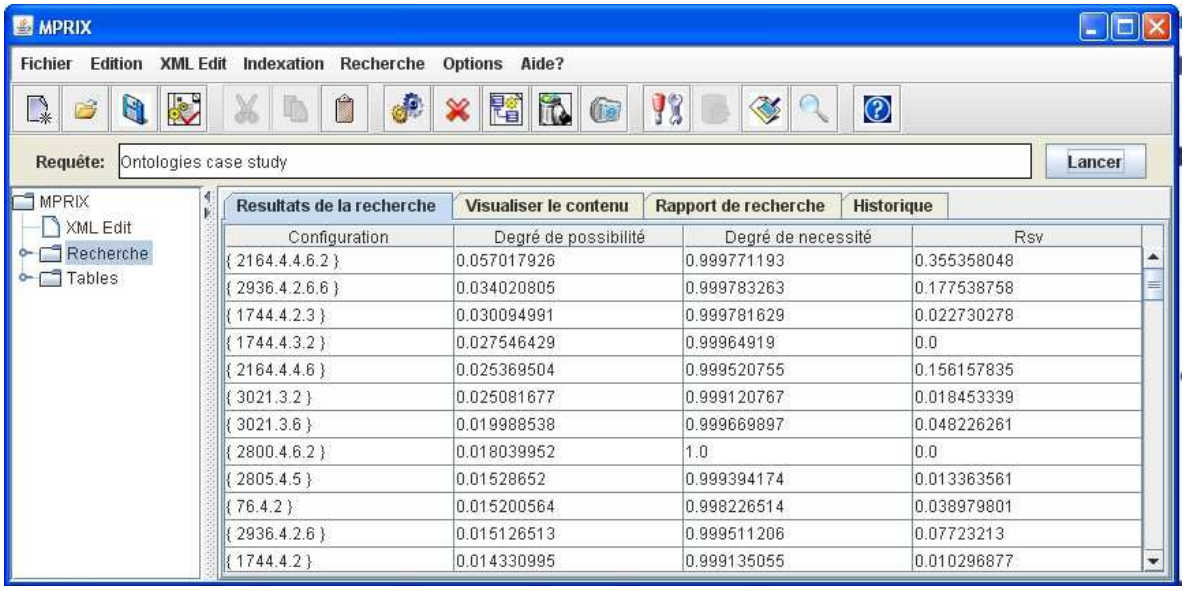
Figure A.1 *- Interface de recherche du prototype MPRIX*

Fonctionnement :

(1) Zone d'édition de requête

(2) Lancement de la recherche

(3) L'onglet Résultats de la recherche, contient les configurations des éléments retournés selon leur ordre de pertinence (possibilité et nécessité de pertinence) (voir figure A.2).



Configuration	Degré de possibilité	Degré de necessité	Rsv
{ 2164.4.4.6.2 }	0.057017926	0.999771193	0.355358048
{ 2936.4.2.6.6 }	0.034020805	0.999783263	0.177538758
{ 1744.4.2.3 }	0.030094991	0.999781629	0.022730278
{ 1744.4.3.2 }	0.027546429	0.99964919	0.0
{ 2164.4.4.6 }	0.025369504	0.999520755	0.156157835
{ 3021.3.2 }	0.025081677	0.999120767	0.018453339
{ 3021.3.6 }	0.019988538	0.999669897	0.048226261
{ 2800.4.6.2 }	0.018039952	1.0	0.0
{ 2805.4.5 }	0.01528652	0.999394174	0.013363561
{ 76.4.2 }	0.015200564	0.998226514	0.038979801
{ 2936.4.2.6 }	0.015126513	0.999511206	0.07723213
{ 1744.4.2 }	0.014330995	0.999135055	0.010296877

Figure A.2 *- Onglet Résultats de recherche*

(4) L'onglet Visualiser le contenu, permet à l'utilisateur de consulter le contenu de chaque unité retournée. En effet, en cliquant sur une configuration dans l'onglet ''Résultats de recherche'', Un document XML est alors créé contenant le contenu et les informations associées à la configuration comme montré dans la figure A.3 suivante :

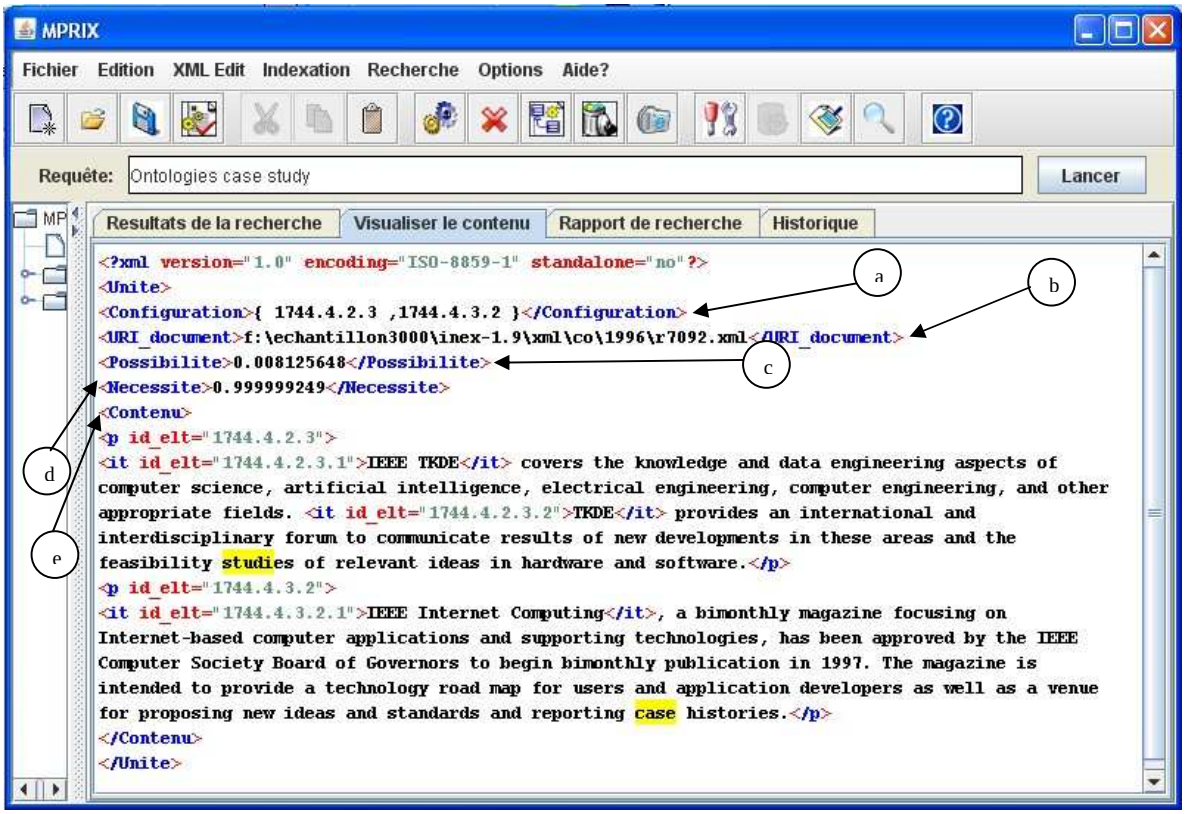


Figure A.3 *- Onglet Visualiser le contenu*

La configuration visualisée contient :

(a) La liste des identifiants des éléments constituant cette unité d'information retournée.

(b) L'emplacement du document concerné par la configuration visualisée.

(c) Le degré de possibilité de la configuration visualisée.

(d) Le degré de nécessité de la configuration visualisée.

(e) Le contenu textuel des éléments retournés.

(5) L'onglet Rapport de recherche (voir figure A.4), contient le rapport de recherche de la dernière requête, tel que le nombre total d'unités retournées, le nombre d'unités retournées ayant un degré de nécessité non nul, le temps pris dans le processus de recherche en seconde, etc.

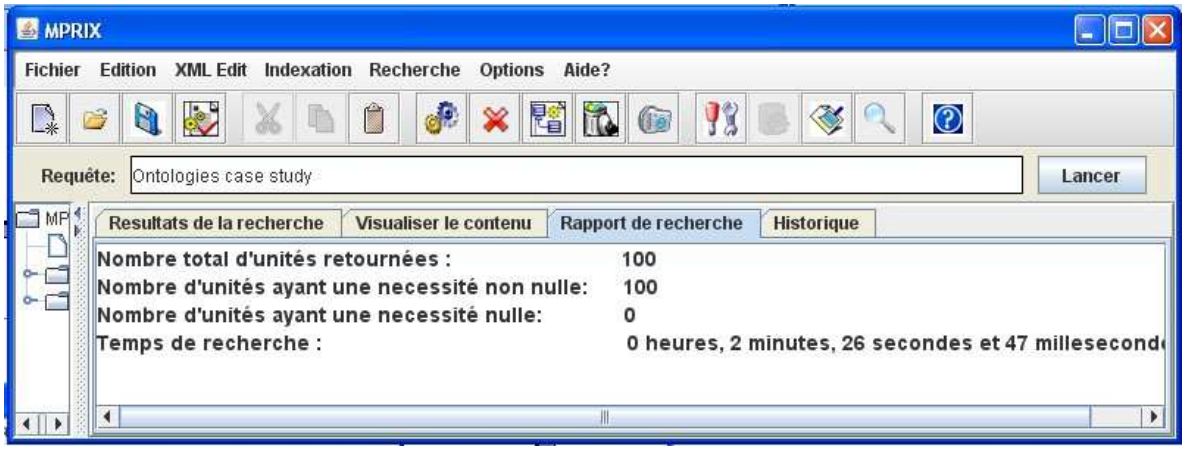


Figure A.4 *- Onglet Rapport de recherche*

(6) L'onglet Historique, garde trace des requêtes précédemment soumises au système de recherche.

Figure A.5 *- Onglet Historique*

4.2 Description de la barre des Menus

La barre des Menus présente les différentes fonctionnalités du système MPRIX tels que l'indexation, la recherche et le traitement textuel des documents XML, à travers les menus Fichier, Indexation, Recherche et XML Edit.

4.2.1 Menu fichier

Figure A.6 - *Menu fichier*

1. Connexion : permet la connexion à la base Index, en spécifiant les différents paramètres relatifs à la base de données, comme c'est illustré dans la fenêtre de connexion de la figure A.7 suivante:

Figure A.7 - *Connexion à la base Index*

2. Déconnexion : permet à l'utilisateur de fermer la connexion déjà établie avec la base de données.

3. Quitter : permet de sortir de l'application.

4.2.2 Menu Indexation

L'interface d'indexation est résumée dans le menu Indexation de la figure A.8 suivante:

Figure A.8 - *Menu Indexation*

1. Vider Index : permet de vider la base Index en supprimant les descripteurs des documents indexés, le vidage est réalisé après confirmation de cette action, comme c'est montré dans la figure A.9 suivante :

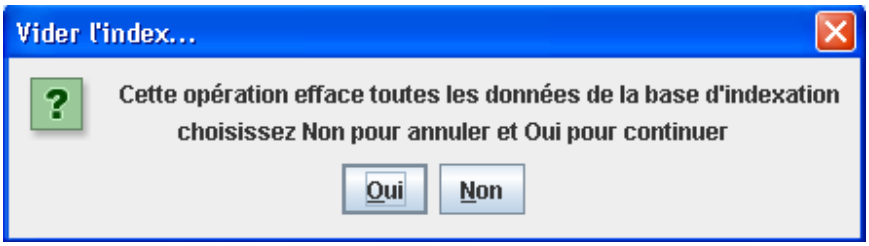


Figure A.9 *- Confirmation de l'action «Vider index »*

Figure A.10 *- Vidage de la base Index*

2. Indexation : permet d'ajouter de nouveaux documents à la base Index, l'ajout peut être un document ou bien une collection de documents comme le montre la figure A.11**.**

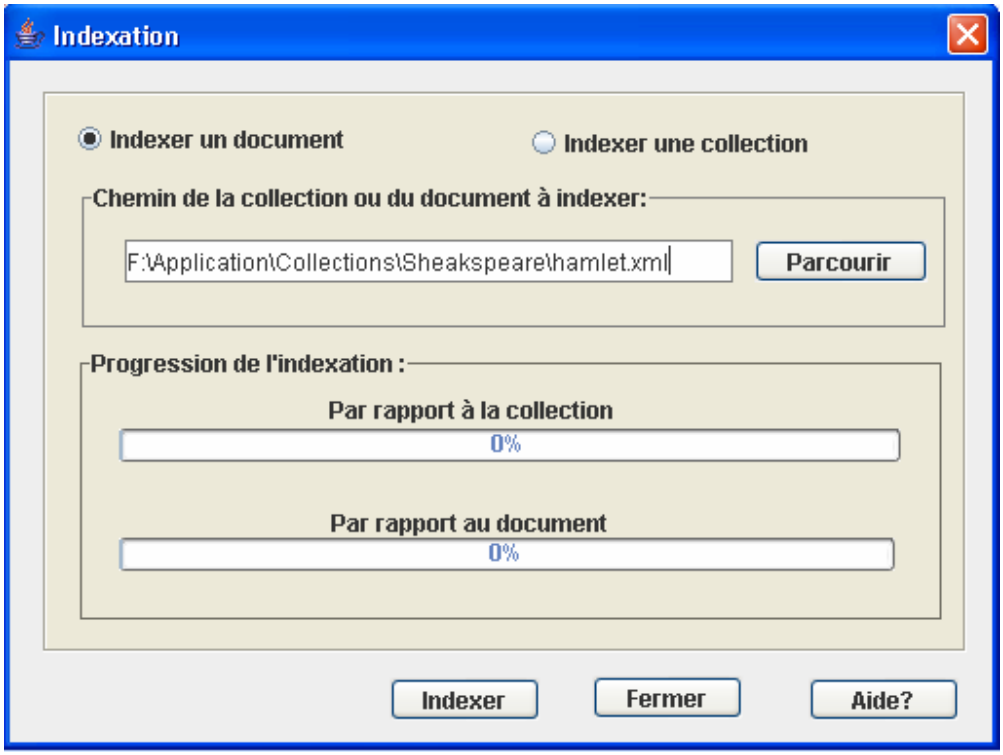


Figure A.11 *- Fenêtre d'indexation d'un document ou d'une collection de documents XML*

3. Supprimer : permet la suppression d'un document déjà indexé, par la spécification de son chemin de sauvegarde, comme le montre la fenêtre de la figure A.12 suivante **:**

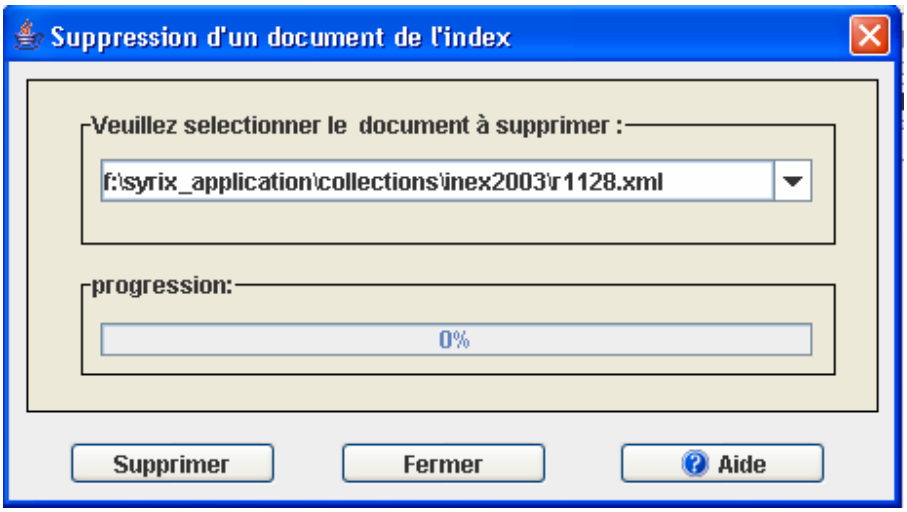


Figure A.12 *- Fenêtre de suppression d'un document dans la base d'indexation*

4. Pondération : elle permet de calculer le poids des termes des documents indexés, elle permet d'associer à chaque terme d'un document son poids au niveau de l'élément où il apparaît. La pondération est réalisée séquentiellement document par document, en sélectionnant à chaque fois le document qui sera pondéré.

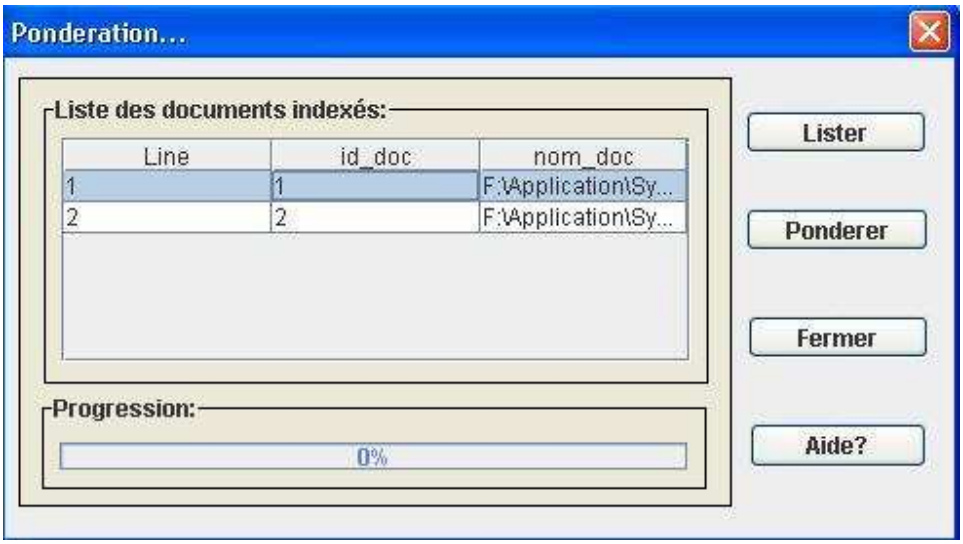
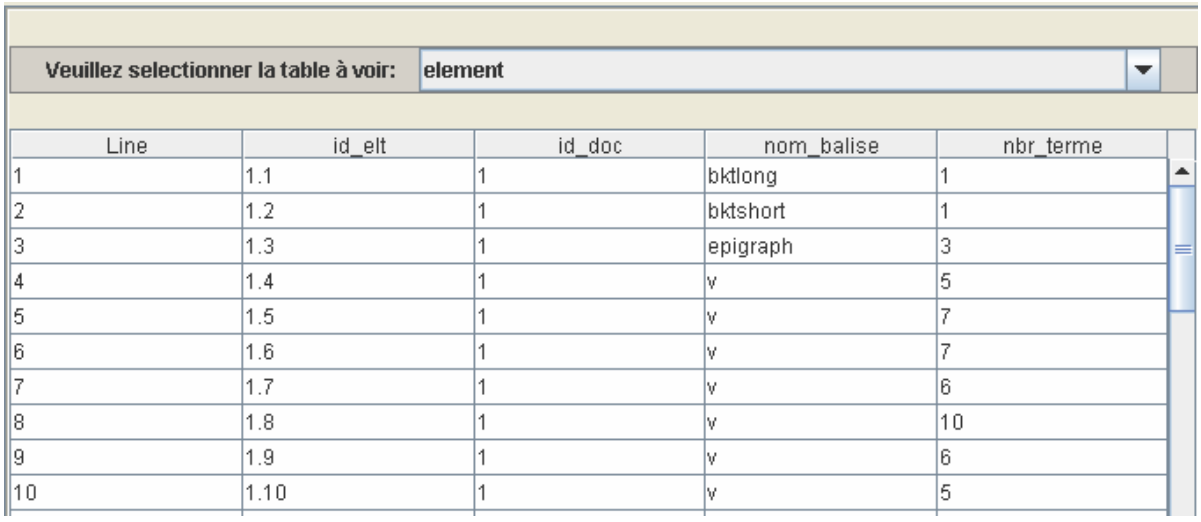


Figure A.13 *- Fenêtre de la pondération*

5. Voir Index : permet de visualiser le contenu des tables composant la base Index. L'accès est direct en sélectionnant le nom de la table à visualiser (voir figure A.14)



Line	id_elt	id_doc	nom_balise	nbr_terme
1	1.1	1	bktlong	1
2	1.2	1	bktshort	1
3	1.3	1	epigraph	3
4	1.4	1	v	5
5	1.5	1	v	7
6	1.6	1	v	7
7	1.7	1	v	6
8	1.8	1	v	10
9	1.9	1	v	6
10	1.10	1	v	5

Figure A.14 *- Visualisation du contenu de la table élément*

4.2.3 Menu Recherche

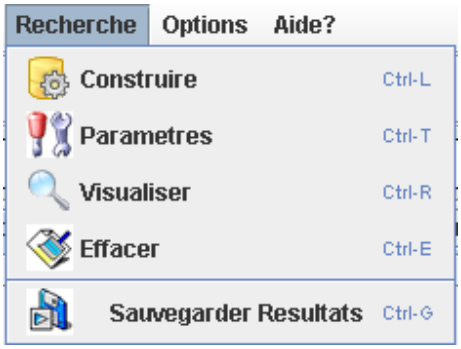


Figure A.15 *- Menu Recherche*

1. **Paramètres :** permet de spécifier les paramètres de la recherche

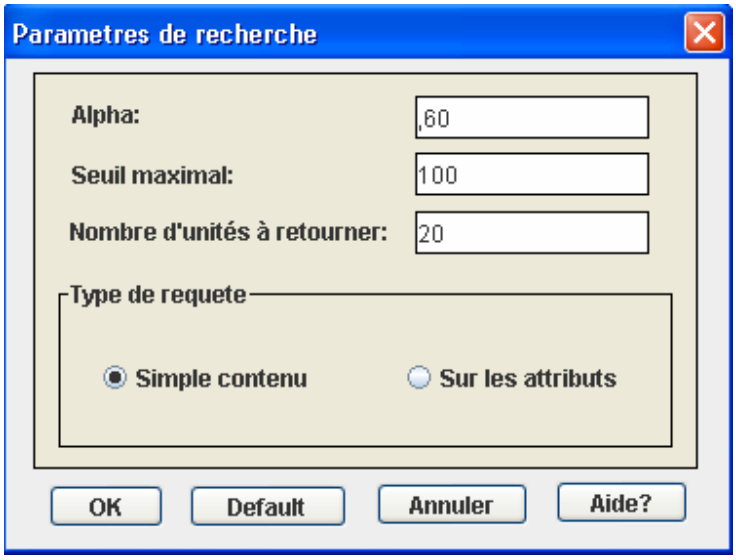


Figure A.16 *- Fenêtre paramètres de recherche*

Avec :

- **Alpha :** paramètre permettant de quantifier l'importance de la distance séparant les nœuds balises (éléments structurels du document) à la racine du document.
- **Seuil maximal :** le nombre maximum de configurations à prendre en considération lors de l'appariement.

- **Nombre d'unités à retourner :** pour limiter le nombre maximum d'unités à retourner lors de l'appariement.
- **Type de requête :** spécifier le type de requête, la recherche peut être par mots clés ou bien une recherche sur attributs.

2. **Visualiser :** Visualise le contenu d'une configuration sélectionnée parmi les configurations retournées comme réponse à la requête.
3. **Effacer :** Efface toutes les zones d'affichage qui concerne la recherche précédente.
4. **Sauvegarder Resultats :** Sauvegarde de l'unité d'information actuelle affichée dans l'onglet «Visualiser le contenu».

4.2.4 Menu XML Edit

XML Edit est un simple éditeur de traitement de documents XML qui nous permet de créer de nouveaux documents XML. Il sert aussi à afficher ou modifier des documents XML. Enfin, il permet de vérifier la validité d'un document XML.

***Figure A.17** - Menu XML Edit*

1. **Nouveau :** créer un nouveau document XML
2. **Ouvrir :** ouvrir un document XML en sélectionnant son emplacement
3. **Enregistrer :** enregistrer le document XML courant
4. **Enregistrer sous :** enregistrer le document XML courant dans un autre emplacement spécifié par l'utilisateur

5. **Analyser :** Analyse le document XML actuel, pour vérifier sa conformité ou sa validité (si une DTD lui est associée).

Voici dans la figure suivante, un exemple de traitement d'un fichier XML à l'aide de cet éditeur :

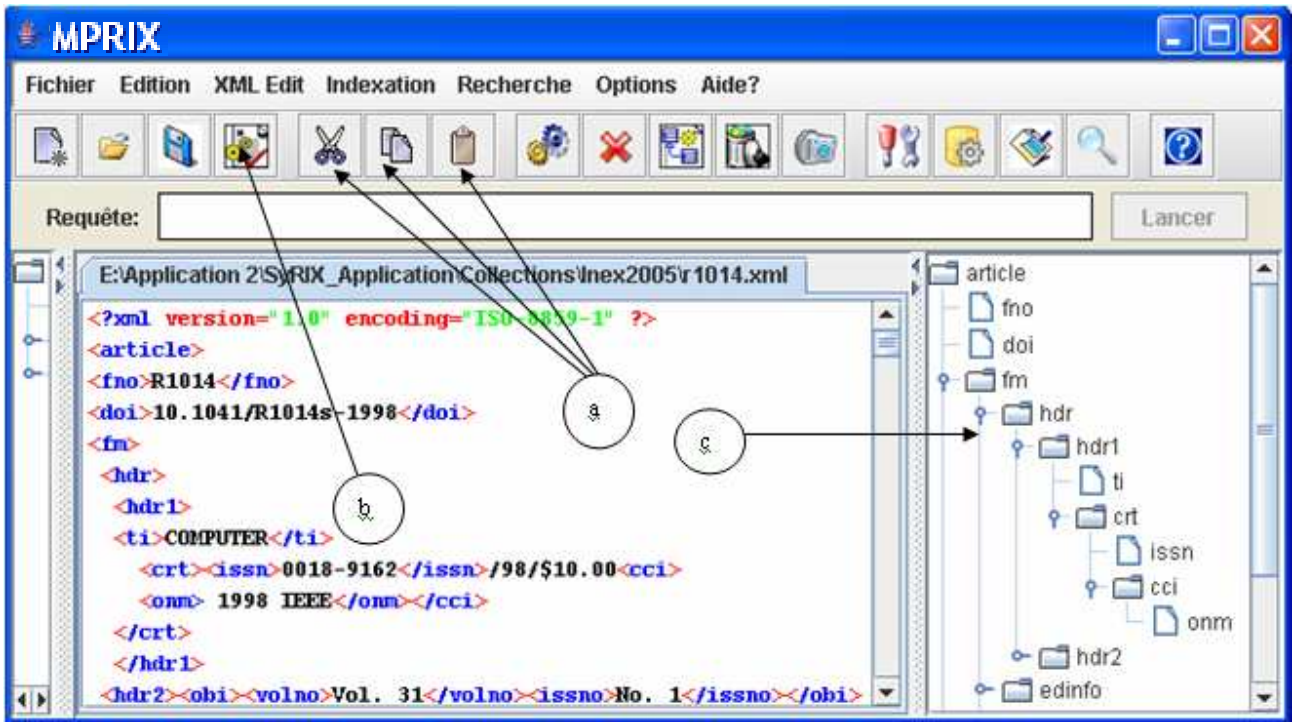


Figure A.18 *- Onglet Edit XML*

(a) Boutons de la barre d'édition.

(b) Boutons d'analyse du document XML.

(c) Arbre associé au document après analyse (dans le cas où le document est valide).

Annexe B

Questionnaire pour participant 1

Notre travail se situe dans le contexte de la recherche d'information et plus particulièrement la recherche d'information dans des documents semi structurés de type XML. Actuellement, la recherche d'information a évolué de l'accès à un document ou un ensemble de documents vers l'accès à des parties (éléments) de documents XML répondant à un intérêt particulier de l'utilisateur. Dans cette optique, nous proposons un nouveau modèle de recherche d'information dans les documents XML qui permet, contrairement à tous les modèles de la littérature, d'agréger dans un même résultat des éléments complémentaires, non redondants et indépendants provenant de différentes parties du document XML.

Pour valider notre modèle de recherche d'information structurée, le prototype MPRIX a été implémenté. Ce prototype présente une interface conviviale pour l'interrogation et la présentation des résultats à l'utilisateur. Cette interface permet une formulation simple de requête par contenu (mots clés) et présente les résultats sous forme d'une liste triée, par ordre de pertinence, d'agrégats (groupe d'éléments non redondants et indépendants) répondants au mieux au besoin de l'utilisateur.

L'objectif de ce questionnaire est d'évaluer la recherche agrégée c'est-à-dire voir si le résultat sous forme d'agrégat aide à améliorer la pertinence du résultat de recherche. Par conséquent, dans la suite du questionnaire les questions posées concerneront l'agrégat le plus complet (celui qui contient le plus grand nombre d'éléments) et ces disjonctions.

Pour l'évaluation, vous disposez de six (6) requêtes composées chacune de mots clés. Chaque requête sera soumise au système MPRIX et une fois le résultat de la recherche obtenu, vous avez sept (7) questions auxquelles vous devrez répondre. L'analyse de vos réponses nous permettra d'évaluer le modèle MPRIX et plus particulièrement l'apport de l'agrégation du résultat pour la recherche d'information.

Merci pour votre contribution et votre aide à l'évaluation de notre travail.

Nom : Prénom :

Spécialité : Grade :

Requête Q1 : Software engineering

1- Les éléments de l'agrégat sont-ils redondants ?

Oui :

Non :

2- Les éléments de l'agrégat sont-ils indépendants. Autrement dit, sont-ils des éléments qui proviennent généralement de différentes parties (éléments frères et/ou éléments de parents différents) du document XML?

Oui :

Non :

3- Par rapport à la requête, l'agrégat est-il?

Pertinent:

Partiellement pertinent:

Non pertinent:

4- Si on veut prendre qu'une partie de l'agrégat résultat. Par rapport à ce dernier, cette partie est-elle :

Pertinente :

Partiellement pertinente :

Non pertinente :

5- Trouvez-vous que le résultat sous forme d'agrégat contient des informations complémentaires qui permettent d'avoir une vue plus complète sur le résultat de la recherche.

Oui :

Non :

6- Trouvez-vous que le résultat sous forme d'agrégat a aidé à améliorer le résultat de la recherche.

Oui :

Non :

7- Avez-vous des commentaires à faire ?

Requête Q4 : Ontologies case study

1- Les éléments de l'agrégat sont-ils redondants ?

Oui :

Non :

2- Les éléments de l'agrégat sont-ils indépendants. Autrement dit, sont-ils des éléments qui proviennent généralement de différentes parties (éléments frères et/ou éléments de parents différents) du document XML?

Oui :

Non :

3- Par rapport à la requête, l'agrégat est-il?

Pertinent:

Partiellement pertinent:

Non pertinent:

4- Si on veut prendre qu'une partie de l'agrégat résultat. Par rapport à ce dernier, cette partie est-elle :

Pertinente :

Partiellement pertinente :

Non pertinente :

5- Trouvez-vous que le résultat sous forme d'agrégat contient des informations complémentaires qui permettent d'avoir une vue plus complète sur le résultat de la recherche.

Oui :

Non :

6- Trouvez-vous que le résultat sous forme d'agrégat a aidé à améliorer le résultat de la recherche.

Oui :

Non :

7- Avez-vous des commentaires à faire ?

Requête Q7: user-centered design of web sites

1- Les éléments de l'agrégat sont-ils redondants ?

Oui :

Non :

2- Les éléments de l'agrégat sont-ils indépendants. Autrement dit, sont-ils des éléments qui proviennent généralement de différentes parties (éléments frères et/ou éléments de parents différents) du document XML?

Oui :

Non :

3- Par rapport à la requête, l'agrégat est-il?

Pertinent:

Partiellement pertinent:

Non pertinent:

4- Si on veut prendre qu'une partie de l'agrégat résultat. Par rapport à ce dernier, cette partie est-elle :

Pertinente :

Partiellement pertinente :

Non pertinente :

5- Trouvez-vous que le résultat sous forme d'agrégat contient des informations complémentaires qui permettent d'avoir une vue plus complète sur le résultat de la recherche.

Oui :

Non :

6- Trouvez-vous que le résultat sous forme d'agrégat a aidé à améliorer le résultat de la recherche.

Oui :

Non :

7- Avez-vous des commentaires à faire ?

Requête Q11: electronic commerce business strategies

1- Les éléments de l'agrégat sont-ils redondants ?

Oui :

Non :

2- Les éléments de l'agrégat sont-ils indépendants. Autrement dit, sont-ils des éléments qui proviennent généralement de différentes parties (éléments frères et/ou éléments de parents différents) du document XML?

Oui :

Non :

3- Par rapport à la requête, l'agrégat est-il?

Pertinent:

Partiellement pertinent:

Non pertinent:

4- Si on veut prendre qu'une partie de l'agrégat résultat. Par rapport à ce dernier, cette partie est-elle :

Pertinente :

Partiellement pertinente :

Non pertinente :

5- Trouvez-vous que le résultat sous forme d'agrégat contient des informations complémentaires qui permettent d'avoir une vue plus complète sur le résultat de la recherche.

Oui :

Non :

6- Trouvez-vous que le résultat sous forme d'agrégat a aidé à améliorer le résultat de la recherche.

Oui :

Non :

7- Avez-vous des commentaires à faire ?

Requête Q14: brain research differential geometry

1- Les éléments de l'agrégat sont-ils redondants ?

Oui :

Non :

2- Les éléments de l'agrégat sont-ils indépendants. Autrement dit, sont-ils des éléments qui proviennent généralement de différentes parties (éléments frères et/ou éléments de parents différents) du document XML?

Oui :

Non :

3- Par rapport à la requête, l'agrégat est-il?

Pertinent:

Partiellement pertinent:

Non pertinent:

4- Si on veut prendre qu'une partie de l'agrégat résultat. Par rapport à ce dernier, cette partie est-elle :

Pertinente :

Partiellement pertinente :

Non pertinente :

5- Trouvez-vous que le résultat sous forme d'agrégat contient des informations complémentaires qui permettent d'avoir une vue plus complète sur le résultat de la recherche.

Oui :

Non :

6- Trouvez-vous que le résultat sous forme d'agrégat a aidé à améliorer le résultat de la recherche.

Oui :

Non :

7- Avez-vous des commentaires à faire ?

Requête Q17: call for papers conference workshop multimedia

1- Les éléments de l'agrégat sont-ils redondants ?

Oui :

Non :

2- Les éléments de l'agrégat sont-ils indépendants. Autrement dit, sont-ils des éléments qui proviennent généralement de différentes parties (éléments frères et/ou éléments de parents différents) du document XML?

Oui :

Non :

3- Par rapport à la requête, l'agrégat est-il?

Pertinent:

Partiellement pertinent:

Non pertinent:

4- Si on veut prendre qu'une partie de l'agrégat résultat. Par rapport à ce dernier, cette partie est-elle :

Pertinente :

Partiellement pertinente :

Non pertinente :

5- Trouvez-vous que le résultat sous forme d'agrégat contient des informations complémentaires qui permettent d'avoir une vue plus complète sur le résultat de la recherche.

Oui :

Non :

6- Trouvez-vous que le résultat sous forme d'agrégat a aidé à améliorer le résultat de la recherche.

Oui :

Non :

7- Avez-vous des commentaires à faire ?