**REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE**
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
**UNIVERSITE MOULOUD MAMMERI DE TIZI-OUZOU**
FACULTE DE GENIE ELECTRIQUE ET INFORMATIQUE
**DEPARTEMENT INFORMATIQUE**

# Mémoire de fin d'études

**En vue de l'obtention du diplôme de**

# Master

**Domaine : Informatique**

**Option : Systèmes Informatiques**

Thème

# PROPOSITION ET EXPERIMENTATION D'UN MODELE DE LANGUE POUR LA RECHERCHE D'INFORMATION DANS LES DOCUMENTS XML

**Réalisé par :** Mlle Thiziri BELKACEM

**Proposé et dirigé par** : Mme Samia FELLAG

Soutenue publiquement le: *Juin 2015*

**Année universitaire : 2014/2015**

# Remerciements

Je remercie tout d'abord le bon Dieu qui m'a aidé et donné la volonté pour la réalisation de ce travail ;

Mes vifs remerciements accompagnés de tout mon profond respect s'adressent à mon premier exemple dans mes études, ma promotrice Mme S.Fellag pour son suivi et son engagement lors d'élaboration de ce travail, je la remercie pour les orientations et les conseils qui m'ont été efficaces ;

Je remercie vivement les membres du jury qui nous font l'honneur d'accepter d'évaluer ce travail ;

Finalement, je remercie tout ceux qui m'ont aidé et ont contribué de prés ou de loin à la réalisation de ce travail.

Merci.

# Dédicace

*Je dédie ce travaille :*

*A ceux qui m'ont donné tout et qui font le bonheur de ma vie, à ceux qui m'ont appris à aller au bout de mes rêves et qui ont fait de moi ce que je suis maintenant, à mes chers parents ;*

*A la mémoire de mes oncles paternels, les martyrs Rabah et Moh N'Amirouche ;*

*A la mémoire de mes grands parents maternels ;*

*A la mémoire de mon oncle maternel Nouredine ;*

*A la mémoire de mon grand père paternel ;*

*A ma grand-mère paternelle que dieu la garde ;*

*A ma chère grande sœur : Ouardia, sa belle famille et sa chère petite fille la mignonne Liza ;*

*A mes chers et adorables sœurs : Lila et Lynda*

*A mes chers frères : Amirouche, Moh N'Ammirouche et notre petit Rabah*

*A ma promotrice Mme S. Fellag ;*

*A mes chers(es) amis(es) ;*

*Et tous ceux qui m'aiment.*

# LISTE DES FIGURES

**Chapitre I : La recherche d'information et la structure**



**Chapitre III : Un modèle de langue pour la RIS dans les documents XML**

**Chapitre IV : Expérimentation et Résultats**

# LISTE DES TABLEAUX

**Chapitre I : La recherche d'information et la structure**



**Chapitre II : Le modèle de langue**



**Chapitre III : Un modèle de langue pour la RIS dans les documents XML**



**Chapitre IV : Expérimentation et résultats**

# SOMMAIRE

# *Sommaire*

# INTRODUCTION GENERALE

## Introduction générale :

Depuis son existence, l'homme a pensé à laisser trace de ses accomplissements ou de son passage. Cet instinct s'est développé de plus en plus et avec l'apparition des premières sciences, aux siècles de lumière, les savants procédaient à préserver leurs recherches et découvertes dans des papiers ou des planches qui servirons par la suite comme connaissances, ou information nouvelle pour d'autres. Par la suite, les bibliothèques et les encyclopédies apparaissent donnant naissance à ce qui est aujourd'hui une science : « La recherche d'information ». Avec l'apparition et la popularisation des ordinateurs, des documents électroniques de différents types, des supports de stockage et des réseaux de communication d'information apparaissent bouleversant les liens entre l'Homme et l'information, et pour que le premier accède au deuxième, il procède à sa recherche par tous les moyens qui lui sont offerts et qui sont mis à sa disposition.

Abrégée en RI, la recherche d'information est un domaine informatique qui date des années 1940, dès la naissance des ordinateurs, dont les premières applications étaient destinées pour la bibliothéconomie, d'où le nom « Automatisation des Bibliothèques », dont le but est de satisfaire le besoin de l'utilisateur en information, ce qu'on appelle la pertinence du résultat et dans les années 1950, on commençait de petites expérimentations en utilisant des petites collections de documents (références bibliographiques). Des expérimentations plus larges ont été menées, et une méthodologie d'évaluation du système a été développée avec des corpus de test qui par la suite, ont beaucoup collaboré à l'avancement de la RI. A partir de 1995 : les années de l'Internet, la problématique de pertinence s'est élargie. Par exemple, on traite maintenant plus souvent des documents multimédia qu'avant. Cependant, les techniques de base utilisées dans les moteurs de recherche sur le web restent identiques. Ce dernier a fait émerger des nouveaux formats de documents traités par les Systèmes de Recherche d'Information (SRI), ces derniers sont passés de la RI classique ne traitant que des documents textes dits plats, à la RI structurée ou semi-structurée traitant les documents à formats variables et structurés, tel que le format XML, qui est flexible, standard et utilisé par les différents SRI structurés.

La RI pour sa part, a subit un développement approprié qui a fait apparaitre une grande taxonomie de modèles de recherche qui servent à donner des présentations mathématiques pour les concepts de la RI, à savoir : le document, la requête, leur contenu et la liaison entre un document et une requête (similarité), cette grande taxonomie est résumée dans 3 grandes classes :

- Les modèles probabilistes
- Les modèles algébriques
- Les modèles ensemblistes

S'intéressant à la première classe (des modèles probabilistes), un modèle très utilisé est le modèle de langue, qui est une dérivée du modèle probabiliste de base, dont les concepts de la RI sont pris en compte dans des probabilités paramétrées, traitant les caractéristiques d'un

document contenant l'information recherchée, d'une collection contenant ce document ainsi que celles de la requête reçue en terme de mots clés.

Dans le cadre de la RIS, indépendamment du modèle RI utilisé, l'indexation et la pertinence des unités d'information retournées doivent prendre en compte la notion de structure du document xml. Pour cela, la nouvelle granularité d'information est prise en compte dans les formules de score de pertinence, car cette dernière cible un élément mais pas le document entier. De ce fait, dans le cadre des modèles probabilistes généralement, et le modèle de langue précisément, de nouveaux paramètres sont intégrés aux probabilités de similarité entre un document ou un élément de celui-ci (dans le cadre de la RIS) et une requête utilisateur pour satisfaire son besoin en information.

## Contexte du travail :

Ce travail se situe dans le contexte de la recherche d'information structurée dans les documents de format XML. Dans ce contexte, un SRI doit retourner l'élément le plus pertinent à la requête, d'où la nouvelle granularité de l'information tout en obéissant aux exigences de l'utilisateur à savoir le contenu et la structure. Pour cela, deux types de requêtes à traiter : les requêtes CO qui visent le contenu, indépendamment de la structure et les requêtes CAS qui visent le contenu et la structure du résultat recherché. Dans cette optique, plusieurs modèles de la RI classique ont été adaptés à la RI structurée pour les documents XML.

Dans notre contribution, nous nous sommes intéressés au modèle de langue étant une variante du modèle probabiliste de base. Un modèle de langue représente la similarité entre un document est une requête du fait que le modèle du document puisse être généré par celui de la requête. Dans cette vision, nous avons procédé à l'adaptation d'un modèle de langue présenté dans le cadre de la RI classique, à la nouvelle granularité de l'information présentée dans la RIS.

# CHAPITRE I : LA RECHERCHE D'INFORMATIONS ET LA STRUCTURE

## Partie 1 : La recherche d'information traditionnelle

### Introduction :

La recherche d'informations est un domaine informatique ayant pour but la mise en disposition de l'utilisateur un ensemble d'informations, ce dernier doit être capable de satisfaire son besoin en informations et répondre à la majorité de ses requêtes. Pour se faire, un système de recherche d'information se base sur un ensemble de documents qui constituent la collection de recherche ou la base documentaire.

La recherche d'information est un mécanisme qui facilite l'accès à une base d'informations, elle est basée sur le taux de correspondance d'un document à une requête exprimant le besoin en information de l'utilisateur. Qu'appel-t-on un document ? Une requête ? Quels sont les concepts de la RI ? Et quel est le lien entre ces différents concepts ?

Ces différents concepts seront présentés dans le présent chapitre, comme on présentera les modèles piliers de la RI, le mécanisme de recherche avec un SRI ainsi que leur fonctionnement pour satisfaire le besoin en information de l'utilisateur.

### 1. Définition :

Abrégée en RI (Recherche d'Information) ou IR (Information Retrieval), est un domaine, historiquement lié aux sciences de l'information et à la bibliothéconomie qui ont toujours construit le problème de représentation des documents dans le but d'en récupérer des informations recherchées par construction d'index. L'expression « Recherche d'Information » a eu différentes définitions dont :

D'après (l'AFNOR, 1979) [Urfist, 2004], *"La recherche d'information (RI) est un ensemble de méthodes et procédures ayant pour objet d'extraire d'un ensemble de documents, les informations voulues. Dans un sens plus large, la RI est toute opération (ou ensemble d'opérations) ayant pour objet la recherche, la collecte et l'exploitation d'informations en réponse à une question sur un sujet précis"*

D'un point de vue informatique, c'est une branche s'intéressant aux : stockage, organisation et acquisition des informations diverses ;

Du point de vue utilisateur, c'est l'aspect d'acquisition des unités informatives les plus précises à sa requête. [Sauvagnat, 2006]

Nous pouvons résumer que la RI est le domaine informatique qui cherche à mettre en œuvre des outils logiciels, pour garantir un meilleur stockage et organisation, une recherche ciblée, une sélection précise et une acquisition satisfaisante d'un ensemble riche d'informations répondant aux besoins des utilisateurs en utilisant un « Système de Recherche d'Information».

## 2. Système de recherche d'information (SRI) :

### 2.1 Définition :

Un Système de Recherche d'Informations est un outil logiciel utilisé pour retrouver des informations qui répondent à une requête donnée par l'utilisateur, en se basant sur un ensemble de documents.

Un SRI est un ensemble de programmes informatiques coopératifs ayant pour but la sélection des informations dites « pertinentes », répondants aux besoins en information de l'utilisateur, dont le fonctionnement peut être exprimé par le processus de recherche appelé « Processus En U », comme le montre le schéma suivant :

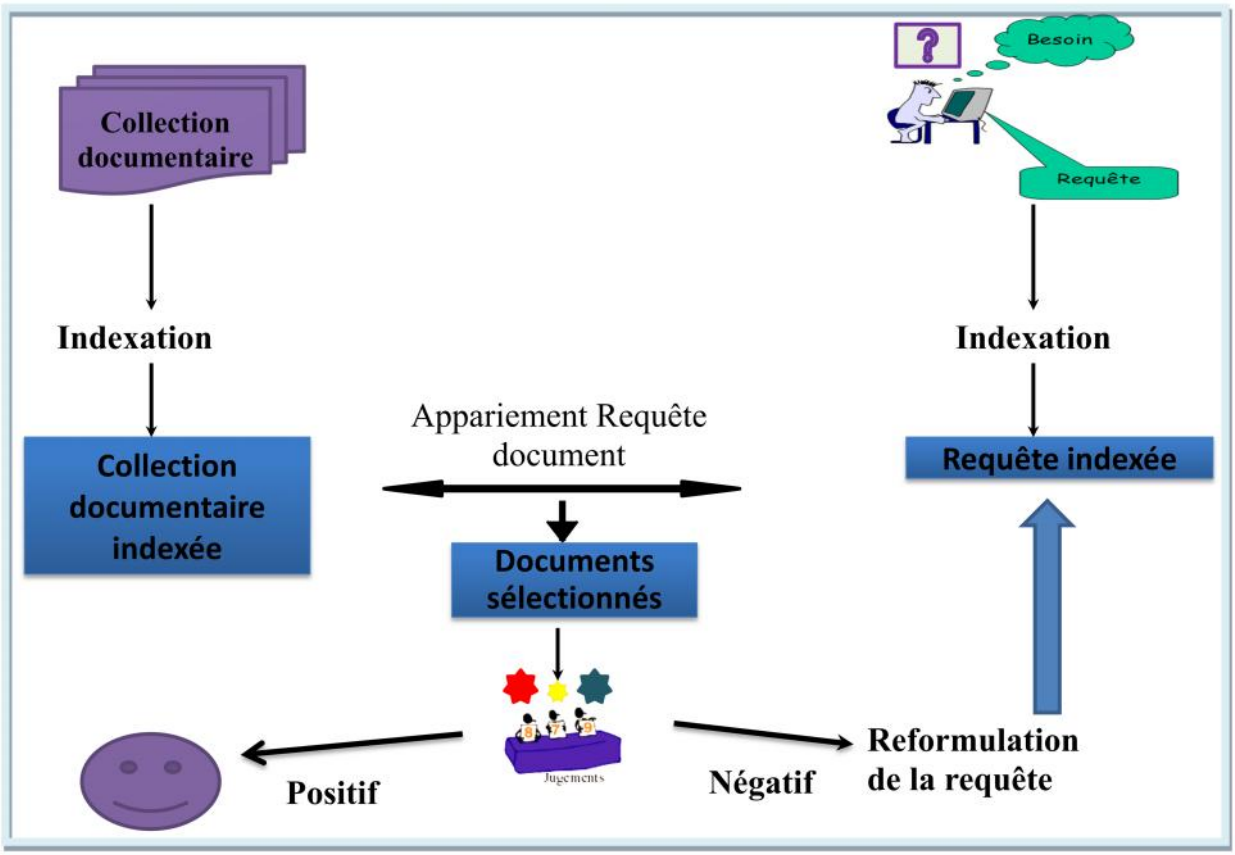



**Figure I-1 :** le processus de fonctionnement en U

### 2.2 Terminologie :

***Document*** **:** C'est l'élément central de tout SRI et un objet en évolution continue, car il est lié aux développements des technologies informatiques, son évolution est motivée par l'évolution du Web.

Suzanne Briet dans [S. Briet, 1951] : *« Un document est une preuve à l'appui d'un fait, [à savoir] tout indice concret ou symbolique, conservé ou enregistré, aux fins de représenter, de reconstituer ou de prouver un phénomène physique ou intellectuel ».*

On peut résumer qu'un document est l'élément primordial d'un SRI, pouvant contenir tout élément porteur de données : paragraphe, texte, image, vidéo, page web…

***Base documentaire :*** Ou *corpus documentaire* est une collection de documents de différentes structures à des contenus diverses, elle sert à fournir une information recherchée par le système en fonction.

***La requête :*** C'est l'expression du besoin de l'utilisateur en information, sous forme d'un ensemble de mots lexicaux regroupés sous différents formats dans un langage d'interrogation. En RI, c'est un ensemble de mots clés issus d'une expression fournie par l'utilisateur, c'est l'élément déclencheur de recherche provenant d'un utilisateur.

***Appariement document/requête :*** C'est la fonction de comparaison entre le document et la requête, elle revient à calculer un score, supposé représentant la pertinence du document vis-à-vis de la requête, ceci en se basant sur un modèle RI bien déterminé **(voir section 4)**. Cette valeur est calculée à partir d'une fonction ou d'une probabilité de similarité notée ***RSV(Q,d)*** *(Retrieval Status Value)*, entre la requête ***Q*** et le document ***d***.

Cette mesure tient compte du poids des termes de la requête dans les documents et retourne pour chacun de ces derniers sa valeur correspondante. La fonction de similarité permet ensuite d'ordonner les documents renvoyés à l'utilisateur. La qualité de cet ordonnancement est primordiale. En effet, l'utilisateur se contente généralement d'examiner les premiers documents renvoyés (les 10 ou 20 premiers). Si les documents recherchés ne sont pas présents dans cette tranche, l'utilisateur considérera le SRI comme mauvais vis-à-vis de sa requête et les informations qu'il retourne ne sont pas pertinentes.

***Notion de pertinence :*** C'est l'objectif d'un SRI pour un document retourné après l'appariement, et est définie dans [Nie, 2004] comme étant une mesure d'informativité d'un document à la requête ou de l'utilité du document pour l'utilisateur.
La pertinence peut être perçue selon deux niveaux : le niveau système et le niveau utilisateur.

*La pertinence système :* est une mesure d'évaluation de la similarité entre le document et la requête, tandis que
*La pertinence utilisateur :* correspond aux jugements utilisateur porté sur un document retourné par le system.

La figure suivante explique ces niveaux de pertinence :

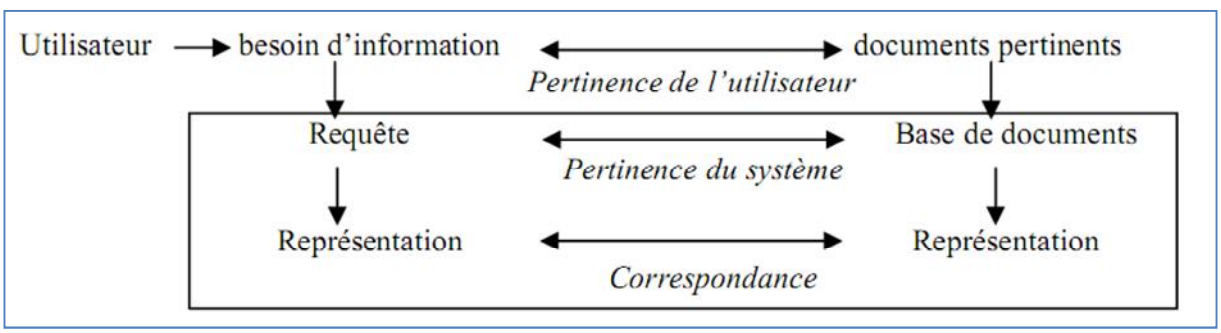



**Figure I-2 :** Illustration des niveaux de pertinence [Nie, 2004]

***La reformulation :*** Comme l'utilisateur exprime son besoin en information sous-forme d'une requête en langage naturel, il est habituel de la soumettre à une reformulation en requêtes SRI plus précise sur laquelle se base le SRI pour retourner des documents plus efficaces aux besoins en informations de l'utilisateur. Cette fonction fait partie du processus d'optimisation du SRI, car elle se base sur des mots de la requête utilisateur et les documents disponibles. On en distingue deux types :

*Reformulation directe :*

Basée sur un thésaurus[1], donc reformulation *Globale*.

Basée sur les résultats de la recherche en cours, donc reformulation *Locale*.

- *Globale*: Ajout des termes issus du thésaurus à base d'une fonction de similarité.
- *Locale* : Usage des informations issues des documents trouvés, terme ou relation associée.

*Reformulation indirecte :* ou par injection de pertinence, est une modification de la requête initiale à base des jugements utilisateur sur la pertinence des documents qui lui sont retournés, tel qu'il s'agit d'un processus itératif sur la requête initiale à laquelle on ajoute les termes issus des documents jugés « pertinents » par l'utilisateur, et de laquelle on restitue les termes appartenant aux documents jugés « non pertinents » par l'utilisateur, pour renforcer l'importance des termes dans cette requête. Selon la formule suivante :

$$Q_1 = \alpha Q_0 + \frac{\beta}{|Dp|}\sum_{D_j \in Dp} D_j - \frac{\delta}{|Dnp|}\sum_{D_j \in Dnp} D_j$$

Tel que : | | : désigne le cardinal d'un ensemble
***$Q_0$*** : Requête initiale
***$Q_1$*** : Nouvelle requête
***Dp*** : Ensemble des documents restitués et jugés pertinents par l'utilisateur
***Dnp*** : Ensemble des documents restitués et jugés non pertinents par l'utilisateur
$\alpha$ =1, $\beta$= 0.5, $\delta$=0.25

**3. L'indexation :**

**3.1. Définition :**

C'est le processus permettant d'extraire d'un document ou d'une requête une représentation paramétrée du contenu sémantique, constituée de mots clés à objectif de minimiser le coût de recherche au niveau d'un SRI et dont le résultat est un descripteur du document ou de la requête. L'index d'un document est une liste de termes recouvrant au mieux son contenu sémantique susceptible d'être comparée avec les requêtes utilisateur.

[1] **Le thésaurus :** est un dictionnaire de mots, représentant les notions de proximité et voisinage entre les mots : synonymie, variante,...

On en cite trois types :

a- *Indexation manuelle :* Il s'agit d'un spécialiste en indexation qui extrait les mots et leurs descripteurs manuellement.
b- *Indexation automatique :* se fait par des systèmes spécialisés dans le domaine.
c- *Indexation semi-automatique :* le système spécialisé retourne les descripteurs qui conviennent et les spécialistes se chargent de choisir ceux qu'ils voient nécessaires.

Le choix d'une méthode doit être fait en fonction du domaine, de la collection et de l'application considérée, car selon [Anderson & Perez-Carballo, 2000] et leur étude comparative faite sur ces types, leurs avantages et inconvénients sont équilibrés.

### 3.2. Processus d'indexation:

Dans les SRI, le processus d'indexation est automatisé et regroupe un ensemble de traitements à effectuer sur un document. On distingue : l'extraction automatique des mots des documents, l'élimination des mots vides, la lemmatisation (radicalisation ou normalisation), le repérage des groupes de mots, la pondération des mots et enfin la création de l'index.

***3.2.1*** ***L'analyse lexicale :*** sert à reconnaitre les mots (séquences de caractères attachés), les caractères de ponctuation et les caractères blancs, puis retourner les termes pouvant être des mots simples (exp : retour) ou composés (exp : retour arrière).

***3.2.2*** ***L'élimination des mots vides*** *:* les mots vides sont les mots les moins significatifs, et qui sont éliminés lors du processus d'indexation par :

- L'utilisation d'un *anti-dictionnaire*, qui est une liste des mots vides d'un document.
- L'élimination des mots dépassant un certain nombre d'occurrences dans la collection.

Le but de cette étape est d'éviter d'avoir des indexes trop longs et une recherche lente.

***3.2.3*** ***La lemmatisation :*** c'est l'opération de réduction de la forme d'un mot à sa racine. Un mot peut être présenté sous différentes formes (exp : mots de la même famille), cette étape permet d'éviter à l'utilisateur de faire introduire les différentes formes du même mot lors d'une recherche (exp : pluriel, singulier ou masculin, féminin.

***3.2.4*** ***La pondération des termes :*** Elle permet de mesurer l'importance d'un terme dans un document, et revient à retrouver les mots qui représentent au mieux le contenu du document. Pour mettre en évidence les diverses contributions d'un terme dans cette représentation, un poids lui est affecté. Cette mesure est souvent calculée en se basant sur des propriétés et interprétations statistiques. D'après Zipf [Zipf, 1949], si on dresse une liste de l'ensemble des mots différents d'un texte quelconque, classés par ordre de fréquences décroissantes, on constate que la fréquence d'un mot, est inversement proportionnelle à son rang de classement dans la liste. Formellement, ceci peut être traduit par la formule suivante :

$$\textit{fréquence} \times \textit{rang} = \textit{constante}$$

La loi de distribution des termes suit alors la courbe présentée dans la figure suivante :

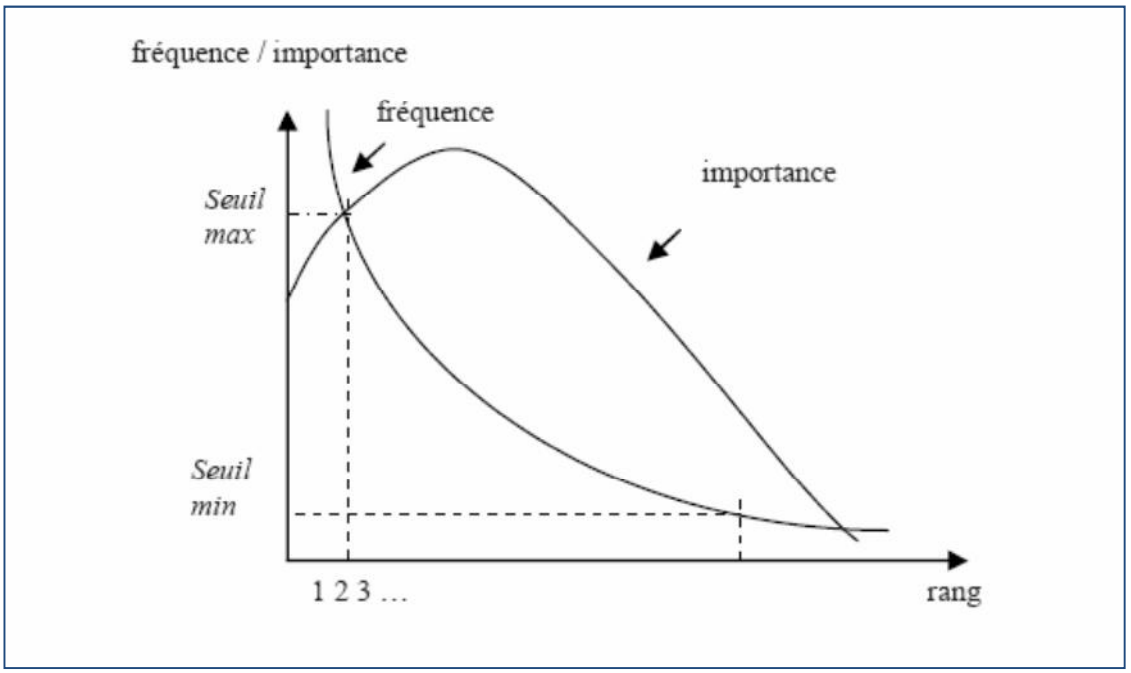



**Figure I-3 :** Importance d'un terme en fonction de sa fréquence d'apparition dans le document

Cela veut dire que le mot à plus grande fréquence et le mot à plus petite fréquence sont à moindre importance. Elle est appelée aussi : *loi de distribution des termes*, elle explique la courbe hyperbolique par ce que George Zipf appel « *Principe du moindre effort* »[2].

Pour une bonne indexation et bonne pondération, les aspects d'exhaustivité (description documentaire la plus complète) et spécificité (meilleur discrimination entre les documents) sont pris en compte et sont équilibrés, l'équilibre entre ces deux caractéristiques pour un document se trouve dans le schéma de pondération « TF*IDF ». Cette approche est basée sur la combinaison des deux facteurs : pondération locale (*tf : term frequency*) qui est la représentativité locale d'un terme au sein du document lui-même et pondération globale (*idf : inversed document frequency*) pour la représentativité globale au sein de la collection du document, en définissant un grand volume de formules telles que :

Pour la pondération locale :

***La fonction brute $tf_{ij}$ :*** est une fonction proportionnelle à la fréquence d'occurrence d'un terme dans un document, elle retourne simplement le nombre d'occurrences du terme $t_i$ dans le document $D_j$.

***La fonction binaire :*** elle vaut 1 si le terme appartient au document en cours et 0 si non.

[2] **Le principe du moindre effort :** est une théorie de George Zipf 1949 concernant les documents et les articles textuels : un auteur préfère la réutilisation des termes que d'en attribuer des nouveaux dans son article.

***La fonction logarithmique :*** elle combine $tf_{ij}$ avec un logarithme dans le but de diminuer l'effet de larges différences entre les fréquences des termes d'un document pour les rapprocher les uns des autres : $\alpha + log(tf_{ij})$ où $\alpha$ est une constante.

Cette fonction est proposée par [Salton & Buckley, 1988] et montre qu'un document à grande fréquence du terme $t_i$ n'est pas plus pertinent qu'un autre qui contient plusieurs termes de la requête un petit nombre de fois.

Ou la pondération globale, donc calcul du facteur *idf* ainsi :

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

Ou :

$$idf = \frac{(N - n_i)}{n_i}$$

Où : $n_i$ est le nombre des documents contenant le terme $t_i$

$N$ est la taille de la collection.

Ou par une combinaison des pondérations où toutes les fonctions sont référencées par *tf*idf*,

*tf* : importance d'un terme pour un document et

*idf* : le pouvoir discriminatoire d'un terme pour un document.

Ainsi, un terme à valeur tf*idf élevée est important dans le document et le distingue dans la collection. Dont on trouve les formules :

- $Wij = tf_{ij} \square \frac{N}{n_i}$
- $Wij = \left(1 + \log\left(tf_{ij}\right)\right) * \log(\frac{N}{n_i})$
- $Wij = \left(0.5 + 0.5 \square \frac{tf_{ij}}{\max tf}\right) * log\frac{(N-n_i)}{n_i}$

A l'addition de ces formules, s'ajoute la formule de [Sparck Jones, 1972] qui a été reprise dans différents moteurs d'indexation sous cette forme :

$$tf \square idf = \log(1 + tf) \square \frac{N}{df}$$

*df* : étant le nombre de documents contenant le terme en question

***3.2.5 Création des index :*** la création des index est le processus de stockage des informations sélectionnées lors du processus d'indexation en utilisant des moyens de

stockage tels que les fichiers inverses[3] et les structures correspondantes telles que les arbres (arbres binaires de recherche, les n-arbres) et les tableaux triés…

### 4. Les modèles de base de la recherche d'information :

Le modèle de recherche représente le noyau d'un SRI. Il remplit les deux rôles principaux suivants:

- Créer une représentation formelle, pour un document ou pour une requête basée sur leurs termes;
- Définir une méthode de comparaison entre une représentation de document et une représentation de la requête, afin de déterminer leur degré de correspondance (*similarité* ou *appariement*).

Pour bien différencier les principaux modèles de la RI, la figure suivante présente une taxonomie de ces modèles :

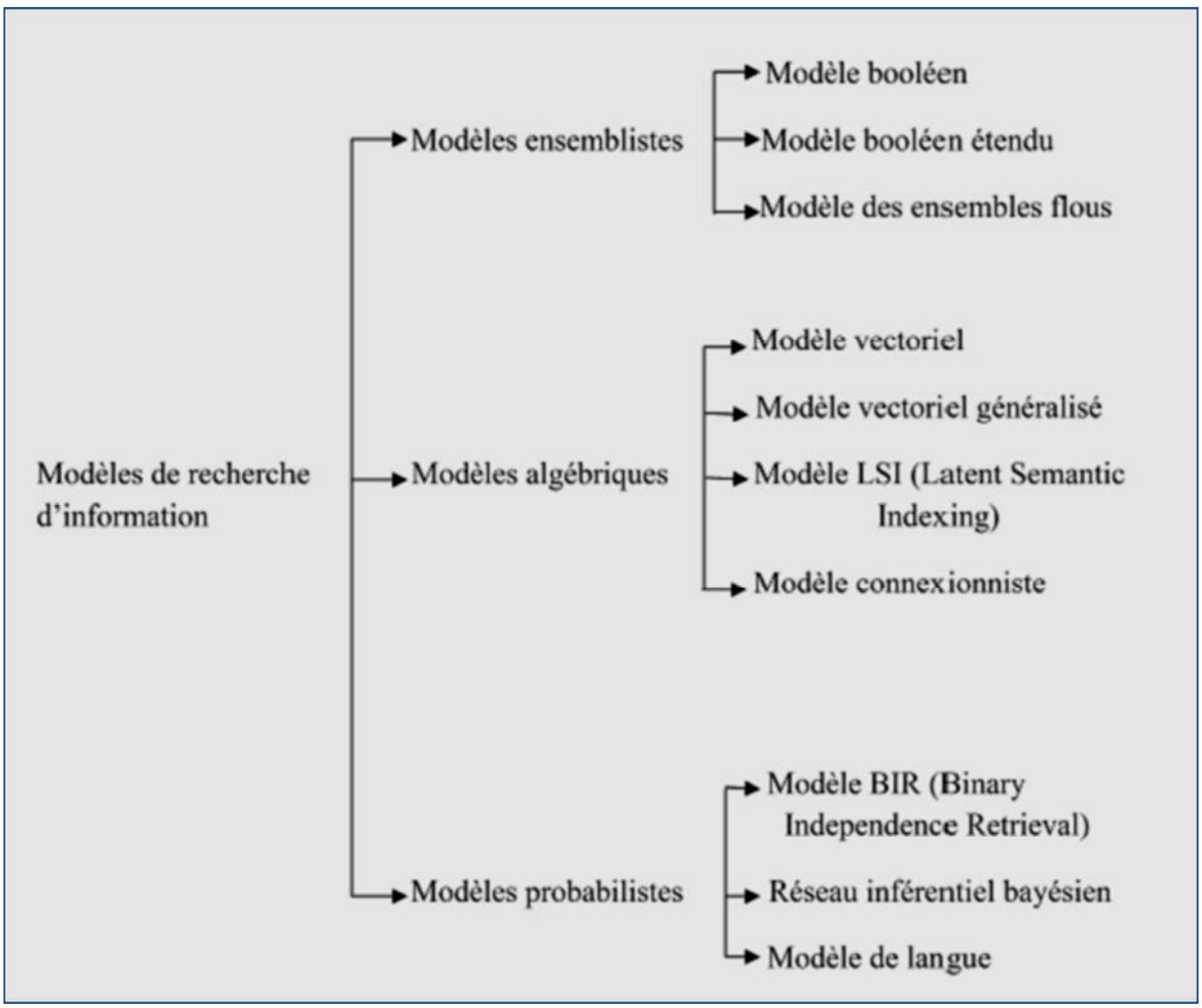



**Figure I-4 :** Taxonomie des modèles RI

[3] **Le fichier inverse :** est un fichier composé d'ensemble des mots associés à un document et une liste de toutes leurs positions (posting).

### 4.1 Le modèle booléen de base :

Le modèle booléen [G. Salton et al, 1993] est le modèle le plus ancien dans la RI. Dans ce modèle, un document est représenté par un ensemble de termes. Une requête est une expression logique composée de termes assemblés par les opérateurs logiques et, ou et non.

La formulation de la requête se base sur les trois opérateurs booléens :

- La conjonction et (^), exige que les termes soient présents simultanément dans la description d'un document,
- La disjonction ou (v), exige qu'au moins un des termes soit présent dans la description des documents à retourner,
- La négation non (¬), utilisée pour écarter les documents qui contiennent un terme.

Le modèle booléen considère dans l'index qu'un terme est présent ou non dans le document, par conséquent le poids d'un terme noté $wi,j \in \{0, 1\}$.

Exemple :

Soit la requête utilisateur suivante :

Requête $Q$ : (cyclisme OR natation) AND NOT dopage

| Le document contient | | | | | Document Pertinent |
|---|---|---|---|---|---|
| Cyclisme | Natation | cyclisme OR natation (1) | Dopage | NOT dopage (2) | (1) AND (2) |
| 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 0 | 0 |

**Tableau I-1:** Exemple de traitement d'une requête du modèle booléen.

**Evaluation des requêtes :** Soit la requête suivante : $Q_k = (tq_{k1} \wedge tq_{k2}) \vee (tq_{k3} \wedge \neg tq_{k4})$ Où : $Q_k$ est la k$^{éme}$ requête, et $tq_{ki}$ est le $i^{éme}$ terme de la requête $Q_k$. Soumettre ce type de requête à un SRI basé sur un modèle booléen, implique que ce dernier doit retourner un ensemble de documents contenant simultanément les termes $tq_{k1}$ et $tq_{k2}$, ou un ensemble de documents contenant le terme $tq_{k3}$ et non pas le terme $tq_{k4}$, cela après avoir traité cette requête de la façon suivante :

| *Requête ($Q_k$)* | *rsv($D_j$ ,$Q_k$)* | *Le résultat de l'évaluation* |
|---|---|---|
| $tq_{ki}$ | *rsv($D_j$ , $tq_{ki}$)* | *= 1 si $tq_{ki}$ □ $D_j$*<br>*= 0 sinon* |
| $tq_{k1} \wedge tq_{k2}$ | *rsv($D_j$ , $tq_{k1} \wedge tq_{k2}$)* | *= 1 si rsv($D_j$ , $tq_{k1}$) = 1 et rsv($D_j$ , $tq_{k2}$) = 1*<br>*= 0 sinon* |
| $tq_{k1}$ v $tq_{k2}$ | *rsv(Dj , $tq_{k1}$v $tq_{k2}$)* | *= 1 si rsv($D_j$ , $tq_{k1}$) = 1 ou rsv($D_j$ , $tq_{k2}$) = 1*<br>*= 0 sinon* |
| $\neg tq_{k1}$ | *rsv(Dj , $\neg tq_{k1}$)* | *= 1 si rsv($D_j$ , $tq_{k1}$) = 0*<br>*= 0 sinon* |

**Tableau I-2:** Evaluation d'une requête dans le modèle booléen classique [Tebri, 2004]

L'inconvénient de ce modèle est que le score binaire diminue le volume des documents partiellement pertinent pouvant être utiles ainsi que ceux retournés ne sont pas triés dû au traitement booléen.

### 4.2 Le modèle booléen étendu :

Le modèle booléen étendu appelé aussi modèle PNorm, a été introduit en 1983 par Salton et d'autres [Salton & al, 1983]. Son principe de base est de compléter le modèle de base, en intégrant des poids d'indexation dans l'expression de la requête et documents. Ceci a pour conséquence, la sélection de documents sur la base d'un appariement rapproché (fonction d'ordre) et non exact.
Considérons un ensemble de termes $t_1,\ldots,t_N$, et soit $d_{ij}$ le poids du terme $t_i$ dans le document $Dj = (d_{1j},\ldots,d_{Nj})$, avec $1 \le i \le N$ et $0 \le d_{ij} \le 1$. La similarité entre le document $D_j$ et une requête $Q_k$ décrite sous une forme conjonctive ou disjonctive est donnée comme suit :

$$\textbf{Opérateur OR}: \qquad RSV(D_j, Q_K) = \left(\frac{\sum_{i=1}^{N} q_{ik}^{p} d_{ij}^{p}}{\sum_{i=1}^{N} q_{ik}^{p}}\right)^{1/p}$$

$$\textbf{Opérateur AND}: \qquad RSV(D_j, Q_K) = 1 - \left(\frac{\sum_{i=1}^{N} q_{ik}^{p} (1 - d_{ij}^{p})}{\sum_{i=1}^{N} q_{ik}^{p}}\right)^{1/p}$$

Où $P$ une constante $0 \le P \le \infty$, et $q_{ik}$ le poids du terme $t_i$ dans la requête $Q_k$.

Dans le modèle booléen étendu, lorsque $p=1$, il n'y a aucune distinction entre les deux connecteurs *ET* et *OU*. Par conséquent, la similarité entre les requêtes et les documents peut être calculée par le produit scalaire entre leurs termes pondérés.
La littérature rapporte, qu'aucune méthode formelle n'est proposée pour la détermination de la valeur du paramètre $P$ [Ponte, 1998].

### 4.3 Le modèle vectoriel :

C'est le plus utilisé dans la RI. Il repose sur l'aspect quantitatif des documents et requêtes par rapport aux termes, dont l'idée de base est d'utiliser une représentation géométrique pour classer les documents par ordre de pertinence par rapport à une requête :

Les documents et requêtes sont représentées dans un espace vectoriel à N dimensions (N étant le nombre de termes) par des vecteurs, et la longueur du vecteur suivant l'un des axes (termes) représente le poids du terme dans la requête ou le document : en effet est une relation proportionnelle entre la longueur du vecteur et le poids du terme et il revient à choisir le vecteur (document) ayant le maximum des poids représentés par rapport aux maximum des axes (termes) par similarité avec la requête (vecteur_requête). La pertinence est représentée par un degré de similarité entre le vecteur de la requête et celui du document, calculé par des fonctions différentes : produit scalaire, cosinus de l'angle distance euclidienne… Formellement, dans le modèle vectoriel, la représentation d'un document est vue comme un vecteur :

$$D_j = \{W_{1j}, W_{2j}, \ldots. W_{nj}\}$$

Où $n$ : étant le nombre des termes de l'index

$w_{ij}$ $_{(i=1,n)}$: représentent les poids des termes dans le document $D$

Aussi la requête est vue comme un vecteur :

$$\vec{Q} = \{W_{1q}, W_{2q}, \ldots. W_{nq}\}$$

Une des plus simples mesures de similarité est celle produite par : (Relvance Status Value)

$$RSV(\overrightarrow{D_j}, \vec{Q}) = \sum_{i=1}^{n} \text{Wij} \square \text{Wiq}$$

Mesure de Cosinus :

$$\text{Sim(Dj , Qk)} = \frac{\sum_{i=1}^{n}(\text{WDij} \square \text{WQik})}{\sqrt{\sum_{i=1}^{n}\text{WD}_{ij}^2 \square \sum_{i=1}^{n}\text{WQ}_{ik}^2}}$$

Mesure de Jaccard :

$$\text{Sim(Dj , Qk)} = \frac{\sum_{i=1}^{n}(\text{WDij} \square \text{WQik})}{\sum_{i=1}^{n}\text{WD}_{ij}^2 + \sum_{i=1}^{n}\text{WQ}_{ik}^2 - \sum_{i=1}^{n}(\text{WDij} \square \text{WQik})}$$

Mesure de Dice :

$$\text{Sim(Dj , Qk)} \frac{\sum_{i=1}^{n}(\text{WDij} \square \text{WQik})}{\sum_{i=1}^{n}(\text{WD}_{ij}^2 + \text{WQ}_{ik}^2)}$$

Dans toute les fonctions de similarité, on cherche toujours à mettre en évidence la correspondance document/requête par rapport aux termes en utilisant les notions représentées par les vecteurs (document/requête) et les axes (termes), et plus les termes associent le document à la requête plus le score est augmenté, et inversement.

Exemple **:** soient les termes $t_1$, $t_2$ et $t_3$, le document D et la requête Q. Le schéma suivant représente les différents poids x, y et z des termes $t_i$ dans le document D et la requête Q :

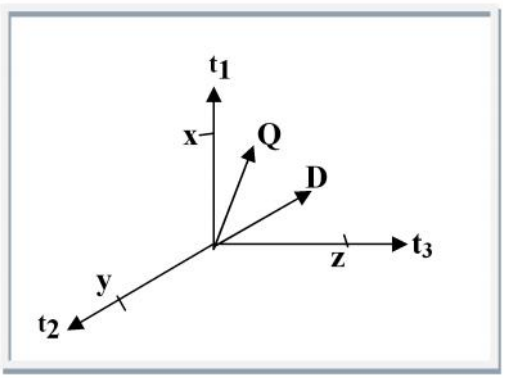



**Figure I-5 :** graphe des poids.

L'inconvénient dans ce modèle est que tous les termes sont considérés indépendant même s'ils ne le sont pas (les axes), et les requêtes sont moins expressives (pas d'expressions comme dans le booléen).

### 4.4 Le modèle des réseaux Bayésiens :

Un réseau d'inférence Bayésien est un graphe de dépendance orienté et sans cycles dont : Les nœuds représentent des variables proportionnelles (des termes, requêtes ou documents) et les arcs représentent les relations de dépendance entre ces variables.

Exemple : Soit une requête Q composée de trois termes $t_1$, $t_2$ et $t_3$

telle que :

Q = 1 si :

Il existe une combinaison des termes $t_i$ pour laquelle la requête Q est valide.

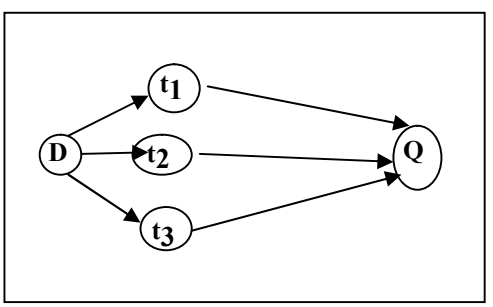



**Figure I-6 :** Illustration de l'exemple en réseau inférentiel

Ainsi que pour le document D et le but final est de trier les documents D selon leurs pertinences et on utilise pour cela la probabilité de réalisation de la requête pour le document :

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

Ou :

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

**4.5 Le modèle de langue :**

Les SRI utilisant les modèles de langages suivent une approche différente des autres modèles. En effet, dans la plupart des modèles, on cherche à comparer une représentation de la requête de l'utilisateur avec une représentation du document recherché, pour évaluer la pertinence de celui-ci. Ici, on part de l'observation que, l'utilisateur crée la requête à partir d'une représentation hypothétique qu'il se fait du document recherché. La requête est donc, inférée par l'utilisateur à partir des documents voulus. Le but du modèle de langage est de générer des requêtes à partir des documents, et comparer celles-ci avec la requête de l'utilisateur. La pertinence d'un document est estimée en calculant la probabilité que la requête utilisateur soit inférée par celui-ci, donc : comparaison entre requêtes mais pas entre une requête et un document. Nous revoyions ce modèle en détailles dans le prochain chapitre.

## 5. Evaluation des systèmes de recherche d'informations :

Un système de recherche d'information est le moyen qu'utilise un utilisateur depuis l'apparition de ce domaine informatique, donc il s'avère nécessaire de penser à l'évaluation de tels systèmes pour garantir la satisfaction de l'utilisateur par le renvoie de l'information pertinente. Pour se faire, des mécanismes d'évaluations ont étés établis pour comparer, évaluer et juger la qualité des SRIs qui doivent implémenter de bonnes indexations représentant le contenu sémantique de la collection et des requêtes, utiliser de bonnes formules de pondération pour y aboutir et se baser sur de bonnes formules de calcul de similarité. Pour juger de tels concepts, on définit les mesures du rappel et précision.

**Rappel et précision :**

Ce sont deux parmi d'autres mesures permettant l'évaluation correcte de la capacité d'un SRI de retourner des documents pertinents satisfaisant l'utilisateur. L'objectif de ces métriques est de permettre de : retrouver tous les documents pertinents et rejeter tous les documents non pertinents. Afin de bien introduire ces notions, la figure suivante exprime le partitionnement de l'ensemble des documents restitués par le SRI en deux sous-ensembles :

Documents pertinents A et Documents non pertinents B.

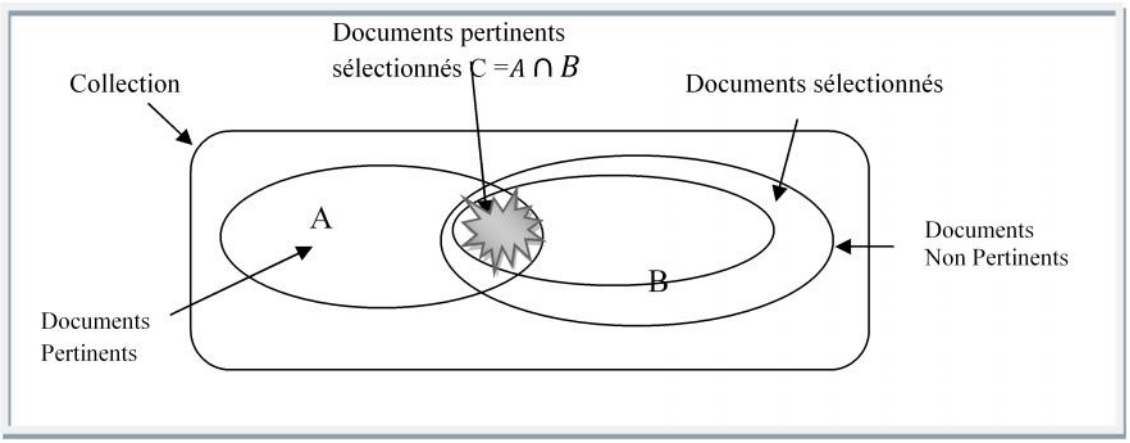



**Figure I-7 :** Partition d'une collection pour une requête [Mataoui, 2007].

Ainsi, on aura :

$$\text{Rappel} = \frac{|C|}{|A|} \qquad \text{et} \qquad \text{Précision} = \frac{|C|}{|B|}$$

Les deux métriques ne sont pas indépendantes. Il y a une forte relation entre elles: *quand l'une augmente, l'autre diminue*. [Fellag, 2006]
Le comportement d'un système peut varier en faveur de précision ou en faveur de rappel (au détriment de l'autre métrique). Ainsi, pour un système, on a une courbe de précision-rappel qui a en général l'aspect suivant:

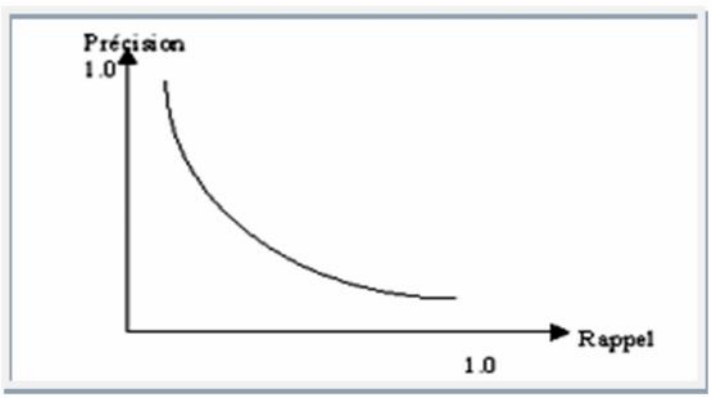



**Figure I-8 :** Courbe de rappel/précision [Nie, 2004]

- Un bon SRI est celui qui est capable de renvoyer les bons documents en évitant à retourner les documents non pertinents (en faisant le moindre bruit possible), et de restituer le maximum de documents pertinent (silence du système).

**Compagnes d'évaluation et collections de référence :**

Depuis les années 1970s, plusieurs projets ont vu le jour dont le but est la construction d'une collection de références qui permettent l'évaluation des SRIs dans des compagnies spécialisées. Ces projets ont fourni plusieurs collections en RI dont : la collection CACM, ou

encore la compagne CLEF[4](Cross Language Evaluation Form) et la collection ISI. Parmi les plus importants projets, on trouve celui initié par la DARPA (Defense Advanced Research Project Agency), co-organisé par le NIST[5] (Nationnal Institut of Standards and Technology). Ce projet est appelé : la compagnie d'évaluation TREC[6] (Text Retrieval Conference) qui a commencé en 1992, dans le but d'encourager le domaine de la recherche documentaire sur des grandes collections de test. L'évaluation des SRI dans la compagnie TREC se fait ainsi :

On aura un ensemble de SRIs participants à l'évaluation auxquels est fournie la même documentation (base documentaire) et même ensemble de requêtes puis comparer les résultats renvoyés par chacun ; les documents retournés des différents systèmes participant sont jugés à propos de leurs pertinences par le NIST, tel que : l'ensemble des documents pertinents pour chaque requête est obtenu en prenant les *K* documents les mieux classés des différents SRI participant à la compagnie d'évaluation et ils sont ensuite montrés à des juges qui décident finalement à propos de la pertinence de chaque document. Les participants à TREC disposent de la liste des documents pertinents pour chaque requête, et peuvent ainsi évaluer les performances de leurs SRI respectifs.
De TREC-1 à TREC-6, les recherches étaient centrées sur deux tâches principales: la tâche de routage et la tâche ad hoc.

**Conclusion :**

En conclusion, la recherche d'informations est née avec différentes sciences émergentes de la curiosité de l'être humain et s'est développée au cours du temps. La RI est développée et automatisée pour garantir plus de flexibilité en recherche, ce qui a donné naissance à des concepts, des protocoles des modèles et des systèmes de recherche d'informations, ces dernier sont en interaction avec l'utilisateur via le web. La naissance du web a vu émerger des documents diversifiés et de plus en plus développés, donc les processus de recherche doivent prendre en compte ces nouvelles technologies et satisfaire l'utilisateur avec les informations pertinentes.

[4] **CLEF :** est une compagne d'évaluation qui a établi l'union avec la compagnie INEX pour faciliter le transfert de connaissance entre les forums d'évaluation, on peut trouver une description plus fine dans : http ://www.clef-campaign.org
[5] National Institue of Standards and Technology (www.nist.gov)
[6] http ://trec.nist.gov

## Partie 2 : La recherche d'information structurée et semi structurée

### Introduction :

L'objectif principal d'un SRI classique est de retourner pour toute requête utilisateur les documents les plus pertinents pour satisfaire son besoin en information. Dans cette optique, les documents sont représentés par un ensemble de mots clés désignés par l'index du document, dans le but de comparer cette présentation avec une présentation de la requête ce qui revient à calculer une valeur de similarité RSV en se basant sur le poids des termes de la requête dans le document en question. Avec le développement du web, l'entité « Document » est vue progresser des documents texte vers les documents structurés ou semi-structurés XML, qui est un format variant à présentation arborescente du contenu de document. De grandes collections documentaires sont vues apparaitre et des SRI modernes ont vu le jour pour s'adapter à ce nouveau format, dans l'objectif est de répondre à la question : Comment satisfaire l'utilisateur avec l'information pertinente en profitant des avantages de la structure documentaire ? Pour répondre à une telle problématique, il va falloir définir un ensemble de notions préliminaires dans la RI classique pour la RI structurée et comment adapter les notions d'indexation, pondération, appariement et recherche à la nouvelle technologie XML ?

### 1. Définitions des notions de structure et contenu:

**La structure** est la caractéristique de tout article ou document. C'est la définition d'un format logique pour représenter le corps d'un document. Dans les documents électroniques, la structure est définie par des balises bien imbriquées encadrant des portions d'information (texte), dont deux types de structure : la structure physique qui traite les propriétés typographiques (police, taille, couleur …) associée à chaque élément du texte, et la structure logique qui décrit la nature des éléments et les relations hiérarchique qui les relient.

**Le contenu** d'un document structuré fait référence à son contenu textuel compris entre ses balises, et qui constituent des éléments porteurs d'information.

Voici un exemple de document structuré :

```
<Document>
<Titre>Le titre du document</Titre>
<Section>
<Texte>Ceci est un exemple</Texte>
</Section>
</Document>
```

**Figure I-9 :** Exemple de document structuré.

## 2. Documents structurés XML:

La notion de document a évolué pour passer du concept de document plat à un concept où le document est devenu un objet pouvant contenir plusieurs sources d'informations (vidéo, texte, image …). Pour exploiter ces sources d'une manière rigoureuse, le format de document est défini par une structure logique qui est décrite par plusieurs normes internationales (SGML, XML). Dans ce chapitre nous nous intéressons au format XML car il constitue l'objectif de ce travail.

La structure logique d'un document XML est un arbre. Chaque nœud de l'arbre est un élément XML qui est décrit par une balise ouvrante est une balise fermante. Un élément peut avoir un ou plusieurs attributs XML dans l'arbre du document XML.

### 2.1 Présentation de XML :

Abrégé en XML (eXtensible Markup Language), est un langage à balises extensibles (format variable), est un nouveau standard de représentation et d'échange de données sur internet, il sépare le contenu d'un document des instructions de sa représentation, car comme le HTML, XML utilise des balises et des attributs à seule différence que le HTML définit la signification de chaque balise et chaque attribut (souvent : la manière dont le texte ou l'image apparaitra dans le navigateur) et XML utilise le balisage pour délimiter les éléments de données tandis que leurs interprétations s'effectuent par l'application qui les lit.

Xml est dit aussi un Métalangage, ie : langage pour écrire d'autres langages. Il est à la base d'une collection des nouveaux langages comme : XHTML, MathML …etc., et de nombreuses technologies comme : XSL, Query, XLink …ect. Comme il est illustré dans la figure suivante :

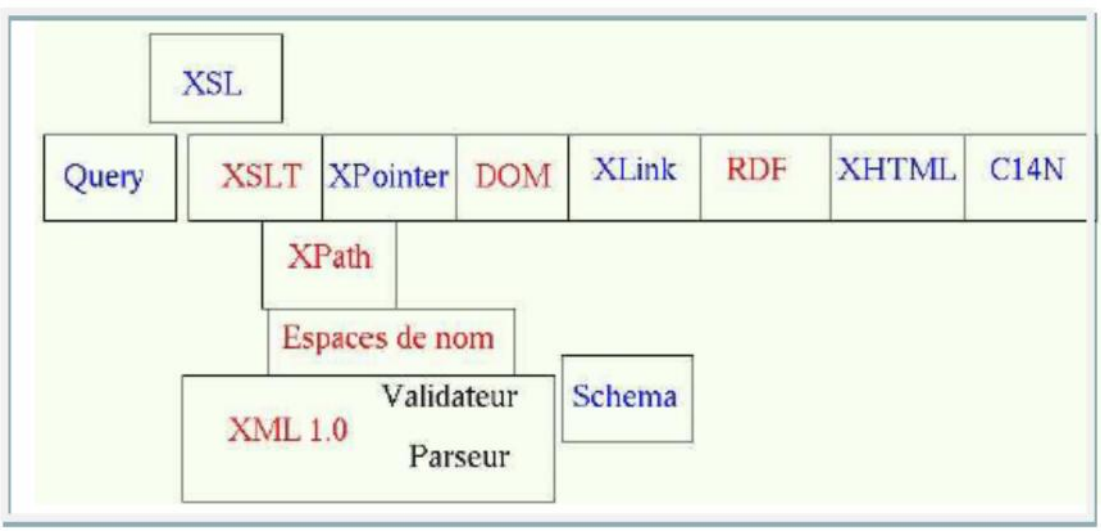



**Figure I-10:** Technologies de la famille XML [Vieillard, 2000]

### 2.2 Construction d'un document XML

XML repose sur trois paramètres de construction fondamentaux : les éléments, les attributs et les valeurs associées aux attributs s'ils existent. Ce sont ces blocs qui constituent le document XML. Un document XML *bien formé* est un document XML qui respecte les *règles lexicales* et *syntaxiques* suivantes :

- Il existe un seul élément (racine) qui contient tous les autres,
- A chaque balise ouvrante est associée une balise fermante bien imbriquée (pas de chevauchements entre les balises),
- Les noms des balises ne doivent pas commencer par « XML », mais par une lettre ou le caractère souligné « _ » et pas de caractères spéciaux,
- Si un élément a un attribut, ce dernier doit être unique et possédera obligatoirement une valeur.

Cependant, tout document XML est soumis au contenu suivant :

- Les commentaires pour la documentation du programmeur,
- Un arbre d'éléments décrivant le contenu sémantique du document et
- Un prologue (facultatif mais fortement conseillé) : contient toute les déclarations XML nécessaires (version, format du codage et une information sur les déclarations externes), types des données du document (DTD : règles de structure, les éléments inclus ainsi que leurs types et les valeurs par défaut) pour garantir la validité du document et des instructions nécessaires pour le traitement du document,

Voici un exemple de document semi-structuré XML :

```
<movie>
      <title>Primer</title>
      <overview>
            <releasedates>
                  <releasedate>
                        France 21 February 2004
                  </releasedate>
            </releasedates>
      </overview>
      <cast>
      <actors>
            <actor>
                  <name>
                        Shane Carruth
                  </name>
                  <character>
                        Aaron
                  </character>
            </actor>
      </overview>
      <plot>
            Four fledgling entrepreneurs[...], wrestle over their invention.
      </plot>
</movie>
```

**Figure I-11 :** Exemple de document XML [Cyril, 2013].

**2.2.1 Le commentaire :** est une information destinée pour le constructeur lui même, il a le format suivant :

```
< !—Ceci est un ommentaire-->
```

**2.2.2 L'élément :** est la plus petite unité dans le corps du document XML. Est une unité sémantique composée des balises d'ouverture et de fermeture portant le nom de l'élément et son contenu, ayant la syntaxe suivante :

```
<Elément> Ceci est un élément</Elément>
```

Un élément vide est représenté par : <nom_élément/>

**2.2.3 L'attribut :** est une information secondaire associée à une balise d'ouverture, suivant la syntaxe suivante :

```
<Elément attribut=''valeur  chaine  caractéres''>contenu</Elément>
```

**2.2.4 Le prologue :**

Contient généralement :

- Une déclaration XML, par exemple :

<?xml version = ''1.0'' encoding = ''ISO-8859-1'' standalone = ''yes'' ?>

Indique au programme traitant que la version d'XML utilisée est 1.0, le codage utilisé est : ISO-8859-1 et que ce document possède des déclarations extérieures.

- Des instructions de traitement, sont facultatives mais utilisées par l'application en vue de déclencher certains traitements.
- Une déclaration de type du document (DTD), établit les règles de définition de la structure documentaire XML, et est déclarée au sein d'un document (DTD interne) ou comme un document différent (DTD externe). On dit qu'un *document* est *valide* s'il contient une DTD. Voici un exemple d'une DTD correspondante au document de la figure 11 :

```
<!ELEMENT movie (title, overview?, casts?)>
<!ATTLIST movie xmlns:xlink CDATA #FIXED "http://www.w3.org/1999/xlink">
<!ELEMENT title (#PCDATA)>
<!ELEMENT overview (directors, release) >
<!ELEMENT director (name, movie)>
<!ELEMENT release (#PCDATA)>
<!ELEMENT casts (actors?)>
<!ELEMENT actors (actor+)>
<!ELEMENT actor (name, character?)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT character (#PCDATA)>
```

**Figure I-12 :** DTD correspondante à une liste de filmes

### 2.3 Représentation graphique des documents XML:

L'organisation hiérarchique des éléments structurels des documents XML nous permet de représenter ces derniers sous la forme d'un graphe connexe acyclique orienté : un arbre.

Dans un arbre XML, les éléments sont représentés par des nœuds. Les arcs traduisent l'imbrication des différents éléments du document. Le contenu textuel sera quant à lui localisé dans les nœuds feuilles. Voici un exemple explicatif :

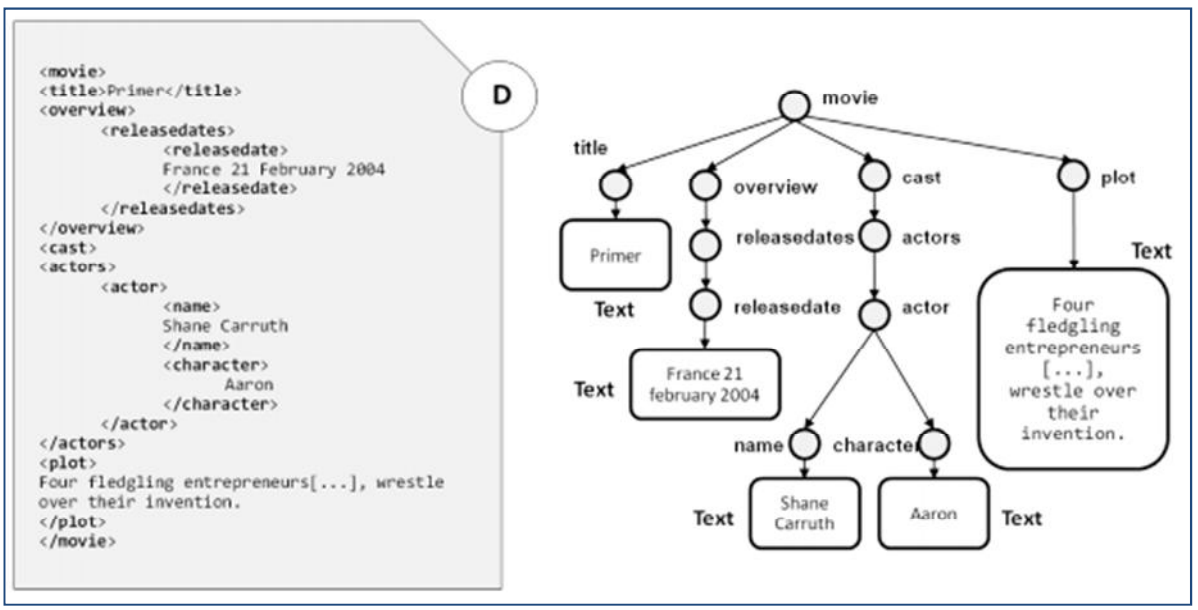



**Figure I-13 :** Exemple d'une présentation arborescente d'un document XML

### 2.4 Analyseur du document XML :

XML n'est pas un langage de programmation, mais c'est uniquement un langage de structuration et de représentation de données. Il ne comporte pas d'instructions de contrôle et ne permet donc pas d'exploiter directement les données, pour cela, il faut disposer d'un analyseur, encore appelé parser en anglais qui permet de récupérer dans une structure XML, des balises**,** leur contenu**,** leurs attributs et de les rendre accessibles. Il existe deux types d'analyseurs : le parser SAX (Simple API for XML) produisant un flux d'évènements et le parser DOM (Document Object Model) produisant un graphe d'objets en mémoire.

#### 2.4.1 SAX :

SAX (Simple Api for Xml) est un analyseur orienté évènement standardisé par le groupe XML-DEV, il fournit une interface évènementielle pour parcourir un document XML, renvoi à l'application les ouvertures des balises et leurs fermetures, le contenu textuel … ce qu'on appel « Evènement », ceci via des méthodes de gestion des évènements : startDocument(), endDocument(), startElement(), endElement() … ect.

### 2.4.2 DOM :

**DOM** (***D**ocument **O**bject **M**odel ou modèle objet de document*) est une API (***A**pplication **P**rogramming **I**nterface ou interface de programmation d'application*) permettant de représenter un document XML sous forme d'arbre d'objets reliés entre eux. Chaque objet représente une entité (document, élément, attribut, texte,…) d'un document XML.
DOM permet une navigation aisée dans un document XML mais nécessite le chargement complet en mémoire de sa structure arborescente. Le modèle DOM est une recommandation du W3C depuis octobre 1998.

Voici un exemple de document XML et sa représentation DOM correspondante :

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

**Figure I-14 :** Exemple de document XML *article.xml*

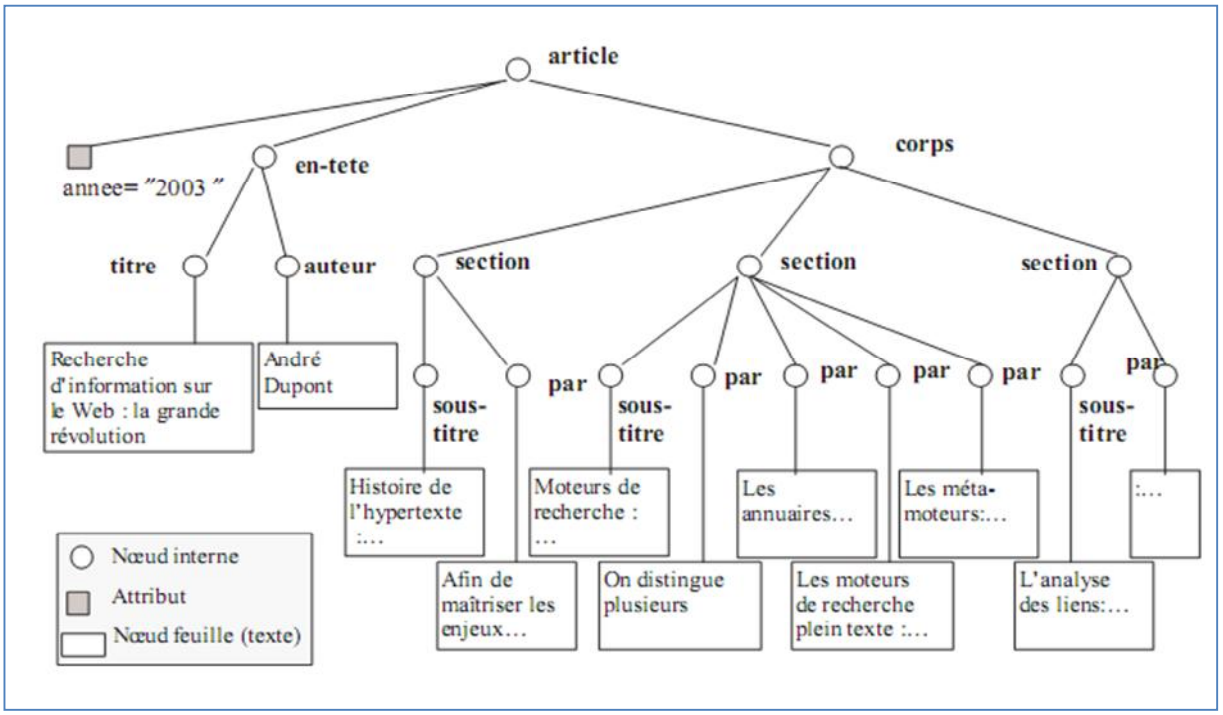



**Figure I-15 :** Représentation DOM du document XML précédent

## 3. Problématiques liées à la RIS :

### 3.1 Discussion :

Le passage de la RI traditionnelle à la RI structuré a impliqué la mise en exploit d'un nouveau format de documents, connu sous le nom de documents XML structurés ou semi-structurés. Donc, au lieu de représenter le document par uniquement une suite de chaine de caractères ce qu'on appelle document textuels, une nouvelle notion est ajoutée et est l'information structurée : l'information dans un document n'est plus un simple paragraphe mais c'est une donnée complexe regroupant l'information textuelle et son emplacement ou encore son niveau. C'est ce qu'on appelle *la granularité de l'information,* ce qui a posé des problématiques sur le traitement des données et informations retournées à l'utilisateur répondant à sa requête : le nouveau format des documents impose une représentation arborescente où l'information textuelle se localise sur les nœuds feuilles, donc ce qui doit être retourné à l'utilisateur n'est pas obligatoirement un document tout entier, mais une partie de celui-ci : un élément ou un ensemble d'éléments. Le but des SRI est alors d'identifier des parties du document les plus pertinentes à une requête et qui sont des unités d'informations supposées répondre aux requêtes fournies, en implémentant les fonctions d'indexation, pondération et recherche de façon à prendre en compte la nouvelle granularité de l'information :

- *L'indexation,* qui se situe au niveau de l'information structurelle, car dans les documents traditionnels, ce processus se base sur la pondération des termes qui représentent le contenu du document tout entier et dans la RI structurée, la dimension structurelle s'ajoute au contenu : Que doit-on indexer de la structure du document ? et comment relier cette structure au contenu texte du document ?

- *L'interrogation des documents,* il s'agit de permettre à l'utilisateur d'exprimer des besoins en informations diversifiés (contenu textuel et/ou structurel).
- *La pertinence* des unités retournées se justifie par la comparaison de l'arbre de la requête fournie et celle extraite de l'unité retournée tout en respectant le contenu informatif exigé et pour ce faire doit-on pondérer les termes d'indexation ? Et en fonction de quel niveau doit-on procéder à cette pondération ?

Afin d'aboutir aux notions de pertinence, les techniques de la RI traditionnelle doivent être adaptées et/ou de nouvelles méthodes doivent être proposées pour l'indexation, l'interrogation, la recherche et le tri des unités informatives.

### 3.2 Pondération des termes :

Pour pondérer les termes d'indexation des documents XML, plusieurs travaux ont été présentés pour répondre à la problématique d'évaluation de la similarité entre une partie d'un document et une requête utilisateur, pour cela les chercheurs en RI ont adapté les formules de pondération des documents plats aux documents XML. En étudiant la structure des documents : les nœuds, les éléments, les nœuds feuilles… et en ajoutant des paramètres qui en tiennent compte aux formules de pondération déjà proposées en RI traditionnelle tout en se basant sur l'approche *tf*idf*.

### 3.3 L'indexation des documents :

Le processus d'indexation consiste en extraction des clés de recherche des documents. Dans le cas des documents textes « plats », le contenu textuel des documents est traité afin de trouver et de pondérer les termes les plus représentatifs du contenu du document. Dans le cas des documents semi-structurés, cette fonction doit prendre en considération la structure du document, ce qui est une problématique posée.

Pour répondre à cette problématique, la fonction d'indexation est traitée en deux fonctions [Sauvagnat, 2005]:

*L'indexation de l'information textuelle* qui se base sur le contenu situé au niveau des nœuds feuilles toute en exploitant les mots clés de cette information,

*L'indexation de l'information structurelle* qui revient à prendre en compte de la structure du document en intégrant des paramètres dans les formules de pondération qui traduisent la structure documentaire.

### 3.4 La notion de granularité :

Bien avant l'apparition de XML, des travaux concernant la granularité de l'information à renvoyer à l'utilisateur ont été présentés. Ces travaux se basent sur le fait qu'une demande d'information ne correspond pas forcément à la recherche d'un document mais peut avoir pour cible des éléments de granularité plus fine, nous pouvons trouver dans [Fellag, 2006] des exemples de travaux développés dans ce contexte, ces travaux sur la segmentation de texte ont été motivés par le fait que l'information recherchée est souvent plus petite qu'un document

entier. Toutefois ils se heurtent à la définition de l'unité de recherche élémentaire. D'après [Piwowarski, 2003], pour un corpus de documents XML, on peut distinguer trois différents types d'unités d'information :

### 3.4.1. Document XML entier :

Tel que, la requête :

- Est exprimée comme un ensemble de termes à pondérer, ou
- Exprime des contraintes structurelles (exigences utilisateur sur la forme) et il faut en tenir compte.

Exemple : rechercher un document historique qui traite le sujet du cout de Chechnek en Egypte il y 2963 ans ainsi que des événements associés.

### 3.4.2 Un élément :

Cela signifie que la demande d'information est spécifiée de l'une des deux façons suivantes :

- Indiquer clairement le type d'unité recherchée (image, paragraphe, vidéo,…) : généralement booléenne.

Ou :

- Trouver l'élément qui résume au mieux les sections pertinentes du document.

### 3.4.3 Un ensemble d'éléments :

Où l'information n'est pas précisément définie (pas d'exigences de structure ou d'imbrication d'éléments les uns dans les autres), dans ce cas on procède à retourner l'unité la plus pertinente possible et la plus petite possible.

Cas d'une requête de type : « Section qui parle des origines Algériennes », et la solution cherche à mettre en œuvre l'unité la plus cohérente pour la requête.

## 4. Evaluation des SRI structurés :

Tout comme les systèmes de recherche d'informations dans la RI traditionnelle, les SRI dans la RI structurée sont aussi évalués dans des compagnes spécifiées, en fournissant un ensemble de documents et requêtes à des systèmes candidats, puis évaluer les résultats retournés par chacun des systèmes, en utilisant un ensemble de mesures pour juger la pertinence des résultats retournés et la précision du système qui les a retourné. Dans cette optique, on entend parler toujours de la compagne spécifiée dans l'évaluation des SRI structurés.

### 4.1 La compagne INEX :

INEX (INitiative for the Evaluation of Xml retrieval) est créée en 2002 dans le but d'évaluer les systèmes de recherche d'information structurée. Le but était le développement d'une méthodologie et des outils (collection de test, requêtes et mesures d'évaluation), pour tester des systèmes et approches de recherche XML, en utilisant un langage de requêtes qui

est NEXI[7] (Narrowed Extend XPath I) et un ensemble de tâches telles que : ad-hoc, la tâche multimédia, la tâche « Relevance Feedback » et la tâche hétérogène, et par la suite plusieurs compagnies INEX dérivées de celle-ci à plusieurs évolutions et améliorations sont utilisées pour évaluer les SRI participant, telles que :

- INEX 2005-2006 : En 2005 la collection de base est une collection d'outils provenant de la « IEEE Computer Society », ciblée au format XML. Elle est de volume d'environ 500 à 700 Mo composée de 8 millions éléments, équivalent à 12000 articles, d'environ 1500 éléments à profondeur moyenne de 6,9 (nombre de niveaux), provenant de 18 magasines ou revues différentes. A partir de 2006, la collection de base utilisée pour les tests étant la collection *Wikipedia,* qui est utilisée dans la plupart des tâches. Cette collection de 6 Go, est composée de 659.388 documents d'une profondeur moyenne de 6.72. Le nombre moyen de nœuds XML par document est 161,35.
- INEX 2007 : Le changement principal dans INEX 2007 concerne la permission de retourner des parties arbitraires d'un document et l'évaluation de la pertinence d'un texte d'un élément en fournissant des requêtes diverses. L'évaluation s'effectue par l'étude des trois tâches ad hoc : Focused task(la tâche concentrée), Incontext task (tâche de mise en contexte) composée des deux tâches : Relvant in context et Best in context. [INEX 2007 Evaluation Measures (Draft)]
- INEX 2012 : En 2012, INEX et en collaboration avec CLEF, ont étudié les différents aspects pour l'accès à l'information concentrée et ils ont établi les tâches de base pour l'évaluation des SRI en se basant sur des collections appropriées.
  [INEX repport, Repport on INEX 2012]

### 4.2 Les éléments d'évaluation :

#### 4.2.1 Les requêtes :

Sont appelées aussi *Topics*, elles sont créées par des différents participants et représentent les demandes moyennes de l'utilisateur, ainsi elles sont alors de différents types :

- **Les CO (Content Only) :** Des requêtes en langage naturel dont les mots-clés sont reliés par (+) pour impliquer l'obligation de présence du mot et (-) pour impliquer l'interdiction de sa présence, comme le montre l'exemple suivant :

[7] Vous pouvez trouver une description générale d'INEX dans : *http://inex.is.informatik.uni-duisburg.de/2006.*

```
<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 I-16:** Exemple de requête CO, issue du jeu de test 2003

- **Les CAS (Content And Structure) :** A l'indication du contenu bref voulu, on ajoute des contraintes de structure telles que les exigences de contenu des éléments du document XML

Pour chaque Topic, est construite une présentation sous forme de différents champs pour expliciter sa portée :

- Title : définition de la forme générale (CO)
- Key words : donne l'ensemble des mots clés de la requête qui ont permit l'exploration du corpus.
- Les champs Description et Narrative : expression en langage naturel des désirs de l'utilisateur.
- Le champ Castitle pour la forme structurée CAS.

Et voici un exemple de requête orientée structure et contenu :

```
<inex_topic topic_id="205" query_type="CO+S" ct_no="12">
<InitialTopicStatement>McLuhan</InitialTopicStatement>
<title>marshall mcluhan</title>
<castitle>//bdy//*[about(., "Marshall McLuhan")]</castitle>
<description>
Find information about the relevance of Marshall McLuhan's ideas for current
digital technologies.
</description>
<narrative>
I am writing an essay on the inuence of new media icon Marshall McLuhan
on digital technologies. I'm seeking information describing how McLuhan's views
have inuenced current digital technologies. To be relevant, a retrieved item should
discuss some aspect of Marshall McLuhan's visionary ideas or famous one-liners
in the context of current digital technologies. Retrieved elements that merely cite
some of McLuhan's work are non-relevant, as are elements that discuss ideas not
originating from McLuhan.
</narrative>
```

**Figure I-17:** Exemple de requête CAS (Topic 205 d'INEX 2005)

### 4.2.2 Les tâches [INEX 2005]:

**- La tâche ad-hoc :**

C'est une recherche qui se fait en simulation de l'utilisateur d'une bibliothèque composée de documents XML, qui est interrogé par des requêtes utilisateur conçues dans la compagnie et portent sur le contenu et la structure. Cette principale tâche est fragmentée en trois sous tâches distinctes :

**a- La tâche CO : (Content Only task)**
Sert à répondre aux requêtes utilisateur de type CO par des granules d'information XML. Dans cette tâche, aucune indication de structure n'aide les SRI à savoir le type d'unité à retourner.

**b- La tâche SCAS : (Strict Content And Structure task)**
Répondre aux requêtes CAS avec des granules XML de manière stricte, ie : obéir aux exigences de structure et contenu à la fois, dont le champ Title des requêtes est basé sur une syntaxe XPath.

**c- La tâche VCAS : (Vague Content And Structure task)**
Répondre aux requêtes CAS de manière vague. ie : avec des granules satisfaisant globalement les requêtes.

### 4.2.3 L'évaluation :

L'évaluation de pertinence des SRI XML passe premièrement par une validation des éléments/documents qui sont jugés à la main. Par la suite, en 2002 une échelle à deux dimensions a été proposée et basée sur les degrés de pertinence et couverture, qui ont été remplacée par les notions d'exhaustivité et de spécificité, depuis 2003. Telles que :

L'exhaustivité est mesurée selon une échelle à quatre niveaux, un élément est :

- Pas exhaustif : il ne traite pas du tout du sujet de la requête.
- Marginalement exhaustif : il traite peu d'aspects du sujet de la requête.
- Assez exhaustif : il traite de nombreux aspects du sujet de la requête.
- Très exhaustif : il traite la plus part ou tout les aspects du sujet de la requête.

La spécificité décrit à quel point l'élément est focalisé sur la requête, ie : la couverture du document/élément au sujet et elle est aussi mesurée sous quatre niveaux, l'élément peut être à :

- Pas de couverture : le thème traité n'a rien à avoir avec celui de la requête, ie : n'est pas du tout spécifique.
- Couverture trop large : ou marginalement spécifique où le thème de la requête est traité exactement dans un sous élément.
- Petite couverture : si l'élément renvoyé contient juste une partie de l'information pertinente.
- Couverture exacte : ou l'élément est très spécifique et le seul sujet qui traite est celui de la requête.

L'usage de l'échelle à deux dimensions, est impliqué par le besoin de mesurer la pertinence d'un élément par rapport à son descendant, et lorsqu'un élément n'est pas pertinent, il n'a pas de couverture et inversement.

### 4.2.4 Mesures d'évaluation :

Jusqu'à 2004, les seules mesures appliquées dans l'évaluation des SRI étaient le rappel et la précision, par la suite dans les compagnes INEX 2005 et 2006 d'autres mesures ont été définies pour mettre une meilleure évaluation des SRI en RI structurée [Xavier] :

- **Le gain cumule (xCG) :**

Cumulation des scores de pertinence des éléments de la liste des résultats. Etant donnée une liste d'éléments triée par ordre décroissant dans laquelle, les éléments sont présentés par leurs scores de pertinence :

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

i : le rang de l'élément dans la liste

xCG(i) : somme des scores de pertinence des documents j ($j = \overline{1,i}$)

xG(j) : le score du document de rang j

Après avoir calculé le gain cumule des éléments, pour chaque requête on calcule un vecteur de gain idéal xCGI à partir de la base de rappel, et le xCG peut être alors comparé au xCI avec le nxCG :

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

tel que : pour l'élément de rang i, le score de pertinence cumulé acquis sur le score de pertinence idéale voulu nous donne une valeur de norme comprise entre 0 et 1 tel que :

si nxCG(i) ⟶ 0 élément non pertinent

si nxCG(i) ⟶ 1 élément pertinent

Et il reflète le gain relatif que l'utilisateur accumule jusqu'à ce rang si le système avait produit une liste triée optimale.

- **L'effort précision (ep) :**

Elle représente l'effort (en nombre de liens à visiter) qu'un utilisateur doit fournir pour parvenir à un gain donné *r*, $e_{system}$ (respectivement $e_{ideal}$) est le rang auquel le gain *r* est atteint par le système (respectivement par la liste optimale).
Cette mesure dépend du gain, car elle est calculée par :

$$ep(r) = \frac{e_{idéal}}{e_{system}}$$

Tel que : *r* : c'est le gain

*e* : le rang correspondant au gain

$e_{idéal}$ : le rang auquel le gain est idéal

$e_{system}$ : le rang auquel le gain est celui retourné par le système évalué.

- **Mesure de precal :**

Elle a été utilisée lors de la compagnie d'évaluation 2002 pour définir la probabilité qu'un élément retrouvé et retourné à l'utilisateur soit pertinent, est calculée par :

$$P(pert/retr)(x) = \frac{x.n}{x.n + esl_{x.n}}$$

Tel que : *pert* : document *x* pertinent

*retr* : document retrouvé

$esl_{x.n}$ : nombre attendu d'éléments non pertinents retrouvés jusqu'à ce qu'un point de rappel *x* soit atteint

*n* : le nombre de documents pertinents dans la collection par rapport à une certaine requête.

**Conclusion :**

La RI étant une nouvelle branche de l'informatique, présente à l'utilisateur des opportunités diversifiées lui permettant d'accéder à l'information pertinente, ceci par le billet des SRI qui implémentent les différents concepts que nous avons présenté au cours de ce chapitre. Que ce soit en RI traditionnelle ou en RI structurée, le but étant le même : satisfaire le besoin de l'utilisateur en information pertinente, il reste d'actualité de mettre en œuvre tous les moyens permettant à un SRI d'atteindre un tel objectif. Dans ce chapitre, nous avons présenté les concepts de base de la recherche d'information traditionnelle et semi-structurée, nous avons illustré les différences principales et le passage des documents plats vers les documents XML ainsi que les problématiques émergées par ce nouveau format. Quelque soit le SRI, traditionnel ou semi-structuré, il se base sur un ensemble de présentations sémantiques des concepts de la RI : document, requête, éléments, mots clés…, pour cela, il se base sur un modèle RI basique dans ces représentations ainsi que sa démarche de recherche et de calcul, parmi ces modèles, il existe un modèle probabiliste qui a été défini par « Le modèle de langue ». Au cours du prochain chapitre, nous allons voir les concepts de ce modèle, son utilisation et les travaux basés sur ce modèle.

# CHAPITRE II : LE MODELE DE LANGUE

# Partie 1 : Le modèle de langue dans la RI classique

## Introduction :

A l'origine, la recherche d'informations se base sur un ensemble de concepts mathématiques et analytiques pour représenter théoriquement l'aspect de comparaison entre les requêtes utilisateur (ou système) et les documents à la disposition d'un SRI, pour retrouver l'information pertinente, cet ensemble de concepts : présentation schématique, formules mathématiques, lois,… est connu en RI sous l'appellation : *Modèle de recherche d'information.* Tel que nous l'avons vu dans le premier chapitre, il existe une grande taxonomie de modèles RI englobant trois grandes classes :

- Les modèles ensemblistes : tels que le modèle booléen et ses dérivés.
- Les modèles algébriques : tels que le modèle vectoriel et ses dérivés.
- Les modèles probabilistes : tels les réseaux inférentiels et le modèle de langue.

Dans ce chapitre, nous nous intéresserons au modèle de langue comme étant une dérivée du modèle probabiliste de base et notre contribution exploite largement ce modèle, pour bien comprendre ce modèle, nous présenterons un ensemble de connaissances le concernant et qui répondent à des questions du type : Quelle est l'approche à l'origine de la naissance de ce modèle ? Quelles sont ses spécificités par rapport aux autres modèles ? Et quels sont ses métriques adaptées dans le processus de recherche ?

## 1. Le modèle probabiliste de base :

Le modèle probabiliste de base est à l'origine d'un ensemble de dérivées telles que le modèle de langue. Le premier modèle probabiliste était proposé par Maron Kuhns en 1960. Ils ont utilisé une approche mathématique basée sur la théorie des probabilités, d'où le processus de sélection des documents pertinents par un SRI se base sur le résultat de calcul d'une probabilité que le résultat soit pertinent. Il consiste en présentation des résultats de recherche dans un ordre basé sur cette probabilité, Robertson résume ce critère d'ordre dans [Robertson, 1977] par ce qu'il appelle le « Principe de classement probabiliste » désigné aussi par PRP (Probability Ranking Principle). Dans ce modèle de base, la réponse à une requête par un SRI commence par la spécification d'un ensemble de propriétés relatives à un ensemble de documents de sa collection nommé « Ensemble de réponse idéal »[1], il contient uniquement et rien que les documents pertinents. Il reste bien entendu qu'au moment de la requête, ces propriétés ne sont pas connues, mais elles sont fixées par des tentatives de recherche permettant de générer une première description probabiliste de cet ensemble, en suite, il faudra une interaction avec l'utilisateur pour améliorer cette description [Robertson, 1977]. Pour mesurer la pertinence des documents renvoyés aux utilisateurs, le modèle

[1] **Ensemble de réponse idéal :** ensemble de documents de la collection qualifiés de conteneurs des informations pertinentes par apprentissage.

probabiliste se base sur la distribution des termes dans les documents d'apprentissage [2] selon deux hypothèses :

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

Le processus de recherche donc se traduit par le calcul du degré de pertinence d'un document pour une requête par le billet de deux probabilités :

$P(w_{ij}/Pert)$ = Probabilité que le terme $t_i$ apparait dans le document $d_j$ sachant qu'il est pertinent,

$P(w_{ij}/\neg Pert)$ = Probabilité que le terme $t_i$ apparait dans le document $d_j$ sachant qu'il n'est pas pertinent,

Puis trier les documents selon leur probabilité de pertinence en utilisant une fonction de classement RSV :

$$RSV(d,q) = \frac{P(Per|q,d_i)}{P(\neg Per|q,d_i)} \qquad (1)$$

Avec : $P(Per|q,d_i)$ est la probabilité que le document $d_i$ soit pertinent pour la requête $q$ et $P(\neg Per|q,d_i)$ est la probabilité qu'il soit non pertinent.

Donc : il revient à sélectionner les documents ayant à la fois une forte probabilité d'être pertinents est une faible probabilité d'être non pertinents. En appliquant la formule de bayes qui est une **:** formule mathématique utilisée pour le calcul des probabilités conditionnelles, elle a la forme suivante :

$$p(A/B) = \frac{p(B/A) \cdot p(A)}{p(B)}$$

on obtient :

$$P(Per|q,d_i) = \frac{P(Per|q)P(d_i|Per,q)}{P(d_i)} \quad et \quad P(\neg Per|q,d_i) = \frac{P(\neg Per|q)P(d_i|\neg Per,q)}{P(d_i)}$$

Avec :

$P(d_i)$ Est la probabilité de choisir le document $d_i$ on considère qu'elle est constante,

$P(Per|q,d_i)$ Respectivement $P(\neg Per|q,d_i)$ est la probabilité que le document $d_i$ fait partie des documents pertinents ou non pertinents pour la requête *q,*

$P(Per|q)$ Respectivement $P(\neg Per|q)$ est la probabilité de pertinence ou non pertinence d'un document quelconque pour la requête *q,*

[2] **Documents d'apprentissage :** des documents sélectionnés de la collection par plusieurs tentatives de traitement de la même requête, pour les faire partie d'un ensemble de réponse idéal.

Avec : $P(Per|q) + P(\neg Per|q) = 1$

Par présentation binaire, soient : un document $D = (D_1, D_2 \dots, D_n)$ avec $D_{i(i=\overline{1,n})}$ des termes de ce document, tels que :

$$D_i = \begin{cases} 1 & si\ D\ le\ contient \\ 0 & sinon \end{cases}$$

Donc, il revient à déterminer les probabilités *P(L=1|D)* (probabilité que le document *D* soit pertinent) et *P(L=0|D)* (probabilité que le document *D* soit non pertinent). D'où la probabilité de pertinence revient à calculer la probabilité $P(L=1|D_1 , D_2 , \dots D_n)$ que le document *D* contient les termes $D_{i(i=\overline{1,n})}$. D'où la fonction de similarité (1) devient :

$$RSV(d,q) = \frac{P(L = 1|D1 , D2 , \dots Dn)}{P(L = 0|D1 , D2 , \dots Dn)}$$

L'un des systèmes qui ont implémenté le modèle probabiliste est le modèle OKAPI [Robertson & Walker, 1999], en utilisant la fonction BM25[3] pour calculer la probabilité de pertinence d'un document *D* vis-à-vis de la requête *q* :

$$RSV(D,Q) = \sum_{t\epsilon Q} w^{(1)} \frac{(K_1 + 1)tf}{K + tf} \frac{(K_3 + 1)Qtf}{K_3 + Qtf} \quad et\ K = K_1\left(1 - b + \frac{b * dl}{avdl}\right)$$

Avec :

*tf* : fréquence d'apparition du terme *t* dans le document *D*,

*Qtf* : fréquence d'apparition du terme *t* dans la requête *Q*,

$k_1$, *b* et $k_3$ : paramètres dépendants de la nature des requêtes et du corpus

*dl* : la longueur du document *D*,

*avdl* : la longueur moyenne des documents,

$w^{(1)}$: poids de Robertson-Sparck Jones du terme *t* dans la requête *Q* calculé ainsi :

$$w^{(1)} = log\left(\frac{\frac{r + 0.5}{R - r + 0.5}}{\frac{n - r + 0.5}{N - n - R + r + 0.5}}\right)$$

Où :

*r* : nombre de documents pertinents contenant le terme *t*,
*R* : nombre total de documents pertinents,
*n* : nombre de documents contenant le terme *t*,
*N* : nombre total de documents.

[3] **BM25 :** vous pouvez trouver des explications détaillées de cette formule ainsi que son utilisation dans [Belkacem, 2013]

L'avantage dans le modèle probabiliste de base est que la notion d'indépendance entre les termes est préservée par une loi de distribution appliquée aux termes et aux documents. Par contre, il présente l'inconvénient majeur qui est le problème lié au calcul des différentes probabilités pour tous les documents puis leur distribution. Ainsi que ce modèle de base représente une limitation majeure c'est qu'il ne tient pas compte de la fréquence d'un terme dans un document, or que ce critère est indispensable dans le calcul des scores de pertinence, et les méthodes qui en tiennent compte ont un impact négatif sur cette mesure, en effet : si dans un document un terme apparait 15 fois et dans un autre apparait 14 fois, le modèle de base considère que les deux fréquences sont différentes, or qu'elle sont sensiblement identiques [Benaouicha, 2009]. Dans ce modèle, l'idée de base est d'essayer de rapprocher la requête de l'un des documents de la collection, en calculant des valeurs probabilistes brèves basées sur les termes de la requête et ceux d'un document, avec l'estimation d'un ensemble de documents pertinents appelé « Documents d'apprentissage », voir aussi les limitations associées.

Pour pallier à ces problèmes, plusieurs dérivées du modèle probabiliste de base sont apparues, telles que le modèle de langue.

## 2. Les modèles de langue :

Les modèles de langue sont basés sur l'hypothèse suivante : « Un utilisateur en interaction avec un système de recherche, fournit une requête en pensant à un ou plusieurs documents qu'il souhaite retrouver » [Mataoui, 2007]. L'observation est que l'utilisateur crée la requête à partir d'une représentation hypothétique qu'il se fait du document recherché donc sa requête est inférée à partir des documents voulus. De ce fait, la requête est fournie au système de recherche en utilisant une langue précise et dans ces modèles, il revient à créer un modèle pour chaque langue, donc créer un modèle sur lequel se baser pour générer des requêtes à partir des documents de même langue et les comparer en suite au requête utilisateur.

Le domaine de la RI s'est beaucoup inspiré du succès des méthodes statistiques en linguistique informatique [4] [Boughanem, 2000]. Les premiers travaux qui ont porté sur l'utilisation des modèles de langue sont basés sur les techniques de l'informatique linguistique, car ce dernier et la RI ont en commun la spécificité de manipulation de grandes masses de texte.

### 2.1 Le modèle de langue en linguistique informatique :

Le but des modèles de langue est de déterminer la probabilité que la requête utilisateur Q puisse être inférée par le document D, notée *P(Q/D)* : ce qui est l'idée de base du modèle probabiliste [Maron, 1960], à l'exception de la façon dont cette probabilité est calculée ainsi que les paramètres qui y sont intégrés. Ce que nous allons voir dans la suite de ce travail. Cependant, le terme « Modèle de Langue » est emprunté de la linguistique informatique où un modèle de langue sert à contrôler la convenance et la justesse linguistique par des fonctions probabilistes.

[4] **La linguistique informatique :** est le domaine qui sert à analyser une langue par un ensemble de probabilités qui calculent l'estimation d'inférer une séquence de mots à partir des mots d'une langue donnée.

### 2.2 L'idée de base :

La modélisation statistique d'une langue est un processus de distribution de probabilité sur toutes les séquences possibles « *S* » dans cette langue. Ce processus génératif revient à déterminer une fonction probabiliste *P* qui va assigner une probabilité *P(S)* à *S* pouvant être un mot ou une séquence de mots de cette langue, une fois le processus est valide pour un mot, la fonction peut être distribuée sur une séquence pour estimer la probabilité de génération de n'importe quelle autre séquence de cette langue, autrement dit : la probabilité de générer une séquence de mots du modèle de langue correspondant.

Soit la séquence des mots suivants : $S = m_1, m_2, \dots m_n$

La probabilité *P(S)* associée à cette séquence est :

$$P(S) = \prod_{i=1}^{l} P(m_i | m_1, m_2 \dots m_{i-1})$$

Avec *l* est le nombre de mots prédécesseurs de $m_i$

Dans cette probabilité, un problème lié à la multiplicité des paramètres surgit, d'où les études nous ont conduites à la simplification suivante :

$$P(S) = \prod_{i=1}^{l} P(m_i | m_{i-n+1}, \dots m_{i-1}) \qquad (2)$$

Dont le mot $m_i$ ne dépend que de ses *n-1* prédécesseurs directs.

Dans ce cas, ce modèle est dit *n-gramme* (séquence à *n* mots). Dans le cas où l'on considère la génération d'un mot d'une langue, on parle de modèle *uni-gramme* d'où l'indépendance entre les mots de cette langue et la fonction (2) devient :

$$P(S) = \prod_{i=1}^{l} P(m_i)$$

De même pour :

Bi-gramme : $P(S) = \prod_{i=1}^{l} P(m_i | m_{i-1})$

Tri-gramme : $P(S) = \prod_{i=1}^{l} P(m_i | m_{i-2} m_{i-1})$

Ce que l'on doit estimer dans le modèle de langue en informatique linguistique sont les uni-grammes et les tri-grammes, pour une langue donnée. Cependant, il est difficile voir impossible d'estimer ces probabilités dans l'absolue, il faut par contre se baser sur un corpus textuel *C*, puis si le corpus est suffisamment grand, nous pourrons ainsi dire que le modèle de

langue établit est approximativement le modèle de cette langue : *p(.|c )* par hypothèse qu'il reflète cette langue en général.

Dans l'ensemble, ces probabilités sont calculées à base d'une estimation de vraisemblance maximale ML (Maximum Likelihood) qui est calculée selon les fréquences d'occurrence d'un n-gramme α dans le corpus *c* : la probabilité que la séquence α soit inférée du corpus *c* est estimée par :

$$P_{ML} = \frac{|\alpha|}{\sum_{\alpha_j} |\alpha|} = \frac{|\alpha|}{|c|}$$

Avec :

|α| : fréquence d'occurrence du n-gramme α dans le corpus *c*

$\alpha_j$ : n-gramme de même langueur que α

|*c*| : taille du corpus par nombre de mots = somme de toutes les occurrences de tout les termes

*Exemple :* [Boughanem, 2000]

Soit le corpus contenant les mots suivants : « le, un, prof, ML, dit, aime, de, langue, modèle, RI » représentant une langue.

Soit à estimer la probabilité *p(s)* d'inférer la séquence *s* de ce corpus avec *s* = «le prof aime le ML ».

Table de vraisemblance maximale :

| Mot | le | un | prof | ML | dit | aime | de | langue | modèle | RI |
|---|---|---|---|---|---|---|---|---|---|---|
| *freq.* | 3 | 2 | 2 | 1 | 2 | 1 | 4 | 2 | 1 | 2 |
| *P(.\|c)* | 0.15 | 0.1 | 0.1 | 0.05 | 0.1 | 0.05 | 0.2 | 0.1 | 0.05 | 0.1 |

**Tableau II-1 :** Table de vraisemblance maximale

Donc : |*c*| = 20 = $\sum freq.$

D'où :

$$P(s) \approx P(s|c) = P("le"|c) \times P("prof"|c) \times P("aime"|c) \times P("le"|c) \times P("ML"|c)$$
$$= 0.15 \times 0.1 \times 0.05 \times 0.15 \times 0.05 = 0.0000056$$

*Problème :* dans les grands corpus textuels, beaucoup de n-grammes n'apparaissent pas et ceci engendre l'attribution de probabilités nulles à l'estimation d'une séquence, car cette dernière est calculée à base d'un produit des probabilités des mots de la séquence. Ainsi, la séquence *s* = « le prof dit non » aura une probabilité nulle même si la majorité des mots sont présents dans le corpus (cas de l'exemple précédent) à cause du mot « non » qui n'y est pas présent. Pour remédier à ce problème, des techniques sont invoquées et exploitées pour éviter

à pénaliser les documents pouvant être pertinents dans la RI et la linguistique informatique, c'est éviter à attribuer une estimation de vraisemblance nulle pour un corpus. En se basant sur les caractéristiques de la collection d'entrainement et les probabilités d'y inférer la séquence en question. Ces techniques sont connues par : *techniques de lissage.*

### 2.3 Les techniques de lissage :

Le lissage est une fonction qui cherche à attribuer des probabilités non nulles à des séquences contenant des mots absents dans quelques éléments d'un corpus d'entrainement, afin de généraliser les modèles de langue, c'est une façon d'éviter le surentrainement d'un modèle sur un corpus et d'aboutir à un modèle à grande capacité de généralisation. Plusieurs méthodes de lissage existent dans la littérature, dont les méthodes classiques suivantes :

#### 2.3.1 Lissage de Laplace :

Ou méthode « ajouter-un » qui consiste à incrémenter de 1 toutes les fréquences des n-grammes, donc pour le n-gramme α :

$$P_{Laplace}(\alpha|c) = \frac{|\alpha| + 1}{\sum_{\alpha_i \in V}(|\alpha_i| + 1)}$$

Avec : *V* est le vocabulaire de la langue, et les autres paramètres sont tel déjà expliqués.

#### 2.3.2 Lissage Good-Turing :

Cette méthode se focalise sur la fréquence d'occurrence *r* observée pour un n-gramme α, on obtient la fréquence $r^*$ suivante :

$$r^* = (r + 1)\frac{n_{r+1}}{n_r}$$

Avec : $n_r$ est le nombre des n-grammes apparus *r* fois dans le corpus

$n_{r+1}$ est le nombre des n-grammes apparus *r+1* fois dans le corpus

D'où l'estimation suivante :

$$P_{GT}(\alpha) = \frac{r^*}{\sum_{\alpha_i \in c}|\alpha_i|}$$

Ainsi, la fréquence des n-grammes leur est appliquée une diminution d'ordre $\frac{r^*}{r}$

#### 2.3.3 Lissage de Backoff :

Consiste à utiliser un modèle du même ordre que le n-gramme recherché dans le corpus d'entrainement, si le premier est observé dans le dernier et un modèle d'ordre inférieur si non. Cas du lissage Katz qui combine un modèle bi-gramme avec un modèle uni-gramme :

$$P_{katz}(m_i|m_{i-1}) = \begin{cases} P_{GT}(m_i|m_{i-1}) \; si \; |m_i \, m_{i-1}| > 0 \; (bi-gramme) \\ \alpha(m_{i-1}) P_{katz}(m_i) \qquad\qquad\qquad\qquad sinon \end{cases}$$

Avec : $\alpha(m_{i-1})$ est un paramètre de normalisation par distribution des fréquences diminuées dans $P_{GT}$, calculé par :

$$\alpha(m_{i-1}) = \frac{1 - \sum_{m_i:|m_{i-1}\, m_i|>0} P_{katz}(m_i|m_{i-1})}{1 - \sum_{m_i:|m_{i-1}\, m_i|>0} P_{ML}(m_i)}$$

### 2.3.4 Lissage par interpolation :

Le lissage par interpolation tel appliqué par Jelinek Mercer, consiste à combiner un modèle d'ordre *n* avec un ou plusieurs modèles d'ordre inférieur, que d'utiliser ce dernier uniquement lorsque la séquence en question est de fréquence nulle ce qui est le cas dans le lissage de Backoff. Ainsi, on obtient le modèle suivant pour une combinaison des deux modèles bi-gramme et uni-gramme :

$$P_{JM}(m_i|m_{i-1}) = \lambda_{m_{i-1}} P_{ML}(m_i|m_{i-1}) + (1 - \lambda_{m_{i-1}}) P_{ML}(m_i) \qquad (1)$$

Avec : $\lambda_{m_{i-1}}$ est un paramètre dépendant du mot $m_{i-1}$ ou il peut avoir une valeur fixe pour tout les mots de la séquence et sert à mettre en valeur la séquence recherchée.

Typiquement, dans le cas de la RI, on ne considère que les uni-grammes, de ce fait, la probabilité $P_{JM}$ devient :

$$P_{JM}(m_i|d) = (1 - \lambda) P_{ML}(m_i|d) + \lambda P_{ML}(m_i|c) \qquad (2)$$

Ce modèle est conçu en combinant le modèle du document avec le modèle de la collection, par estimation maximale de vraisemblance ML.

### 2.3.5 Lissage de Dirichlet :

Comme le lissage précédent ne tient pas de la taille des échantillons d'entrainement, ou de la taille des documents (cas de la RI), Dirichlet a proposé une implémentation du modèle (2) en exploitant les valeurs de λ (2) en fonction de la taille des échantillons, et obtenir la forme suivante :

$$\begin{aligned} P_{Dir}(m_i|d) &= \frac{|d|}{|d| + \mu} P_{ML}(m_i|d) + \frac{|d|}{|d| + \mu} P_{ML}(m_i|c) \\ &= \frac{|d| P_{ML}(m_i|d) + \mu P_{ML}(m_i|c)}{|d| + \mu} \qquad (3) \end{aligned}$$

Or que :

$$P_{ML}(m_i|d) = \frac{|m_i|}{\sum_{m\, j \epsilon d} |m_j|} = \frac{tf(m_i, d)}{|d|}$$

D'où, (3) devient :

$$P_{Dir}(m_i|d) = \frac{tf(m_i, d) + \mu P_{ML}(m_i|c)}{|d| + \mu}$$

Avec : $|d|$ est la taille du document $d$.

$tf(m_i, d)$ est la fréquence brute du terme $m_i$ dans le document $d$

Remarque :

Plusieurs recherches dans le domaine de la RI ont montré que le chois de la méthode de lissage a un grand impact pour les SRI se basant sur les modèles de langue. D'où une conclusion a était faite par les recherches de [Zhai & Lafferty, 2001] sur le modèle de lissage de Dirichlet comme étant le modèle uni-gramme qui a donné les meilleurs résultats d'après leurs expérimentations. Ainsi, les modèles de langue ont été appliqués pour résoudre plusieurs problèmes dans la reconnaissance de la parole [Jelinek, 1997] et d'autres domaines linguistiques tels que la traduction automatique [Brown, 1993].

De ce fait, les modèles de langue sont appliqués aussi avec succès dans la RI, donc : Quelle est la performance ajoutée par ces modèles dans le monde de la RI ? Et comment sont traités ces modèles ? Ce sont des questions auxquelles répond la section suivante.

## 3. Les modèles de langue en RI :

### 3.1 Principe :

Dans le modèle probabiliste de base, la pertinence d'un document est estimée en fonction d'occurrence ou non-occurrence d'un ensemble de termes de la requête dans ce document (Section 1.), or que le modèle de langue qui est une dérivée de ce modèle de base, nous amène à voir la pertinence d'un document sous un autre angle.

Le principe des approches utilisant le modèle de langue est différent des autres modèles : on ne tente pas de modéliser directement la notion de pertinence dans le modèle, mais on la considère face à une requête, ie : la probabilité que cette requête puisse être générée par le document en question après avoir créer un modèle de langue spécifique à ce dernier. Donc : on suppose qu'à chaque document est associé un modèle de langue typique *Md* et soit la requête fournie est *q*, donc : la pertinence du document *d* vis-à-vis de la requête *q* est déterminée par : *P(q|Md)* qu'on lit : « probabilité de génération de la requête *q* par le modèle de langue du document *d* »

### 3.2 Approches d'exploitation du modèle de langue en RI :

En se basant sur la représentation des documents et des requêtes, ainsi que la fonction de similarité par calcul de probabilité, les approches de modélisation de langue pour la RI se classe en trois grandes catégories desquelles plusieurs méthodes sont dérivées, ces classes se résument ainsi :

1. Génération de la requête par le modèle de document (Query Likelihood Models) qui sont des méthodes appliquant le principe expliqué ci-dessus, donc par association d'une probabilité *P(q|Md)* à chaque document.

2. Génération de document à partir de la requête (Document Likelihood Models) contrairement à l'approche précédente, cette approche associe à chaque requête un modèle et calcule la probabilité de génération du contenu document de cette requête.
3. Similarité document-requête qui consiste à établir un modèle pour chacun des deux entités (document et requête) et calculer une probabilité de similarité entre les deux modèles, par calcul de l'entropie croisée KL (Kullback-Leiber).

Dans ces trois approches, les documents résultats sont classés par rapport à la probabilité présentée. Dans ce qui suit, des détails de chacune des approches.

### 3.2.1 Génération de la requête par le modèle de document : [Ponte & Croft, 1998]

Les auteurs voient que la requête utilisateur est créée à base d'une idée à priori qu'il se fait sur le document $d$ ou sont modèle $M_d$, donc il choisit les termes de ça requête qui doit être générée à partir d'un document pertinent, avec la probabilité suivante qui représente donc le score de pertinence de ce document :

$$Score(d,Q)=P(Q|M_d)$$

Dans les modèles de la RI, la requête est considérée comme étant une suite de mots clés : $Q = t_1, t_2, \dots, t_n$ et pour simplifier, les termes sont considérés indépendants les uns des autres. Dans cette approche, pas uniquement les termes de $Q$ présents dans la requête qui sont pris en compte, mais on fait la distribution de ces termes aux termes absents, par le modèle Bernoulli :

- Supposons que les termes de $Q$ sont : $t_1, t_2, \dots, t_n$ et les termes $t_{n+1}, t_{n+2}, \dots, t_l$ sont absents de la requête.

Donc, la probabilité $P(Q|M_d)$ est réexprimée ainsi :

$$P(Q|M_d) = P(t_1, t_2, \dots, t_n|M_d) \times P(\neg t_{n+1}, \neg t_{n+2}, \dots, \neg t_l|M_d)$$

$$= \prod_{i=1}^{n} P(t_i|M_d) \times \prod_{j=n+1}^{l} P(\neg t_j|M_d)$$

$$= \prod_{t_i \in Q} P(t_i|M_d) \times \prod_{t_i \in Q} (1 - P(t_i|M_d))$$

Puis ils ont proposé de calculer la probabilité $P(t_i|M_d)$, en s'inspirant du lissage de Backoff comme suit :

$$P_{PC}(t_i|M_d) = \begin{cases} P_{ML}(t_i|d)^{1-\hat{R}(t_i,d)} \times P_{avg}(t_i)^{\hat{R}(t_i,d)} & si\ tf(t_i,d) > 0 \\ P_{ML}(t_i|c) & sinon \end{cases}$$

Avec : $tf(t_i,d)$ est la fréquence brute du terme $t_i$ dans le document $d$.

$P_{ML}(t_i|d)$ est la probabilité par vraisemblance maximale de $t_i$ dans $d$.

$$\hat{R}(t_i, d) = \frac{1}{1+\bar{f}_{t_i}} \times \left(\frac{\bar{f}_{t_i}}{1+\bar{f}_{t_i}}\right)^{tf(t_i,d)}$$

$\bar{f}_{t_i}$ est la fréquence moyenne de $t_i$ dans les documents qui le contient.

$\hat{R}$ est le risque d'apparition du terme $t_i$ dans le document *d.*

$P_{avg}(t_i)$ est la probabilité moyenne du terme $t_i$ dans les documents qui le contient.

$P_{ML}(t_i|c)$ est la probabilité par vraisemblance maximale du terme $t_i$ dans la collection *c*

L'avantage dans cette approche est qu'elle met en évidence l'aspect de « spécificité du document » car elle considère les termes absents dans la requête.

### 3.2.2 Génération de document à partir de la requête : [Lavrenko & Croft, 2001]

Dans ce modèle, les auteurs ont proposé de voir la pertinence du document construite à base du sujet abordé par la requête, ie : estimer le modèle du document à partir de celui de la requête sans utiliser aucun corpus d'entrainement, c'est ce que les auteurs appellent le modèle de pertinence, noté $\theta_R$.

Donc, la probabilité devient $P(t|\theta_R)$ qui est la probabilité de générer un terme à partir du modèle de pertinence $\theta_R$, mais ce dernier n'est pas en corps connu, donc suggestion d'exploiter les documents retournés les mieux classés (feedback documents). Ce modèle est alors défini comme suit :

$$P(t|\theta_R) = \sum_{\bar{d}\in\bar{R}} \frac{P(t|d)P(q|d)P(d)}{p(q)} = \sum_{\bar{d}\in\bar{R}} P(t|d)P(q|d)$$

Avec : *R* est l'ensemble des documents feedback.

*P(q|d)* est le score de pertinence du document *d* vis-à-vis de la requête *q*.

### 3.2.3 Similarité modèle de document-modèle de requête :

Les modèles que nous venons de voir utilisent tous le modèle de 1$^{er}$ ordre (uni-gramme), ils se basent sur les mots simples, par hypothèse d'indépendance entre termes, cependant en réalité, les termes dans un document sont reliés ou bien dépendants les uns des autres, donc il s'avère évident de penser à un modèle combinant les termes par extension de l'ordre (utiliser les bi-grammes par exemple), pour mettre en évidence les relations potentielles entre termes. Ces relations sont de deux types :

#### 3.2.3.1 Les relations de proximité :

Appelées aussi les relations de surface ou de dépendance entre termes, elles sont établies en deux directions :

- Prise en compte des mots composés dans le modèle de langue, dont plusieurs travaux ont exploité ce type de mots. On trouve dans [Hammache, 2014] un aperçu de l'ensemble de ces travaux et le principe d'utilisation des mots composés dans le modèle de langue en RI.
- Prise en compte des proximités entre termes, dont le principe se base sur l'intuition que « un bon document est celui dans lequel les termes de la requête apparaissent proches les uns des autres », les travaux qui se basent sur cette direction se distinguent de la manière dont chacun intègre la proximité dans son modèle de langue. On peut trouver dans [Hammache, 2014] plus de détails sur cette approche.

#### 3.2.3.2 Les relations sémantiques :

Les modèles de langue qui prennent en compte les relations sémantiques entre les termes, soit modifient le modèle de la requête en ajoutant les termes liés, ce qu'on appelle « L'expansion du modèle de la requête », soit ils modifient le modèle du document en attribuant une plus grande probabilité aux termes liés aux termes du document par rapport à celles attribuées à ceux qu'ils ne le sont pas, ce qu'on appelle « L'expansion du modèle du document », ou exploiter ces deux approches conjointement.

### 4. Le modèle de langue et la longueur du document : [Achmoukh, 2006]

### 4.1 L'idée de base :

Ce modèle se base sur l'approche de génération de la requête utilisateur par le modèle de langue du document [Ponte & Croft, 1998] (Section 3.2.1), repris ainsi :

$$score(d, Q) = P(d/Q)$$

*Score*(*d,Q*) = score de pertinence du document *d* pour la requête *Q*

*P*(*d/Q*) = probabilité conditionnelle

Par la formule de Bayes, cette probabilité devient ainsi :

$$P(d/Q) = \frac{P(Q/d)P(d)}{P(Q)}$$

*P*(*Q*) est la probabilité de la requête *Q*

En partant du principe que l'utilisateur génère ça requête indépendamment du document *d*, ie : *Q* n'a aucune influence sur la pertinence de *d*, d'où *P*(*Q*) sera ignorée ainsi :

$$P(d/Q) = P(Q/d)P(d)$$

De ce fait, la longueur du document est incorporée dans sa probabilité *P*(*d*) ainsi :

$$P(d) = \frac{|d|}{\sum_{i=1}^{n}|d_i|} = \frac{|d|}{|C|}$$

Avec : $|d|$ est la longueur du document *d* et $|C|$ est la longueur de la collection.

Donc, on peut dire que dans cette approche, la probabilité d'obtenir le document *d* dépend de sa longueur, car la première est proportionnelle à la deuxième.

La probabilité conditionnelle $P(Q/d)$ est calculée ainsi :

Soit la requête $Q = m_1, m_2, \dots, m_n$ (suite de mots), donc :

$$P(Q/d) = \prod_{i=1}^{n} P(m_i/d)$$

Avec *d* étant un ensemble de mots représentant son modèle de langue *Md*, qui permet de calculer la probabilité de générer le mot $m_i$ par ce modèle par l'estimation de vraisemblance (section 2.2), notée :

$$P_{MLE}(m_i/Md) = \frac{tf(m_i,d)}{|d|}$$

Mais, d'après les études de pondération des termes (Loi de Zipf), les mots les plus fréquents dans un document sont des mots vides, car ils ne permettent pas la distinction entre les documents d'une collection du fait qu'ils sont distribués sur tous les documents de la collection. De ce fait, la distribution des termes dans la collection documentaire textuelle est prise en compte dans cette approche et est notée $P_I(m_i/Md)$ : la probabilité d'importance du mot $m_i$ dans le modèle de langue *Md* du document *d*. Elle est calculée ainsi :

$$P_I(m_i/Md) = \frac{\frac{tf(m_i,d)}{|D|}}{\sum_{k=1}^{n}\frac{tf(m_i,d_k)}{|d_k|}} = \frac{P_{MLE}(m_i/Md)}{\sum_{k=1}^{n}P_{MLE}(m_i/Md_k)}$$

Avec :

$\sum_{k=1}^{n}\frac{tf(m_i,d_k)}{|d_k|}$ est le facteur de distribution du terme $m_i$ dans la collection

*n* est la taille de la collection en nombre de documents

$tf(m_i,d_k)$ est la fréquence du terme $m_i$ dans le document $d_{k(k=1\dots n)}$

Cette mesure met le point sur la distribution d'un terme dans les documents d'une collection, car plus que le terme occure dans la plupart des documents avec une fréquence élevée, plus qu'il est moins important pour le document en cours. Et les mots les moins fréquents dans la collection sont les plus importants pour un document dont leur probabilité

$P_{MLE}$ est importante. Cette conclusion est illustrée par des exemples explicatifs dans [Achmoukh, 2006].

Après normalisation des probabilités associées à chacun des documents de la collection pour un terme $m_i$ (leur somme est égale à 1), la probabilité conditionnelle $P(Q/Md)$ est calculée ainsi :

$$P(Q/Md) = \prod_{i=1}^{n} P_I(m_i/Md)$$

Dont le problème de la probabilité nulle.

Pour pallier à ce problème, la mesure qui a été prise s'est basée sur la normalisation d'une technique de lissage, en se basant sur la distribution des probabilités.

### 4.2 Lissage et redistribution des probabilités :

La forme générale de la technique de distribution des probabilités est :

$$P(m_i/Md) \begin{cases} P_s(m_i/Md) & si\ tf(m_i, d) > 0 \\ P_u(m_i/Md) & si\ tf(m_i, d) = 0 \end{cases}$$

Avec :

$P_s(m_i/Md)$ probabilité du mot $m_i$ de la requête présent dans le document $d$

$P_u(m_i/Md)$ probabilité du mot $m_i$ de la requête absent dans le document $d$

Ainsi, on obtient :

$$P(Q/Md) = \prod_{i:tf(m_i,d)>0} P_s(m_i/Md) \times \prod_{i:tf(m_i,d)=0} P_u(m_i/Md)$$

Avec :

$P_s(m_i/Md) = P_I(m_i/Md)$

$P_u(m_i/Md) = P_{MLE}(m_i/Mc)$ tel que $Mc$ est le modèle de langue de la collection, où : $P_{MLE}(m_i/Mc) = \frac{tf(m_i,C)}{|C|}$ et $|C|$ est la taille de la collection en nombre de termes.

Par principe que l'importance d'un mot absent dans n'importe quel document de la collection est liée à sa fréquence dans celle-ci indépendamment de sa distribution dans les documents où il apparait, cette distribution est prise en compte dans le calcul d'importance d'un mot dans la collection :

$$P_I(m_i/Mc) = \frac{\frac{tf(m_i, C)}{|C|}}{\sum_{k=1}^{n} \frac{tf(m_i, d)}{|d_k|}}$$

Vous trouverez des exemples explicatifs dans [Achmoukh, 2006].

### 4.3 Expansion du modèle de document avec le modèle de la collection :

Vu que le document a généralement un modèle instable dû à l'absence de certains mots, il est habituellement combiné avec le modèle de la collection pour solutionner ce problème. Dans cette approche, la combinaison que l'auteur avait proposé est reliée à deux paramètres : $P_I(m_i/Md)$ et $P_I(m_i/Mc)$ définis précédemment, et on a aboutis à la formule suivante :

$$P_I(m_i/d) = \lambda_1 P_I(m_i/Md) + \lambda_2 P_I(m_i/Mc)$$

Ce qui donne :

$$P_I(Q/d) = \prod_{m_i \in Q} (\lambda_1 P_I(m_i/Md) + \lambda_2 P_I(m_i/Mc))$$

Avec :

$\lambda_1$ et $\lambda_2$ sont des paramètres de pondération de chacun des modèles *Md* et *Mc* dont : $\lambda_1 + \lambda_2 = 1$

Après développement, on obtient la formule suivante:

$$P_I(Q/d) = \prod_{mi \in Q_i} \left(\lambda_1 \frac{tf(m_i,d)}{\sum_{k=1}^{n} \frac{tf(m_i,d_k)}{|d_k|} \times |d|} + \lambda_2 \frac{tf(t_i,C)}{\sum_{k=1}^{n} \frac{tf(m_i,d_k)}{|d_k|} \times |C|}\right)$$

D'où la formule de score suivante :

$$score(Q,d) = \frac{|d|}{|C|} \times \left( \prod_{mi \in Q_i} \left(\lambda_1 \frac{tf(m_i,d)}{\sum_{k=1}^{n} \frac{tf(m_i,d_k)}{|d_k|} \times |d|} + \lambda_2 \frac{tf(t_i,C)}{\sum_{k=1}^{n} \frac{tf(m_i,d_k)}{|d_k|} \times |C|}\right)\right)$$

Le modèle que nous avons présenté dans cette section (section 4.) est un modèle conçu en 2006 par Farida Achmoukh, c'est un modèle ayant pour but la prise en compte de la taille du document dans la construction du modèle de langue correspondant à un document, pour calculer le score de pertinence suite à une requête utilisateur, car il prend en compte la distribution des mots dans les documents de la collection, ce modèle est comparé au modèle de langue basique dans [Achmoukh, 2006] et les résultats montrent son efficacité. C'est pour cela que nous nous sommes basé sur ce modèle dans la réalisation de notre contribution, après avoir étudié les modèles de langue appliqués aux documents XML dans la partie suivante.

## 5. Evaluation des modèles de langue :

Comme dans tout domaine d'étude et de science, les solutions et les approches mises en place doivent subir une évaluation de qualité pour pouvoir en choisir la meilleure. Dans la

RI, tout comme les systèmes de recherche d'information, les modèles de langue sont évalués et comparés. La qualité d'un modèle de langue peut être exprimée par une valeur dite de « perplexité » ou indécision [Estève, 2002]. Cette mesure de valeur *k* s'interprète par le fait que le modèle puisse déduire d'un historique de *k* termes introduits, celui qui est le prochain émis par un utilisateur dans sa requête.

Cette valeur notée *PP* est calculée par : $PP = 2^{LP}$

Avec : *LP* ou *LogProb* désigne l'approximation moyenne d'émission d'une séquence de mots *s* de l'historique des mots *h,* sa valeur est donnée par :

$$PL = -\frac{1}{n} logP(s) = -\frac{1}{n} \sum logP(m_i/h)$$

*Interprétation :*

La perplexité représente le degré d'incertitude du modèle de langue évalué, donc plus sa valeur est petite plus le modèle est performant, ie : sa valeur de prédiction est importante. Deux types de perplexité existent :

- Perplexité calculée sur le corpus d'apprentissage du modèle, elle permet de décider la qualité des approximations ou probabilités utilisées pour définir le modèle ainsi que la qualité de ses paramètres.
- Perplexité calculée sur le corpus de test, elle permet de juger la capacité à la généralisation associée au modèle.

**Conclusion :**

Le modèle de langue est une variante du modèle probabiliste de base, nous avons vu au cours des paragraphes précédents que c'est un modèle adapté premièrement dans la linguistique informatique, l'idée de base est de calculer la probabilité que la requête *Q* soit générée par le corpus *C*, ce qui est appliqué en recherche d'information traditionnelle (dans les documents plats). Des problèmes ont été constatés en appliquant les modèles de langue à la RI, tels que les probabilités nulles, mais des études ont été faites pour pallier à ces problèmes qui surviennent lors de la distribution des probabilités aux termes de la requête *Q*, ainsi que des problèmes liés à la taille variable des documents, en appliquant des techniques dites de lissage ou de combinaison des modèles de langue du document et de la collection, ce qui a fait du modèle de langue un ensemble riche de métriques de recherche, dans le but de satisfaire le besoin de l'utilisateur en information pertinente. De ce fait, comment ces métriques sont-elle adaptées dans la recherche d'information structurée et semi-structurée appliquée aux documents XML ? La partie suivante résume l'ensemble de ces concepts.

**Partie 2 : Le modèle de langue et la RI structurée**

**Introduction :**

La RI structurée a évolué dans le but d'approcher la pertinence système de la pertinence utilisateur pour répondre à ses requêtes, cette évolution a inclus l'utilisation de différents modèles de la RI classique tout comme le modèle de langue.

Les principes du modèle de langue fut développés en linguistique informatique et appliqués dans la RI classique, sont adapté à la RI structurée pour les appliquer aux documents structurés et semi-structurés de type XML. Dans cette partie, nous allons présenter les modèles de langue du point de vu RI structurée dans les documents XML (RI XML): Comment la structure du document est elle exploitée ? Sur quoi se base-t-on dans le processus d'appariement document-requête ? Et comment les facteurs et les paramètres des formules sont ils appliqués ?

1. **Utilisation de la structure dans le processus d'appariement :**

Dans les modèles de langue appliqués à la RI traditionnelle, le modèle de langue résume la présentation du document *d* : *Md* ainsi que la proximité d'aboutir à une séquence de mots de ce modèle. Cependant, les modèles de langue appliqués à la RI XML remplacent le modèle du document *Md* par le modèle de l'élément *Me*, car la RI XML représente une nouvelle granularité de l'information, l'unité de réponse ciblée n'est plus un document tout entier mais une partie de celui-ci. De ce fait, le modèle *Me* est la plateforme basique de l'estimation d'une requête *Q* à partir d'un document *d* est donnée par : [Baeza & Ribeiro, 2011].

$$P(Q/Me) = P_{t_i \in Q}(t_1, t_2, \dots, t_n|Me) = \prod_{i=1}^{n} P(t_i|Me)$$

De ce fait, la probabilité d'aboutir à un élément en réponse à une requête est évalué dans [Sigubjörnsson & al., 2004] au travers d'un lissage de Jelinek-Mercer (Voir partie 1 section 2.3.4) pour obtenir la formule suivante qui fait l'interpolation entre la probabilité du terme *t* dans l'élément *e* et sa probabilité dans la collection *C* :

$$P(t/Me) = \lambda P(t|e) + (1 - \lambda)P(t|c)$$

λ: est une constante de pondération de la séquence de mots recherchée.

D'autres auteurs, en suite, ont intégré la taille de l'élément lors de la création du modèle, dans la probabilité associée à un élément donnée [Lalmas, 2009] :

$$P(e) = \frac{|e|}{\sum_{i \in C} |e_i|}$$

$|e|$ : taille de l'élément en nombre de termes

*C* : collection de test ou d'apprentissage

Dans [Li & Vander Weide, 2010] les auteurs séparent les conditions de contenu et les contraintes structurelles du composant résultat, et les combinent pour calculer le score du document entier, par croisement entre le modèle de la requête et celui du document normalisé sur la taille de celui-ci.

## 2. Utilisation de la structure du document dans les modèles de langue :

Dans les modèles de langue appliqués à la RIS, nous identifions deux grandes familles d'approches pour l'utilisation de la structure :

- La contextualisation des éléments : revient à faire l'interpolation sur le modèle d'un élément pour couvrir le contenu d'un document,
- L'agrégation : sert à considérer l'élément comme une agrégation de ses descendants.

### 2.1 La contextualisation des éléments :

Selon [Cyril, 2013], c'est un ensemble de techniques qui cherchent à évaluer la pertinence d'une unité de texte, au moyen d'un ensemble d'informations obtenu en dehors de cette unité et au travers d'autres unités qui l'entourent. Donc, la contextualisation d'un nœud ou bien d'un élément évalue la pertinence de ce dernier par combinaison de son score avec les scores d'autres éléments de niveaux différents (nœuds feuilles, lien parental…). Il considère que les techniques de lissage appliquées à la RI XML est une sorte de contextualisation, du fait que des paramètres qui prennent en compte les caractéristiques structurelles sont intégrés dans les formules de calcul d'estimation des éléments.

En 2010, Mulhem et Chevalet proposent dans [Mulhem & Chevalet, 2010] d'uniformiser tout les modèles des éléments en fonction de leur taille, car ils considèrent l'élément selon son type (label), donc considérer les termes associés à des éléments de même label particulier par rapport aux termes de la collection, ils ont proposé le modèle suivant :

$$P_{\mu,l}(t_i|e_l) = \frac{tf(t_i, e_l) + \mu P(t_i, C_l)}{|e_l| + \mu}$$

Avec : μ une constante et $e_l$ l'élément de label particulier.

$C_l$ est la collection dont les éléments sont de label $l$

$t_i$ est le terme recherché.

On trouve aussi dans [Ganguly & al., 2011] l'extension du concept de recherche par élément introduit par [Hiemstra, 2003] par l'interpolation entre le modèle de l'élément en cours et celui de son parent, favorisant la pertinence du document en présence d'éléments non pertinents dans celui-ci, dans une collection XML :

$$P(t_i|Me) = \lambda \times P(t_i|e) + (1 - \lambda) \times (\beta \times P(t_i|D) + (1 - \beta) \times P(t_i|C))$$

Avec λ et $\beta$ des paramètres de pondération de chacun de : document et collection respectivement.

### 2.2 L'agrégation :

Le principe est de combiner des représentations des éléments de niveau inférieur (fils) avec celle de l'élément en cours en fonction de la structure du document [Larson, 2006]. De ce fait, si l'on considère que seuls les nœuds feuilles sont munis de l'information textuelle, alors, cette dernière liée aux nœuds parents (éléments internes) est obtenue par l'agrégation des éléments fils.

En effet, cette fonction identifie les relations structurelles entre les éléments d'un document en mettant en évidence l'information textuelle, le score est alors évalué par l'interpolation entre les modèles de langue des fils qui y sont associés. La formule est la suivante :

$$P(t_i|Me) = \lambda_0 P(t_i|Me) + \sum_{j=1}^{l} \lambda_j P(t_i|Me_j)$$

Avec : *e* est l'élément ayant *l* fils,

*Me* est le modèle associé à *e*,

$Me_{j(j=\overline{1,l})}$ sont les modèles associés à chacun des fils $e_j$ de *e*,

λ est un paramètre d'influence de chaque modèle d'un élément, tel que :

$$\lambda_0 + \sum_{j=1}^{l} \lambda_j = 1$$

## 3. Traitement des contraintes structurelles des requêtes CAS:

D'après [Cyril, 2013] et les classifications de [Trotman, 2009] et [Lalmas, 2009], les contraintes de structure que l'utilisateur puisse exiger dans sa requête, peuvent être traitées de trois manières :

Par :

- Classification[5] des éléments selon leur label ;
- Calcul du score selon les conditions de structure : générer une suite de probabilités associées à chacun des éléments répondants à chacune des contraintes de score puis faire l'interpolation ;

[5] **Approche de classification des éléments :** construire des classes d'éléments équivalents en fonction de leur label (balise type). Vous trouverez dans [Ramirez, 2011] un exemple explicatif.

- Prise en compte de la notion de l'hiérarchie et de distance entre les éléments, pour déterminer quels modèles éléments *Me* à prendre en compte dans l'interpolation dans le processus de calcul de score par estimation.

Dans cette optique, des auteurs partisans de la troisième approche, considèrent que les éléments pertinents ont tendance à apparaitre au début du document XML, ils calculent la probabilité d'un élément de la manière suivante : [Huang & al., 2007] [Huang, 2008]

$$P(e) = \frac{1}{5 + |e_{location}|} \times \frac{1}{3 + |e_{path}|}$$

Avec :

$e_{location}$ est une valeur d'emplacement dans un chemin XPath[6] et $|e_{location}|$ est le nombre d'éléments de même emplacement,

$e_{path}$ est la longueur du chemin vers l'élément *e* à partir de la racine et $|e_{path}|$ est le nombre des éléments de même longueur du chemin,

*Exemple :*

Soit un élément de label = « *paragraphe* » et que son chemin depuis la racine est : *//article [1]//section[2]//paragraphe[1]*

Donc : $e_{location} = 1$ et $e_{path} = 3$

Une autre approche élémentaire [Zhao & Callan, 2009] qui est une extension du modèle de langue de *Indri*[7] [Strohman & al., 2005] a l'intuition que la requête est une procédure de génération des termes à partir d'un document, ie : pour construire sa requête, l'utilisateur au lieu de s'inspirer de la structure dans la détermination de l'emplacement des termes, la requête prend en considération la manière dont chacun des éléments génère les termes de la requête.

4. **Utilisation explicite de la structure des documents vis-à-vis des contraintes des requêtes :**

Dans les approches utilisant les modèles de langue comme plate forme d'appariement, l'idée de base est d'inclure la structure du document dans la construction du modèle associé à un document, du fait que les approches de comparaison d'arbres document n'y sont pas facilement intégrables. Ce qu'on appelle *le processus de relaxation de structure.*

De ce fait, dans [Chihebddine, 2011] l'auteur utilise la relaxation d'arcs de [Ben Aouicha & al., 2010] afin d'obtenir un modèle de document composé d'un ensemble de liens entre tout les nœuds à relation de type : ancêtre-descendants deux à deux et propose d'utiliser deux fonctions de lissage : Dirichlet et Jelinek-Mercer (voire partie 1 : Techniques de lissage).

[6] **XPath :** voir chapitre I partie 2 : « la galaxie XML »
[7] **Indri :** est un moteur de recherche construit par Strohman et d'autres en 2005, en se basant sur les modèles de langue.

Cette approche est restée dans le cadre théorique puisqu'elle n'a été ni implémentée ni testée [Cyril, 2013].

**5. Un modèle de langue basé sur la structure de document :** [Cyril, 2013]

**5.1 Prise en compte de la structure du document XML :**

La faible utilisation de la structure dans la RIS pour les modèles de langue, d'après Cyril s'explique par le fait que l'expression du besoin en information de l'utilisateur sous forme de requête, porte en priorité sur le contenu souhaité plutôt que sur l'emplacement de ce dernier dans l'arbre du document XML.

L'auteur conduit sa contribution dans l'objectif de proposer un modèle qui prend en compte l'utilité de la structure d'un document et l'utiliser efficacement. Donc, une requête cible un élément à balise (label) spécifique, et le score de pertinence du nœud répondu est donné par la formule suivante :

$$RSV(n,Q) = \begin{cases} 0 & si\ l(n) \neq EC(Q) \\ \lambda \times score_c(n,Q) + (1-\lambda) \times score_s(n,Q) & si\ non \end{cases}$$

Avec :

$score_c$ est score de contenu pour le nœud *n*

$score_s$ est le score de similarité structurelle pour *n*

$\lambda$ □ [0,1] est le paramètre d'apport du $score_c$ et $score_s$

$EC(Q)$ est la fonction qui retourne la balise spécifique par la requête *Q*

$l(n)$ est la balise du nœud *n*

Cette approche commence par l'évaluation des nœuds feuilles qui contient éventuellement l'information textuelle, ceci par le calcul d'une mesure de score dite « *score de contenu* » basé sur l'approche *tf*idf* [Sparck Jones, 1973], ainsi :

$$score_c(x) = \sum_{t_i \square Q} tf(t_i|x) \times idf_{t_i}$$

Avec *x* est un nœud feuille de la collection considérée.

Un score est attribué ensuite aux nœuds interne, par une double propagation [Laitang &Pinel-Sauvagnat, 2011] [Laitang & al., 2011] :

- *Bottom up* : propagation du score *P(x)* à la racine, ainsi chaque nœud interne *n*, obtiendra un score *P(n)* par cumulation des scores qui le traverse (du bas vers le haut).
- *Top down* : propagation des scores des ancêtres jusqu'aux nœuds feuilles par intuition qu'un nœud pertinent appartient à un document qui l'est aussi.

D'où, le score de contenu brut d'un nœud *n* est donné par la formule de propagation suivante :

$$bs_c(n) = \begin{cases} I + V + A & si\ n \neq racine \\ I & si\ non \end{cases}$$

$$I = \frac{P(n)}{|feuilles()|} \quad score\ intermèdiaire\ de\ contenu$$

$$V = \frac{P(a_1) - P(u)}{|feuilles(a_1)|} \quad score\ de\ voisinage$$

$$A = \frac{bs_c(a_1, Q) - \frac{P(a_1)}{|feuilles(a_1)|}}{|fils(a_1)|} \quad score\ des\ noeuds\ \text{ancêtres}$$

$bs_c(a_1, Q)$ est le score brut du nœud $a_1$à la requête *Q*

Avec :

$$P(n) = \frac{\sum_{x \in feuilles(n)} P(x)}{|feuilles(n)|}$$

$|feuilles()|$ : est le nombre de nœuds feuilles du document

$a_1$ : nœud père

$P(a_1)$ : score intermédiaire de $a_1$

$|feuilles(a_1)|$ : nombre de nœuds feuilles de $a_1$

Après normalisation, le score final est obtenu par le maximum *max()* des scores de contenu brut des nœuds $n_i \in C$ obtenus en réponse à la requête *Q* :

$$score_c(n, Q) = \frac{bs_c(n, Q)}{\max(bs_c(n_i)_{n_i \in C})}$$

Avec :

$score_c(n, Q) \sum_{q_i \in Q} score_c(n, q_i)$ et $Q = q_1, q_2, \dots q_i, \dots q_k$

Le score de contenu brut d'un nœud interne est calculé par la somme des scores de contenu pour chaque sous-requête :

$$bs_c(n, Q) = \sum_{q_i \in Q} \frac{score_c(n, q_i)}{\max(score_c(n_i, q_i)_{n_i \in C})}$$

Ce qui est du score de similarité structurelle : $score_c(n, Q)$ du nœud *n* est donnée par :

$$score_c(n, Q) = bs_c(n, Q) = sim_s(S, Q)$$

Avec :

*S* est un sous-arbre extrait du document pour répondre à la contrainte structurelle.

$sim_s(S, Q)$ est la probabilité entre la requête Q et le sous-arbre *S* calculée par modèle de langue en utilisant l'une des formules proposées par l'auteur.

**5.2 Estimation de la probabilité de génération de la structure de la requête :**

L'auteur considère que le document étant un arbre XML est généré par un modèle de langue $M_T$ : modèle de langue associé à l'arbre *T* du document considéré, et les arbres candidats $T_i$ pour *i* chacun des candidats, sont classés selon leur probabilité de génération de la requête à partir de leur modèle de langue $M_{T_i}$. D'où : $sim_s(M_T, Q)$ est la probabilité de similarité entre la requête *Q* et l'arbre *T*, donnée par la probabilité suivante :

$$P(Q|M_T) = \prod_{e_{i,j} \in Q} P(e_{i,j}|M_T)^{w(e_{i,j},Q)}$$

Avec :

. $e_{i,j}$ est l'arc $u \rightarrow v$ de la requête pour lequel on calcule la probabilité $P(e_{i,j}|M_T)$ que $e_{i,j} \in T$

. $w(e_{i,j}, Q)$ est le poids calculé selon la distance entre les nœuds *u* et *v*, calculé par :

$$w(e_{i,j}, Q) = \sum_{u,v \in Q / \ label(u)=i \ et label(u)=j} w(e_{i,j}^{u \to v}) = exp^{1-d(u,v)} \quad (1)$$

Tel que : *d(u,v)* est la distance séparant *u* et *v* dans l'arbre de la requête en nombre d'arcs.

### 5.3 Techniques de lissage basées sur la structure :

Etant donné ce modèle, classe les arbres à partir de la probabilité d'occurrence des arcs de la requête $e_{i,j} \in T$ et que les arcs absents se verront assigner une probabilité nulle, un arc manquant peut disqualifier le sous-arbre dans son ensemble en lui assignant une probabilité nulle. Pour cela, l'auteur à utilisé des techniques de lissage des probabilités pour pallier au problème des probabilités nulles :

- **Lissage de Jelinek-Mercer :**

Le modèle d'arbre $M_T$ est lissé à partir de tous les arcs compris dans la collection :

$$P(e_{i,j}|M_T) = \left(\alpha \times P(e_{i,j}|M_T)\right) + (1 - \alpha) \times P(e_{i,j}|M_C)^{w(e_{i,j},Q)}$$

Avec :

$P(e_{i,j}|M_T) = \frac{w(e_{i,j},T)}{w(T)}$ est la probabilité de pertinence d'un arc dans le document à arbre *T* telle que : $w(e_{i,j}, T)$ calculé par (1) avec substitution de *Q* par *T*

$w(T) = \sum_{e_{i,j} \in T} w(e_{i,j}, T)$ et $P(e_{i,j}|M_C) = \frac{w(e_{i,j},C)}{w(C)}$ avec $w(C) = \sum_{e_{i,j} \in C} w(e_{i,j}, C)$ et $w(e_{i,j}, C)$ est calculé par (1) avec substitution de *Q* par *C*

α: est une constante de normalisation.

- **Lissage de Dirichlet :**

En se basant sur la longueur de document :

$$P(e_{i,j}|M_T) = \frac{w(e_{i,j}, T) + \mu \times \frac{w(e_{i,j}, M_D) + \lambda P(e_{i,j}|M_C)}{w(D) + \lambda}}{w(T) + \mu}$$

Avec :

$\lambda$ et $\mu$ sont des paramètres de lissage utilisés pour contrôler l'influence de la collection sur le score final,

$w(D) = \sum_{e_{i,j} \in D} w(e_{i,j}, D)$ dont $w(e_{i,j}, D)$ est calculé par (1) avec substitution de *Q* par *D*.

## 6. Discussion :

La recherche d'information représente une taxonomie riche de modèles, dans ce travail nous nous somme intéressés aux modèles de langue, bien précisément les modèles de langue appliqués à la RIS dans les documents XML (RI XML).

En effet, le modèle de langue étant une variante du modèle probabiliste de base, son principe est que la requête utilisateur peut être inférée par un document particulier, de ce fait, les travaux accomplis dans le cadre de la RI traditionnelle proposent des techniques d'appariement diversifiées entre un document de la collection et la requête de l'utilisateur fournie à l'interface du moteur de recherche, cela en implémentant les formules d'estimation du score et qui sont établies dans le modèle probabiliste de base, en partant de la connaissance des modèles de documents de la collection textuelle et du modèle de la requête étant une séquence de mots.

Dans la RI XML, sont peu les travaux présentés pour les modèles de langue. Mais, d'après l'étude des travaux proposés dans le cadre de la RIS par modèle de langue, il reste d'actualité que le principe d'adaptation du modèle de langue à la RIS est que le modèle du document doit inclure sa structure, car l'information portée par un document n'est plus simple, mais c'est une information composée : l'information textuelle et l'information structurelle. De ce fait, le modèle de document est traduit par l'ensemble des modèles de ses éléments le composant, du fait que l'élément est l'unité la plus miniaturisée du document structuré ou semi structuré et c'est lui qui nous conduit à l'information pertinente recherchée. Dans cette optique, les auteurs de la RI XML s'intéressant au modèle de langue, ont présenté des approches qui font l'interpolation entre le modèle d'un élément (comme mesure d'inclusion de la structure), le modèle du document (comme mesure de mise en faveur l'information textuelle) et le modèle de la collection (comme mesure de généralisation). De ce fait, la recherche d'une séquence de termes dans un document par modèle de langue XML met en évidence la pertinence de l'élément, ses prédécesseurs et le document tout entier, ie : c'est une adaptation de la nouvelle granularité d'information au modèle d'appariement, soit par contextualisation d'un élément dans le modèle du document, soit par l'agrégation des informations présentes dans chacun des éléments pour construire le modèle de langue.

Dans toutes les approches d'adaptation du modèle de langue à la RI XML, les auteurs prenaient en compte le modèle de l'élément soit par inclusion de la taille de ce dernier et la fréquence d'un terme dans celui-ci dans le modèle du document, soit par établissement des liens parentales entre les éléments d'un document par relation « ancêtre-descendant » dans le modèle du document, soit par inclusion de l'indexation de la structure ou autres, mais les relations entre : *les termes et les éléments, la taille d'un élément en nombre de termes, la taille d'un document en nombre d'éléments et la taille de la collection en nombre d'éléments*, ne sont pas inclues, ce qui est un ensemble de paramètres qui peuvent générer une méthode efficace dans l'exploitation du modèle de langue à la RIS.

Notre contribution est construite dans cette optique et se base sur l'appariement entre le modèle du document et celui de la requête, pour la construction d'un modèle de longue efficace à la RI XML.

**Conclusion :**

Dans cette partie, on a vu que le modèle de langue été utilisé au préalable dans la linguistique informatique, puis suite aux succès de ces modèles, ils sont appliqués à la RI traditionnelle, dont le point commun entre ces deux domaines est la manipulation des gros volumes textuels. Vue que la RI, avec le développement du web, exploite les documents XML, les auteurs s'intéressant au modèle de langue ont fait adapter ce dernier à la RI moderne, pour trouver de nouvelles solutions qui peuvent réunir les requêtes de l'utilisateur et son idée sur les documents qui lui seront retournés avec le modèle de langue des documents de la collection de recherche.

Au cours de ce chapitre, nous avons remarqué que plusieurs auteurs ont présenté des travaux et des approches pour les modèles de langue appliquées à la RIS, mais les méthodes proposées sont si nombrables. C'est pour cela que nous nous sommes inspirés des méthodes existantes pour proposer notre contribution de développement d'un modèle de langue appliqué à la RI XML. Ce qui est l'objet du chapitre suivant.

# CHAPITRE III : UN MODELE DE LANGUE POUR LA RI DANS LES DOCUMENTS XML

## Partie 1 : Approche proposée

### Introduction :

La RI XML (Recherche d'Information dans les documents XML), comme la RI traditionnelle (dans les documents plats), exploite l'ensemble des modèles existants dans la littérature, présentés dans le Chapitre I dont le but est de réaliser des approches et des systèmes de recherche d'informations (SRI) flexibles et exploitant au mieux la structure du document XML pour répondre à une requête utilisateur. Ainsi, la RIS exploitant le modèle de langue, doit aussi exploiter cette structure pour calculer la probabilité que le modèle structuré puisse générer cette requête, ce qui lui attribue un bon score de pertinence.

Dans cette partie, nous présenterons la problématique liée à l'adaptation du modèle de langue à la nouvelle granularité de l'information, ainsi que notre contribution par la présentation d'un modèle de langue exploitant la structure d'un document XML, pour présenter par la suite les testes et les résultats auxquels nous avons aboutis.

### 1. Problématique :

Dans la RI, le but de tout modèle est de rapprocher la pertinence système de la pertinence utilisateur, donc c'est d'utiliser le maximum de facteurs qui traduisent au mieux le contenu du document recherché. Dans les modèles probabilistes généralement et les modèles de langue précisément, plusieurs facteurs de pertinence ont été étudiés et mis en œuvre, sous forme de paramètres intégrés dans les formules d'appariement et de calcul de score de pertinence d'un document recherché.

Dans le chapitre précédent, nous avons étudié les paramètres à prendre en considération dans un modèle de langue, car ce dernier perçoit la pertinence sous un format différent dans le calcul des probabilités : contrairement au modèle de base, le modèle de langue se base sur la probabilité qu'une requête utilisateur soit générée par un modèle de document : en général. Dans cette optique, plusieurs travaux ont été présentés dans la littérature pour la RI dans les documents plats pour le modèle de langue, dont l'approche se basant sur la taille du document [Achmoukh, 2006] résumée dans le chapitre précédant en Partie 1. Mais concernant la RI XML, nous avons constaté dans la Partie 2 que certains auteurs se sont intéressés à proposer des approches combinant à la fois la contrainte structurelle d'un document XML et le principe d'appariement dans les modèles de langue.

Dans la RI classique, un modèle de langue présente le modèle du document, le modèle de la requête et celui de la collection en terme des mots les constituant, donc en se basant sur le fait qu'ils sont des suites de mots clés. Concernant la RI XML, un modèle de langue d'un document XML combine les modèles de langue des éléments le constituant, et le modèle de la collection se base sur les modèles des éléments de ses documents, tel qu'tel qu'un document XML est une arborescence d'éléments : le nœud feuille (qui contient l'information textuelle qui est exploitée dans le processus d'appariement), le nœud interne (ayant des descendants) et le nœud unique racine (qui représente le document XML complet). Donc, dans une approche

se basant sur les modèles de langue dans la RI XML, un auteur doit adapter les caractéristiques de ces modèles à la nouvelle granularité de l'information présentée dans les documents XML.

De ce fait, le processus de calcul du score de pertinence dans la RI XML, doit prendre en compte ces notions :

- Un document XML est une arborescence d'éléments,
- Un nœud feuille est un ensemble de termes,
- Un élément peut être un nœud interne ou feuille,
- La collection XML est un ensemble de documents XML, donc un ensemble d'arborescences.

C'est précisément dans cette optique que nous avons proposé notre contribution. Le calcul des probabilités de pertinence d'un composant retourné à l'utilisateur doit prendre en compte la pertinence d'un élément (d'un nœud feuille ou d'un nœud racine) et l'importance d'un terme dans une collection de documents structurés sera le résultat de combinaison de sa probabilité dans les éléments cités ci-dessus, ainsi, l'importance des termes ou la probabilité d'avoir la requête utilisateur à partir d'un document XML sera le résultat de distribution de l'importance des termes la constituant.

## 2. Contribution :

Dans la réalisation de notre travail, nous avons étudié l'ensemble des paramètres à prendre en compte dans l'adaptation du modèle de langue à la nouvelle granularité d'information, celle contenue dans les documents semi-structurés XML, car selon la structure d'un document et son contenu textuel que nous pourrions en extraire un contenu informatif combinant à la fois la structure et l'information recherchée par l'utilisateur. Nous avons abouti à cet ensemble de paramètre que nous utiliserons par la suite dans l'établissement de notre approche:

*dcel* = La taille d'un document en nombre d'éléments

$|e_j|$= La taille de l'élément $e_j$ en nombre de termes

$\Delta|e|$= La taille moyenne d'un élément dans le document en cours, calculée par :

$$\frac{taille\ du\ document\ en\ nombre\ de\ termes}{taille\ du\ document\ en\ nombre\ d'éléments} = \frac{\sum_{i=1}^{n}|e_i|}{dcel}$$

$tf_{ji}$ = Fréquence du terme $t_i$ dans l'élément $e_j$

*Nf* = Nombre de nœuds feuilles dans le document en cours

*Nfc* = Nombre de nœuds feuilles dans la collection

$|D| = \sum_{j=1}^{Nf} |e_j|$ La taille du document en nombre de termes

$|ed_i|$ = Nombre d'éléments qui contiennent le terme $t_i$ dans le document en cours

$|ec_i|$= Nombre d'éléments qui contiennent le terme $t_i$ dans la collection des documents XML.

## 3. Modèle de langue pour la RI XML :

### 3.1 Idée de base :

Dans notre travail, nous nous sommes inspirés du travail présenté dans [Achmoukh, 2006] exploitant la taille d'un document plat (texte). Par adaptation à la nouvelle granularité d'information.

Pour commencer la recherche, et répondre à une requête utilisateur, nous devons examiner les nœuds constituant un document de la collection, pour cela, nous calculerons la probabilité d'avoir l'élément *ej* comme résultat ;

Ensuite, nous allons estimer une probabilité conditionnelle reliant la requête *Q* à un élément *ej*, c'est la probabilité de similarité élément-requête, en deux phases :

1. Calculer la probabilité d'importance d'un terme $t_i$ dans un élément *ej,* au biais de deux algorithmes différents ;
2. Calculer le produit de ces probabilités pour tous les termes de la requête : c'est la distribution aux termes de la requête *Q ;*

Enfin, nous calculerons le produit entre la valeur de similarité ainsi obtenue et la probabilité associée à l'élément *ej,* ce qui nous donne le score de pertinence de l'élément *ej* pour la requête *Q* et nous retournerons le résultat *ej* si le score de celui-ci n'est pas nul.

### 3.2 Détail de l'approche :

Pour commencer, on peut estimer les scores des documents de la collection de recherche pour effectuer le tri des documents, avant la recherche de l'élément adapté à la requête. Pour cela, on utilise la formule suivante :

$$score(D, Q) = \prod_{i=1}^{n} \frac{|ed_i|}{dcel} \qquad (1)$$

$|ed_i|$ = Nombre d'éléments qui contiennent le terme $t_i$ de la requête $Q$ dans le document $D$

$n$ = Nombre de termes de $Q$

- Nous pouvons aussi utiliser ce score comme un lissage des probabilités nulles, dans le cas d'absence d'un terme de la requête $Q$ dans un élément du document $D$.

***But :*** Un document dont la majorité des nœuds contiennent la majorité des termes de la requête est sémantiquement pertinent, donc les éléments le constituant peuvent répondre au besoin de l'utilisateur en information. Ce tri nous indiquera à quel point la requête fournie pourra être satisfaite ou non par notre système, donc nous donnera une image de l'ensemble des éléments pertinents pouvant être retournés. De plus, si on veut faire une recherche dans une collection de documents pertinents, on propose d'utiliser cette formule pour sélectionner les documents pertinents (Cette mesure n'a pas été implémentée dans ce travail, mais c'est envisagé en perspective).

En suite, pour répondre à une requête utilisateur, le SRI XML retournera un élément de score calculé par la formule présentée dans le chapitre II (Partie 1 section 4.1) et que nous avons adapté à la RI XML, du fait que dans un document XML l'unité recherchée n'est plus un document entier mais c'est l'élément répondant à la requête, donc :

$$score(e_j, Q) = P(e_j/Q) = P(Q/e_j) \times P(e_j) \qquad (2)$$

Avec $P(e_j)$ est la probabilité d'avoir l'élément $e_j$ comme résultat pour une requête, cette probabilité peut être calculée de deux façons :

Par :

$La\ formule\ de\ base$: $$P(e_j) = \frac{|e_j|}{|D|} \qquad (3.1)$$

Ou par :

$La\ variante$: $$P(e_j) = \frac{|e_j|}{\Delta|e| \times Nf} \qquad (3.2)$$

Du fait que l'importance d'un élément ne dépend pas seulement du rapport entre sa taille et celle du document le contenant mais peut dépendre des autres éléments et des nœuds feuilles

dans le document vue que l'information textuelle est présente dans les nœuds feuilles ainsi que par hypothèse de dépendance entre les éléments du même document, ie : les éléments d'un même document appartiennent au même contexte même s'ils ne traitent pas forcément du même sujet mais leurs thèmes sont en relation.

Ainsi que la probabilité conditionnelle d'avoir la requête utilisateur à partir de l'élément en cours :

$$P(Q/e_j) = \prod_{i=1}^{n} P(t_i/e_j) \qquad (4)$$

C'est la distribution des probabilités des termes de la requête $Q$ étant une suite de mots : $Q = t_1, t_2, \dots, t_n$ et le modèle de l'élément $e_j = t'_1, t'_2, \dots, t'_n = Me_j$

Ainsi, la probabilité d'avoir le terme $t_i$ dans le modèle de l'élément $e_j$ dans (4) est donnée par son importance dans celui-ci :

$$P_l(t_i/Me_j) = P(t_i/e_j) = \frac{P_{MLE}(t_i/Me_j)}{\sum_{k=1}^{N} P_{MLE}(t_i/Me_k)} \qquad (5)$$

Avec : $N = dcel$ et $P_{MLE}$ est la probabilité de vraisemblance maximale que nous avons adapté pour la granularité « élément XML » ainsi :

- Par la formule de base :

$$P_{MLE}(t_i/Me_j) = \frac{tf_{ji}}{|e_j|} \qquad (5.1)$$

- Ou bien, en exploitant la formule de pondération *F4* proposée par [fellag, 2006] pour calculer le poids du terme $t_i$ dans l'élément $e_j$ dans sa probabilité de génération :

$$P_{MLE}(t_i/Me_j) = \frac{F4}{|e_j|} \qquad (5.2) \qquad avec\ F4 = tf_{ji} \times ief \times \frac{1}{h_1 + h_2 \times \frac{|e_j|}{\Delta|e|}}$$

$$ief = \log\left(\frac{Nf}{ef_i} + \alpha\right) \ et\ \alpha = 0.5\,, h_1 = 0.7\ \ h_2 = 0.3$$

### 3.3 Problème des probabilités nulles :

De la formule (4), on peut synthétiser que certains termes de la requête $Q$ peuvent ne pas figurer dans l'élément $e_j$, ce qui leurs attribue une probabilité nulle dans (5), donc pénaliser toute la séquence de termes $Q$ ainsi que l'élément $e_j$ par une probabilité nulle. Pour pallier à ce problème, nous adapterons la technique de lissage utilisée dans [Achmoukh, 2006] à notre modèle pour XML :

- **Le lissage et distribution des probabilités :**

Cette technique nous permet de calculer la probabilité d'importance d'un terme $t_i$ dans le modèle $Me_j$ de l'élément $e_j$ : $P_I(t_i / Me_j)$ selon sa présence ou son absence dans l'élément :

$$P_I(t_i/Me_j) = P(t_i/Me_j) = \begin{cases} P_s(t_i/Me_j) & si\ tf(t_i/e_j) > 0 \\ P_U(t_i/Me_j) & si\ tf(t_i/e_j) = 0 \end{cases} \qquad (6)$$

D'où, (4) devient :

$$P(Q/e_j) = \prod_{t_i \in e_j} P_S(t_i/Me_j) \times \prod_{t_i \notin e_j} P_U(t_i/Me_j) \qquad (4')$$

Avec :

$P_S(t_i/Me_j) = P_I(t_i/Me_j)$ Calculée par (5),

$P_U(t_i/Me_j)$ = la probabilité associée à un terme absent dans l'élément $e_j$, pour son estimation, nous avons utilisé le modèle du document, car l'importance d'un terme absent dans un élément du document est représentée par son importance dans le document tout entier, donc sa présence dans des éléments voisins ou lointains à l'élément en cours :

$$P_U(t_i/Me_j) = P(t_i/Md) = \frac{\sum_{j=1}^{Nf} tf(t_i, e_j) + 1}{\sum_{j=1}^{Nf} |e_j| + 1} \qquad (7)$$

Qui est le rapport entre : la fréquence du terme $t_i$ dans tous les nœuds feuilles du document $d$, à laquelle nous avons appliqué un deuxième lissage étant le lissage de Laplace pour la fréquence « ajouter-un » et la taille du document en nombre de termes augmenté de un « 1 » pour éviter à avoir des probabilités supérieure à un « 1 ». Vu que dans un document, les termes ne sont pas distribués équitablement entre les éléments le constituant, il s'avère nécessaire d'appliquer une distribution pour les termes.

Dans ce qui suit, une adaptation de la distribution des termes que nous avons appliqué aux documents d'une collection XML, qui est présentée dans [Achmoukh, 2006] dont nous avons adapté la granularité « élément » dans le calcul de la probabilité d'importance d'un terme $t_i$ dans le modèle du document $Md$ :

$$P_I(t_i/Md) = \frac{P(t_i/Md)}{\sum_{j=1}^{Nf} P(t_i/Me_j)} \qquad (8)$$

Avec $P(t_i/Md)$ calculée par (7) et $P(t_i/Me_j)$ calculée par (6)

### 3.4 Expansion du modèle de l'élément avec le modèle du document et celui de la collection :

Du fait que ce ne sont pas tous les mots d'une requête qui sont présents dans un élément, or qu'ils peuvent apparaitre dans d'autres éléments voisins ou un peu loin dans l'arbre du document XML, ou encore n'apparaitre dans aucun de ses éléments donc ne plus apparaitre dans le document entier, mais apparaitre quelque part dans un élément des arbres de la collection.

De ce fait, dans la RI classique, les auteurs ont adapté une expansion au modèle de la collection pour éviter à pénaliser toute séquence (requête) ayant un terme absent dans le document plat. Dans [Achmoukh, 2006], l'auteur à proposé un modèle combinant le modèle du document texte et celui de la collection le contenant :

$$P_I(t_i/D) = \lambda_1 P_I(t_i/Md) + \lambda_2 P_I(t_i/Mc) \quad (9)$$

Avec : $\lambda_1 + \lambda_2 = 1$

Dans notre cas, nous avons proposé ce qui suit :

$$P_I(t_i/e_j) = \lambda_1 P_I(t_i/Me_j) + \lambda_2 P_I(t_i/Md) + \lambda_3 P_I(t_i/Mc) \quad (10)$$

Avec : $\lambda_1 + \lambda_2 + \lambda_3 = 1$ soit : $\lambda_1 = 0.5$ , $\lambda_2 = 0.3$ $et$ $\lambda_3 = 0.2$

- Les valeurs des constantes $\lambda_1$, $\lambda_2$ et $\lambda_3$ ont été fixées ainsi sans avoir testé leurs effets, mais du fait que dans notre cas, on s'intéresse beaucoup plus au modèle de l'élément : $\lambda_1 = 0.5$ puis le modèle du document$\lambda_1 = 0.3$ du fait qu'un élément fait partie d'un document puis le modèle de la collection $\lambda_1 = 0.2$ que nous avons utilisé juste comme lissage des probabilités nulles.

Ainsi que :

$$P_I(t_i/Mc) = \frac{\sum_{j=1}^{Nfc} tf(t_i, e_j)}{\sum_{j=1}^{Nfc} |e_j|} \times \frac{1}{|ec_i|} \quad (11)$$

Avec *Nfc* est le nombre de nœuds feuilles dans la collection et $ec_i$ est le nombre d'éléments contenant le terme $t_i$ dans la collection.

D'où, (2) devient :

$$score(e_j, Q) = P(e_j/Q) = \prod_{t_i \in \overline{Q}} P_I(t_i/e_j) \times P(e_j) \quad (12)$$

## 4. Algorithmes de recherche :

Notre approche propose de procéder en commençant par le calcul des scores de similarité entre chacun des éléments des documents de la collection et la requête utilisateur, pour cela, le SRI se basant sur ce modèle doit suivre les algorithmes suivants :

### 4.1 Algorithme (1): « Calcul des scores »

Cet algorithme englobe les autres algorithmes, car il permet de lancer la recherche en utilisant notre modèle, il implémente la formule (3) sa structure est la suivante :

*dcel:* est la taille du document en nombre d'éléments

**Entrées** = {collection C de documents XML , requête Q}

**Sorties** = {$élément_i$ , $score_i$}

Debut

Tant que C n'est pas vide faire

Pour tout document $d_i$ contenant *dcel* éléments faire

Pour (j=*1* à $dcel(d_i)$) faire

Debut

Calculer la probabilité $P(e_j)$ *par la formule 3.1 ou 3.2*

Calculer la probabilité d'inférer Q à partir de $e_j$: $P(Q/e_j)$ algorithme (2) pour la formule (4) ou algorithme (6) pour la formule (4')

//selon le lissage choisi

Calculer la probabilité de score de $e_j$ pour Q: formule (12)

Fin

Fait

Fait

Fait

Fin

### 4.2 Algorithme (2) : « Calcul de similarité »

Cet algorithme nous permet de calculer la probabilité de similarité entre la requête $Q$ est l'élément $e_j$ représentant la probabilité d'inférer la requête $Q$ de l'élément $e_j$, il implémente la formule (4) et sa structure est la suivante :

**Entrées** = {*élément* $e_j$ , *requête* $Q = t_1, t_2, \dots t_n$}

**Sortie** = valeur de similarité $P$

Constantes : $\lambda_1 = 0.5$ , $\lambda_2 = 0.3$ , $\lambda_3 = 0.2$

Variable : $P$

Debut

Tant que $Q \neq$ null faire

Debut

Pour le terme $t_i$ faire

Debut

Calculer $P_I(t_i/Me_j) = P_{Ie}$ par l'algorithme (3)

Calculer $P_I(t_i/Md) = P_{Id}$ par l'algorithme (4)

Calculer $P_I(t_i/Mc) = P_{Ic}$ par l'algorithme (5)

$P_i := \lambda_1 P_{Ie} + \lambda_2 P_{Id} + \lambda_3 P_{Ic}$

*// c'est la probabilité* $P_I(t_i/e_j)$ *calculée par la formule (10)*

Fin

Fait

$Q := Q \setminus \{ t_i \}$

Fin

Fait

Calculer et retourner le produit des $\prod P_{i(\overline{1, \dots n})}$

Fin

### 4.3 Algorithme (3) : « Calcul d'importance d'un terme dans un élément donné »

Cet algorithme nous permet de calculer pour un terme donné de la requête, son importance dans le modèle associé à un élément de l'arborescence du document XML concerné, en utilisant la formule (6), sa structure est la suivante :

**Entrées** = {un modèle d'élément : $Me_j$ , un terme $t_i$}

**Sortie** = valeur de $P_I(t_i/Me_j)$

Variables : P,S

Debut

Pour le terme $t_i$ faire

Debut

Si tf($t_i/e_j$) ≠ 0 alors

Debut

P := $P(t_i/ej)$ calculé par la formule (5)

Fin

Si non

P := $P(t_i/Md)$ calculé par la formule (7)

Fsi

Retourner P

Fin

Fait

Fin

### 4.4 Algorithme (4) : « Calcul d'importance d'un terme dans un document »

Cet algorithme nous permet de calculer la probabilité d'importance d'un terme $t_i$ dans le modèle *Md* associé à un document *d* donné, par la formule (7), sa structure est la suivante :

$Nf(d)$ : est le nombre de nœuds feuilles du document *d*

**Entrées** = { document $d$ = ensemble d'éléments $e_j$ , le terme $t_i$ }

**Sortie** = valeur de $P_I(t_i/Md)$ = $P_I$

Variables : $Ptd$, $Pte$ = 0

Debut

$Ptd$ := $P(t_i/Md)$ calculée par la formule (7)

Pour k := 1 à Nf(d) faire

$Pte$ := $Pte$ + $P_I(t_i/Me_j)$

*//avec $P_I(t_i/Me_j)$ calculée par l'algorithme (3)*

Fait

$P_I$ := $Ptd$ / $Pte$

Retourner $P_I$

Fin

### 4.5 Algorithme (5) : « Calcul de l'importance d'un terme dans la collection »

Cet algorithme nous permet de calculer la probabilité d'importance d'un terme de la requête utilisateur dans le modèle de la collection de documents XML, il implémente la formule (11), sa structure est la suivante :

$Nf(C)$ : est le nombre de nœuds feuilles dans toute la collection $C$

**Entrées** = { ensemble des nœuds feuilles $e_j$ de la collection XML , un terme $t_i$ }

**Sortie** = valeur de $P_I(t_i / Mc) = P$

Variables : $Stf_C = o$ , $St = o$

// *Stfc :* Somme des tf du terme dans tous les nœuds de la collection

// *St :* somme des tailles des nœuds feuilles de la collection en nombre de termes

Debut

Pour j := 1 à Nf(C) faire

$Stf_C := Stf_C + tf\ (t_i / e_j)$

//avec $tf\ (t_i / e_j)$ *est la fréquence brute de* $t_i$ *dans* $e_j$

Fait

Pour k := 1 à Nf(C) faire

$St := St + |e_k|$

Fait

$P := Stf_C / (St * |ec_i|)$

Retourner $P$

Fin

### 4.6 Algorithme (6) : « Calcul de la probabilité $P(Q/e_j)$ »

Cet algorithme nous permet de calculer la probabilité conditionnelle $P(Q/e_j)$ pour chacun des éléments de la collection XML, dans le cas du lissage par distribution des probabilités, il implémente la formule (4'), sa structure est la suivante :

**Entrées** = { *requête* $Q = (t_1, t_2, \ldots t_n)$, *élément* $e_j$ }

**Sortie** = valeur de $P(Q / e_j)$

Variable $P := 1$

Debut

Pour $i := 1$ à $n$ faire

Si ( $t_i \in e_j$) alors $P := P * P_I(t_i / Me_j)$

//*avec* $P_I(t_i / Me_j)$ calculée par la formule (5) algorithme (3)

Si non

$P := P * P_I(t_i / Md)$

//*avec* $P_I(t_i / Md)$ calculée par l'algorithme(4)

Fsi

Fait

Retourner $P$

Fin

**Conclusion :**

Dans cette partie, nous avons présenté une approche ascendante dans l'exploitation du modèle de langue à la RIS XML, notre approche exploite complètement la structure du document XML, en se basant sur : la taille d'un document en nombre de termes (document XML vu comme document plat), la taille d'un document en nombre d'éléments (document XML comme arbre d'éléments) et la taille d'un document en nombre de nœuds feuilles (car l'information textuelle se trouve au niveau des nœuds feuilles). Ainsi, nous avons adapté l'approche exploitant la taille d'un document plat dans le modèle de langue présenté dans [Achmoukh, 2006] à la nouvelle granularité d'information en RIS XML, en présentant des formules de calcul de score d'un élément pour une requête ainsi que les algorithme à suivre dans le processus de recherche par notre modèle. En outre, l'adaptation des techniques de lissage par distribution des probabilités et en s'inspirant de la technique de Laplace pour les documents plats et que nous avons appliqué au modèle de l'élément instable, combiné à chacun des modèles du document et de la collection, pour retourner l'élément le plus pertinent possible pour une requête utilisateur.

Dans la partie suivante, nous présenterons un exemple d'application aux documents XML, tout en suivant la démarche de recherche par notre modèle et que nous avons présenté au cours de cette partie, nous montrerons aussi l'application de l'ensemble des algorithmes et les résultats obtenus dans l'application.

## Partie II : Application aux documents XML

**Introduction :**

Dans le but de montrer l'utilisation de notre approche, nous avons procédé à son application pour une mini-collection de documents XML, en utilisant une requête simple. Dans cette partie, nous allons montrer la démarche à suivre dans le processus de recherche en utilisant notre modèle, le calcul des scores ainsi que la sélection des nœuds pertinents.

1. **Présentation des éléments d'application :**

**1.1 Requête :**

Pour la réalisation de l'exemple d'application, nous avons utilisé une requête simple de type CO, soit la requête : *« Influence des réseaux sociaux »*, après élimination des mots vides : « *des* », on obtient la requête *Q= « influence réseaux sociaux »*, après la lemmatisation des termes de *Q* :

*influence* → *influ*

*réseaux* → *réseau*

*sociaux* → *socia*

D'où, on obtient la requête de recherche suivante : *query= « influ réseau socia »*

**1.2 Documents :**

Notre mini-collection est constituée de 5 documents XML, dont le contenu de chacun d'eux est différent des autres. Voici la présentation de chaque document ainsi que leurs arbres représentatifs :

```
<XML version="1.0" encoding="iso-8859-1"?>
<article>
<en-tete>
<titre>Armée électronique syrienne</titre>
<preface>
L'Armée électronique syrienne ou Syrian Electronic Army (SEA) est un groupe de pirates informatiques dont la création remonte au début de la guerre civile syrienne, en 2011. La SEA est responsable de nombreuses cyberattaques envers des médias occidentaux qu'ils estiment hostiles à Bachar el-Assad, y compris des agences de presse et des institutions des droits de l'Homme.
</preface>
</en-tete>
<corps>
<revendication>
La plupart des experts en piratage les présentent comme des jeunes qui ne sont pas forcément des pro-régimes. Le chercheur à l'Université de Toronto Helmi Noman se demande au contraire si le groupe n'est pas tout bonnement lié au régime syrien, et justifie son argument par les efforts salués par Bachar al-Assad lui-même, au cours du mois de juin 2011.
</revendication>
<attaques>
La SEA s'est principalement illustrée en piratant le compte Twitter de l'agence de presse américaine Associated Press. Le 23 avril 2013, un tweet de l'agence annonce 2 explosions à la Maison Blanche et la blessure du président Barack Obama. L'information, diffusée peu de temps après les attentats du marathon de Boston, avait parut suffisamment crédible et a provoqué un affolement général à Wall Street qui a fait perdre 136 milliards de dollars de capitalisation boursière. Cette affaire a remis en cause l'influence grandissante des réseaux sociaux sur les marchés boursiers. L'armée électronique syrienne pirate également des boîtes mails. Les hackers se  disent « honorés de collecter des infos en piratant les boîtes mails de certains pays qui sont devenus les ennemis de la Syrie »
</attaques>
</corps>
</article>
```

**Figure III-1:** Doc1.XML

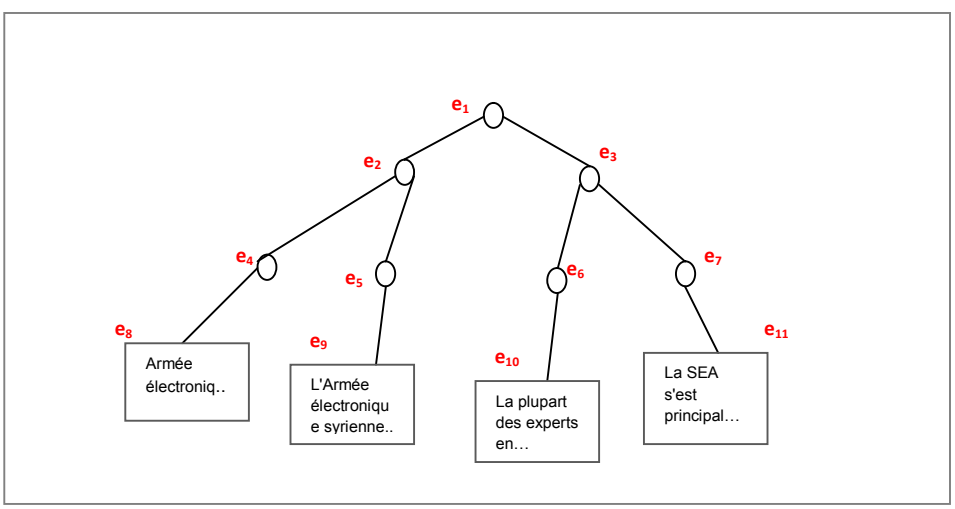



**Figure III-2 :** Arbre correspondant à Doc1.XML

```
<XML version="1.0" encoding="iso-8859-1"?>
<article>
<en-tete>
<titre>Les réseaux sociaux</titre>
<definition>
Un réseau social, c'est un groupement d'utilisateurs d'internet qui échangent des informations et qui permet de partager des photographies, des vidéos ou même son humeur du jour. Par ailleurs, le terme de réseaux sociaux désigne généralement l'ensemble des sites internet permettant de se constituer un réseau d'amis ou de connaissances professionnelles.
</definition>
</en-tete>
<corps>
<presentation>
<apparition>
Les réseaux sociaux n'existent pas depuis longtemps. En effet, c'est en 1995 que nous assistons à la naissance de Classmates, le premier réseau social crée par Randy Conrads. L'objectif de ce dernier était de remettre en contact des anciens camarades de classe. Classmates existe encore, mais a été détrôné par l'arrivée fracassante de nombreux autres réseaux sociaux tels que Facebook ou Twitter. Les réseaux sociaux ont par la suite formé une nouvelle génération de consommateurs de l'information, toujours plus désireux de savoir vite, bien et globalement.
</apparition>
<communication>
La communication est devenue la clé de voûte de notre monde ultra connecté et les nouveaux médias font ce que les médias traditionnels ne font qu'avec plus de restrictions: communiquer à l'intérieur ainsi qu'à l'extérieur d'un pays, joignant des millions de personnes de tous horizons. Par ailleurs, les réseaux sociaux ont complètement bouleversés les modes de vie et de penser des gens et sont devenus presque indispensables à cette nouvelle génération en quête perpétuelle de progrès. Ces derniers ont donc une influence considérable sur notre société
</communication>
</presentation>
</corps>
</article>
```

**Figure III-3 :** Doc2.XML

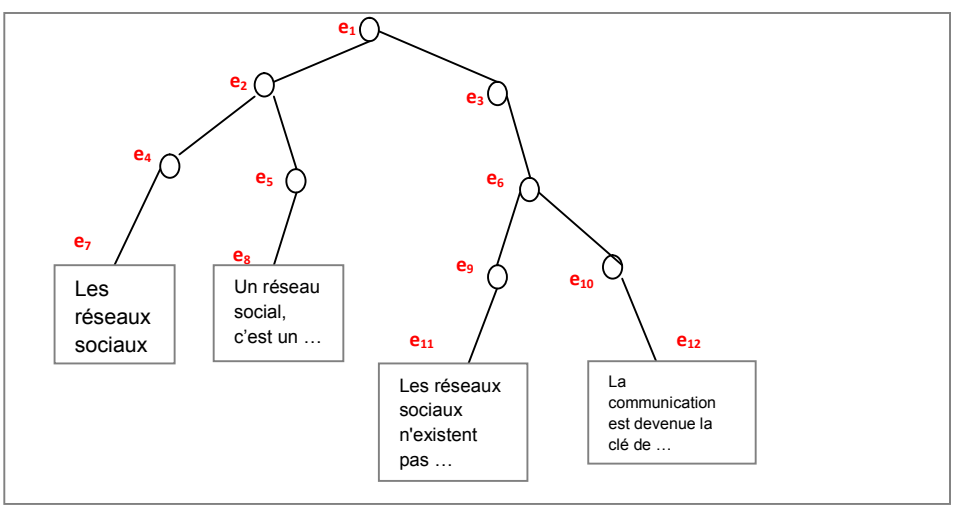



**Figure III-4 :** Arbre correspondant à Doc2.XML

```
<XML version="1.0" encoding="iso-8859-1"?>
<article>
<en-tete>
<titre>influence (psychologie)</titre>
<definition>
L'influence est le processus par lequel une personne fait adopter un point de vue par une autre.
L'influence opère une inflexion : celui qui aurait pensé ou agi autrement s'il n'était pas influencé se
dirige dans le sens que souhaite l'influent de façon apparemment spontanée.
</definition>
</en-tete>
<corps>
<processus>
Le processus d'influence est notamment à la base du leadership, la capacité d'obtenir que les
autres fassent ce que vous voulez ou coopèrent à vos objectifs sans utiliser de sanction ou de
promesse. Le charisme et les qualités particulières que l'on prête à un chef — celles qui font que
l'on désire le seconder avec enthousiasme — peuvent être considérés comme un phénomène
d'influence, pas toujours délibéré.
</processus>
<domaine>
<géopolitique>
zones d'influence, politique d'influence, par opposition à politique de puissance
</géopolitique>
<politique>
pour désigner des phénomènes de pouvoir qui ne reposent pas sur la détention d'une autorité
légale (l'influence des intellectuels, des médias ou des autorités morales, par exemple)
</politique>
<sociologique>
les groupes d'influence sont des organisations qui exercent une certaine emprise sur les décisions
des autorités et les réorientent dans un sens favorable à leurs intérêts
</sociologique>
<intelligence économique>notamment à propos du lobbying
</intelligence économique>
</domaine>
</corps>
</article>
```

**Figure III-5 :** Doc3.XML

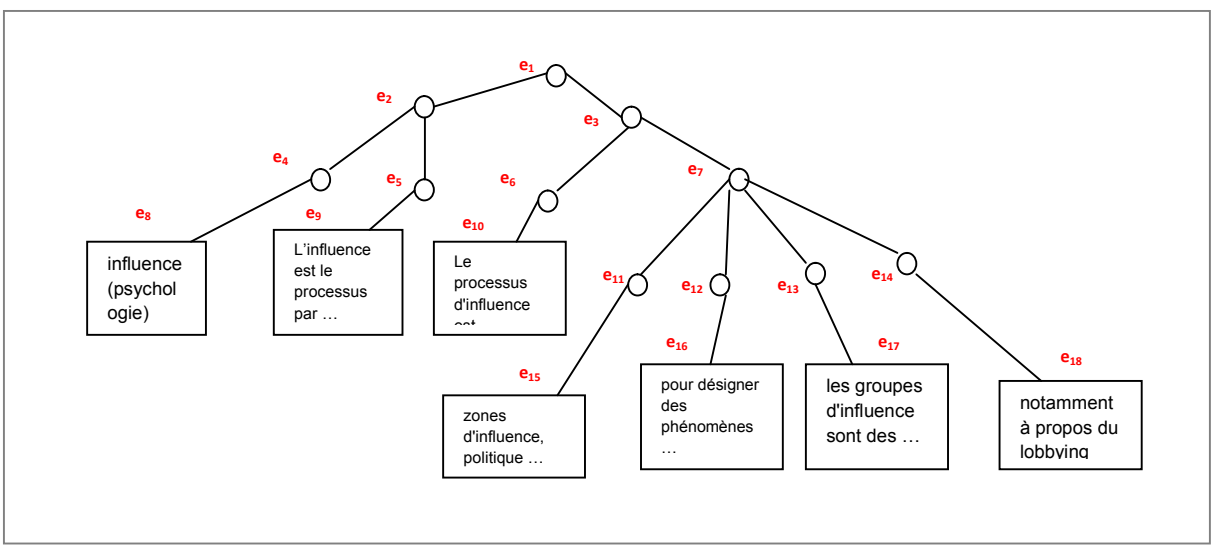



**Figure III-6 :** Arbre correspondant à Doc3.XML

```
<XML version="1.0" encoding="iso-8859-1"?>
<article>
<titre>réseaux d'influence d'entreprise</titre>
<presentation>
<p>
Un réseau d'influence est l'ensemble des contacts qu'une entreprise ou un individu a la possibilité de solliciter afin d'obtenir, directement ou indirectement, officiellement ou officieusement, l'aide nécessaire à la réussite de sa démarche. L'appartenance à certaines organisations crée un certain réseau d'influence.
</p>
<p>
Il ne faut pas le confondre avec un groupe d'influence, une organisation dont le but est d'influencer les décisions politiques, comme les lobbys ou les think tank.
</p>
</presentation>
</article>
```

**Figure III-7 :** Doc4.XML

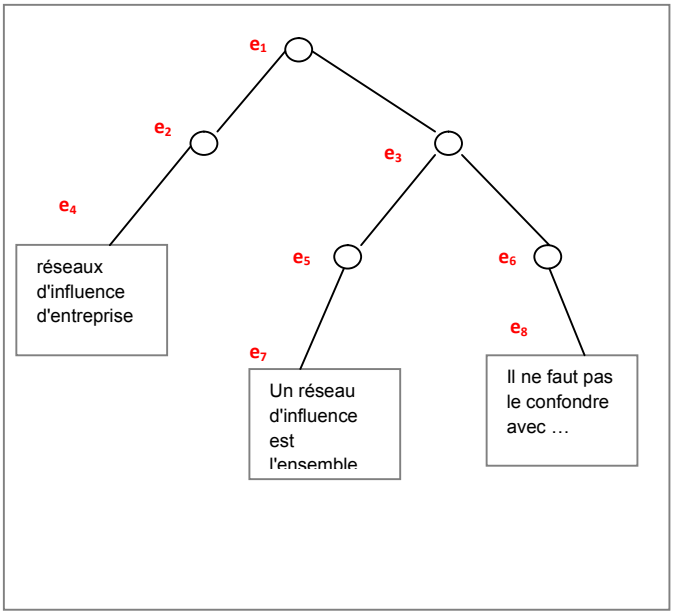



**Figure III-8 :** Arbre correspondant à Doc4.XML

```
<XML version="1.0" encoding="iso-8859-1"?>
<article>
<en-tete>
<titre>influence des réseaux sociaux</titre>
<introduction>
Aujourd'hui, les réseaux sociaux sont de plus en plus nombreux sur le web et l'on remarque que cela devient dangereux vis à vis de la vie privée de ces utilisateurs qui se
dévoile. Tout d'abord, un réseau social est une plate-forme électronique permettant, par la création gratuite d'un espace personnel de présentation de soi que l'on appelle « profil », de nouer des relations avec des membres du réseau. Notons que ce type de réseau en ligne, liant virtuellement et moralement les personnes, a vu le jour avec le site Classmates.com en 1995.
</introduction>
</en-tete>
<corps>
<exemples>
<Facebook> est un réseau de partage très influant, sur lequel nous pouvons rester en contact ou encore prendre contact avec les personnes du réseau. De plus, l'utilisateur peut publier différentes informations comme des photos, des textes ou encore des liens.
</Facebook>
<Twitter> est une plate-forme sociale de blogage permettant d'envoyer des messages brefs appelés tweets afin de partager des opinions, des passions et de découvrir ce qui se
passe en temps réel dans le monde.
</Twitter>
<MySpace> est un site web permettant de créer un espace personnalisé. Ce site est majoritairement connu pour un partage musical.
</Myspace>
</exemples>
<influence>
<foyer>
<p>
L'ordinateur est devenu l'objet de tous les foyers, créant une passerelle entre les différentes générations qui sont décalées dans le temps mais pas dans les technologies, en influençant leurs relations. Effectivement les utilisateurs des réseaux sociaux sont répartis sur plusieurs générations.
</p>
```

```
<p>
L'influence de ces réseaux est que la moitié des personnes âgées entre 16 et 29 ans voient leurs familles,75 % leurs amis. Cependant, les personnes âgées de plus de 29 ans voient plus leurs familles que leurs amis. De plus, on remarque que plus l'âge augmente et plus la différence de fréquence augmente. Donc on en déduit que ce résultat est influencé par les réseaux sociaux qui favorise les rencontres en amis plutôt que la famille.
</p>
</foyer>
<scolarisation>
Pour voir l'influence sur la scolarité des jeunes utilisateurs de ces réseaux, des questions ont été posées aux élèves du lycée et CEM, sur les réseaux sociaux, concernant leurs scolarités. D'après les sondages, 73,1% de personnes font leurs devoirs avant d'utiliser les réseaux sociaux alors que 11,5% les font après avoir utiliser ces plate-forme électroniques. De ce fait, la conclusion est que les réseaux sociaux n'influencent pas autant la scolarité de ces élèves, même si une très petite minorité fait ses devoirs après ou encore pire en même temps d'utiliser ces réseaux sociaux.
</scolarisation>
<sociale>
L'influence des réseaux sociaux sur la société en générale, est que des idéologies sont échangées entre les citoyens, ainsi que des sensibilisations peuvent êtres faites en s'échangeant des conseils et des avis. De ce fait, un réseau social influence positivement sur la société, mais sans nier que l'excès d'utilisation de ces réseaux, spécialement pour les inconscients, influence négativement en causant des problèmes à l'intérieur des familles.
</sociale>
</influence>
</corps>
</article>
```

**Figure III-9 :** Doc5.XML

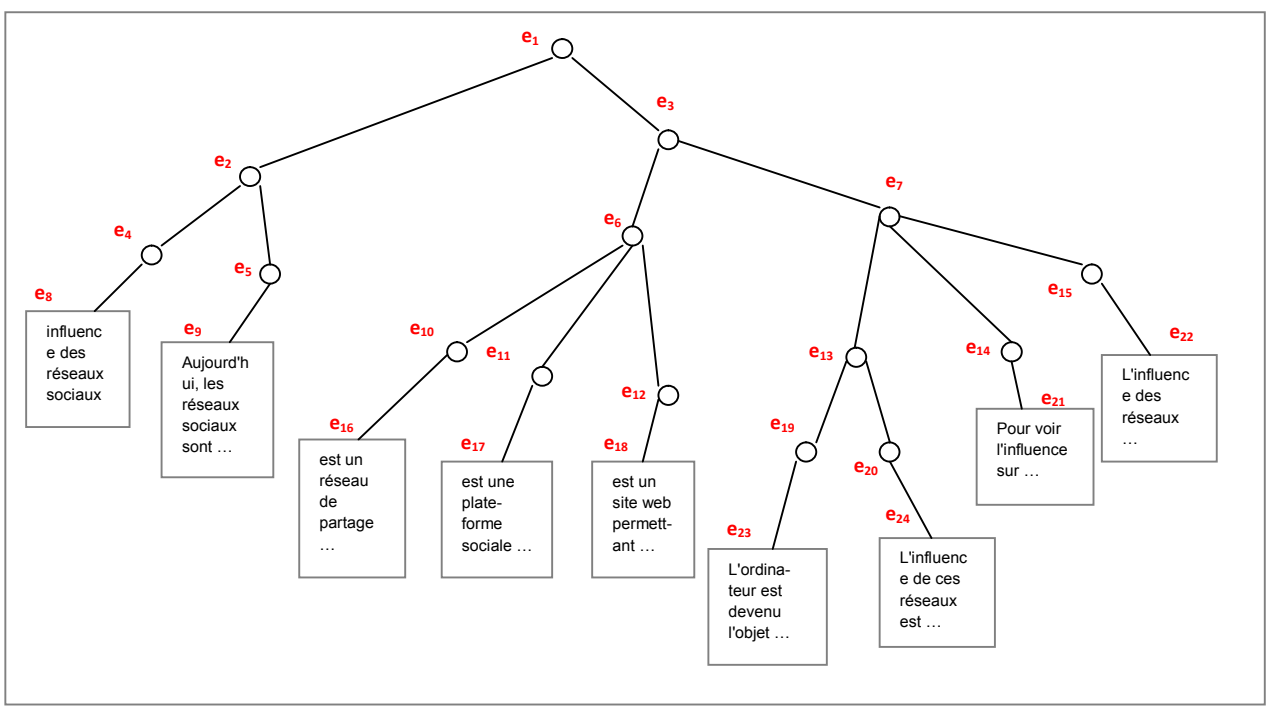



**Figure III-10 :** Arbre correspondant à Doc5.XML

## 2. Estimation des scores des documents:

Telle qu'on l'a présenté dans la partie 1, l'estimation des scores des documents nous donne une idée à priori sur la satisfaction de la requête de recherche. Pour cela, on a utilisé la formule de score : formule (1), on a aboutis aux résultats suivant :

| query / $Doc_i$ | **dcel** | « *influ* » | « *réseau* » | « *socia* » | **score** |
|---|---|---|---|---|---|
| | | **$ef_i$** | | | |
| $Doc_1$ | 11 | 5 | 5 | 5 | **0.094** |
| $Doc_2$ | 12 | 5 | 12 | 12 | **0.417** |
| $Doc_3$ | 18 | 14 | 0 | 0 | **0.000** |
| $Doc_4$ | 8 | 8 | 6 | 0 | **0.000** |
| $Doc_5$ | 24 | 18 | 20 | 20 | **0.521** |

**Tableau III-1 :** Estimation des scores des documents de la collection pour la requête *query*

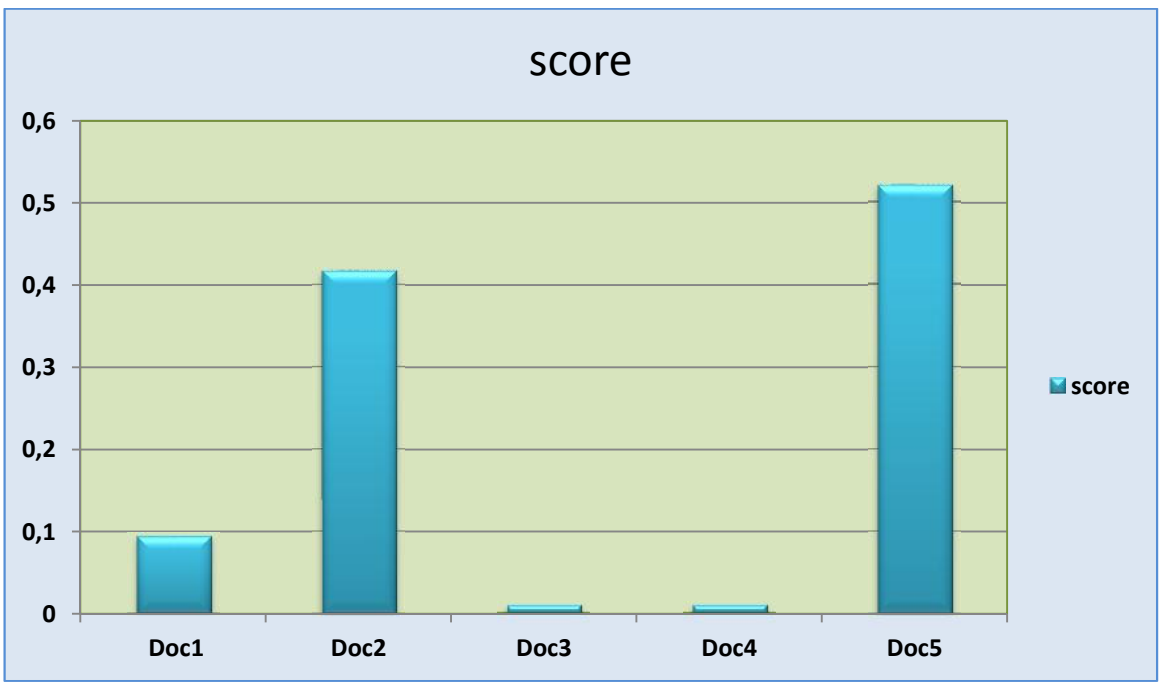



**Figure III-11 :** Histogramme des valeurs de score de chacun des documents XML

*Remarque :* A partir de l'histogramme, on peut déduire les documents pertinents ou semi pertinents et ceux qui ne traite pas du tout le sujet de la requête. On peut déduire que le document $Doc_5$ est assez pertinent ainsi que $Doc_2$ qui traite du sujet de la requête, puis $Doc_1$ qui traite une petite partie du sujet abordé par les termes de la requête *query*. Par contre, $Doc_3$ et $Doc_4$ ne sont pas pertinents pour la requête, en rappelant du sujet de Doc3 concernant

« l'influence en psychologie » et Doc4 qui traite «*les réseaux d'influence des entreprises* », on acceptera les valeurs de score nulles.

### 3. Processus de recherche par le modèle de langue :

Pour retourner un élément comme résultat de recherche pour une requête utilisateur, un score doit être attribué aux éléments constituant les documents de la collection, pour cela la démarche à suivre commence par le calcul de la probabilité de score pour tous les éléments de la collection, en utilisant l'algorithme (1) :

#### 3.1 Calcul de la probabilité $P(e_j)$ des éléments :

Pour tous les éléments de la collection, en utilisant la formule (3.1) de base, et sa variante (3.2), nous avons aboutis aux résultats suivants :

| **$Doc_1$** | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $e_j$ | $e_1$ | $e_2$ | $e_3$ | $e_4$ | $e_5$ | $e_6$ | $e_7$ | $e_8$ | $e_9$ | $e_{10}$ | $e_{11}$ |
| $\|e_j\|$ | 151 | 41 | 110 | 3 | 38 | 36 | 74 | 3 | 38 | 36 | 74 |
| $P(e_j)$ par (3.1) | 1 | 0.272 | 0.728 | 0.02 | 0.252 | 0.238 | 0.49 | 0.02 | 0.252 | 0.238 | 0.49 |
| $P(e_j)$ par (3.2) | 0.898 | 0.244 | 0.696 | 0.018 | 0.226 | 0.214 | 0.44 | 0.018 | 0.226 | 0.214 | 0.44 |

**Tableau III-2 :** les probabilités associées à chacun des éléments du document $Doc_1$

Pour les autres documents, vous trouverez les tableaux associés dans l'ANEXE III.

#### 3.2 Calcul de la probabilité $P(Q/e_j)$ pour chaque élément :

Pour calculer le score de pertinence des éléments de la collection, nous devons calculer la probabilité $P(Q/e_j)$ de similarité des éléments $e_j$ pour la requête $Q$, ceci en se basant sur les probabilités d'importance de chacun des termes de la requête dans l'élément. Donc, on calcule le produit des probabilités d'importance de chacun des termes dans chacun des éléments en utilisant l'algorithme (2), ou l'algorithme (6).

Dans ce qui suit, une application de l'algorithme (2) :

##### 3.2.1 Calcul de la probabilité $P_l(t_i / Me_j)$ de chaque terme :

Pour chaque terme de la requête, nous avons calculé sa probabilité d'importance dans chacun des éléments de la collection en utilisant l'algorithme (3). On a obtenu les valeurs suivantes pour chaque élément :

| termes / ($Doc_i$ , $e_j$) | | *« influ »* | *« réseau »* | *« socia »* |
|---|---|---|---|---|
| **$Doc_1$** | $e_1$ | 0.0131 | 0.0131 | 0.0131 |
| | $e_2$ | 0.0131 | 0.0131 | 0.0131 |
| | ... | … | … | … |

**Tableau III-3 :** les probabilités d'importance de chacun des termes de la requête dans chacun des éléments des documents de la collection XML

### 3.2.2 Calcul de la probabilité $P_I(t_i / Md)$ de chaque terme :

Pour chaque terme de la requête *query* , nous avons calculé sa probabilité d'importance dans le modèle de chacun des documents de la collection XML, en utilisant l'algorithme (4), nous avons obtenu les résultats suivants :

| Termes / $Doc_i$ | *« influ »* | *« réseau »* | *« socia »* |
|---|---|---|---|
| **$Doc_1$** | 0.228 | 0.09 | 0.139 |
| **$Doc_2$** | 0.078 | 0.045 | 0.122 |
| **$Doc_3$** | 0.039 | 0.056 | 0.056 |
| **$Doc_4$** | 0.104 | 0.058 | 0.12 |
| **$Doc_5$** | 0.026 | 0.03 | 0.03 |

**Tableau III-4 :** les probabilités d'importance de chacun des termes de la requête dans le modèle associé à chacun des documents de la collection XML

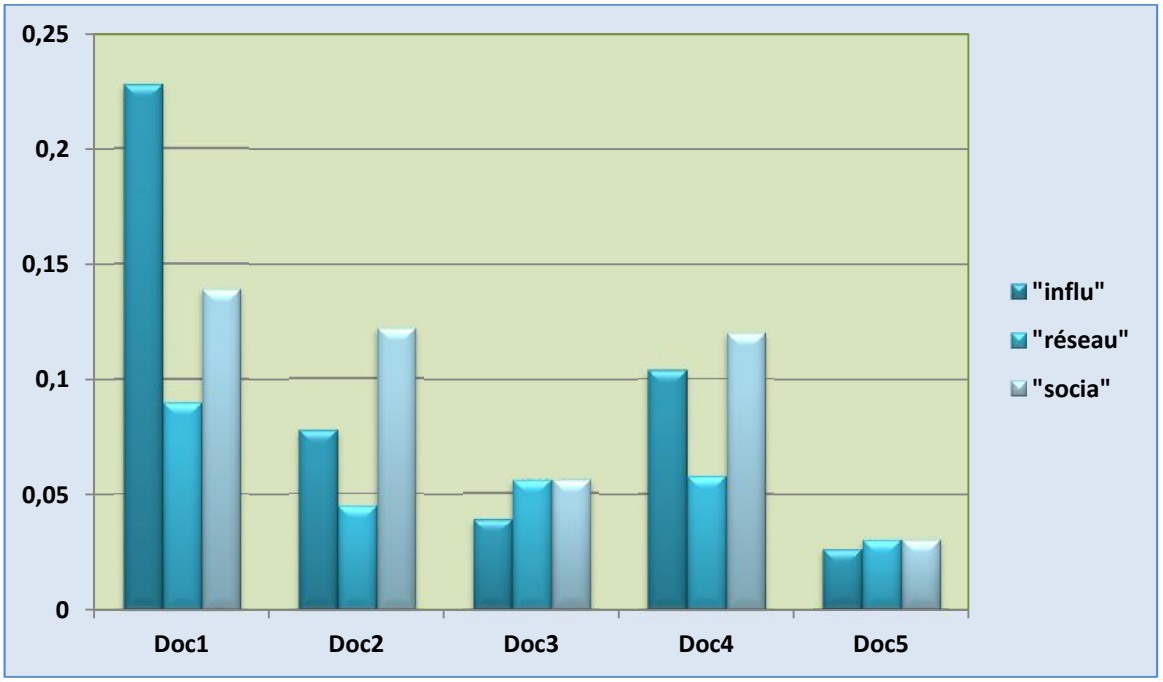



**Figure III-12 :** Histogramme des probabilités d'importance de chacun des termes de la requête dans chacun des documents XML

*Remarque :* d'après cet histogramme, on peut remarquer l'écart entre l'importance de chacun des termes dans le même document, de ce fait, on peut déduire le document pouvant être le plus pertinent à la requête, par principe que le document le plus informatif est celui dont l'écart entre l'importance de chacun des termes de la requête est faible, de ce fait, on peut déduire à priori que le document $Doc_5$ est le plus informatif.

### 3.2.3 Calcul de la probabilité $P_I(t_i / Mc)$ de chacun des termes :

Pour chacun des termes de la requête, nous avons calculé sa probabilité d'importance dans le modèle de la collection, en suivant l'algorithme (5). On a aboutis aux résultats suivants :

| **Termes** | *« influ »* | *« réseau »* | *« socia »* |
|---|---|---|---|
| $P_I(t_i / Mc)$ | 0.007 | 0.008 | 0.006 |

**Tableau III-5 :** les probabilités d'importance de chacun des termes de la requête dans le modèle de la collection XML

Puis, nous avons calculé la probabilité d'importance de chacun des termes $t_i$ dans chacun des éléments $e_j$ de la collection : $P_I(t_i/e_j)$ en utilisant l'algorithme (2) et nous avons obtenu les résultats suivants :

| $Doc_i$ | $e_j$ | $P_I("influ"/e_j)$ | $P_I("réseau"/e_j)$ | $P_I("socia"/e_j)$ |
|---|---|---|---|---|
| $Doc_1$ | $e_1$ | 0.076 | 0.037 | 0.051 |
| | $e_2$ | 0.076 | 0.037 | 0.051 |
| | ... | … | … | … |

**Tableau III-6 :** les probabilités d'importance de chacun des termes de la requête dans chacun des éléments des documents de la collection XML

Vous trouverez l'ensemble des valeurs associées aux différents termes de la requête dans tous les documents, dans l'ANNEXE III .

### 3.3 Calcul des scores de similarité requête-élément :

Une fois les probabilités $P(e_j)$ de chacun des éléments est calculée, ainsi que $P(Q/e_j)$ en utilisant les algorithmes précisés précédemment, nous allons calculer les scores de similarité entre chaque élément de chaque document et la requête *query* = *Q*, en utilisant la formule (12), pour ensuite sélectionner ceux qui peuvent répondre avec pertinence à la requête. Dans le cas de notre exemple, nous avons obtenu les résultats suivants :

| $Doc_1$ | | | |
|---|---|---|---|
| $e_j$ | ***score*** | $e_j$ | ***score*** |
| $e_1$ | 0.0014 | $e_7$ | 0.0030 |
| $e_2$ | 0.0040 | $e_8$ | 0.0030 |
| $e_3$ | 0.0100 | $e_9$ | 0.0200 |
| $e_4$ | 0.0030 | $e_{10}$ | 0.1000 |
| $e_5$ | 0.0340 | $e_{11}$ | 0.0400 |
| $e_6$ | 0.0700 | | |

**Tableau III-7 :** les scores de similarité entre les éléments du document $Doc_1$ et la requête *query*

Pour les scores de similarité des éléments des autres documents et la requête, vous trouverez les tableaux associés dans l'ANNEX III.

## 4. Renvoie des résultats :

Pour renvoyer les résultats pertinents pour l'utilisateur, un seuil de score minimal doit être fixé à priori, pour pouvoir filtrer les nœuds de la collection et en choisir ceux qui peuvent satisfaire la requête, car un nœud dont le score de pertinence est très proche du zéro ne peut apporter un contenu informatif par rapport à la requête. Vu que notre approche est dérivée d'un modèle probabiliste, les scores de pertinence sont de valeurs comprises entre zéro et un,

pour procéder au filtrage des résultats, nous allons fixer un seuil proche de zéro, soit « 0.020 ». Donc, tout nœud dont le score de pertinence est supérieur ou égale à 0.020 sera renvoyé, ainsi on obtiendra les résultats dont l'ordre croissant des scores ainsi :

| Document | élément | score |
|---|---|---|
| $Doc_2$ | $e_1$ | **0.130** |
| $Doc_5$ | $e_3$ | **0.130** |
| $Doc_1$ | $e_{10}$ | **0.100** |
| $Doc_4$ | $e_4$ | **0.100** |
| $Doc_5$ | $e_4$ | **0.100** |
| $Doc_5$ | $e_{15}$ | **0.100** |
| $Doc_5$ | $e_{21}$ | **0.100** |
| $Doc_5$ | $e_{14}$ | **0.080** |
| $Doc_5$ | $e_{16}$ | **0.080** |
| $Doc_5$ | $e_{22}$ | **0.080** |
| $Doc_5$ | $e_{20}$ | **0.073** |
| $Doc_5$ | $e_{24}$ | **0.073** |
| $Doc_1$ | $e_6$ | **0.070** |
| $Doc_5$ | $e_{17}$ | **0.070** |
| $Doc_3$ | $e_5$ | **0.064** |
| $Doc_5$ | $e_{12}$ | **0.050** |
| $Doc_5$ | $e_{18}$ | **0.050** |
| $Doc_5$ | $e_{23}$ | **0.050** |
| $Doc_5$ | $e_{19}$ | **0.046** |
| $Doc_2$ | $e_9$ | **0.045** |
| $Doc_2$ | $e_{11}$ | **0.045** |
| $Doc_2$ | $e_5$ | **0.041** |
| $Doc_1$ | $e_{11}$ | **0.040** |
| $Doc_2$ | $e_4$ | **0.040** |
| $Doc_5$ | $e_6$ | **0.040** |
| $Doc_5$ | $e_{11}$ | **0.040** |
| $Doc_2$ | $e_6$ | **0.038** |
| $Doc_3$ | $e_{16}$ | **0.037** |
| $Doc_3$ | $e_8$ | **0.035** |
| $Doc_1$ | $e_5$ | **0.034** |
| $Doc_2$ | $e_7$ | **0.032** |
| $Doc_4$ | $e_3$ | **0.032** |
| $Doc_4$ | $e_1$ | **0.024** |
| $Doc_5$ | $e_7$ | **0.023** |
| $Doc_5$ | $e_9$ | **0.021** |
| $Doc_1$ | $e_9$ | **0.020** |
| $Doc_2$ | $e_3$ | **0.020** |
| $Doc_3$ | $e_3$ | **0.020** |
| $Doc_4$ | $e_5$ | **0.020** |
| $Doc_4$ | $e_7$ | **0.020** |

| | | |
|---|---|---|
| $Doc_5$ | $e_8$ | **0.020** |
| $Doc_5$ | $e_{10}$ | **0.020** |

**Tableau III-8 :** Tableau des nœuds résultats classés par ordre décroissant des scores de pertinence pour la requête *query*

Voici l'histogramme correspondant, pour bien illustrer l'ensemble des nœuds pertinents résultants de la recherche :

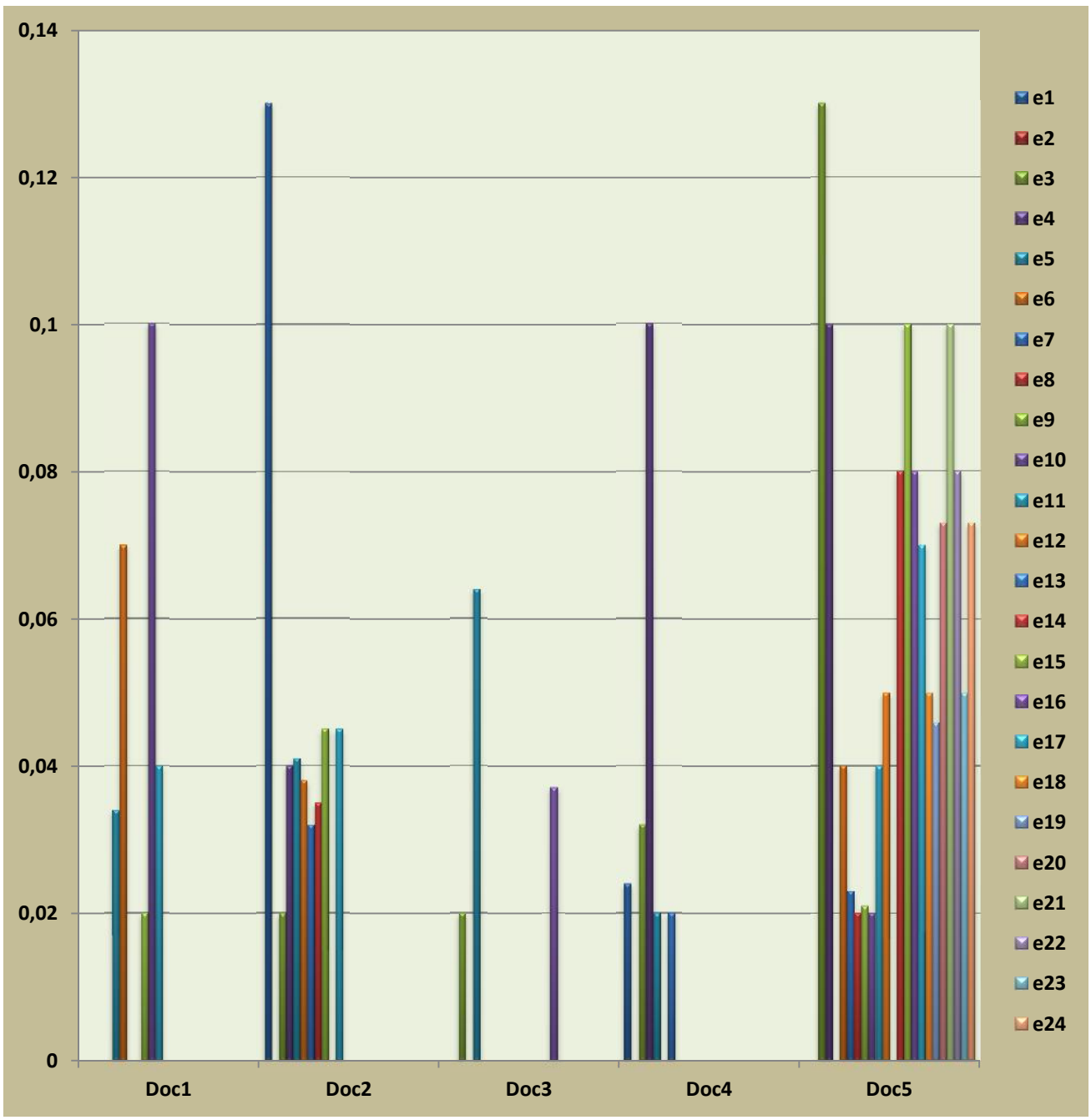



**Figure III-13 :** Histogramme des scores des résultats retenus pour la requête *query*

**Interprétation :**

Rappelons que les thèmes de chacun des documents de la mini-collection d'application sont : « `Armée électronique syrienne` », « `Les réseaux sociaux` », « `influence (psychologie)` », « `réseaux d'influence d'entreprise` » et « `influence des réseaux sociaux` » respectivement pour : $Doc_1$ , $Doc_2$ , $Doc_3$ , $Doc_4$ et $Doc_5$ avec la requête *query* = « `influ réseau socia` ».

On remarque sur l'histogramme que l'ensemble des éléments à score de pertinence le plus élevé appartiennent aux documents dont le thème est proche de celui de la requête : ça sous entend $Doc_1$ , $Doc_2$ et $Doc_4$ , ou ceux dont le thème est celui abordé par la requête : ça sous entend $Doc_5$. Ainsi que la majorité des nœuds résultats retenus font partie de $Doc_2$ et $Doc_5$, dont le contenu est informatif pour la requête : littérairement, et d'après les sujets traités par chacun de ces documents, on constate l'évidence des résultats obtenus par cette approche, car l'histogramme nous montre le regroupement des résultats, dont les scores sont les meilleurs, dans les documents qui sont exhaustifs pour la requête, d'où la déduction sur la pertinence considérable des éléments résultats.

**Conclusion :**

Dans cette partie, nous avons montré la démarche à suivre pour l'application de notre modèle de recherche, dans les documents structurés XML. En se basant sur une mini-collection de cinq documents XML et une requête d'application, nous avons appliqué les algorithmes de recherche et de calcul de chacune des probabilités pour arriver enfin à retourner des résultats pertinents pour la requête de recherche. Dans le dernier paragraphe, nous avons interprété les résultats de recherche et leurs scores de pertinence pour la requête utilisée. Nous avons comparé les sujets abordés par chacun des documents et celui traité par la requête, du point de vu score de pertinence et nous avons conclu sur l'efficacité des résultats, du point de vu littérature : par rapport aux sujets traités.

Donc, ce chapitre résume notre approche de recherche par modèle de langue, ainsi que la démarche à suivre dans le processus d'appariement, avec illustration de notre contribution par un exemple d'application explicatif. Le prochain chapitre résumera l'expérimentation de cette méthode et les résultats obtenus.

# CHAPITRE IV : EXPERIMENTATION ET RESULTATS

# Partie 1 : Environnement et outils d'implémentation

## Introduction :

Dans le but d'affirmer les résultats de l'application effectuée dans le chapitre précédent pour notre modèle, nous avons procédé à son implémentation, pour cela, nous avons utilisé un ensemble d'outils logiciel pour offrir une plateforme adéquate à nos tests. Dans cette partie, nous présenterons les outils de développement ainsi que la plateforme d'accueil utilisée pour exécuter nos programmes de recherche dans le cadre du modèle de langue proposé.

Cette application a été réalisée dans une plateforme Unix (UBUNTU 10.10) montée dans une VMware Workstation, en utilisant les outils suivants : module d'indexation Xfirm, Eclipse comme environnement de développement, Java comme langage de programmation, Oracle 10g XE comme SGBD pour la gestion de la BDD du système contenant l'index des documents de la collection et les pilotes ODBC & JDBC pour la connexion à la BDD du système Xfirm pour récupérer les données nécessaires à la recherche.

### 1. VMware Workstation :

VMware Workstation est un environnement de test et de développement qui permet aux administrateurs système de créer et d'exécuter des machines virtuelles (VM) directement sur un bureau. Il permet aussi l'évaluation des hyperviseurs[1].

La version utilisée, VMware Workstation 9, est optimisée pour fonctionner avec les systèmes d'exploitation 64 bits et Windows 8. La VMware utilise une interface Web pour connecter les utilisateurs de machines virtuelles locales et serveur hébergé depuis un PC, un smartphone ou une tablette. La figure suivante montre l'interface de la VMware Workstation9 utilisée :

[1] **Un hyperviseur :** est aussi appelé un gestionnaire de machine virtuelle, est un programme qui permet à plusieurs systèmes d'exploitation de partager une foule matérielle unique. Chaque système d'exploitation semble avoir le processeur de l'hôte, la mémoire et d'autres ressources à lui tout seul.

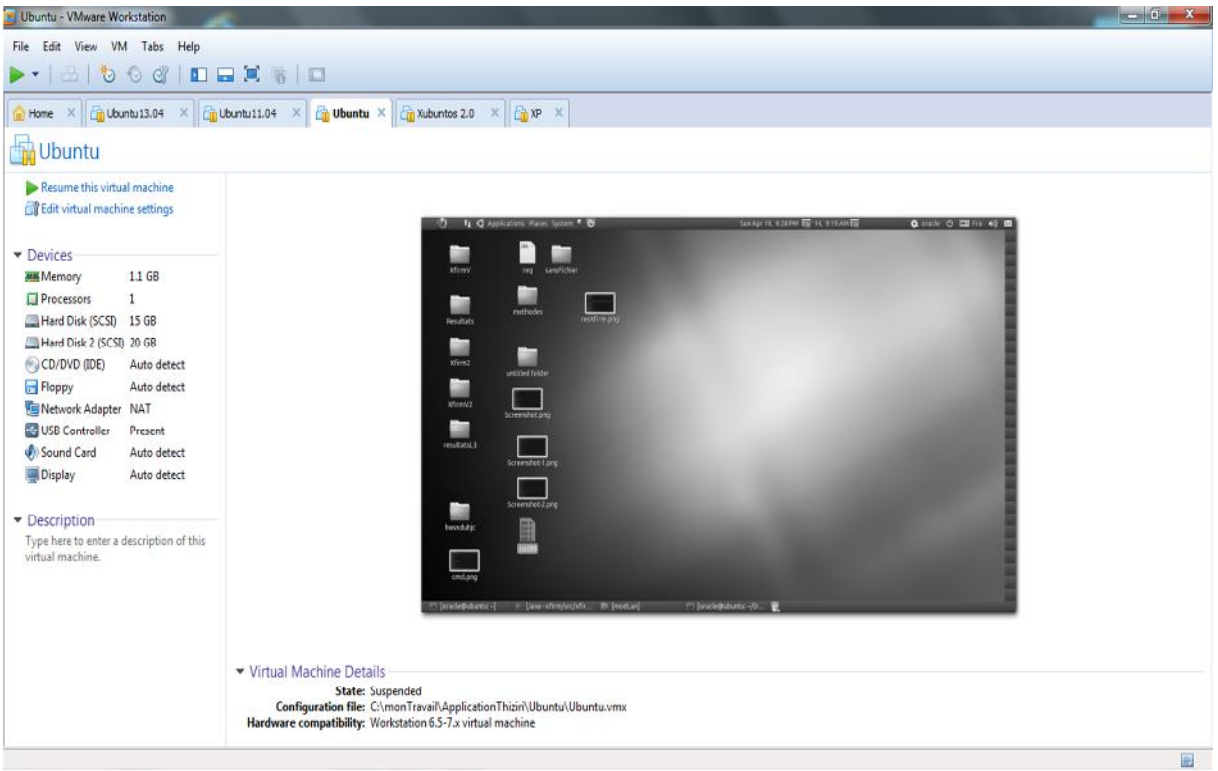



**Figure IV-1 :** Interface de la VMware Worksation 9

## 2. Le système de recherche Xfirm :

### 2.1 Présentation :

C'est un système de recherche d'informations développé en 2005 par Karen Sauvagnat, ce système se base sur le modèle vectoriel [Sauvagnat, 2005] et est orienté vers la RI structurée dans des corpus documentaires XML. Il a apporté de la flexibilité dans la recherche car il utilise une représentation des documents par un modèle générique de données (description des formats des documents dans une base de données) et un langage de requêtes en permettant l'expression du besoin en information de l'utilisateur avec des simples expressions du langage naturel puis une reformulation des requêtes.

XRIRM est un système orienté pertinence, il repose sur la propagation de pertinence à travers les éléments constituants un document : un premier score est calculé pour les nœuds feuilles ensuite il est propagé dans l'arbre du document et il est diminué durant la propagation pour répondre au critère de spécification d'une unité pour une requête reçue.

Pour assurer un fonctionnement cohérent, il présente un ensemble de modules qui ont donné lieu au développement d'un prototype permettant l'indexation de la collection documentaire XML. Ce prototype est entièrement réalisé en langage JAVA en utilisant des API (Application Progrming Interface) telles que SAX (Simple API for XML) pour parser les documents XML et un JDBC pour l'accès aux BDD du système. Ses composantes principales ainsi que leurs liaisons et structures sont présentées par l'architecture du prototype présentée dans la figure suivante :

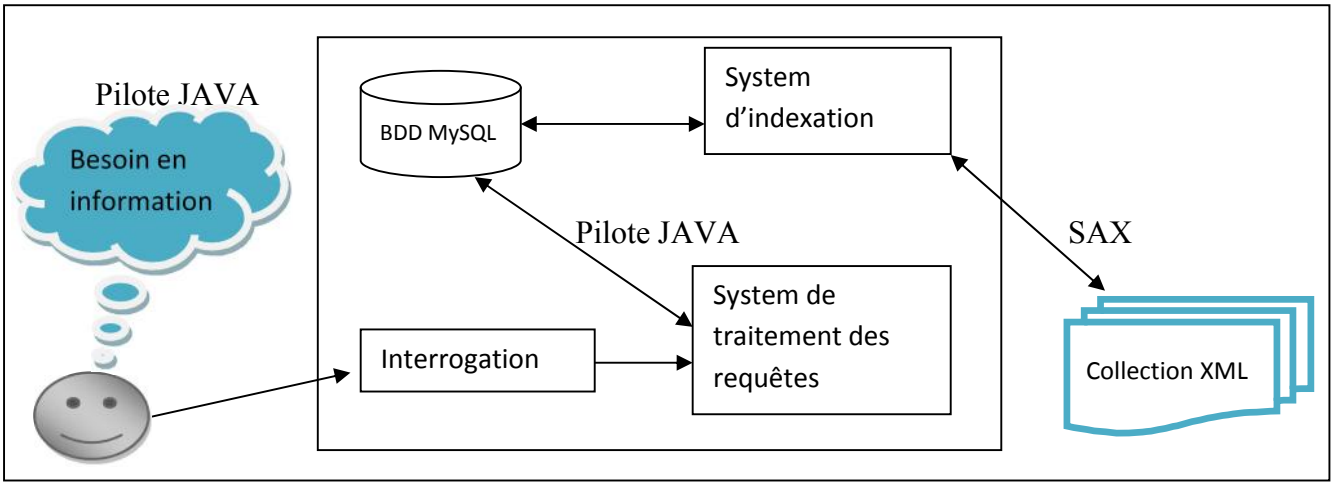



**Figure IV-2:** Architecture de base du model XFIRM [Sauvagnat, 2005] système

Pour plus de détails, ce système est présenté dans l'ANNEXE IV.

### 2.2 Modules exploités

Dans l'implémentation de notre travail, nous avons exploité le module d'indexation constitué du système d'indexation, la BDD et le parser SAX, tel c'est mentionné dans la figure ci-dessus, pour indexer notre collection de test constituée que de 1000 documents XML. Pour interroger notre système ci obtenu et effectuer la recherche, nous avons développé le module d'interrogation et de recherche, tel que ce dernier exploite la BDD contenant les index, en accédant aux tables suivantes :

- **Collection** : contient des informations générales sur toute la collection : nombre de documents, nombre de termes, nombre d'éléments …
- **Document** : contient des informations sur tous les documents de la collection : nombre de termes, nombre de niveau …
- **Terme** : contient des informations sur les termes de la collection : fréquence dans la collection, nombre d'éléments le contenant …
- **Path** : contient des informations sur les nœuds de la collection : document conteneur, l'attribut, taille …
- **NodeToTerm** : contient des informations sur les nœuds feuille de la collection : les termes présents dans un nœud donné …

## 3. L'environnement de développement Eclipse :

C'est un environnement de développement intégré (IDE) développé par I.B.M. pour des applications telles que java, dans le site[2] d'I.B.M associé à Eclipse on trouvera la définition suivante : *«Le projet Eclipse est un projet de développement de logiciels open source dédié à fournir un robuste, complet, une qualité commerciale et plate-forme de l'industrie pour le développement d'outils hautement intégrés »*. Donc, par définition, Eclipse est une plate-forme ouverte pour l'intégration de l'outil, par l'intégration des plugins à un IDE. La question a été confondue car une force industrielle complète, y compris la fonction

[2] *http://www.eclipse.org*

IDE Java est fournie avec la plate-forme Eclipse, sous la forme de plugins qui étendent les installations du cadre de base de l'Eclipse.En outre, les plugins Eclipse peuvent s'étendre d'autres plug-ins. Quand une application basée sur Eclipse démarre, il découvre et active tous les plug-ins qui ont été configurés pour le poste de travail. La plate-forme Eclipse est littéralement la somme de ses parties, car il est capable d'effectuer n'importe quelle fonction qui a été ajoutée à elle par les plug-ins qu'il contient pour le moment.
Eclipse est dit universel et polyvalent, car il permet de créer des projets de développement, qui mettent en œuvre n'importe quel type de langage de programmation (Java,
C++, PHP, JavaScript, …). La figure suivante présente l'interface de l'IDE Eclipse sous linux :

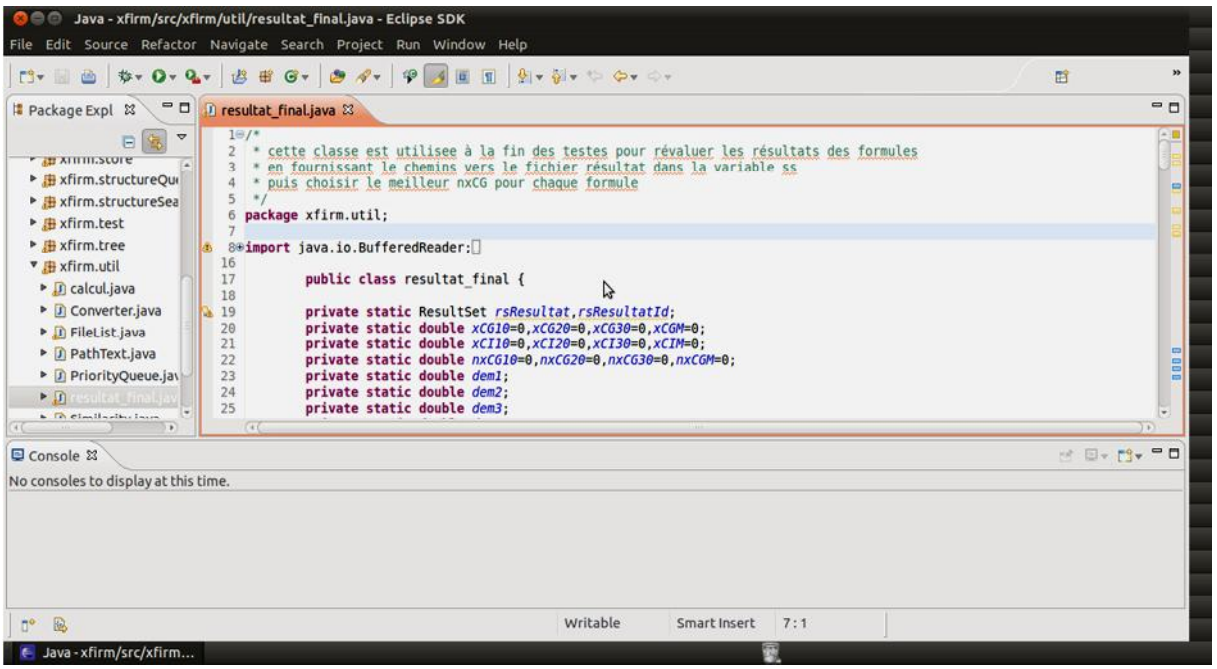



**Figure IV-3 :** L'interface de l'IDE Eclipse sous linux

## 4. Le langage Java :

Le choix de java comme langage de programmation s'est imposé vue que le code source d'XFIRM est entièrement écrit en java.
On peut faire remonter la naissance de Java à 1991. À cette époque, des ingénieurs de chez **Sun Microsystems** ont cherché à concevoir un langage applicable à de petits appareils électriques (*code embarqué*). Pour ce faire, ils se sont fondés sur une syntaxe très proche de celle de C++, en reprenant le concept de machine virtuelle déjà exploité auparavant par le PascalUCSD. L'idée consistait à traduire d'abord un programme source, non pas directement en langage machine, mais dans un pseudo langage universel, disposant des fonctionnalités communes à toutes les machines. Ce code intermédiaire, dont on dit qu'il est formé de *bytecodes*, se trouve ainsi compact et portable sur n'importe quelle machine, il suffit simplement que cette dernière dispose d'un programme approprié (on parle alors de *machine virtuelle*) permettant de l'interpréter dans le langage de la machine concernée. Il est dit aussi un langage multiplateforme selon la célèbre maxime «Write once, run everywhere ».

Java est un langage de programmation à usage général mais avec des fonctionnalités qui le rendent plus simple :

- Création des applications web, les servlets,

- Les applets,
- Programmation procédurale, événementielle et implémentation flexible de l'orienté objet,
- Bibliothèques riches de méthodes.
- …

Dans notre cas, on s'est basé sur la programmation procédurale par l'exploitation des différentes bibliothèques permettant l'implémentation des différentes phases de recherche.

Aujourd'hui, Java trouve une nouvelle niche dans la création d'applications RIA(Rich Internet Applications), des applications qui proposent des fonctionnalités, notammentdes interfaces, plus évoluées à la fois sur Internet et sur les téléphones portables. Le langageJavaFx est un langage agile dérivé de Java, sous le contrôle de Sun Microsystems, qui met àprofit la portabilité de Java ainsi que les vastes bibliothèques déjà disponibles dans le langagejava.

## 5. Le SGBD Oracle10g EX :

Un SGBD (Système de Gestion des Bases de Données) est un logiciel permettant la création, manipulation et modification des BDD, ainsi que le contrôle et la confidentialité des données, dans notre cas, le SGBD utilisé est Oracle 10g :

### 5.1 Définition :

Oracle est un SGBD écrit en langage ℂ et édité par la société[3] du même nom, qui est un leader mondial des bases de données et a été créée en 1977 par Lawrence Ellison, Bob Miner, et Ed Oates. Elle s'appelle alors *Relational Software Incorporated (RSI)* et commercialise un Système de Gestion de Bases de données relationnelles (SGBDR ou RDBMS pour *RelationalDatabase Management System*) nommé *Oracle*.
En 1984 la première version d'Oracle(Oracle4) est commercialisée sur les machines IBM. Et en 1985 Oracle 5 permet une utilisation client-serveur grâce au middleware *SQL*Net*. Et en 1988 Oracle 6 est disponible sur un grand nombre de plates-formes et apporte de nombreuses nouvelles fonctionnalités ainsi qu'une amélioration notable des performances.
Et ce n'est qu'en 1992, qu'Oracle 7 sort sur les plates-formes UNIX (elle ne sortira sur les plates-formes Windows qu'à partir de 1995). Cette version permet une meilleure gestion de la mémoire, du CPU et des entrées-sorties. La base de données est accompagnée d'outils d'administration (SQL*DBA) permettant une exploitation plus aisée de la base.

### 5.2 Les fonctionnalités d'Oracle :

Oracle est un SGBD permettant d'assurer :

- La définition et la manipulation des données
- La cohérence des données
- La confidentialité des données
- L'intégrité des données
- La sauvegarde et la restauration des données
- La gestion des accès concurrents (transactions)

[3]Oracle Corporation *http://www.oracle.com*

### 5.3 L'interface *SQL*PLUS* :

Oracle propose également de nombreux outils de développement permettant d'automatiser la création d'applications s'interfaçant avec la base de données. Parmi ces outils on a utilisé le *SQL*PLUS* qui est une interface interactive permettant d'envoyer des requêtes SQL et PL/SQL à la base de données, la créer et de manipuler interactivement des objets de la base via une interface en ligne de commandes à travers le SGBD Oracle. SQL*PLUS est habituellement utilisé pour formuler une requête SQL et obtenir, sur un écran et de façon immédiate, le résultat attendu.

La figure suivante montre cette interface sur la plateforme linux :

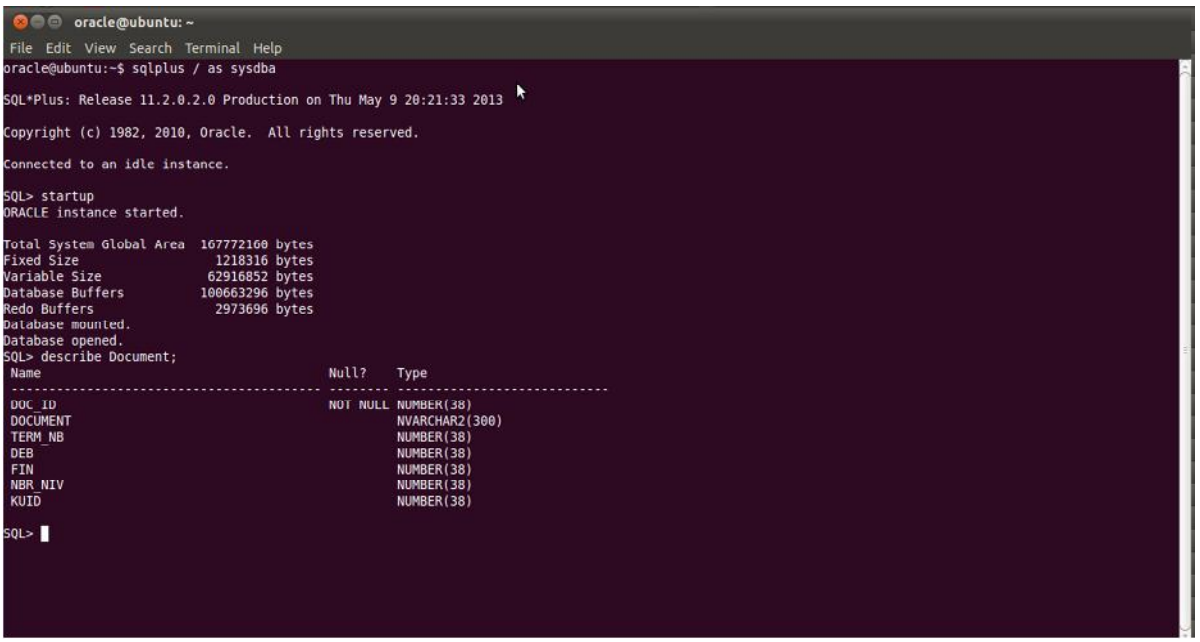



**Figure IV-4 :** Illustration de l'interface SQL*PLUS sous linux.

### 5.4 Les outils de connexion au SGBD et à la BDD :

#### 5.4.1 Le pilote de connexion au SGBD Oracle :

La classe permettant la connexion au SGBD est : *java.sql.DriverManager* qui se charge de la gestion, du contrôle et de la connexion au SGBD en fournissant les méthodes principales suivantes :

- staticvoidregisterDriver(Driver driver): enregistre le driver pour un type de SGBD particulier ;
- staticConnectiongetConnection (String *url*, String *user*, String *password*): Crée une connexion permettant d'utiliser une base de données.

Avec:
*driver* : est un pilote dépendant du SGBD utilisé, dans notre cas il s'agit du pilote d'oracle sous linux : *jdbc:oracle:thin:@oracleserv.irit.fr:1521:test*
*url* : identification de la base considérée sur le SGBD à format dépendant du SGGB utilisé
*user* : nom de l'utilisateur qui se connecte à la base
*password* : mot de passe de l'utilisateur

### 5.4.2 L'ODBC/JDBC :

- **Le JDBC :**(Java Data Base Connectivity)

C'est un outil de connexion à la BDD conçu par Sun. JDBC est un Framework pour le langage Java permettant l'accès aux bases de données relationnelles dans un programme Java indépendamment du type de la base utilisée (mySQL, Oracle, Postgres...) et seule la phase de connexion au SGBDR change. Il permet de faire tout type de requêtes : sélection des données dans des tables, création et insertion d'éléments dans les tables et gestion des transactions.
Les packages java le configurant : *java.sql*et *javax.sql*

- **L'ODBC :**

Abrégé en ODBC Open DatabaseConnectivityest un outil de connexion à une BDD, il construit un pont « *Bridge »* entre le pilote JDBC et la BDD configurée. Tel que le JDBC communique avec l'interface ODBC et non directement avec la BDD. L'intérêt réside dans le caractère standard de l'ODBC qui est utilisé dans les SGBD comme *Microsoft SQLServer*ou *Microsoft Access*. Ces différents drivers permettent d'accéder indifféremment à la plupart des SGBD.

- **Principales classes pour accéder à une BDD**

*La classe Driver Manager*charge et configure le driver de la base de données qui permet de gérer l'accès à un type particulier de SGBD.
*La classe Connection*réalise la connexion et l'authentification à la base de données.
*La classe Statement(et PreparedStatement*) contient la requête SQL et la transmet à la base de données.
*La classe ResultSet*permet de parcourir les informations retournées par la base de données dans le cas d'une sélection de données.

**Conclusion :**

Dans cette partie, nous avons présenté les outils utilisés pour la mise en place de notre approche de recherche, ce sont des outils de développement logiciel utilisés pratiquement pour la réalisation de différents systèmes informatiques, dans notre cas c'est le système de recherche d'information qui nous intéresse. Nous avons procédé par l'adjonction d'un module de recherche d'information par modèle de langue XML que nous avons proposé, en développant un ensemble de méthodes JAVA qui implémentent nos formules utilisées pour le calcul des scores de similarité entre un éléments XML et une requête utilisateur, puis nous avons exploité ces méthodes dans les algorithmes de recherche que nous avons expliqué dans le chapitre précédent.

La partie suivante montrera les éléments de tests, les tests effectués dans le cadre de recherche sur un échantillon de collection INEX ainsi que les résultats obtenus et leur évaluation par une mesure d'évaluation appropriée.

## Partie 2 : Tests et évaluation des résultats

### Introduction :

Dans le but d'évaluer la qualité d'un SRI ou d'une approche de recherche, cette dernière doit subir une implémentation et un ensemble de tests qui vont éclaircir sa qualité, sa pertinence ainsi que sa précision, et à quel point que cette approche pourra agir sur la satisfaction du besoin de en information de l'utilisateur. Dans cette optique, nous avons procédé à la réalisation de l'implémentation de notre approche de recherche par modèle de langue, en utilisant la plateforme précisée dans la partie précédente, nous avons réalisé les tests que nous présenterons au long de cette partie.

Cette partie montrera les éléments de tests utilisés, la procédure d'implémentation, les tests ainsi que les résultats obtenus et leur évaluation en utilisant des mesures de la compagne d'évaluation INEX.

### 1. Protocole d'évaluation :

Pour la réalisation de nos tests, nous avons utilisé des échantillons des éléments d'évaluation de la compagne INEX :

#### 1.1 Les requêtes de tests :

Pour effectuer des tests de recherche dans le cadre d'INEX de notre approche et pouvoir l'évaluer, nous avons utilisé un échantillon de 20 requêtes CO (Content Only) extraites des 124 requêtes de la compagne INEX 2006 qu'on a pris aléatoirement de quatre sous-ensembles de ces requêtes.
Le tableau suivant montre l'ensemble des requêtes utilisées :

| Le numéro de la requête | Le titre de la requête |
|---|---|
| **289** | emperor "Napoleon I" Polish |
| **290** | "genetic algorithm" |
| **295** | software intellectual property patent license |
| **297** | "cool jazz" "West coast" musician |
| **321** | buildings designed Antoni Gaudi Barcelona architect |
| **322** | castles kasteel in the netherlands |
| **323** | founder ikea |
| **324** | composition of planet rings |
| **325** | "Cirque du Soleil" shows |
| **327** | cloning animals accepted "United States of America" |
| **328** | NBA European basketball player |
| **362** | effect nuclear power plant accident |
| **364** | +mushroom poisonous poisoning |
| **365** | economy peru international investment tourism |
| **366** | Fourier transform applications |
| **367** | true story films best director or movie award |
| **393** | Wireless devices "Health Hazards" |
| **408** | "electroconvulsive therapy" depression |
| **409** | Hybrid Vehicles -biology "fuel efficiency" "fuel sources" model engine |
| **413** | Coordinates and Population of capital cities of Europe |

**Tableau IV-1 :** Les 20 requêtes utilisées dans les tests des formules de pondération

### 1.2 La collection de test :

Le tableau suivant montre les caractéristiques principales de la collection totale d'INEX 2006 :

| La taille de la collection | 4.6 Go |
|---|---|
| Le nombre de documents dans toute la collection | 659388 répartis en 22 ensembles |
| Le nombre d'éléments dans toute la collection | 50.000.000 |

**Tableau IV-2 :** Caractéristiques de la collection INEX 2006

Pour nos tests, nous avons utilisé un échantillon de 1000 documents différents, extraits de toute la collection :

| La taille de la collection | 17,1 Mo |
|---|---|
| Le nombre de documents dans la collection de test | 1000 |
| Le nombre d'éléments dans la collection de test | 191549 |

**Tableau IV-3 :** Caractéristiques de la collection de test

## 2. Procédure d'implémentation :

Pour implémenter notre modèle, nous avons procédé en suivant certaines étapes, pour une mise en place des algorithmes de recherche que nous avons montré dans le chapitre précédent. Pour ce faire, tel que c'est mentionné dans la partie précédente de ce chapitre, nous avons utilisé le module d'indexation « *BaseWriter* » et la Base De Données qui contient les index du système XFIRM. Concernant la recherche, ainsi que le calcul de similarité et le renvoie des résultats, nous avons développé des modules correspondant à notre méthode, comme la classe de recherche « *InexLang* » pour une recherche dans le cadre d'INEX par le modèle de langue proposé, ainsi que les méthodes de calcul des probabilités de similarité associées à notre modèle, dans la classe « *Similarity* » et d'autres méthodes qui servent à lire l'index et récupérer des informations telles que les nœuds d'un document, dans la classe « *BaseReader* ».

Cette procédure peut être divisée en deux phases :

### 2.1 Phase d'indexation :

Tel que nous avons mentionné dans ci-dessus, notre système utilise l'indexation XFIRM utilisant un parser SAX pour analyser les documents XML. Nous avons rempli la BDD avec l'index de 1000 documents XML de la compagne d'évaluation INEX 2006. Les index sont

stocké dans une BDD Oracle 10g EX s'exécutant sous Linux (UBUNTU 10.10), pour permettre par la suite la recherche des termes d'une requête fournie, par le parcours de la BDD qui fait correspondre à chaque document de la collection un index contenant ses éléments et ses termes, permettant ainsi la récupération flexible des données de recherche.

### 2.2 Phase de recherche :

C'est la phase la plus cruciale de notre implémentation, car elle consiste à trouver tout les nœuds de tout les documents de la collection pouvant répondre à la requête utilisateur, donc nous avons procédé à récupérer tout les nœuds d'un document de la collection, puis leurs calculer les scores associés par rapport à la requête. En effet, cette méthodologie est résumée dans l'*Algorithme(1)* de recherche dans le chapitre précédent, pour sa mise en place, nous avons développé un module de recherche approprié exploitant des méthodes JAVA appropriées pour le calcul des probabilités associées.

## 3. Lancement des tests :

Pour pouvoir lancer un test dans le cadre d'INEX, notre système exécute le module « *InexLang* », pour ce faire, nous devrons tout d'abord lancer un terminal pour démarrer oracle, par les commandes suivantes :

*~$ sqlplus / as sysdba*

// pour lancer la console sqlplus en administrateur

*SQL > startup*

// lancer le serveur de BDD

La figure suivante montre les résultats de chacune des commandes ci-dessus :

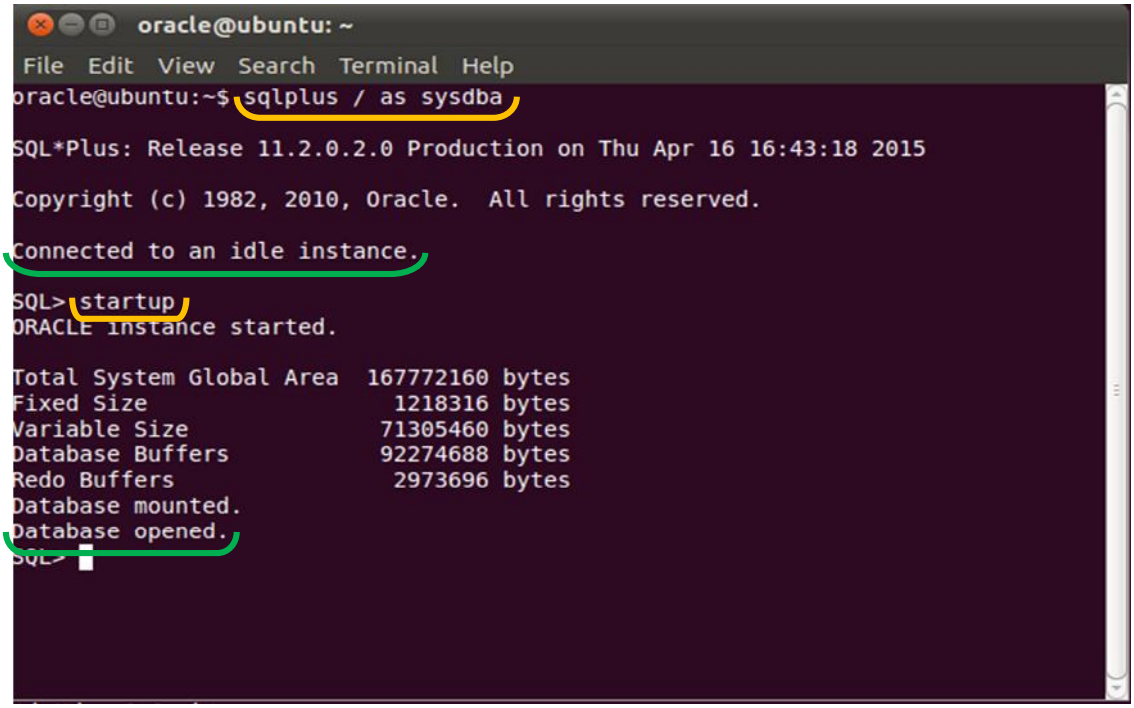



**Figure IV-5 :** Illustration des commandes de lancement du SGBD du système en lignes de commande

A partir d'un deuxième terminal, se placer dans l'environnement d'exécution et configurer un *listener* pour les requêtes de la BDD, par la commande suivante :

*~$ netca*

La figure suivante montre son utilisation :

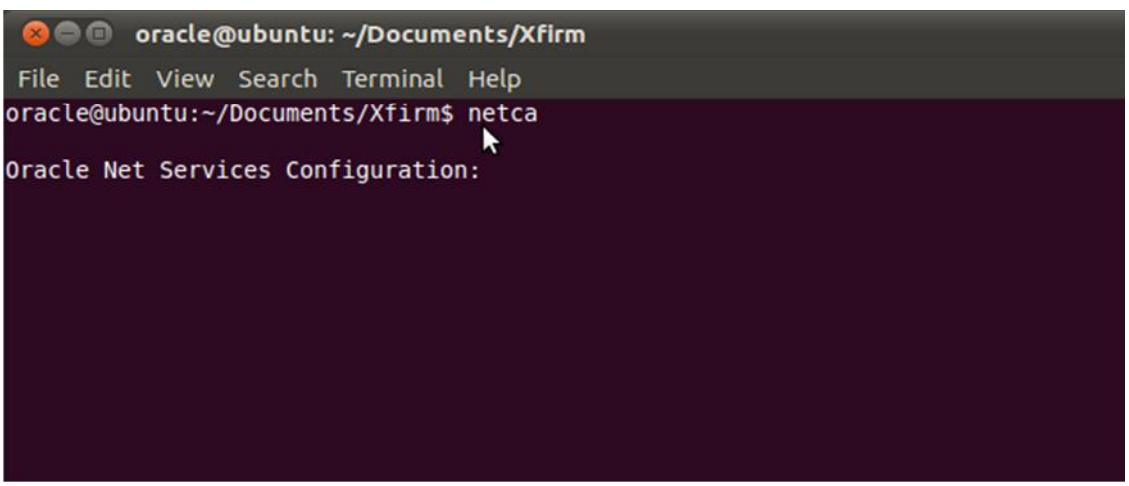
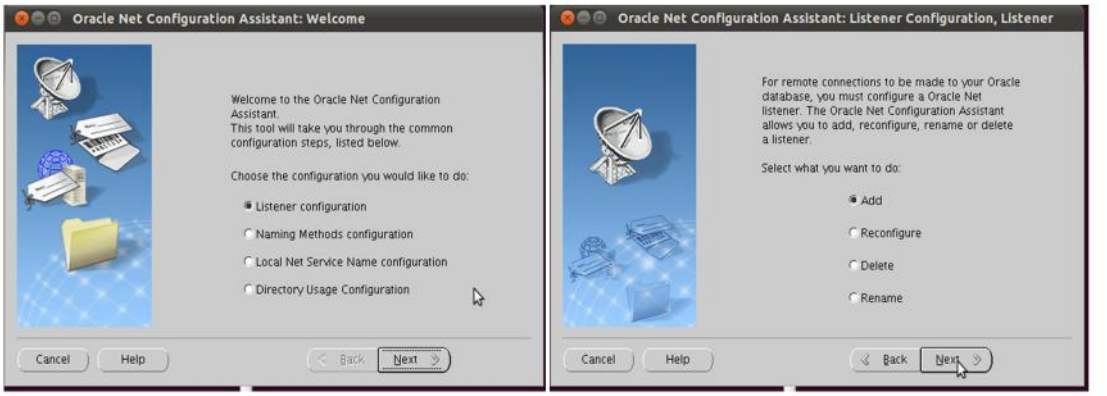



**Figure IV-6 :** Utilisation de la commande « netca » à partir d'un terminal

Suivre ensuite les boites de dialogue suivantes :




(1)




(2)


(3)


(4)


(5)


(6)

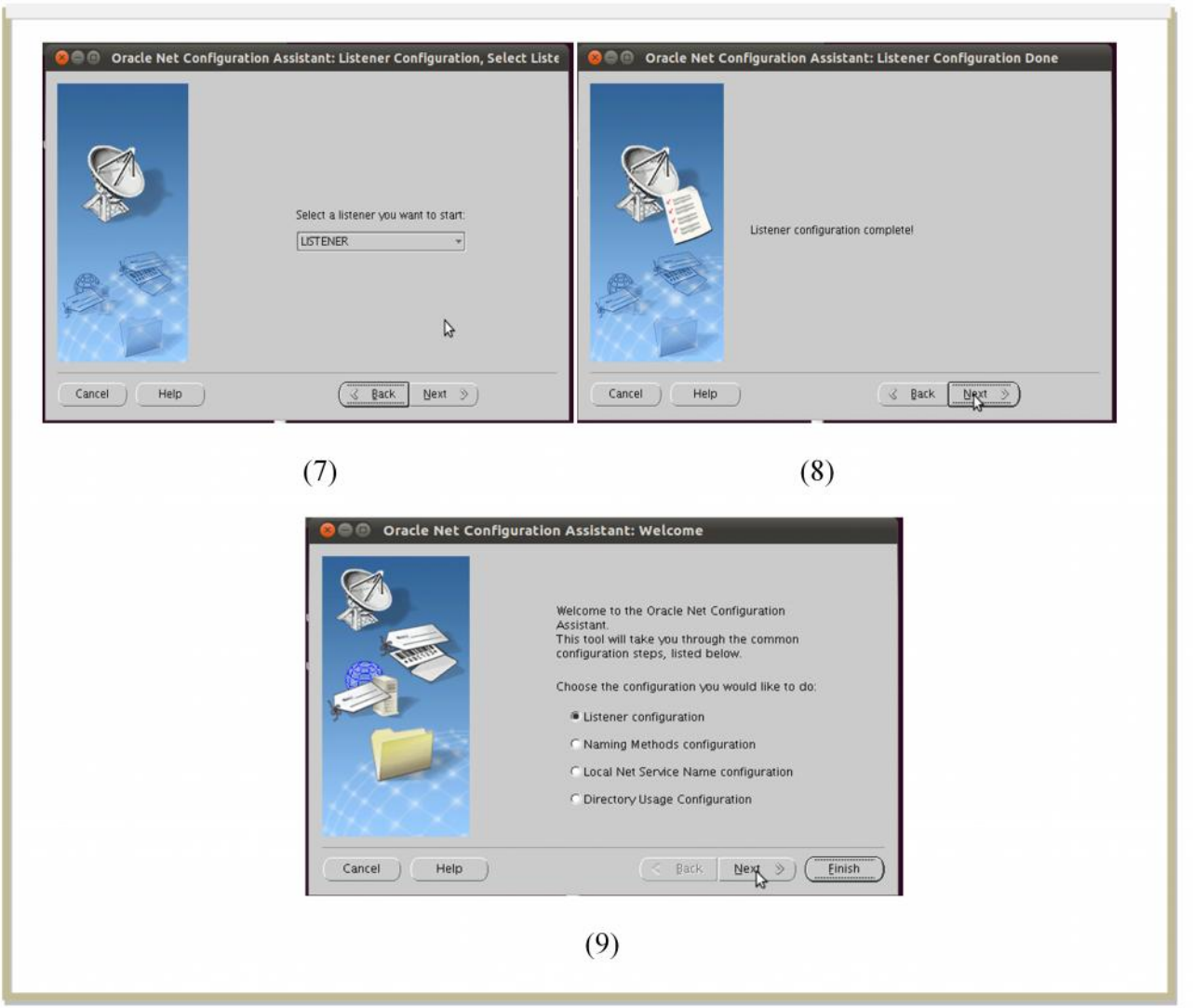



**Figure IV-7 :** Illustration des étapes à suivre pour la configuration d'un *listener* sous Linux

Enfin, nous pouvons lancer notre module de recherche par la commande suivante :

*˜$ java xfirm/inex/InexLang p1 p2 p3 p4*

Avec :

*p1* : chemin vers le fichier de configuration de la BDD Oracle

*p2* : chemin vers le fichier contenant les requêtes de test à exécuter

*p3* : chemin vers le fichier résultat qui va contenir les résultats de recherche

*p4* : le nom de la collection de recherche indexée dans la BDD

La figure suivante montre un exemple de lancement d'une requête de test :

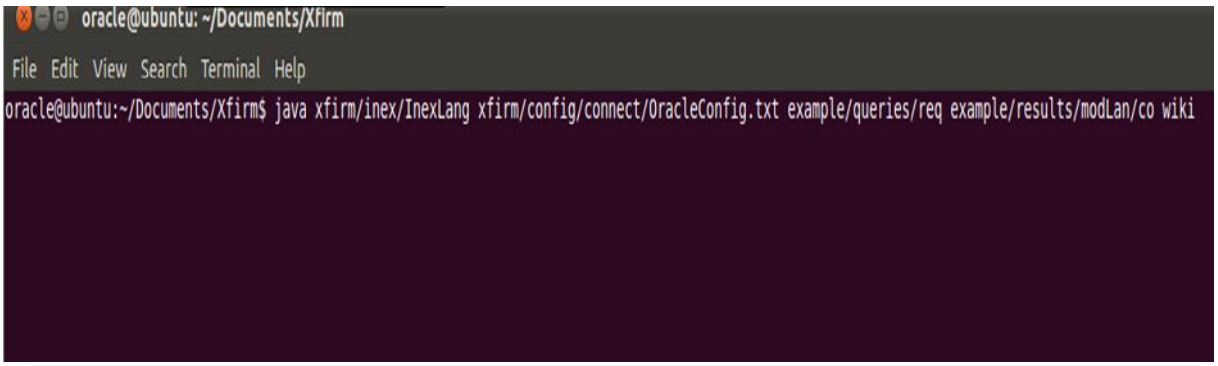



**Figure IV-8 :** Exemple d'une requête de test par le modèle de langue proposé

### 4. La mesure d'évaluation :

Afin d'évaluer notre approche, nous avons utilisé les métriques d'évaluation de INEX 2006 (Voir chapitre I).

Nous avons utilisé la classe *Resultat_final* pour calculer la valeur nxCG des scores :

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

Le principe consiste à calculer la valeur nxCG pour 5 rangs, ainsi *i* aura les valeurs (5, 10, 25, 50, max) qui représentent respectivement les nombre des premiers résultats considérés et la totalité des résultats.

Cette valeur est calculée pour le système que nous avons conçu à base de notre modèle, ainsi que pour le système XFIRM en utilisant les mêmes éléments de test, dans le bute de comparer notre approche à une approche différente.

**xCG** représente la somme des scores des *i* premiers résultats obtenu par le système considéré.

**xCI** représente la somme des *i* premiers scores obtenus pour les résultats de INEX 2006 (les scores idéales)

### 5. Tests et résultats :

Dans le but de comparer notre approche à une approche différente, nous avons lancé les mêmes tests sur le système XFIRM, et notre système qui implémente notre modèle, et voici les résultats obtenus :

- Les résultats obtenus pour le système XFIRM :

| Gain / Requête | Gain à 5 | Gain à 10 | Gain à 25 | Gain à 50 | Gain sur la totalité |
|---|---|---|---|---|---|
| **289** | 0.36797 | 0.12397 | 0.05778 | 0.03023 | 0.00681 |
| **290** | 0.60411 | 0.23964 | 0.12739 | 0.07775 | 0.06238 |
| **295** | 0.40824 | 0.15318 | 0.07372 | 0.03898 | 0.00733 |
| **297** | 0.50477 | 0.21346 | 0.11045 | 0.05830 | 0.01213 |
| **321** | 0.63571 | 0.23271 | 0.12725 | 0.069530 | 0.01171 |
| **322** | 0.67522 | 0.25722 | 0.12906 | 0.06941 | 0.02469 |
| **323** | 0.33505 | 0.11225 | 0.05208 | 0.02722 | 0.00866 |
| **324** | 0.36292 | 0.12378 | 0.05831 | 0.03055 | 0.00540 |
| **325** | 0.70369 | 0.26034 | 0.12965 | 0.06833 | 0.01193 |
| **327** | 0.60822 | 0.23544 | 0.12568 | 0.06730 | 0.01179 |
| **328** | 0.38126 | 0.12929 | 0.06096 | 0.03205 | 0.00742 |
| **362** | 0.54916 | 0.19030 | 0.09532 | 0.05227 | 0.00928 |
| **364** | 0.64507 | 0.31998 | 0.34118 | 0.29901 | 0.30054 |
| **365** | 0.43273 | 0.17188 | 0.10611 | 0.05774 | 0.01145 |
| **366** | 0.58469 | 0.21195 | 0.10621 | 0.05745 | 0.01534 |
| **367** | 0.44465 | 0.15308 | 0.07429 | 0.03926 | 0.00661 |
| **393** | 0.47278 | 0.17107 | 0.08973 | 0.04872 | 0.00985 |
| **408** | 0.60835 | 0.23324 | 0.14181 | 0.08681 | 0.03423 |
| **410** | 0.45752 | 0.17556 | 0.09701 | 0.05409 | 0.00974 |
| **413** | 0.64888 | 0.25607 | 0.14534 | 0.08611 | 0.01719 |
| **La moyenne** | **0.52155** | **0.19822** | **0.11247** | **0.06755** | **0.02922** |

**Tableau IV-4 :** Les résultats de test obtenus pour le système XFIRM

Voici l'histogramme associé :

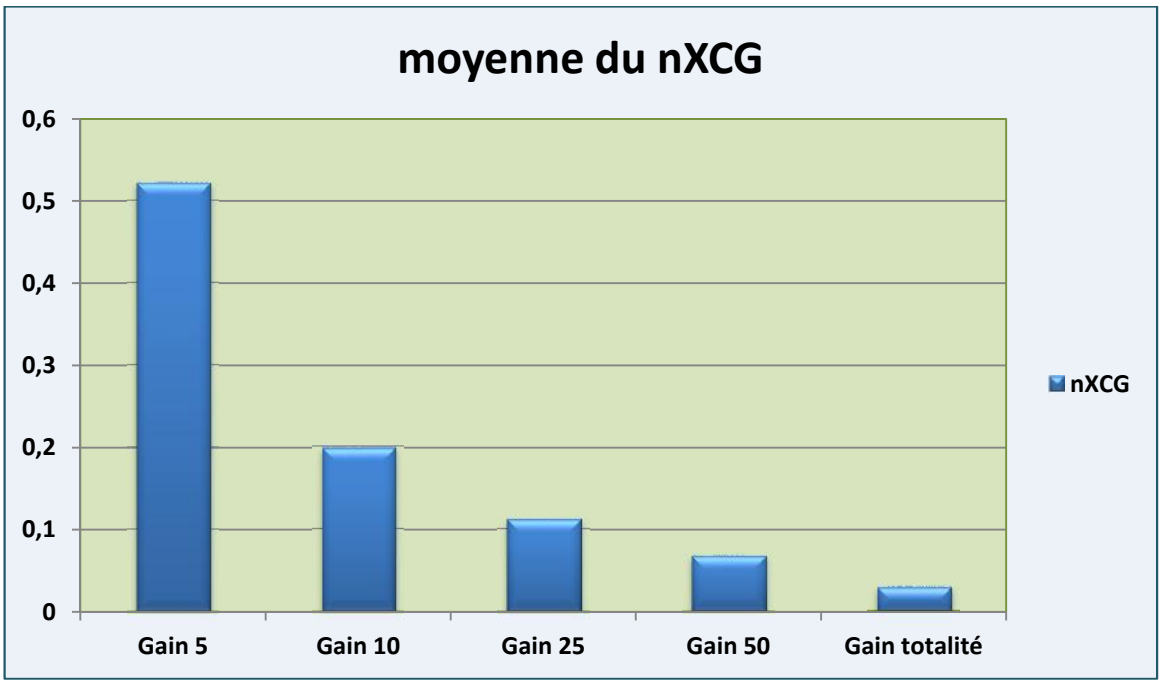



**Figure IV-9 :** L'histogramme associé aux valeurs du gain pour les différents niveaux pour le système XFIRM

- Les résultats obtenus pour notre modèle :

| Gain / Requête | Gain à 5 | Gain à 10 | Gain à 25 | Gain à 50 | Gain sur la totalité |
|---|---|---|---|---|---|
| **289** | 0.88885 | 0.42375 | 0.36727 | 0.31054 | 0.14579 |
| **290** | 0.88203 | 0.41243 | 0.27368 | 0.16782 | 0.01596 |
| **295** | 0.98779 | 0.54145 | 0.54619 | 0.44362 | 0.06047 |
| **297** | 0.92249 | 0.47197 | 0.49806 | 0.45314 | 0.06696 |
| **321** | 0.88545 | 0.45121 | 0.47225 | 0.40077 | 0.02529 |
| **322** | 0.87796 | 0.42614 | 0.41614 | 0.39219 | 0.01249 |
| **323** | 0.79078 | 0.31713 | 0.18170 | 0.10812 | 0.01295 |
| **324** | 0.97754 | 0.51608 | 0.48031 | 0.40376 | 0.01952 |
| **325** | 0.95183 | 0.49412 | 0.37576 | 0.21860 | 0.03147 |

| | | | | | |
|---|---|---|---|---|---|
| **327** | 0.99155 | 0.53353 | 0.45880 | 0.36502 | 0.15996 |
| **328** | 0.85660 | 0.38572 | 0.28185 | 0.18329 | 0.03867 |
| **362** | 0.94555 | 0.50263 | 0.51589 | 0.45183 | 0.06939 |
| **364** | 0.93064 | 0.48351 | 0.50430 | 0.44836 | 0.01434 |
| **365** | 0.95067 | 0.49943 | 0.46518 | 0.38192 | 0.06259 |
| **366** | 0.67806 | 0.29945 | 0.18137 | 0.11526 | 0.02762 |
| **367** | 1 | 0.60024 | 0.56116 | 0.55533 | 0.04227 |
| **393** | 0.93967 | 0.48133 | 0.43653 | 0.37080 | 0.03130 |
| **408** | 0.99145 | 0.52034 | 0.45976 | 0.32450 | 0.02467 |
| **410** | 0.98779 | 0.57840 | 0.53628 | 0.51469 | 0.35453 |
| **413** | 0.92344 | 0.49047 | 0.42967 | 0.27896 | 0.02085 |
| **La moyenne** | **0.91826** | **0.46741** | **0.42617** | **0.34443** | **0.06186** |

**Tableau IV-5 :** Les résultats de test obtenus pour notre modèle

Et voici l'histogramme associé :

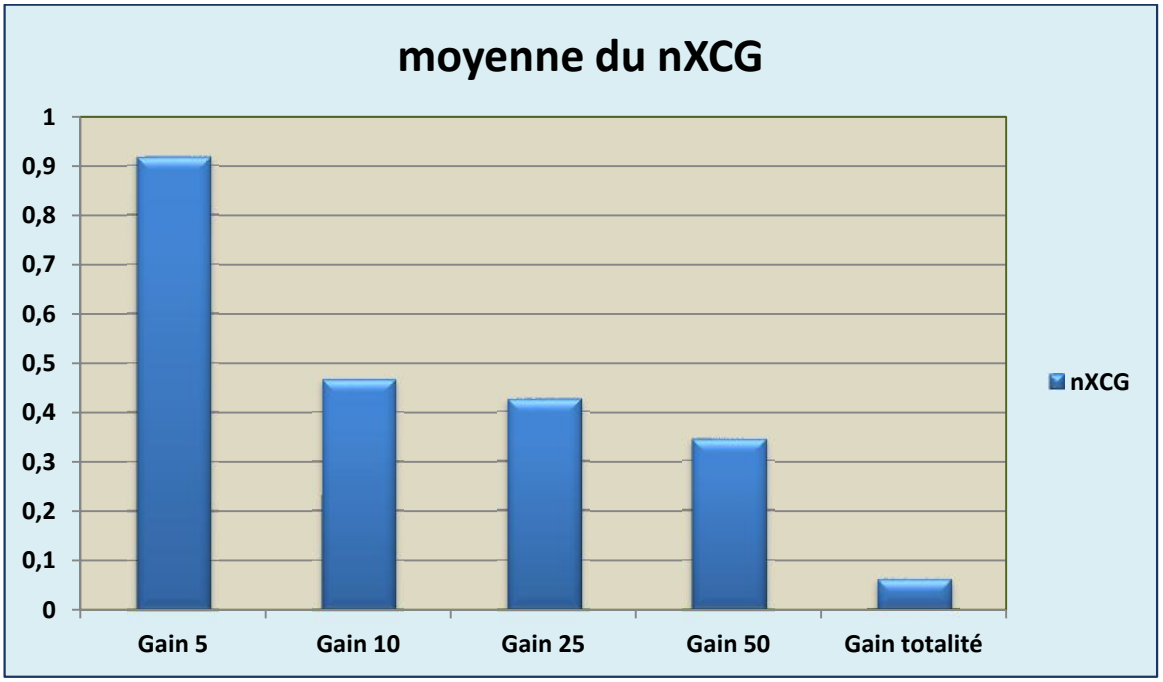



**Figure IV-10 :** L'histogramme associé aux valeurs du gain pour les différents niveaux pour notre modèle

Tableau comparatif :

| Niveau / Système | Gain à 5 | Gain à 10 | Gain à 25 | Gain à 50 | Gain sur la totalité |
|---|---|---|---|---|---|
| **XFIRM** | 0.52155 | 0.19822 | 0.11247 | 0.06755 | 0.02922 |
| **Notre Modèle** | 0.91826 | 0.46741 | 0.42617 | 0.34443 | 0.06186 |
| **Tôt d'amélioration dans notre modèle** | **+39.671%** | **+26.919%** | **+31.37%** | **+27.688%** | **+3.264%** |

**Tableau IV-6 :** Tableau comparatif des résultats obtenu pour XFIRM et notre modèle

Voici l'histogramme associé :

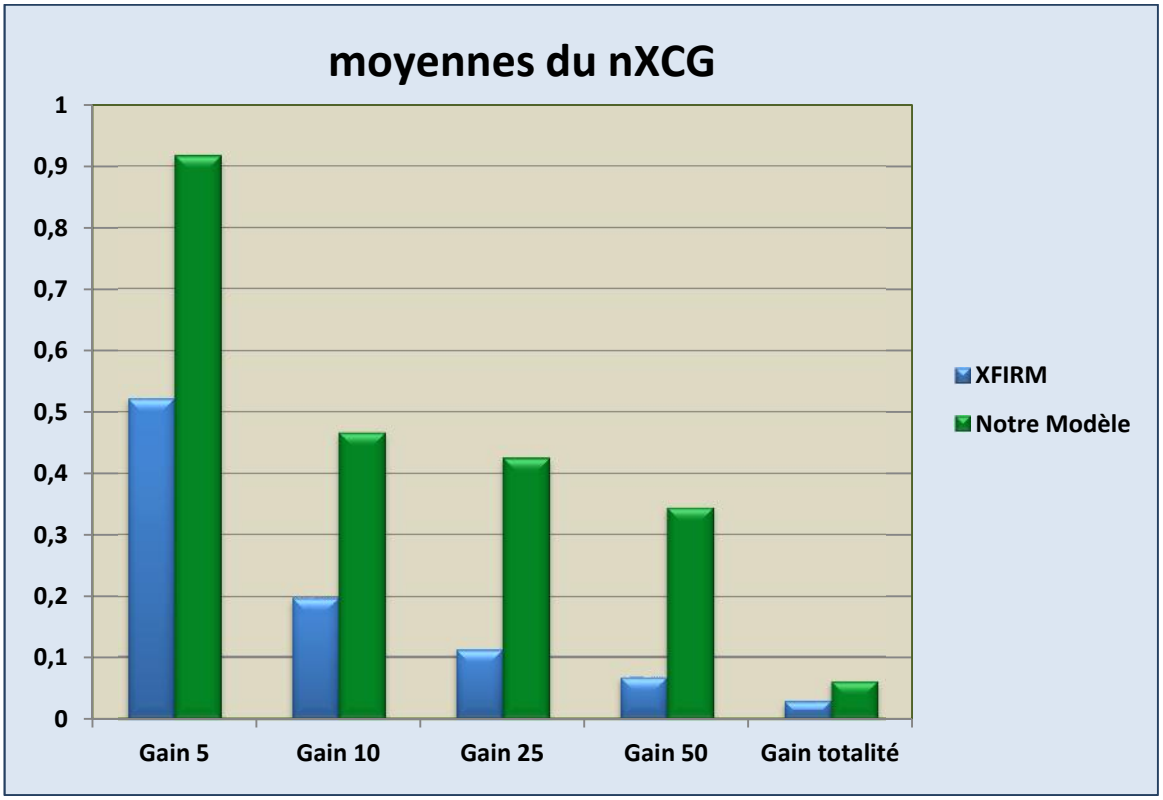



**Figure IV-11 :** L'histogramme associé au tableau comparatif entre notre modèle et XFIRM

**Interprétation :**

L'objectif de la comparaison au système xfirm, est de juger de la qualité des résultats obtenus, avec le modèle de langue proposé, par rapport à un modèle différent et qui a prouvé son efficacité, pour cela, nous avons montré au-dessus la démarche suivie pour effectuer les tests et pouvoir par la suite comparer nos résultats aux résultats obtenus avec le système xfirm qui est un système très utilisé dans les implémentation des approches de recherche, en fournissant les mêmes éléments de tests aux deux systèmes : le système xfirm et le système qui implémente notre modèle.

Le tableau et l'histogramme associé ci-dessus, montrent les différences entre les gains obtenu sur cinq (05) niveaux (les 5, les 10, les 25 et les 50 premiers résultats ainsi que la totalité), on remarque que notre modèle à pu aboutir jusqu'à plus de 39.6% d'amélioration par rapport au modèle implémenté par xfirm cela sur les cinq (05) premiers résultats, ainsi que plus de 26.9% d'amélioration du gain par rapport aux dix (10) premiers résultats et plus de 3.26% pour la totalité des résultats.

Donc, en comparant notre modèle à celui d'xfirm, nous avons aboutis à des améliorations remarquables en utilisant le modèle de langue proposé.

Rappelons que notre modèle a adapté la nouvelle granularité d'information offerte par la recherche dans les documents XML à un modèle proposé dans la RI classique, ce dernier

prends en compte la taille d'un document texte dans l'établissement des probabilités de similarité ainsi que les techniques de lissage [Achemoukh, 2006]. Nous avons développé notre approche de recherche en utilisant ce principe, en prenant en compte un ensemble de paramètres associé : un élément résultat est un nœud de l'arbre XML représentant le document, ainsi qu'un terme présent dans un tel élément peut ne pas l'être dans un autre élément voisin ou un peu loin dans le même document ou apparaitre dans un autre document carrément.

De ce fait, dans notre approche, les probabilités de similarité et les techniques de lissage que nous avons adapté, prennent en compte ces notions par l'intégration des paramètres illustrant l'importance de chacune de ces caractéristiques dans les probabilités de calcul des scores, ce qui nous a permis d'aboutir à des tels améliorations et résultats de recherche.

**Conclusion :**

Dans cette partie, nous avons présenté notre démarche d'implémentation et tests, ainsi que l'ensemble des résultats obtenus évalués et comparés au système xfirm. Nous avons constaté que notre modèle a présenté une amélioration du gain sur les différents niveaux considérés, par un modèle de langue adapté à la nouvelle granularité d'information, du fait que le paramétrage des probabilités de similarité prend en compte cette caractéristique.

En conclusion, pour qu'un modèle RI généralement et dans le cadre de la RI XML précisément, soit réussi, il doit prendre en compte les caractéristiques de son domaine d'application, dans notre cas : les probabilités de similarité doivent prendre en compte la nouvelle granularité d'information présentée en RIS, tel qu'un résultat est un élément pouvant être un nœud feuille ou interne ou racine, ainsi qu'un élément s'il est pertinent cela impliquera la pertinence de ses prédécesseurs d'où le document tout entier, de plus, un terme de la requête peut être présent dans un nœud feuille (texte) du document XML donc présent dans ses prédécesseurs d'où sa présence dans le document, ou contrairement, absent dans un élément ou dans tout le document mais apparait dans un élément d'un autre document de la collection.

# Conclusion générale

# Conclusion générale :

## 1. Objectif :

Notre travail se situe dans le contexte de la recherche d'information structurée dans les documents XML (RI XML). L'objectif est de proposer une adaptation d'un modèle de langue à la nouvelle granularité de l'information présentée par la RIS. Dans ce contexte, le résultat peut être soit un élément ou un ensemble d'éléments ou encore un document tout entier.

Dans cette optique, il s'avère nécessaire d'adapter les techniques de recherche dans la RI classique à la nouvelle granularité d'information présentée. De plus, l'information portée par un document de la collection n'est plus simple, mais elle est composée d'information textuelle ainsi que d'information structurelle. De ce fait, la collection est vue comme un ensemble d'arborescences XML dont les nœuds sont porteurs d'information. Ce qui implique qu'un modèle de recherche, dans le cadre de la RI XML, doit prendre en compte les nouvelles caractéristiques de recherche.

Dans le cadre des modèles de langue, nous avons remarqué au cours du chapitre II, qu'un modèle de langue perçoit la pertinence sous un nouvel angle, car les formules de similarité calculent la probabilité qu'une requête utilisateur puisse être inférée du modèle du document. Plusieurs travaux ont été présentés, parmi lesquels on trouve celui qui prend en compte la taille d'un document [Achemoukh, 2006]. L'approche de recherche par modèle de langue est adaptée à la nouvelle granularité d'information XML, par substitution du modèle du document tout entier par celui d'un élément de celui-ci. Au cours du chapitre II, nous avons remarqué que les approches proposées pour les modèles de langue dans le cadre de la RI XML sont peu nombrables. C'est pour cela que nous avons contribué à proposé une adaptation dans ce cadre. Parmi les travaux qui ont été proposés : utilisation de la structure dans le processus d'appariement [Baeza & Ribeiro, 2011], la contextualisation des éléments [Cyril, 2013], l'uniformité des éléments par rapport à leurs tailles [Mulhem & Chevalet, 2010] et l'extension du concept de recherche par le processus d'interpolation entre le modèle de l'élément en cours et celui de son parent [Ganguly & al, 2011]…

Dans toutes les approches résumées dans ce travail, les auteurs prenaient en compte le modèle de l'élément soit par inclusion de sa taille ainsi que la fréquence d'un terme dans celui-ci, dans le modèle du document, ou par relations « ancêtre-descendant » entre les éléments. Mais les termes d'une requête présents dans un élément influencent énormément sa pertinence, ainsi que la taille d'un document en nombre d'éléments le constituant ainsi que la taille de la collection en nombre d'éléments.

## 2. Synthèse de la proposition et des résultats :

Ce travail contient une proposition d'un modèle de langue pour la RI XML, par adaptation du modèle présenté dans [Achemoukh, 2006] et qui prend en compte la taille d'un document texte, dans le cadre de la RI classique.

Dans un premier temps, on peut choisir de faire une recherche dans une collection de documents pertinents, en utilisant une formule qui estime les scores associés aux documents de la requête, le but est d'en tirer les documents de la mini-collection pertinente et nous permettre d'avoir une idée à priori sur le taux de satisfaction d'une requête.

L'idée de base est de considérer le document comme un arbre XML selon sa taille en fonction du nombre d'éléments le constituant et prise en compte de leurs probabilités de similarité avec un terme de la requête utilisateur, indépendamment du document tout entier, puis appliquer la distribution par rapport à tous les termes de la requête dans un élément donné. Pour cela, nous avons proposé deux lissages : le premier calcule la probabilité d'importance d'un terme selon sa présence ou absence dans un élément et le deuxième est un lissage par interpolation du modèle de l'élément par celui du document et celui de la collection.

Nous calculons aussi des probabilités associées aux éléments d'un document de la collection indépendamment des termes de la requête, c'est la probabilité d'avoir un tel élément comme résultat à une requête. Les probabilités associées aux éléments sont combinées par la suite avec la distribution appliquée aux termes de la requête pour en déduire la probabilité finale : le score de pertinence d'un élément pour une requête.

Après implémentation et tests de l'approche proposée, nous avons comparé nos résultats aux résultats d'un système différent et qui ne se base pas sur un modèle de langue : le système XFIRM [Sauvagnat, 2005]. Au cours du chapitre IV, nous avons présenté l'approche d'implémentation ainsi que les éléments de tests (requêtes et collection XML) de la compagne d'évaluation INEX 2006. Nous avons par la suite évalué les résultats obtenus, en utilisant la mesure du gain que nous avons calculé sur cinq différents niveaux.

La remarque principale est que les valeurs du gain obtenues été proches de 1, en comparaison, nous avons pu aboutir à un gain de plus de +3% sur la totalité des résultats jusqu'à plus de +39% sur les cinq premiers résultats. On rappelle que le but de cette comparaison est d'affirmer la similitude de nos résultats ainsi que l'efficacité du modèle proposé.

En conclusion, le modèle de langue proposé a pu aboutir à des résultats satisfaisants grâce à l'exploitation des différentes caractéristiques d'un document XML, que nous avons prise en compte sous forme de paramètres intégrés dans les formules de calcul de score ainsi que les algorithmes de recherche, comme adaptation d'un modèle probabiliste à la nouvelle granularité d'information.

## 3. Perspectives :

Les perspectives à court et moyen termes, que nous envisageons à la suite de notre travail, sons les présentes :

- Comme nous avons proposé deux formes pour la probabilité associée à un élément résultat (la formule 3.1 et sa variante 3.2), et que nous avons juste testé notre approche avec la formule de base 3.1, nous envisageons de lancer des tests avec la variante 3.2, en utilisant les mêmes éléments de tests et comparer les résultats pour décider de la meilleur proposition ;
- Comme nous avons présenté deux formules pour le calcul de la probabilité de vraisemblance maximale, nous envisageons de tester la deuxième (5.2) et comparer ses résultats à ceux de la formule de base (5.1) que nous avons utilisé dans nos tests ;
- Comme nous avons proposé deux algorithmes de lissage et que nous n'avons testé que le deuxième algorithme (par interpolation), nous envisageons tester le lissage implémenté par l'algorithme (6) et comparer ses résultats à ceux obtenus avec le lissage implémenté par l'algorithme (2).

Avec ces tests, nous pourrions fixer les formules et algorithmes adéquats pour notre modèle, pour :

- Tester par la suite l'approche proposée en utilisant toute la collection INEX 2006 sur toutes les requêtes et évaluer les résultats avec d'autres mesures ;
- Tenir compte de l'informativité des éléments résultats, par définition d'un seuil minimal pour la taille d'un élément résultat ;
- Faire varier les paramètres intégrés dans les probabilités de score.

# BIBLIOGRAPHIE

***BIBLIOGRAPHIE***


- **[Ait Ali & Amrouche, ENSI]** Réseaux bayésiens jumelés et noyau de Fisher pondéré pour la classification de documents XML : Ait Ali Yahia Yassine et Amrouche Karima, Ecole Nationale Supérieure d'Informatique
- **[Alilaouar, 2007**] THÈSE Présenté devant l'Université Paul Sabatier de Toulouse en vue de l'obtention du Doctorat de l'Université Paul Sabatier par Abdeslame ALILAOUAR « Contribution à l'interrogation flexible de données semi-structurées » 2007
- **[Achemoukh, 2006] :** contenant un modèle de langue pour la RI classique se basant sur la taille d'un document. LARI-UMMTO 2006
- **[Amirouche, 2008]** « Contribution à la définition de modèles de recherche d'information flexibles basés sur les CP-Nets », Fatiha BOUBEKEUR-AMIROUCHE : Toulouse, 2008
- **[Anderson & Perez-Carballo, 2000]** "The nature of indexing"
- **[Arnaud & al., 2012]** Une approche de recherche d'information structurée fondée sur la correction d'erreurs à l'indexation des documents
  Arnaud Renard, Sylvie Calabretto, Béatrice Rumpler : Lyon/INSA-Lyon 2012
- **[Belbachir & Boughanem, 2013] :** Faiza Belbachir et Mohand Boughanem, *« Modèles de langue pour la détection d'opinions dans des blogs »*
- **[Ben Aouicha, 2009]** « Une approche algébrique pour la recherche d'information structurée », Mohamed BEN AOUICHA : Université Paul Sabatier de Toulouse, 2009.
- **[Baziz, 2005]** « INDEXATION CONCEPTUELLE GUIDEE PAR ONTOLOGIE POUR LA RECHERCHE D'INFORMATION », Mustapha BAZIZ : Paul Sabatier 2005
- **[Bessai & Boughanem, 2006] :** Fatma-Zohra BESSAI MECHMACHE et Mohand BOUGHANEM**, «** *Vers un modèle possibiliste pour la recherche d'information dans des documents structurés »*. Université Paul Sabatier IRIT-SIG
- **[Boughanem, 2003] :** Mohand Boughanem, *« Modèle de langue pour la recherche d'information »*. Institut de recherche en informatique de Toulouse
- **[Boughanem & al]** Recherche d'Information Structurée : « Vers un modèle possibiliste pour la recherche d'information dans des documents structurés » Fatma-Zohra BESSAI MECHMACHE et Mohand BOUGHANEM
- **[Boughanem, EARIA 2006 : cours]** Introduction à la Recherche d'Information : M. Boughanem Université Paul Sabatier de Toulouse Laboratoire IRIT (EARIA' 2006)
- **[Boughanem & al, IRIT/Paul Sabatier]** « Un modèle de recherche d'information sociale dans les microblogs : cas de Twitter » Lamjed Ben Jabeur, Lynda Tamine et Mohand Boughanem *IRIT, Université Paul Sabatier*
- **[Boughanem, 2000] :** Mohand Boughanem, « *Modèles de langue pour la recherche d'information* **».**Institut de Recherche en Informatique de Toulouse.
- **[Caroline Tambellini & CatherineBerrut labo : Clips-Imag]** Pondération des données incertaines dans les systèmes de recherche d'informations : une première approche expérimentale.
  Caroline Tambellini, Catherine Berrut : laboratoire CLIPS-IMAG
- **[Chargheri & al., 2012]** « Classification de documents combinant la structure et le contenu ». 2012
  Samaneh Chagheri*, Catherine Roussey**, Sylvie Calabretto*, Cyril Dumoulin***

**Université de Lyon, CNRS, LIRIS UMR 5205, INSA de Lyon, 7 avenue Jean Capelle, Villeurbanne, France*
***Irstea/Cemagref, Campus des Cézeaux, Clermont Ferrand, 24 avenue des Landais, Aubière, France*
****27, rue Lucien Langénieux, Roanne, France*

- **[Chebili, 2011] «** Agrégation des résultats dans la recherche d'information Semi-Structurée » Hicham CHEBILI, ESI 2011
- **[Chinenyanga & N.Kushmerick]** "An expressive and efficient language for XML IR"
- **[Christophe Brouard, AMA Grenoble]** « Comparaison du modèle vectoriel et de la pondération tf*idf associée avec une méthode de propagation d'activation » Christophe Brouard *UPMF-Grenoble2/CNRS LIG UMR 5217/équipe AMA Grenoble, France Coria 2013*
- **Cours 2** « Les technologies XML »: Transformation et Formatage de documents XML : Cours 2.2 : XSLT *Novembre 2010 - Version 3.2 –*
- **[Cyril, 2013] :** Cyril Laitang, *« Impact de la structure des documents XML sur le processus d'appariement dans le contexte de la recherche d'information Semi-Structurée »*, Université Toulouse.
- **[E.Kotsakis :** SIR in XML documents**]**
- **[Elayeb, 2009]** « SARIPOD: Système multi-Agent de Recherche Intelligente POssibiliste de Documents Web » : Elayeb Bilel, Toulouse 2009
- **[Eyrolls, 2008]** « Programmer en JAVA : Best of Java », Claude Delannoy :Eyrolls 2008
- **[Fellag, 2006]** MEMOIRE DE MAGISTER SPECIALITE: SYSTEMES INFORMATIQUES. Présenté par: Mme Samia FELLAG née BERCHICHE 2006
- **[Fernandez & al., 2003]** Fernandez, Malhotra, Marsh et Walsh : « Modèle de données XQuery 1.0 et XPath 2.0 » : Rapport technique, World Wide Web Consortium (W3C) May 2003.
- **[Franklin & al., 2002]** « Accelerating XPath Location Step », In M.J.Franklin, B.Moon, and A.Ailamaki, editors, Proceeding of the 2002 ACM SIGMOD Internationnal Conference on Management of Data, Madison, Wisconsin, USA, Pages 109-120.ACM, 2002.
- **[George Badr, 2007]** « Recherche d'Information sur le Web: Impact de la structure des documents sur la pertinence des résultats » Par Georges BADR , Equipe SIG 2006-2007
- **[George Zipf, 1949]** « *Human Behaviour and the Principle of Least Effort"*. Addison-Wesley, 1949.
- **[Géry Mathias & al., Lyon/Saint-Etienne]** BM25t : une extension de BM25 pour la Recherche d'Information ciblée Mathias Géry, Christine Largeron, Franck Thollard : Lyon, Saint-Étienne, France (Laboratoire Hubert Curien).
- **[Hammache, 2014] :** Arezki Hammache, « Recherche d'information : un modèle de langue combinant mots simples et mots composés ». *LARI*-UMMTO
- **[Hammache & al, 2013] :** Arezki Hammache, Mohamed Boughanem et Rachid Ahmed-Ouamer, *« Pseudo-réinjection de pertinence basée sur un modèle de langue mixte combinant les termes simples et composés »*, LARI-UMMTO, IRIT-Paul Sabatier.
- **[Hlaoua, 2007]** « Reformulation de Requêtes par Réinjection de Pertinence dans les Documents Semi-Structurés », Lobna Hlaoua : Toulouse 2007

- **[INEX, 2012]** "Report on INEX 2012"
  P. Bellot, T. Chappell, A. Doucet, S. Geva, S. Gurajada, J. Kamps, G. Kazai, M. Koolen, M. Landoni, M. Marx, A. Mishra, V. Moriceau, J. Mothe, M. Preminger, G. Ramírez, M. Sanderson, E.Sanjuan, F. Scholer, A. Schuh, X. Tannier, M. Theobald, M. Trappett A. Trotman Q. Wang
- **[INEX, 2007]** « INEX 2007 Evaluation Measures (Draft) »
  Jovan Pehcevski1, Jaap Kamps2, Gabriella Kazai3, Mounia Lalmas4, Paul Ogilvie5, Benjamin Piwowarski6, and Stephen Robertson3
  1 INRIA Rocquencourt, France
  2 University of Amsterdam, The Netherlands
  3 Microsoft Research Cambridge, United Kingdom
  4 Queen Mary, University of London, United Kingdom
  5 Carnegie Mellon University, USA
  6 Yahoo! Research Latin America, Chile
- **[J.E.Wolff & al]**: J.E.Wolff, H.Florke and A.B.Cremers : Searching and browsing collections of structural information
- **[Keddache & Kecili, 2012]** « Intégration de formules de pondération au système de recherche dans les documents XML : XFIRM »
  Réalisé par : Melle KEDDACHE Fadhila et Melle KECILI Amel 2011/2012
- **[Lalmas & Vannoorenberghe, London]** « Indexation et recherche de documents XML par les fonctions de croyance »
  Mounia Lalmas et Patrick Vannoorenberghe
  *Department of Computer Science :Queen Mary University of London, London E1 4NS, United Kingdom Perception Systèmes Information, FRE 2645 CNRS*
  *Université de Rouen, Faculté des Sciences*
- **[Lalmas & Vannoorenberghe, London/Roueen]** Indexation et recherche de documents XML par les fonctions de croyance : Mounia Lalmas et Patrick Vannoorenberghe, *Department of Computer Science Queen Mary University of London Perception Systèmes Information et Université de Rouen, Faculté des Sciences….F12-2048..(pondérationXML)*
- **[Laure Soulier & al, 2012] :** Laure Soulier, Lamjed Ben Jabeur, Lynda Tamine et Wahiba Bahsoun, *« Modèle de langue pour l'ordonnancement conjoint d'entités pertinentes dans un réseau d'informations hétérogènes »*. *IRIT, Université Paul Sabatier*
- **[L.Maisonnasse & al, 2008] :** L.Maisonnasse, E.Gaussier et J-P.Chevallet, *« Modèle de relations dans l'approche modèle de langue en recherche d'information »*. Université de Grenoble et IPAL-I2R
- **[Louis, 2002]** « Les technologies XML » Denis MONASSE Lycée Louis le Grand 29 avril 2002
- **[Maïsonnasse, 2006] :** Lïc Maïsonnasse, *« Intégration de connaissances syntaxiques dans les modèles de langue pour la RI »*, CLIPS-IMAG.
- **[Mulhem & Chevallet, 2009] :** Philippe Mulhem et Jean-Pierre Chevallet, *« Un modèle de contexte documentaire par doxels pondérés »,* UJF-Grenoble, UPMF-Grenoble / GrenobleINP / CNRS, LIG UMR 5217, Grenoble, F-38041
- **[Mulhem & Chevallet, 2008] :** Philippe Mulhem et Jean-Pierre Chevallet, *« Modèle de langue par type de doxel pour l'indexation de documents structurés »*, LIG-CNRS, Université de Grenoble

- **[Nait Djoudi, 2012]** "Implémentation et expérimentation d'une méthode de propagation des termes », Nait Djoudi Brahim, UMMTO 2012.
- **[Pierre Pompidor & al., 2009] «** Indexation de co-occurrences dans des corpus de documents structurés et production de cartes sémantiques interactives »
  Pierre Pompidor , Boris Carbonneill , Michel Sala 2009.
- **[Popovici & al, 2007] :** Eugen Popovici, Gildas Ménier et Pierre-François Marteau, *« Interprétation vague des contraintes structurelles pour la RI dans des corpus de documents XML »*, VALORIA, Bretagne
- **[Rana, 2006]** Université Paul Sabatier– I.R.I.T. DEA d'Informatique
  « ANALYSE DES PARAMETRES DE PONDERATION DANS LE CADRE DE COLLECTIONS VOLUMINEUSES »
  Rana EL CHARIF Année universitaire 2005/2006
- **[Rijsbergen, 1979]** C. van Rijsbergen. *Information retrieval*. Butterworths, 1979.
- **[Robertson & al., Microsoft Research]** "Simple BM25 Extension to Multiple Weighted Fields" Stephen Robertson, Hugo Zaragoza and Michael Taylor Microsoft Research. Cambridge, U.K.
- **[Robertson et.al, 1981]:** Robertson, S., van Rijsbergen, C., and Porter, M. Probabilistics models of indexing and searching. Information retrieval research, (Ed. W.R. Oddy et al). London :Butteworths (1981)
- **[Romaric Besançon, Paris 13 : cours]** Recherche d'information - Modèles en Recherche d'Information : Cours Master Recherche Paris 13 « Recherche et extraction d'information » A. Rozenknop source : Romaric Besançon CEA-LIST/LIC2M
- **[Sauvagnat, 2005 ]** Karen SAUVAGNAT « Modèle flexible pour la Recherche d'Information dans des corpus de documents semi-structurés » 2005
- **[Sauvagnat & Boughanem, IRIT – Equipe SIG]** Le langage de requêtes XFIRM pour la recherche d'information dans les documents XML : Karen Sauvagnat, Mohand Boughanem : IRIT – Equipe SIG / RI
- **[Said L'hadj & al, 2006] :** Lynda Said L'hadj, Mohand Boughanem et Karima Amrouche, *«Vers un modèle de langue mixte concepts-mots pour la recherche d'information »*, ESI, IRIT.
- **[Said L'hadj & Boughanem, 2006] :** Lynda Said L'hadj et Mohamed Boughanem, *« Modèle de Langue à base de Concepts pour la Recherche d'Information »*, STIC-ESI, IRIT-Paul Sabatier.
- **[Savoy & Berger, 2004] :** Jacques Savoy et Pierre Yves Berger, *« Recherche bilingue et multilingue d'information »,* Université de Neuchâtel, Pierre-à-Mazel
- **[S. Briet, 1951] :** Suzanne Briet , « *Qu'est ce que la documentation ? ».* Paris. EDIT. 1951
- **[S.Cohen & J.Mamon & Y.Kanza & Y.Sagiv]** : XSErch : A semantic search engin for XML.
- **[Tebri, 2004]** Formalisation et spécification d'un système de filtrage incrémental d'information : Hamid TEBRI, 15/12/2004.
- **[Torjmen, 2009]** « Approches de Recherche Multimédia dans des Documents Semi-Structurés Utilisation du contexte textuel et structurel pour la sélection d'objets multimédia » Mouna TORJMEN, 2009.
- **[T.Schlieder & H.Meuss]** "Querying and Ranking XML Documents"
- **Xavier Tannier, Pari-sud11 : cours]** « Indexation et Recherche d'Information : Modèles de Recherche et Évaluation »
  Xavier Tannier Université Paris-sud 11

- **[Ughetto & al, 2011] :** L.Ughetto, V.Claveau et R.Harastani, *« Différentes interprétations d'un modèle de RI à base d'inclusion graduelle ». CORIA 2011.*
- **[V.N.Anh & A.Moffat]** "Compression and an IR approch to XML retrieval"
- **[Yannick Loiseau, 2004 ]** Recherche flexible d'information par filtrage fou qualitatif : Yannick Loiseau, 16/12/2004
- **[Yuanhua & ChengXiang, 2011]** " Adaptive Term Frequency Normalization for BM25" Yuanhua Lv , ChengXiang Zhai Urbana, IL 61801: 2011
- **[Zargayouna.H, 2004]** Contexte et sémantique pour une indexation de documents semi-structurés.


***Webographie*:**


- http://www.eclipse.org
- http://www.commentcamarche.net/contents/702-oracle-introduction-au-sgbd-oracle
- http://msdn.microsoft.com/fr-fr/library/ms256190%28v=VS.80%29.aspx
- http ://www.Wikipedia.com
- http://www. Wikilingue_Encydia.com
- http://hal.inria.fr/hal-00954051
- http://www.iro.umontreal.ca/~nie/IF

# ANNEXE

**Annexe I :**

**Partie 1 :**

**Les arbres :** un **arbre** est une structure de données qui peut se représenter sous la forme d'une hiérarchie dont chaque élément est appelé nœud, le nœud initial étant appelé *racine*. Dans *un arbre binaire*, chaque élément possède au plus deux éléments fils au niveau inférieur, habituellement appelés *gauche* et *droit*. Du point de vue de ces éléments fils, l'élément dont ils sont issus au niveau supérieur est appelé *père*. Les *arbres binaires de* recherche sont des arbres binaires dont le fils gauche est inférieur au père qui est inférieur au fils droit. *Les n-arbre* sont des arbres dont le maximum de fils pour un nœud est *n*

**Autres modèles RI :**

- **Le modèle des ensembles flous :** La théorie des sous-ensembles flous est une théorie mathématique du domaine de l'algèbre abstraite. Elle a été développée par Lotfi Zadeh en 1965 afin de représenter mathématiquement l'imprécision relative à certaines classes d'objets et sert de fondement à la logique floue. Une partie floue (ou sous-ensemble flou) d'un ensemble $E$ est une application de $E$ dans [0,1] :

$$E \rightarrow [0,1]$$
$$x \rightarrow v_{(0<v<1)}$$

- **Modèle flou :**
  La représentation des ensembles reflète partiellement les contenus sémantiques des documents et des requêtes. Par conséquent, la correspondance d'un document avec les termes d'une requête est approximative, d'où l'extension du modèle booléen basée sur les ensembles flous qui a été proposée par Salton [Salton, 1989]. Cette extension vise également à tenir compte de la pondération des termes dans les documents. Un poids d'un terme exprime le degré d'appartenance de ce terme à un ensemble. Ainsi, un document peut être représenté par un ensemble de termes pondérés comme suit :

$$D_j = \{(td_{1j}, a_{1j}), \ldots, (td_{nj}, a_{nj})\}$$

Où $td_{ij}$ est le $i^{éme}$ terme du document $D_j$, et $a_{ij}$ est le degré d'appartenance (une valeur comprise entre 0 et 1) du $i^{éme}$ terme au document $D_j$.
L'évaluation d'une requête $Q_k$ par rapport à un document $D_j$ peut prendre plusieurs formes. Un exemple de requêtes floues est présenté dans le tableau suivant :

| Requête $(Q_k)$ | $rsv(D_j, Q_k)$ | Le résultat de l'évaluation |
|---|---|---|
| $tq_{ki}$ | $rsv(D_j, tq_{ki})$ | $= a_{ij}$ |
| $tq_{k1} \wedge tq_{k2}$ | $rsv(D_j, tq_{k1} \wedge tq_{k2})$ | $= min\ (rsv(D_j, tq_{k1}), rsv(Dj, tqk2))$ |
| $tq_{k1} \vee tq_{k2}$ | $rsv(D_j, tq_{k1} \vee tq_{k2})$ | $= max\ (rsv(D_j, tq_{k1}), rsv(D_j, tq_{k2}))$ |
| $\neg tq_{ki}$ | $rsv(D_j, \neg tq_{ki})$ | $= 1 - a_{ij}$ |

**Tableau :** Evaluation des requêtes dans le modèle booléen/ensembles flous [Tebri, 2004]

- **Le modèle vectoriel généralisé :**

Wong et al. [Wong & al., 1985] ont proposé en 1985 le modèle vectoriel généralisé dans le but de représenter les dépendances entre les termes de l'index, cependant tous les modèles présentés dans la littérature traitent les termes de l'index d'une manière indépendante, ce modèle a essayer d'établir un cadre formel dans lequel les dépendances entre les termes peuvent être facilement représentées. Le modèle vectoriel généralisé est caractérisé par sa complexité et sa lenteur par rapport au modèle vectoriel classique.

- **Le modèle LSI :**

(Latent Semantic Indexing) ce modèle est dérivé du modèle vectoriel dont le but est la correction des défauts de ce dernier. Réalisé par (Dumais 1995, Foltz 1990) le nombre de dimensions est réduits car il utilise la technique de l'analyse en composante principale sur l'espace des termes. Ce modèle réduit l'espace de représentation des documents en ne considérant que les axes porteurs d'information, ie : qui expriment un poids critique du terme dans la requête ou le document.

Ce modèle se base essentiellement sur la décomposition en valeur singulières, désignée par SVD (Singular Value Decomposition) de la matrice représentante : en ligne les termes et en colonnes les documents. Un élément de la matrice représente le poids d'un terme dans un document. La SVD permet d'une part de réduire l'espace des termes d'indexation, et d'autre part, de représenter les documents et les requêtes dans un espace qui ne dépend pas des termes d'indexation mais des concepts contenus dans les documents.
Plusieurs applications de ce modèle ont été proposées en recherche d'information, et également pour le filtrage d'information et la recherche documentaire multilingue
[Dumais et al., 1996] [Foltz and Dumais, 1992].

- **Le modèle connexionniste :**

Il utilise la formalisation des réseaux des neurones dont les neurones expriment les objets de la RI. Cette représentation permet de modéliser facilement les relations qui existent entre les différents éléments du système :

- Entre termes : relation de synonymie
- Entre documents : similarité, classification
- Entre termes et documents : poids du terme dans le document.

Dans l'ensemble, La valeur de pertinence d'un document vis-à-vis d'une requête est calculée par une fonction $Sim(D_j, Q_k)$ d'un document $D_j$ pour la requête $Q_k$ :

$$Sim(D_j, Q_k) = \alpha \cdot RSV_{D_j} + (1 - \alpha) \cdot RSV_{Q_k}$$

Avec : $RSV_{D_j} = \sum_k S(qtf_k / L_q) \cdot W_{dk}$

$$W_{dk} = \log[tf_k / (L_d - tf_k) \cdot (N_w - L_d - F_k + tf_k) / (F_k - tf_k)]$$

$$RSV_{Q_k} = \sum_k S(tf_k / L_d) \square W_{qk}$$

$$W_{qk} = \log[qtf_k / (L_q - qtf_k) \square (N_w - F_k)/F_k]$$

$tf_k$, $qtf_k$ sont les fréquences du terme $t_k$ dans $D_j$ et dans $Q_k$ respectivement ;
$L_d = \sum_k tf_k$ et $L_q = \sum_k qtf_k$ sont les longueurs du $D_j$ et de $Q_k$ ;
$S$ est une fonction **sigmoïde**[1] ;
$F_k = \sum_{docColl} tf_k$ c'est la fréquence du terme $t_k$ dans toute la collection, et
$N_w = \sum_k F_k$ est le nombre de termes de la collection.

- **Le modèle BIR :**

Le modèle BIR (Binary Independence Retrieval), comme dans tous les modèles probabilistes, il cherche à estimer la probabilité qu'un document $D_j$ soit pertinent pour une requête $Q_k$. Pour estimer cette probabilité (notée $P(R|Q_k,D_j)$), une hypothèse sur la distribution des termes dans les documents est considérée. L'hypothèse consiste à considérer que la distribution des termes dans les documents pertinents et non pertinents est différente. Cette hypothèse, appelée aussi « clustering hypothesis », est validée expérimentalement par Van Rijsbergen et Sparck-Jones [Rijsbergen and Sparck-Jones, 1973].
L'idée de base de ce modèle est de représenter les termes des documents par des valeurs binaires (0 ou 1) : un terme apparait dans un document ou non. Ainsi, un document Dj peut être représenté par un vecteur de termes pondéré : $\omega_i \square \{0,1\}\ et\ i \square \{1, \dots N\}$, où N est le nombre de termes du document. Ainsi, la similarité entre le document et la requête est calculée par :

$$\sum_{t_i \square D_j \cap Q_k} log \frac{P_{ik} \square (1 - q_{ik})}{q_{ik} \square (1 - P_{ik})}$$

Les paramètres $p_{ik}$ et $q_{ik}$ sont estimés pour tous les termes de la requête. Robertson et Sparck-Jones proposent quatre principes pour estimer ces paramètres. Deux principes se basent sur les hypothèses concernant l'indépendance des termes et les deux autres concernent l'ordre des documents :
**I1 :** la distribution des termes dans les documents pertinents est indépendante et leur distribution dans tous les documents est indépendante.
**I2 :** la distribution des termes dans les documents pertinents est indépendante et leur distribution dans les documents non pertinents est indépendante.
**O1 :** la probabilité de pertinence est basée seulement sur la présence des termes recherchés dans les documents.
**O2 :** la probabilité de pertinence est basée simultanément sur la présence des termes recherchés dans les documents et sur leur absence dans les documents.

L'estimation de la probabilité de pertinence $P(R|Q_k,D_j)$ du document $D_j$ , revient à calculer le **odds**[2] du document $D_j$ vis-à-vis de la requête $Q_k$ calculé :

---

[1] La fonction **sigmoïde** (dite aussi *courbe en S*) est définie par : Pour tout réel $x$ $f(x) = \frac{1}{1 + e^{-x}}$ Mais on la généralise à toute fonction dont l'expression est : $f(x) = \frac{1}{1 + e^{-\lambda x}}$ Le nom de « sigmoïde » lui vient de la forme de sa courbe en 'S'.

[2] **Le odds :** est une formule statistique souvent utilisée dans le modèle probabiliste et peut être défini par O(A) = P(A)/(1- P(A)).

*En se basant sur l'hypothèse d'indépendance*:

$$O(R|Q_k, D_j) = O(R|Q_k) \prod_{i=1}^{N} \frac{P(\omega_i|R, Q_k)}{P(\omega_i|\bar{R}, Q_k)}$$

*En se basant sur l'hypothèse de présence du terme dans le document*:

$$O(R|Q_k, D_j) = O(R|Q_k) \prod_{\omega_i=1} \frac{P(\omega_i = 1|R, Q_k)}{P(\omega_i = 1|\bar{R}, Q_k)} \prod_{\omega_i=0} \frac{P(\omega_i = 0|R, Q_k)}{P(\omega_i = 0|\bar{R}, Q_k)}$$

**Le score BM25 :** C'est l'une des plus célèbres instances de la BM25,

- Etant donnée une requête $Q = \{q_1, q_2, \dots q_n\}$ où $q_{i(i=\overline{1,n})}$ sont des mots clés de la requête Q, le score BM25 de pertinence d'un document D vis-à-vis de Q est calculé par :

$$score(D, Q) = \sum_{i=1}^{n} idf(q_i) \cdot \frac{tf_i \cdot (K_1 + 1)}{K_1 \cdot \left((1 - b) + b \cdot \frac{dl}{avdl}\right) + tf_i}$$

Avec : $idf = \log\left(\frac{N}{ni}\right)$ *ou* $\log\left(\frac{N-ni}{ni}\right)$ *ou* $\log\left(\frac{N}{ni}\right) + 1$ ...

Où : $N$ : Nombre total de documents dans le corpus (la collection).
$n_i$ : Nombre total de documents contenant le terme $i$.

**Partie 2 :**

**Entités de référence :** Les références de caractères et d'entités permettent d'inclure des informations dans des documents XML par référence plutôt qu'en saisissant des caractères directement dans le document. Cela peut être utile dans les cas où :

- Des caractères ne peuvent pas être introduits directement dans un document parce qu'ils seraient interprétés comme des balises ;
- Des caractères ne peuvent pas être introduits directement dans un document en raison de limitations de l'appareil de saisie ;
- Des caractères ne peuvent pas être transportés de manière fiable via un processeur limité à des caractères codés sur un octet ;
- Une chaîne de caractères ou un fragment de document apparaît de manière répétée et peut être abrégé ;
- Ou bien les caractères sont illégaux.

Pour représenter le contenu, XML offre un certain nombre de constructions syntaxiques commençant par un 'et' commerciale (&) et se terminant par un point-virgule (;).

Les références de caractères permettent d'insérer des caractères Unicode identifiés par un nombre qui pointe vers un point de code Unicode. Les points de code peuvent être identifiés en notation décimale ou hexadécimale.

*& #value;*

Syntaxe utilisée pour les références décimales.

*&# xvalue;*

Syntaxe utilisée pour les références hexadécimales.

Par exemple, pour insérer le symbole de l'euro (€), un caractère qui manque encore des nombreux claviers, nous pouvons insérer € ou € dans un document.

Le tableau suivant énumère les cinq entités intégrées correspondant aux caractères utilisés pour les balises XML :

| **Entité** | **Référence d'entité** | **Signification** |
|---|---|---|
| lt | < | < (inférieur à) |
| gt | > | > (supérieur à) |
| amp | & | & (éperluette) |
| apos | ' | ' (apostrophe ou guillemet simple) |
| quot | " | " (guillemet double) |

Lorsqu'un caractère risque de provoquer une mauvaise interprétation de la structure du document par l'analyseur XML, on utilise l'entité au lieu d'introduire simplement ce caractère. Les références d'entité ' et " sont très couramment utilisées dans les valeurs des attributs.

**Les technologies XML :**

- **XSL** (eXtensible Stylesheet Language) est le langage qui permet d'écrire des feuilles de style. Une feuille de style est constituée d'un ensemble de règles de transformations, s'appliquant chacune à un ou plusieurs nœuds de l'arbre et permettant de transformer ce nœud en un nouveau nœud de l'arbre résultat. Est un langage déclaratif utilisé pour transformer un document xml en un autre html ou texte ou encore xml.
- **Le langage XSLT** décrit des règles pour transformer un document XML. Ces règles s'appliquent chacune à un ou plusieurs nœuds de l'arbre et spécifient la transformation à effectuer sur un nœud pour le transformer en un nouveau nœud de l'arbre résultat.
- **Le langage XHTML** est destiné à devenir le successeur de HTML. C'est un sous-ensemble de XML, largement compatible avec HTML et qui remédie aux principaux défauts de ce dernier auquel il apporte la rigueur et la clarté. XHTML combine tous les éléments de HTML avec la syntaxe de XML.
- **SVG** est le langage Scalable Vector Graphics (graphiques vectoriels redimensionnables) est un langage permettant de décrire des graphiques à deux dimensions en XML.
- **MathML** il s'agit d'un langage de syntaxe XML permettant de décrire la structure d'un texte mathématiques et de ses formules et d'en permettre l'affichage par un navigateur Web, mais qui permet aussi de décrire un contenu mathématique effectif.

- **Les Espaces de Noms (Namespaces)** La spécification des espaces de noms XML se trouve à l'adresse : http://www.w3/org/TR/Rec-xml-names.
  Les espaces de nom sont spécifiés dans une recommandation du W3C. Ils permettent de distinguer de manière unique des éléments et des attributs portant le même nom lorsqu'ils proviennent d'applications XML différentes.
  Un espace de nom est déclaré au moyen d'un attribut **xmlns** dont la valeur est une adresse $URI_{19}$ (**U**niform **R**esource **I**ndentifier).
- **XPointer (XML Pointer)** est utilisé pour adresser des fragments de documents XML. Xpointer est construit sur le langage Xpath, de qui il a adopté les fonctionnalités et la syntaxe cependant il offre des multiples extensions à Xpath.
- **XLink** est une Recommandation du W3C depuis 2001, il permet la construction de liens XML semblables aux liens HTML mais avec davantage de puissance et d'extensibilité.
- **XQuery** est un langage intéressant avec quelques idées insolites, où tout est une expression qui renvoie une valeur. Un programme ou un script XQuery est juste une expression, avec une fonction facultative et d'autres définitions.
- **RDF :** (Ressource Description Framework) est un standard décrivant les ressources : personnes, lieux, documents… et est un framework contenant un modèle de données : langage et syntaxe. Il a été créé en 1999 en tant que norme sur XML pour les métadonnées. Les métadonnées peuvent être : l'auteur d'une page web, à quelle date une entrée de blog a été publiée, … etc.
- **XPath** est un sous-langage de XSLT, il est venu dans son comme un élément clé de XML. XPath 2.0 a émergé comme un langage robuste deux fois la taille de son prédécesseur, complexe et capable de triompher.

**Les travaux de recherche concernant la granularité d'information dans les documents XML :** ils se basent sur le fait qu'une demande d'information ne correspond pas forcément à la recherche d'un document tout entier mais peut être des éléments de granularité plus fine : Moffat, Sacks-Davis, Wilkinson et Zobel [Moffat & al, 1993] qui considèrent le renvoie des parties documentaires comme réponse à une requête en se basant sur le découpage physique du document (sa structure) ou sur la segmentation en pages physiques (limitées en nombre de caractères) et d'autres comme [Hearst, 1997] qui a proposé une méthode à but d'isoler dans un texte un ensemble de segments homogènes ou bien encore [Salton&al, 1996] qui considère le paragraphe comme étant l'unité d'information élémentaire et tout comme Hearst, il s'intéresse à la définition de l'ensemble des paragraphes qui traitent le même thème et emploie des techniques similaires. Il propose également de grouper les paragraphes entre eux, qu'ils soient contigus ou pas.

**Annexe II :**

**Autres travaux :**

1. **Modèles d'accès à l'information dans les réseaux bibliographiques :**

Le domaine de la RI propose de nombreuses tâches de recherche, dont celle de l'estimation de la pertinence d'information vis-à-vis d'une requête, appelée couramment la recherche d'information ad-hoc. Nous déclinons par la suite les deux grandes approches de recherche ad-hoc citées précédemment dans les réseaux d'informations : l'approche bibliométrique et l'approche basée sur la structure du réseau. L'approche bibliométrique ordonnance les entités bibliographiques en fonction de la qualité des publications scientifiques ou de l'importance des auteurs. Les indicateurs bibliométriques permettent ainsi de synthétiser les informations relatives à la production scientifique [Ibañez et al., 2011]. Ces mesures analysent les liens de citation entre entités [Alonso et al., 2010], [Zhang, 2009], [Egghe, 2006] et [Hirsch, 2005] ou intègrent un aspect temporel supplémentaire lié à la date de publication [Bergstrom et al., 2008], [Garfield, 2006] et [Walker et al., 2006]. Par exemple, Hirsch [Hirsch, 2005] propose d'estimer l'importance d'un auteur en considérant l'ensemble de ses documents publiés et le nombre de citations de ces derniers. Walker et al. [Walker et al., 2006] évaluent le nombre de citations qu'une publication scientifique peut recevoir à l'avenir compte tenu du nombre de citations reçues depuis sa date de publication.

L'approche basée sur la structure du réseau distingue deux principales catégories de modèles consistant, pour la première, à ordonnancer un seul type d'entités en tenant compte de leurs interactions avec l'ensemble des entités [Jabeur et al., 2010], [Kirsch et al., 2006], [Liu et al., 2005] ou, pour la seconde, proposant un ordonnancement d'entités hétérogènes de façon conjointe considérant la structure du réseau [Yan et al., 2010], [Yang et al., 2010], [Tang et al., 2008],[ Zhang et al., 2008] et [Zhou et al., 2007].

D'autres travaux [Yang et al., 2010], [Tang et al., 2008]et [Zhang et al., 2008] intègrent dans leur fonction d'ordonnancement un aspect thématique supplémentaire. Zhang et al. [Zhang et al., 2008] proposent de recommander les entités en combinant un score thématique qui, grâce au modèle de langue, estime la similarité entre la requête et l'entité, et un score d'autorité de l'entité par rapport à l'ensemble du réseau hétérogène grâce à une extension de l'algorithme PageRank. Yang et al. [Yang et al., 2010] proposent d'étendre le modèle Topical PageRank, introduit dans [Nie et al., 2006], afin de mettre en valeur les entités autoritaires représentées par une distribution vectorielle de thèmes. Cet algorithme considère trois comportements pour un utilisateur : "Follow-Stay" lorsqu'un utilisateur navigue dans le réseau en restant dans le même thème, "Follow-Jump" lorsqu'il change de thème et "Jump-Jump" lorsqu'il accède de façon aléatoire à un thème. Le modèle "Author-Topic-Conference" (ACT), proposé dans [Tang et al., 2008], est utilisé par le moteur de recherche Arnetminer 4. Ce modèle représente les entités grâce à une distribution thématique inférée du modèle LDA, "Latent Dirichlet Allocation" [Wei et al., 2006], et ordonnance ensuite les entités document, auteur et conférence en appliquant un algorithme proche de PageRank.

## 2. Affinités lexicales

L'hypothèse d'indépendance entre mots, faite par le modèle uni-gramme, ainsi qu'une grande partie des approches à la RI, n'est pas toujours justifiée. Les modèles bi-gramme (et à fortiori les modèles d'ordre supérieur) tentent en effet de rendre compte des dépendances entre termes; tout en supposant que l'ordre des mots est important. Tandis que cette dernière hypothèse semble raisonnable pour des applications comme la reconnaissance de parole, elle ne s'applique pas nécessairement à la RI. Par exemple, pour une requête "apartment rentals", un document contenant les termes "rent an apartment" ne doit pas être à priori moins bien classé qu'un autre document contant les termes "apartments for rent". Notre réponse à ce problème consiste à baser notre modélisation sur une unité lexicale n'imposant aucune restriction sur l'ordre de ses mots et peu de contrainte sur leur adjacence: l'affinités lexicale. Selon Martin et al. (1983), 98% des relations lexicales dans un texte mettent en jeu des mots dans une fenêtre de 5 mots. Nous adoptons cette propriété pour identifier les unités (paires de mots) sur lesquelles bâtir nos modèles de langue. Par ailleurs, Maarek et al. (1991) introduisent le concept de pouvoir de résolution d'une paire de mots. À l'instar des facteurs tf et idf utilisés dans le modèle vectoriel, l'idée principale derrière le pouvoir de résolution est que les paires de mots qui caractérisent le mieux un document sont celles qui ont en même temps une fréquence élevée dans le document et une fréquence relativement basse dans la collection. Les auteurs suggèrent de calculer le pouvoir de résolution d'une paire < u, v > pour un document d, cd(< u, v >) est la fréquence de la paire dans le document d. Le terme logarith-1. Une liste de 571 mots anglais fournie avec le système S M A RT a été utilisée. Carmen Alvarez, Philippe Langlais, et Jian-Yun Nie mique dans cette équation peut être vu comme une approximation de la quantité d'information véhiculée par la paire, comparable au facteur idf.

$$\rho d(< u, v >) = -cd(< u, v >) \times log(pcorpus(u) \times pcorpus(v))$$

**Annexe III :**

Les probabilités associées à chacun des éléments du document $Doc_2$ :

| **$Doc_2$** | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $e_j$ | $e_1$ | $e_2$ | $e_3$ | $e_4$ | $e_5$ | $e_6$ | $e_7$ | $e_8$ | $e_9$ | $e_{10}$ | $e_{11}$ | $e_{12}$ |
| $\|e_j\|$ | 130 | 32 | 98 | 2 | 30 | 98 | 2 | 30 | 51 | 47 | 51 | 47 |
| $P(e_j)$ par (3.1) | 1 | 0.246 | 0.754 | 0.015 | 0.231 | 0.754 | 0.015 | 0.231 | 0.392 | 0.362 | 0.392 | 0.362 |
| $P(e_j)$ par (3.2) | 0.793 | 0.195 | 0.598 | 0.012 | 0.183 | 0.598 | 0.012 | 0.183 | 0.311 | 0.287 | 0.311 | 0.287 |

Les probabilités associées à chacun des éléments du document $Doc_3$ :

| **Doc3** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $e_j$ | $e_1$ | $e_2$ | $e_3$ | $e_4$ | $e_5$ | $e_6$ | $e_7$ | $e_8$ | $e_9$ |
| $\|e_j\|$ | 109 | 29 | 80 | 2 | 27 | 34 | 46 | 2 | 27 |
| $P(e_j)$ par (3.1) | 1 | 0.266 | 0.734 | 0.018 | 0.248 | 0.312 | 0.422 | 0.018 | 0.248 |
| $P(e_j)$ par (3.2) | 0.779 | 0.207 | 0.571 | 0.014 | 0.193 | 0.243 | 0.329 | 0.014 | 0.193 |
| (suite) | | | | | | | | | |
| $e_j$ | $e_{10}$ | $e_{11}$ | $e_{12}$ | $e_{13}$ | $e_{14}$ | $e_{15}$ | $e_{16}$ | $e_{17}$ | $e_{18}$ |
| $\|e_j\|$ | 34 | 7 | 14 | 15 | 3 | 7 | 14 | 15 | 3 |
| $P(e_j)$ par (3.1) | 0.312 | 0.064 | 0.128 | 0.138 | 0.028 | 0.064 | 0.128 | 0.138 | 0.028 |
| $P(e_j)$ par (3.2) | 0.243 | 0.05 | 0.1 | 0.107 | 0.021 | 0.05 | 0.1 | 0.107 | 0.021 |

Les probabilités associées à chacun des éléments du document $Doc_4$ :

| **Doc4** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $e_j$ | $e_1$ | $e_2$ | $e_3$ | $e_4$ | $e_5$ | $e_6$ | $e_7$ | $e_8$ |
| $\|e_j\|$ | 46 | 3 | 43 | 3 | 25 | 18 | 25 | 18 |
| $P(e_j)$ par (3.1) | 1 | 0.065 | 0.935 | 0.065 | 0.543 | 0.391 | 0.543 | 0.391 |
| $P(e_j)$ par (3.2) | 0.902 | 0.059 | 0.843 | 0.065 | 0.49 | 0.353 | 0.49 | 0.353 |

Les probabilités associées à chacun des éléments du document $Doc_5$ :

| **Doc5** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $e_j$ | $e_1$ | $e_2$ | $e_3$ | $e_4$ | $e_5$ | $e_6$ | $e_7$ | $e_8$ |
| $\|e_j\|$ | 271 | 48 | 154 | 3 | 45 | 56 | 130 | 3 |
| $P(e_j)$ par (3.1) | 1 | 0.238 | 0.762 | 0.015 | 0.223 | 0.278 | 0.643 | 0.015 |
| $P(e_j)$ par (3.2) | 0.534 | 0.126 | 0.407 | 0.01 | 0.12 | 0.15 | 0.344 | 0.01 |

| (suite) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $e_j$ | $e_9$ | $e_{10}$ | $e_{11}$ | $e_{12}$ | $e_{13}$ | $e_{14}$ | $e_{15}$ | $e_{16}$ |
| $\|e_j\|$ | 45 | 24 | 19 | 13 | 70 | 60 | 37 | 24 |
| $P(e_j)$ par (3.1) | 0.223 | 0.119 | 0.094 | 0.064 | 0.346 | 0.297 | 0.183 | 0.119 |
| $P(e_j)$ par (3.2) | 0.12 | 0.063 | 0.05 | 0.034 | 0.185 | 0.159 | 0.407 | 0.008 |
| (suite) | | | | | | | | |
| $e_j$ | $e_{17}$ | $e_{18}$ | $e_{19}$ | $e_{20}$ | $e_{21}$ | $e_{22}$ | $e_{23}$ | $e_{24}$ |
| $\|e_j\|$ | 19 | 13 | 26 | 44 | 60 | 37 | 26 | 44 |
| $P(e_j)$ par (3.1) | 0.094 | 0.064 | 0.129 | 0.218 | 0.297 | 0.183 | 0.129 | 0.218 |
| $P(e_j)$ par (3.2) | 0.05 | 0.034 | 0.068 | 0.116 | 0.159 | 0.407 | 0.068 | 0.116 |

Les probabilités d'importance de chacun des termes de la requête dans chacun des éléments des documents de la collection xml :

| termes / ($Doc_i$ , $e_j$) | | *« influ »* | *« réseau »* | *« socia »* |
|---|---|---|---|---|
| **$Doc_1$** | $e_1$ | 0.0131 | 0.0131 | 0.0131 |
| | $e_2$ | 0.0131 | 0.0131 | 0.0131 |
| | $e_3$ | 0.0131 | 0.0131 | 0.0131 |
| | $e_4$ | 0.0131 | 0.0131 | 0.0131 |
| | $e_5$ | 0.0131 | 0.0131 | 0.0131 |
| | $e_6$ | 0.0131 | 0.0131 | 0.0131 |
| | $e_7$ | 0.0135 | 0.0131 | 0.0131 |
| | $e_8$ | 0.0131 | 0.0131 | 0.0131 |
| | $e_9$ | 0.0131 | 0.0131 | 0.0131 |
| | $e_{10}$ | 0.0131 | 0.0131 | 0.0131 |
| | $e_{11}$ | 0.0135 | 0.0131 | 0.0131 |
| **$Doc_2$** | $e_1$ | 0.0150 | 0.0690 | 0.0620 |
| | $e_2$ | 0.0150 | 0.1250 | 0.0930 |
| | $e_3$ | 0.0150 | 0.0510 | 0.0510 |
| | $e_4$ | 0.0150 | 0.5000 | 0.5000 |
| | $e_5$ | 0.0150 | 0.1000 | 0.0667 |
| | $e_6$ | 0.0150 | 0.0510 | 0.0510 |
| | $e_7$ | 0.0150 | 0.5000 | 0.5000 |

| | | | | |
|---|---|---|---|---|
| | $e_8$ | 0.0150 | 0.1000 | 0.0660 |
| | $e_9$ | 0.0150 | 0.0784 | 0.0784 |
| | $e_{10}$ | 0.0213 | 0.0213 | 0.0213 |
| | $e_{11}$ | 0.0150 | 0.0784 | 0.0784 |
| | $e_{12}$ | 0.0213 | 0.0213 | 0.0213 |
| **$Doc_3$** | $e_1$ | 0.0101 | 0.0090 | 0.0090 |
| | $e_2$ | 0.1720 | 0.0090 | 0.0090 |
| | $e_3$ | 0.0750 | 0.0090 | 0.0090 |
| | $e_4$ | 0.5000 | 0.0090 | 0.0090 |
| | $e_5$ | 0.1480 | 0.0090 | 0.0090 |
| | $e_6$ | 0.0589 | 0.0090 | 0.0090 |
| | $e_7$ | 0.0870 | 0.0090 | 0.0090 |
| | $e_8$ | 0.5000 | 0.0090 | 0.0090 |
| | $e_9$ | 0.1480 | 0.0090 | 0.0090 |
| | $e_{10}$ | 0.0589 | 0.0090 | 0.0090 |
| | $e_{11}$ | 0.2860 | 0.0090 | 0.0090 |
| | $e_{12}$ | 0.0710 | 0.0090 | 0.0090 |
| | $e_{13}$ | 0.0667 | 0.0090 | 0.0090 |
| | $e_{14}$ | 0.1100 | 0.0090 | 0.0090 |
| | $e_{15}$ | 0.2860 | 0.0090 | 0.0090 |
| | $e_{16}$ | 0.0710 | 0.0090 | 0.0090 |
| | $e_{17}$ | 0.0667 | 0.0090 | 0.0090 |
| | $e_{18}$ | 0.1100 | 0.0090 | 0.0090 |
| **$Doc_4$** | $e_1$ | 0.1090 | 0.0650 | 0.0220 |
| | $e_2$ | 0.3330 | 0.3330 | 0.0220 |
| | $e_3$ | 0.0930 | 0.0470 | 0.0220 |
| | $e_4$ | 0.3330 | 0.3330 | 0.0220 |
| | $e_5$ | 0.0800 | 0.0800 | 0.0220 |
| | $e_6$ | 0.1110 | 0.0870 | 0.0220 |
| | $e_7$ | 0.0800 | 0.0800 | 0.0220 |
| | $e_8$ | 0.1110 | 0.0870 | 0.0220 |
| **$Doc_5$** | $e_1$ | 0.0500 | 0.0740 | 0.0440 |
| | $e_2$ | 0.0200 | 0.1000 | 0.0620 |
| | $e_3$ | 0.0400 | 0.0450 | 0.0400 |
| | $e_4$ | 0.3330 | 0.3330 | 0.3330 |
| | $e_5$ | 0.0540 | 0.0890 | 0.0440 |
| | $e_6$ | 0.0190 | 0.0360 | 0.0180 |
| | $e_7$ | 0.0470 | 0.0470 | 0.0480 |
| | $e_8$ | 0.3330 | 0.3330 | 0.3330 |
| | $e_9$ | 0.0540 | 0.0890 | 0.0440 |
| | $e_{10}$ | 0.0420 | 0.0830 | 0.0690 |
| | $e_{11}$ | 0.0540 | 0.0790 | 0.0520 |
| | $e_{12}$ | 0.0540 | 0.0790 | 0.0690 |
| | $e_{13}$ | 0.0430 | 0.0280 | 0.0140 |
| | $e_{14}$ | 0.0330 | 0.0660 | 0.0660 |
| | $e_{15}$ | 0.0810 | 0.0540 | 0.0540 |

| | | | | |
|---|---|---|---|---|
| | $e_{16}$ | 0.0420 | 0.0830 | 0.0690 |
| | $e_{17}$ | 0.0540 | 0.0790 | 0.0520 |
| | $e_{18}$ | 0.0540 | 0.0790 | 0.0690 |
| | $e_{19}$ | 0.0380 | 0.0380 | 0.0380 |
| | $e_{20}$ | 0.0450 | 0.0230 | 0.0230 |
| | $e_{21}$ | 0.0330 | 0.0660 | 0.0660 |
| | $e_{22}$ | 0.0810 | 0.0540 | 0.0540 |
| | $e_{23}$ | 0.0380 | 0.0380 | 0.0380 |
| | $e_{24}$ | 0.0450 | 0.0230 | 0.0230 |

Les probabilités d'importance de chacun des termes de la requête dans chacun des éléments des documents de la collection xml :

| **$Doc_i$** | $e_j$ | $P_I("influ"/e_j)$ | $P_I("réseau"/e_j)$ | $P_I("socia"/e_j)$ |
|---|---|---|---|---|
| **$Doc_1$** | $e_1$ | 0.076 | 0.037 | 0.051 |
| | $e_2$ | 0.076 | 0.037 | 0.051 |
| | $e_3$ | 0.076 | 0.037 | 0.051 |
| | $e_4$ | 0.076 | 0.037 | 0.051 |
| | $e_5$ | 0.076 | 0.037 | 0.051 |
| | $e_6$ | 0.076 | 0.037 | 0.051 |
| | $e_7$ | 0.078 | 0.037 | 0.051 |
| | $e_8$ | 0.078 | 0.037 | 0.051 |
| | $e_9$ | 0.078 | 0.037 | 0.051 |
| | $e_{10}$ | 0.078 | 0.037 | 0.051 |
| | $e_{11}$ | 0.078 | 0.037 | 0.051 |
| **$Doc_2$** | $e_1$ | 0.040 | 0.050 | 0.068 |
| | $e_2$ | 0.040 | 0.080 | 0.086 |
| | $e_3$ | 0.040 | 0.040 | 0.040 |
| | $e_4$ | 0.040 | 0.260 | 0.260 |
| | $e_5$ | 0.040 | 0.070 | 0.072 |
| | $e_6$ | 0.040 | 0.040 | 0.040 |
| | $e_7$ | 0.040 | 0.260 | 0.260 |
| | $e_8$ | 0.040 | 0.070 | 0.070 |
| | $e_9$ | 0.040 | 0.060 | 0.060 |
| | $e_{10}$ | 0.037 | 0.027 | 0.027 |
| | $e_{11}$ | 0.040 | 0.060 | 0.060 |
| | $e_{12}$ | 0.037 | 0.027 | 0.027 |
| | $e_1$ | 0.020 | 0.025 | 0.020 |
| | $e_2$ | 0.110 | 0.025 | 0.020 |
| | $e_3$ | 0.052 | 0.025 | 0.020 |

| | | | | |
|---|---|---|---|---|
| **Doc3** | $e_4$ | 0.260 | 0.025 | 0.020 |
| | $e_5$ | 0.090 | 0.025 | 0.020 |
| | $e_6$ | 0.044 | 0.025 | 0.020 |
| | $e_7$ | 0.060 | 0.025 | 0.020 |
| | $e_8$ | 0.260 | 0.025 | 0.020 |
| | $e_9$ | 0.090 | 0.025 | 0.020 |
| | $e_{10}$ | 0.044 | 0.025 | 0.020 |
| | $e_{11}$ | 0.160 | 0.025 | 0.020 |
| | $e_{12}$ | 0.058 | 0.025 | 0.020 |
| | $e_{13}$ | 0.048 | 0.025 | 0.020 |
| | $e_{14}$ | 0.070 | 0.025 | 0.020 |
| | $e_{15}$ | 0.160 | 0.025 | 0.020 |
| | $e_{16}$ | 0.058 | 0.025 | 0.020 |
| | $e_{17}$ | 0.048 | 0.025 | 0.020 |
| | $e_{18}$ | 0.070 | 0.025 | 0.020 |
| **Doc4** | $e_1$ | 0.090 | 0.053 | 0.050 |
| | $e_2$ | 0.500 | 0.230 | 0.050 |
| | $e_3$ | 0.080 | 0.085 | 0.050 |
| | $e_4$ | 0.500 | 0.230 | 0.050 |
| | $e_5$ | 0.070 | 0.060 | 0.050 |
| | $e_6$ | 0.100 | 0.064 | 0.050 |
| | $e_7$ | 0.070 | 0.060 | 0.050 |
| | $e_8$ | 0.100 | 0.064 | 0.050 |
| **Doc5** | $e_1$ | 0.010 | 0.048 | 0.032 |
| | $e_2$ | 0.020 | 0.060 | 0.040 |
| | $e_3$ | 0.030 | 0.033 | 0.180 |
| | $e_4$ | 0.180 | 0.180 | 0.200 |
| | $e_5$ | 0.037 | 0.060 | 0.020 |
| | $e_6$ | 0.020 | 0.030 | 0.024 |
| | $e_7$ | 0.033 | 0.034 | 0.032 |
| | $e_8$ | 0.180 | 0.034 | 0.200 |
| | $e_9$ | 0.037 | 0.180 | 0.032 |
| | $e_{10}$ | 0.030 | 0.060 | 0.045 |
| | $e_{11}$ | 0.037 | 0.052 | 0.036 |
| | $e_{12}$ | 0.037 | 0.050 | 0.040 |
| | $e_{13}$ | 0.031 | 0.050 | 0.020 |

| | | | | |
|---|---|---|---|---|
| | $e_{14}$ | 0.026 | 0.025 | 0.043 |
| | $e_{15}$ | 0.050 | 0.044 | 0.200 |
| | $e_{16}$ | 0.030 | 0.060 | 0.040 |
| | $e_{17}$ | 0.037 | 0.050 | 0.045 |
| | $e_{18}$ | 0.037 | 0.050 | 0.045 |
| | $e_{19}$ | 0.030 | 0.040 | 0.030 |
| | $e_{20}$ | 0.031 | 0.052 | 0.021 |
| | $e_{21}$ | 0.026 | 0.052 | 0.043 |
| | $e_{22}$ | 0.050 | 0.044 | 0.200 |
| | $e_{23}$ | 0.030 | 0.040 | 0.030 |
| | $e_{24}$ | 0.031 | 0.052 | 0.021 |

Les scores de similarité entre les éléments du document $Doc_2$ et la requête *query* :

| **$Doc_2$** | | | |
|---|---|---|---|
| $e_j$ | ***score*** | $e_j$ | ***score*** |
| $e_1$ | 0.1300 | $e_7$ | 0.0320 |
| $e_2$ | 0.0034 | $e_8$ | 0.0350 |
| $e_3$ | 0.0200 | $e_9$ | 0.0450 |
| $e_4$ | 0.0400 | $e_{10}$ | 0.0080 |
| $e_5$ | 0.0410 | $e_{11}$ | 0.0450 |
| $e_6$ | 0.0380 | $e_{12}$ | 0.0100 |

*L*es scores de similarité entre les éléments du document $Doc_3$ et la requête *query* :

| **$Doc_3$** | | | |
|---|---|---|---|
| $e_j$ | ***score*** | $e_j$ | ***score*** |
| $e_1$ | 0.0100 | $e_{10}$ | 0.0070 |
| $e_2$ | 0.0015 | $e_{11}$ | 0.0020 |
| $e_3$ | 0.0200 | $e_{12}$ | 0.0037 |
| $e_4$ | 0.0023 | $e_{13}$ | 0.0033 |

| | | | |
|---|---|---|---|
| $e_5$ | 0.0460 | $e_{14}$ | 0.0100 |
| $e_6$ | 0.0068 | $e_{15}$ | 0.0050 |
| $e_7$ | 0.0130 | $e_{16}$ | 0.0370 |
| $e_8$ | 0.0023 | $e_{17}$ | 0.0040 |
| $e_9$ | 0.0110 | $e_{18}$ | 0.0010 |

Les scores de similarité entre les éléments du document $Doc_4$ et la requête *query* :

| **$Doc_4$** | | | |
|---|---|---|---|
| $e_j$ | ***score*** | $e_j$ | ***score*** |
| $e_1$ | 0.0240 | $e_5$ | 0.0200 |
| $e_2$ | 0.0100 | $e_6$ | 0.0130 |
| $e_3$ | 0.0320 | $e_7$ | 0.0200 |
| $e_4$ | 0.1000 | $e_8$ | 0.0130 |

Les scores de similarité entre les éléments du document $Doc_5$ et la requête *query* :

| **$Doc_5$** | | | |
|---|---|---|---|
| $e_j$ | ***score*** | $e_j$ | ***score*** |
| $e_1$ | 0.0150 | $e_{13}$ | 0.0100 |
| $e_2$ | 0.0120 | $e_{14}$ | 0.0800 |
| $e_3$ | 0.1300 | $e_{15}$ | 0.1000 |
| $e_4$ | 0.1000 | $e_{16}$ | 0.0800 |
| $e_5$ | 0.0100 | $e_{17}$ | 0.0700 |
| $e_6$ | 0.0400 | $e_{18}$ | 0.0500 |
| $e_7$ | 0.0230 | $e_{19}$ | 0.0460 |
| $e_8$ | 0.0200 | $e_{20}$ | 0.0730 |
| $e_9$ | 0.0210 | $e_{21}$ | 0.1000 |

| | | | |
|---|---|---|---|
| $e_{10}$ | 0.0200 | $e_{22}$ | 0.0800 |
| $e_{11}$ | 0.0400 | $e_{23}$ | 0.0500 |
| $e_{12}$ | 0.0500 | $e_{24}$ | 0.0730 |

**Annexe IV :**

**Le système de recherche XFIRM :** [Sauvagnat, 2006]

XFIRM (XML Flexible Information Retrieval Model) est l'un des modèles de RIS qui présente la possibilité d'implémentation de nombreux modèles de RI, c'est un système orienté pertinence. Tel présenté dans le chapitre IV, XFIRM est constitué principalement d'une BDD (Base De Données) MySQL qui consiste en l'élément centrale de tout le système et contenant ses index, autour de la BDD sont placés des éléments ayant des accès à celle-ci via un pilote JAVA, ces élément sont des modules qui construisent la partie dynamique du système et sont : *Le module d'indexation, le module d'interrogation* et *le module de traitement des requêtes* (orientées contenu et orientées contenu et structure). Pour répondre aux exigences utilisateur en structure et contenu en lui renvoyant une liste triée d'éléments répondants à sa requête. Comme dans notre travail nous avons utilisé le module d'indexation uniquement, et ce dernier est relié à la BDD du système que nous avons exploité énormément, nous avons opté à présenter la structure de cette BDD :

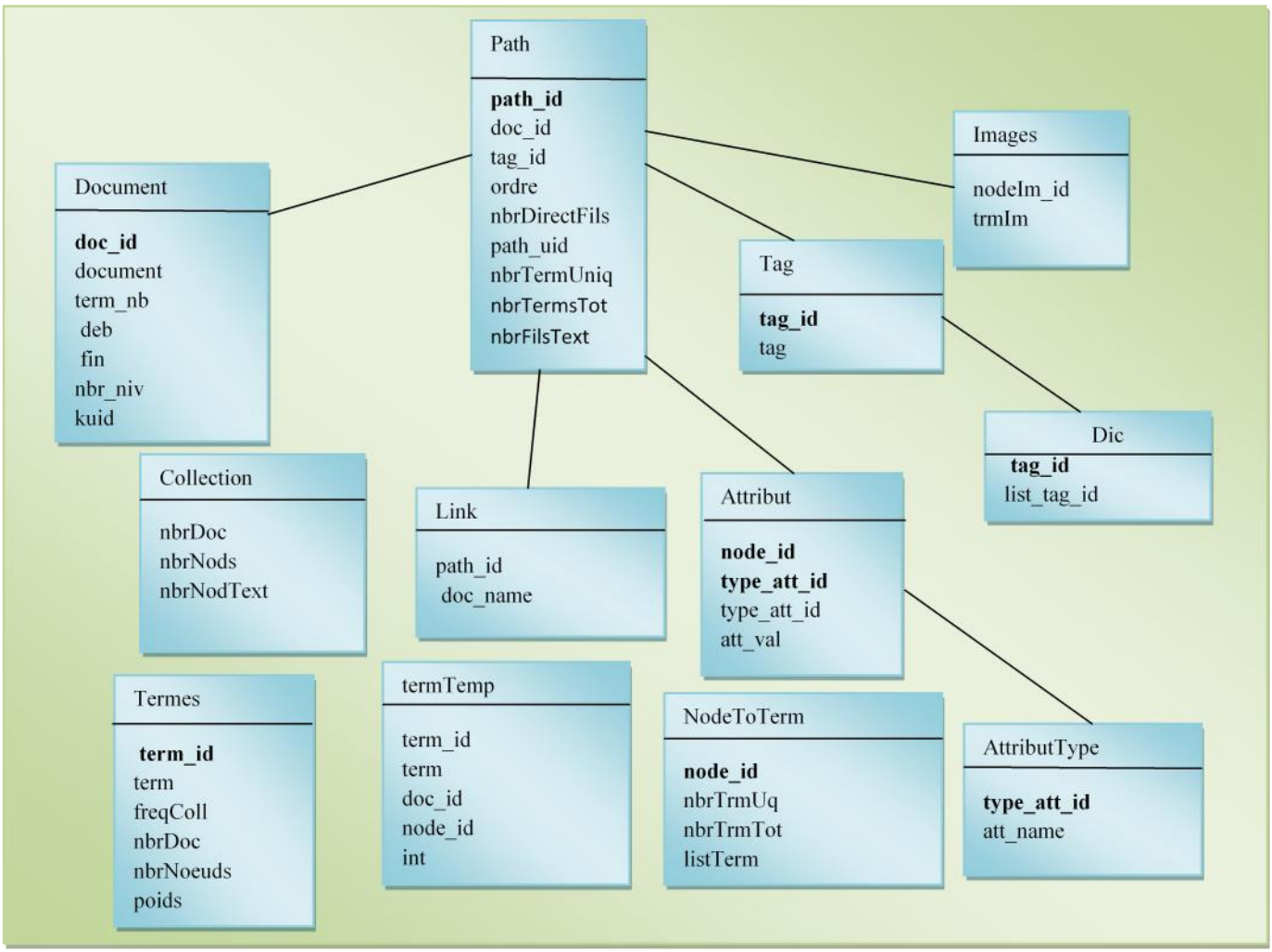



**Figure :** schéma de la BDD XFIRM contenant les index

***Résumé :***

L'objectif principal d'un SRI, dans le cadre de la RI classique, est de retrouver le document répondant à une requête utilisateur donnée. Dans la cadre de la RI dans les documents xml, un SRI doit retourner l'élément correspondant à une requête utilisateur, en se basant sur une collection de documents XML à contenu textuel et structurel.

Quelque soit le domaine de la RI, cette dernière se base sur l'un des modèles d'une taxonomie riche, dans le but d'établir un appariement document/requête dans le cas de la RI classique, ou élément/requête dans le cas de la RI dans les documents xml.

Dans ce travail, nous nous sommes intéressés au modèle probabiliste : *le modèle de langue*, dans le cadre de la RI dans les documents xml. Les premières applications de ce modèle été dans la linguistique informatique, puis des auteurs ont prouvé son applicabilité au domaine de la RI, tout en présentant des approches appropriées pour le modèle de langue dans la RI classique. Dans le chapitre II, nous avons constaté les paramètres pris en compte pour adapter ou proposer des approches de recherche par le modèle de langue, à la nouvelle granularité d'information xml.

Notre contribution est d'adapter une approche de recherche par modèle de langue, se basant sur la taille d'un document, dans le cadre de la RI classique [Achmoukh, 2006] à la nouvelle granularité d'information. Nous avons montré les paramètres pris en compte dans l'adaptation, pour en suite établir les nouvelles probabilités de similarité que nous avons proposé. Notre approche est appliquée à un exemple de recherche dans les documents xml, dans le chapitre III, puis implémentée, testée et évaluée dans le chapitre IV, nous avons aussi comparé les résultats obtenus à des résultats obtenus par le système XFIRM [Sauvagnat, 2005] et nous avons montré les améliorations effectuées par rapport à ce système.

***Mots Clés :*** *recherche d'information, XML, modèle de langue, taille de document, taille de l'élément*

***Abstract:***

The main objective of an IRS within the framework of classical IR is to find the document in response to a given user query. As part of IR in xml documents, an IRS must return the item corresponding to a user query, based on a collection of documents with textual information and structural content.

Whatever the field of IR, this latter is based on a model of a rich taxonomy in order to establish a matching: document / query in the case of traditional IR or element / query in the case of IR in xml documents.

In this work, we studied the probabilistic model: the language model, within the framework of IR in xml documents, its first applications was in computational linguistics and authors have proven its applicability to the field of IR, while having appropriate approaches to language in the model IR classic. In Chapter II, we found the parameters taken in consideration to adapt or propose research approaches by the language model to the new granularity XML of information.

Our contribution is to adapt a research approach by language model, based on the size of a document within the framework of classical IR [Achmoukh, 2006] to the new granularity of information. We showed the parameters taken in consideration while adapting to the new granularity, then established probabilities of similarity that we have proposed. Our approach is applied to an example of research into XML documents in Chapter III and implemented, tested and is evaluated in chapter IV, we also compared our results to those obtained by the system XFIRM [Sauvagnat 2005 ], and we showed the improvements made in relation to this system.

***Key words:*** *information retrieval, XML, language model, document size, element size.*