Intégration de règles d'association pour améliorer la recherche d'informations XML

Cheikh Talibouya Diop — Moussa Lo — Fatou Kamara Sangaré

Equipe BDISIC
Laboratoire d'Analyse Numérique et d'Informatique
UFR de Sciences Appliquées et de Technologie
Université Gaston Berger
BP 234 Saint-Louis (Sénégal)

RÉSUMÉ. *La reformulation de requêtes constitue un moyen d'améliorer la recherche d'informations, en particulier lorsque cela concerne des documents XML. Les approches existantes se basent sur une connaissance du domaine (thésaurus, ontologie) pour étendre la requête initiale. Nous proposons une approche de reformulation automatique basée sur une technique de datamining. Nous intégrons les règles d'association dans le système de recherche d'informations que nous avons développé pour les documents XML. Cela présente l'avantage de ne pas dépendre d'une connaissance du domaine préétablie qui n'est pas toujours disponible mais plutôt de s'appuyer sur une connaissance dynamique et cachée.*

ABSTRACT. *Query expansion allows improving information retrieval, in particular for XML documents. To expand the initial query, we propose an automatic query expansion based on a datamining technique. We integrate association rules in the information retrieval system we developed for XML documents. The advantage of this approach is to rely on a dynamic and hidden knowledge, contrary to an approach using a predefined knowledge that is not always available.*

MOTS-CLÉS : *systèmes de recherche d'informations, règles d'association, reformulation, XML.*

KEYWORDS: *information retrieval systems, association rules, query expansion, XML.*

CORIA 2007, pp. 137–148
Saint-Étienne, 28–30 mars 2007

1. Introduction

La Recherche d'Informations (RI) peut se définir comme l'ensemble des opérations effectuées pour retrouver une information répondant à une requête. Elle est une démarche faite par un utilisateur, pour, à partir de ses besoins informationnels, obtenir à l'aide d'un Système de Recherche d'Informations (SRI) de l'information pertinente.

Les SRI ont pour fonctions essentielles la gestion ainsi que l'accès à des bases documentaires. Ces systèmes permettent non seulement de représenter, de stocker et d'organiser les documents mais également de restituer un ensemble de documents satisfaisant les besoins d'un utilisateur exprimés en fonction d'une requête [BAE 99, IHA 04]. Ainsi dans un SRI, on distingue deux phases principales : le mode de représentation du contenu sémantique des besoins informationnels de l'utilisateur (requête) et des documents (indexation), et le mode de comparaison qui évalue la pertinence des documents par rapport à la requête.

Un SRI dispose d'un modèle de recherche qui intègre les phases citées ci-dessus afin de restituer des documents pertinents avec le minimum de bruit et le maximum de précision. Le bruit et la précision sont deux critères d'évaluation d'un SRI [SAL 83, BAE 99]. Ils notifient respectivement l'ensemble des documents non pertinents restitués et l'ensemble des documents pertinents non restitués. Généralement, un modèle de recherche dérive des trois modèles de recherche classiques qui sont le modèle booléen, le modèle vectoriel et le modèle probabiliste [SAL 83, BAE 99, IHA 04]. Dans notre contexte, nous nous focalisons sur le modèle vectoriel. Introduit en 1970 par Gérard Salton [SAL 71], le modèle vectoriel est le modèle le plus populaire grâce à sa capacité d'ordonner les documents retrouvés, sa robustesse et sa bonne performance dans les tests. Il est basé sur l'espace vectoriel défini par les termes d'index. Dans cet espace, chaque représentation d'un document en fonction des termes d'index est transformée en un vecteur dont les éléments sont des réels appelés poids exprimant l'importance d'un terme dans un document. Le mécanisme de ce modèle permet, à partir des vecteurs obtenus, d'utiliser une fonction de correspondance capable de déterminer le degré de similarité qui existe entre une requête et les documents du corpus.

Nous avons proposé en [SMA 02, LO 05] une méthode de recherche d'informations pertinentes dans une base de documents XML. Cette méthode s'appuie sur une indexation au niveau des feuilles du graphe XML et étend le modèle des fichiers inverses utilisé dans le modèle vectoriel au moyen des expressions de chemin XML. Elle tient compte de la structure d'un document XML et permet de retrouver des parties de documents et de les classer selon leur degré de pertinence.

On note, cependant, dans ce système un silence qui représente la non restitution de documents pertinents. La reformulation de requêtes constitue un moyen de pallier à cette limite. Les solutions de reformulation exploitent généralement une

connaissance du domaine représentée sous la forme d'un thésaurus ou d'une ontologie. Cette connaissance n'est cependant pas toujours disponible.

Nous proposons une approche de reformulation automatique qui s'appuie sur une connaissance non connue à priori mais issue des requêtes antérieures des utilisateurs. Cette connaissance est obtenue au moyen de techniques d'extraction de connaissances et est dynamique car dépend du comportement des utilisateurs.

L'extraction de connaissances permet la découverte de connaissances à partir de gros volumes de données. Elle se donne pour but de retrouver de la connaissance cachée, souvent très utile, dans les données. Cette découverte s'effectue à travers des techniques dont la découverte de règles d'association, l'une des plus utilisées.

Dans ce papier, nous montrons comment nous exploitons les règles d'association pour effectuer une reformulation automatique dans le but de diminuer le silence du SRI XML.

La section 2 est un état de l'art sur la reformulation de requêtes. Dans la partie 3, nous présentons les règles d'association.

Dans la section 4, nous montrons d'abord comment nous intégrons une base de règles d'association dans le SRI pour les documents XML. Ensuite, nous proposons une méthode de reformulation automatique de requêtes qui exploite les règles d'association.

2. Etat de l'art

Le modèle vectoriel présente des limites, en particulier sur les hypothèses faites sur les termes d'index pour pouvoir les associer aux dimensions de l'espace de représentation : l'hypothèse d'indépendance des termes d'index et l'hypothèse qu'un terme est une unité de sens des documents. Or les termes d'index sont en général ambigus et polysémiques, ce qui peut influencer la performance de la recherche [SAL 83]. Plusieurs études ont été menées pour essayer d'intégrer des dépendances entre les termes [SAL 83, BIL 00, SCO 90, HAR 92, WON 85]. Parmi ces modèles nous pouvons citer le modèle Vectoriel Généralisé (SVG) [DUM 94] et le modèle LSI (Latent Semantic Indexing) [SCO 90]. Le but de LSI et SVG est de transformer une représentation par des mots-clés en une autre représentation. Les documents et les requêtes sémantiquement similaires seront plus proches avec la représentation transformée qu'avec les mots-clés. Néanmoins la mise œuvre de ces modèles est plus lente que celle du modèle vectoriel.

Au lieu de définir de nouveaux espaces de représentations, nous conservons le modèle vectoriel en nous focalisant sur la reformulation de la requête qui constitue un moyen d'améliorer la pertinence d'un SRI. La reformulation peut être manuelle ou automatique. Méthode la plus ancienne, la reformulation manuelle est communément appelée la réinjection de la requête (ou relevance feedback) [BAE 99, SCO 90]. Son objectif est de réaliser une première recherche à l'aide des seuls termes de la représentation de la requête initiale. L'utilisateur peut alors indiquer

quels sont les meilleurs documents issus de cette première recherche. Le système utilise cette information pour affiner la recherche ; soit en modifiant les termes de la requête dans sa représentation vectorielle, soit en ajoutant ou en supprimant des termes. On peut noter que la plupart des approches de reformulation de requêtes en RI dans les documents XML [CRO 04, MAS 04, MIH 04, SCH 05, HLA 06] sont basées sur la technique de réinjection. Cela consiste alors à enrichir la requête initiale en spécifiant la structure à partir d'éléments jugés pertinents.

Dans la reformulation automatique [SAL 83], l'expansion de la requête est fonction d'une connaissance du domaine, généralement matérialisée par un thésaurus (voire une ontologie du domaine) qui définit les relations existant entre les différents termes d'index et permet une sélection de nouveaux termes à ajouter dans la représentation initiale de la requête. Cette approche présente l'inconvénient de nécessiter la disponibilité d'une base de connaissance préétablie sur le domaine, or cela n'est pas toujours possible car la création et la maintenance d'un thésaurus ou d'une ontologie nécessite, par exemple, la présence d'un expert.

Dans notre approche, la connaissance utilisée est issue des requêtes formulées antérieurement par les utilisateurs. Il s'agit d'extraire de ces requêtes une connaissance cachée, représentée par des règles d'association. L'utilisation de règles d'association pour améliorer la RI a déjà été abordée dans [LAT 03]. Il s'est agi d'utiliser des règles d'association pour trouver des corrélations de termes dans une RI classique. Les auteurs s'intéressent plus à la découverte de règles d'association dites floues. Dans notre contexte, nous nous intéressons davantage à l'exploitation des règles d'association pour la reformulation qu'à leur découverte. Par ailleurs, nous travaillons sur des documents XML ; ce qui présente l'intérêt de pouvoir restituer des parties de documents satisfaisant une requête et de les classer selon leur degré de pertinence.

Par ailleurs, dans le cadre de la recherche d'informations sur le web, des travaux ont porté sur l'exploitation des fichiers logs pour l'expansion de requêtes [HAN 03, YUN 05]. Dans ces approches, les fichiers logs sont utilisés pour établir des corrélations entre les termes des requêtes et ceux des documents. Dans notre approche, nous nous intéressons plutôt aux corrélations entre termes des requêtes.

3. Les règles d'association

Dans cette partie, nous présentons brièvement les règles d'associations. Le lecteur est invité à consulter [AGR 93, AGR 94] pour de plus amples détails.

La découverte de règles d'associations a été introduite par Agrawal et al. [AGR 93] pour l'analyse du panier de la ménagère. Le problème peut se présenter comme suit : Soit $I = i_1, i_2, \ldots, i_n$ un ensemble d'articles et D un ensemble de transactions, où chaque transaction T est un ensemble d'articles tel que $T \subseteq I$. La *table 1* donne un exemple où une base de données D contient un ensemble de transactions T, et chaque transaction est composée d'un ou de plusieurs articles (items). Un ensemble d'items est appelé un *itemset*.

TID	Items
1	{ pain, beurre, lait }
2	{ pain, beurre, lait, crème }
3	{ crème, fromage }
4	{ batterie, pain, beurre, lait }
5	{ pain, beurre, lait }
6	{ batterie, crème, pain, beurre }

Tableau 1 : Un exemple de base de données

Une règle d'association est une implication de la forme $X \Rightarrow Y$, où $X, Y \subset I$ et $X \cap Y = 0$. Généralement, X est appelé l'*antécédent* et Y le *conséquent*. La règle $X \Rightarrow Y$ a un *support* de s dans l'ensemble de transactions D si $s\%$ des transactions dans D contiennent $X \cup Y$. Le support d'une règle est défini comme $support(X \cup Y)$. La règle $X \Rightarrow Y$ a pour *confiance* c si $c\%$ des transactions de D qui contiennent X contiennent aussi Y. La confiance d'une règle est définie comme $support(X \cup Y)/support(X)$. Par exemple, considérons la base de données du Tableau 1. Lorsque les gens achètent du pain et du beurre, ils achètent aussi du lait dans 66% des cas et 80% des transactions qui contiennent du pain et du beurre contiennent aussi du lait. Une telle règle peut être représentée comme suit :

$$Pain, beurre \Rightarrow lait \qquad support = 0.66, confiance = 0.8$$

Le nombre de règles trouvées pouvant être très important, l'objectif de l'extraction est de ne générer que les règles dont le support et la confiance sont supérieurs respectivement aux seuils minimum de support *(minsup)* et de confiance *(minconf)*. Un itemset dont le support est supérieur à minsup est appelé itemset fréquent. La règle $X \Rightarrow Y$ est une règle *intéressante* ssi $X \cup Y$ est un itemset fréquent et la confiance de la règle est supérieure ou égale à *minconf*. La découverte de règles d'associations se fait en deux étapes :

1. La recherche des itemsets fréquents. Pour cela, l'algorithme utilisé est l'algorithme *Apriori*. Nous parlerons de cet algorithme dans la section 3.1.
2. La recherche à partir de ces itemsets de règles intéressantes.

La performance de l'extraction est surtout déterminée par la première étape qui est la plus coûteuse.

3.1 Algorithme Apriori

L'algorithme Apriori [AGR 94] est un algorithme par niveau qui calcule les itemsets fréquents en partant des itemsets les plus généraux vers les plus spécifiques. Ainsi, au niveau 1, il calcule les 1-itemsets fréquents (i.e de taille 1), passe au niveau 2 et continue ainsi jusqu'à trouver l'ensemble des itemsets fréquents.

Chaque niveau comprend une phase de génération des itemsets candidats et une phase d'évaluation. La phase de génération peut se découper en deux étapes :

1. la phase de jointure : lors de cette phase les itemsets de taille k sont générés en faisant une jointure des itemsets de taille k-1 ;
2. la phase d'élagage, durant laquelle tout itemset X généré par la phase de jointure est supprimé s'il existe un sous-ensemble de X qui est non fréquent. En effet, tous les sous-ensembles d'un itemset fréquent doivent également être fréquents.

La Figure 1 décrit l'algorithme *Apriori*.

Algorithme : Algorithme *Apriori*
Entrée : une base de données et un seuil minimum de support
Sortie : l'ensemble des itemsets fréquents
1. $L_k = \emptyset$; k = 0 ;
2. C_1 = l'ensemble des 1-itemsets dans D
3. L_1 = l'ensemble des itemsets fréquents de C_1
4. while (L_{k+1} non vide)
5. C_{k+1} = Candidate-gen(L_k)
6. L_{k+1} = l'ensemble des itemsets fréquents de C_{k+1}
7. k++
8. return $\cup L_k$

Figure 1 : Algorithme *Apriori*

Il faut dire qu'il existe d'autres algorithmes de recherche des itemsets fréquents tels que [AGA 00, AGA 01, HAN 00, PEI 00, ZAK 00]. Cependant, comme nous l'avons précisé dans la section 2, nous nous intéressons dans ce papier davantage à l'exploitation des règles qu'à leur extraction.

4. Intégration de règles d'association dans un SRI XML

4.1. Architecture du SRI

Notre objectif est de trouver les portions de documents XML satisfaisant une requête et de les classer selon leur degré de pertinence. Cela exige un traitement au niveau des sous-documents lors des phases d'indexation et de recherche. Nous exploitons alors la structure de graphe qui est sous-jacente à chaque document XML [SMA 02]. L'idée consiste alors à exploiter une des caractéristiques fondamentales d'un document XML: l'information ne se trouve qu'au niveau des feuilles de cet arbre; c'est à ce niveau que nous effectuons l'indexation. Ces feuilles sont alors les unités atomiques que nous appelons *unités élémentaires* (ou *sous-documents*). Chaque unité élémentaire est identifiée de façon unique par un chemin indiquant sa position dans le document. Dès lors, l'indexation et la recherche se feront sur l'ensemble des unités élémentaires de tous les documents composant une collection.

Nous présentons ici l'architecture générale du SRI pour les documents XML en montrant ses principaux composants (Figure 2) :

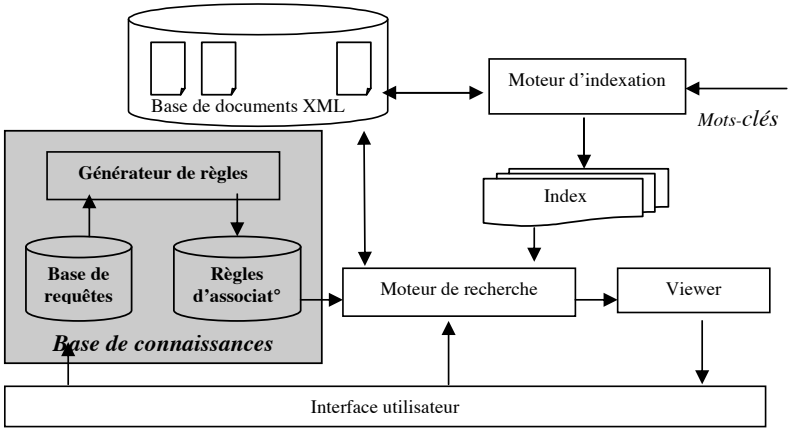

Figure 2 ***: Architecture générale du système de recherche d'information***

Les principaux composants du système de base sont :

- Un moteur d'indexation : pour chaque document XML, le moteur d'indexation crée un index, appelé document-index, en utilisant un ensemble de mots-clés.
- Un moteur de recherche : il permet de retrouver l'information pertinente (documents ou parties de documents) à partir des documents-index et les mots-clés exprimés dans une requête.
- Un viewer : il affiche les documents ou parties de documents retrouvés. Ces résultats sont recomposés (au moyen de XSL) pour être présentés à l'utilisateur final de façon appropriée en HTML.
- Une interface utilisateur : elle est utilisée pour faciliter l'interaction entre l'utilisateur et l'application. Elle permet à l'utilisateur de spécifier sa requête. Elle affiche aussi les documents (ou parties de documents) retrouvés et classés selon leur degré de pertinence.

Nous avons intégré à ces composants de base du SRI une base de connaissances constituée de règles d'association. Cette base de connaissances schématisée en gris dans la Figure 2 permet d'effectuer le processus de reformulation.

4.2. La base de connaissances

Pour diminuer le silence du SRI, nous proposons la reformulation des requêtes à partir des connaissances trouvées dans une base de requêtes provenant de l'enregistrement des précédentes requêtes utilisateurs, et ce pour plusieurs raisons.

La recherche d'informations dans un SRI est basée sur les mots-clés entrés par l'utilisateur. A chaque requête, le moteur de recherche établit une comparaison entre les mots-clés et la base d'index. De ce fait, la même requête retourne toujours le même résultat. Dès lors la pertinence des résultats dépend étroitement du choix de ces mots-clés. Or l'utilisateur, étant à la recherche d'information, ignore souvent quels mots-clés pourraient lui donner les résultats dont il a besoin. Cela vient du fait qu'il ne connaît pas exactement les mots utilisés pour l'indexation. Par ailleurs, il peut exister des corrélations inconnues par l'utilisateur entre certains domaines. Par exemple, un utilisateur qui cherche des documents sur XSL, pourrait être intéressé également par ceux concernant SAX s'il est à la recherche de techniques de manipulation de fichiers XML.

Considérant l'ensemble des requêtes déjà soumises au système, une extraction de connaissances pourrait permettre la découverte de nouveaux mots-clés corrélés à la requête utilisateur. Ceux-ci peuvent être des mots de la même famille que les mots-clés initiaux ou mettre en évidence un besoin non exprimé puisque inconnu comme indiqué précédemment. L'exemple avec XSL et SAX est très illustratif pour ce cas de figure. Cette proposition nécessite donc la génération de règles d'associations entre des mots-clés ; ceux-ci seront soumis au moteur de recherche en fonction de la requête courante.

N°	Requête
1	xml sax xsl xquery
2	xml xsl manipulation
3	sax dom xsl xpath
4	sax xsl xquery
5	xsl xpath sax
6	comparer sax dom xsl xpath xquery

```
<transactions>
  <transaction id="1">
    <items>
      <item>xml</item> <item>sax</item>
      <item>xsl</item> <item>xquery</item>
    </items>
  </transaction>
  <transaction id="2">
    <items>
      <item>xml</item> <item>xsl</item>
      <item>manipulation</item>
    </items>
  </transaction>
…
</transactions>
```

Figure 3 : Document de transactions

La base de connaissances est constituée d'une base de règles d'association extraite à partir d'une base de requêtes.

La base de requêtes, qui correspond à l'ensemble des transactions, est construite en enregistrant les différentes requêtes (partie gauche de la Figure 3) des utilisateurs dans un document XML (partie droite de la Figure 3).

La génération de règles est directement effectuée à partir du document XML en utilisant une implémentation en XQuery de l'algorithme Apriori comme cela a été fait dans [WAN 04]. Soulignons qu'une autre approche serait de transformer d'abord les données XML en données relationnelles et d'appliquer l'algorithme d'extraction. Les règles sont stockées dans un document XML comme le montre la Figure 4 .

```
<rules>
  <rule support="0.83" confidence="0.83">
    <antecedent> <item>xsl</item> </antecedent>
    <consequent> <item>sax</item> </consequent>
  </rule>
  <rule support="0.5" confidence="1">
    <antecedent> <item>sax</item> <item>xquery</item> </antecedent>
    <consequent> <item>xsl</item> </consequent>
  </rule>
  …
</rules>
```

Figure 4 : Document de règles d'association

La génération de règles d'association se fait de manière périodique par l'administrateur du SRI qui fixe les seuils de support et de confiance. Cela veut dire que l'ensemble des règles d'association intéressantes peut évoluer d'une période à une autre. Autrement dit, nous avons ici affaire à une base de connaissances dynamique. L'intérêt d'avoir une base de connaissances dynamique se justifie d'autant plus qu'avec l'évolution rapide des nouvelles technologies, on remarque souvent l'arrivée de nouveaux concepts (exemple : AJAX) utilisés à la place d'autres (exemple : Java Script). Cela engendre une évolution des besoins des utilisateurs en termes de requêtes ; et par conséquent, une évolution de la connaissance.

4.3. Algorithme de reformulation

Nous exploitons la base de connaissances pour effectuer une reformulation automatique de requêtes. La **Figure 5** présente l'algorithme de reformulation.

Soit *T* un ensemble de termes composant une requête et *U(T)* l'ensemble des unités élémentaires satisfaisant à cette requête.

A partir d'une requête utilisateur *T*, le SRI fournit un premier ensemble d'unités élémentaires *U(T)*. La reformulation consiste alors à trouver un ensemble de termes

T' corrélés à ceux de la requête initiale *T* en exploitant la base de règles. Cela permet au SRI de trouver un deuxième ensemble d'unités élémentaires *U(T').*

Entrée : requête *T* : ensemble de termes.

R : ensemble de règles d'association telle que chaque règle est représentée par un couple *(A, C)* où *A* est l'ensemble des termes *antécédent* et *C* l'ensemble des termes *conséquent*.

Sortie : ensemble *T'* de termes (à utiliser pour reformuler la requête *T*)

Début

- Trouver l'ensemble *E* des règles $Ri = (Ai, Ci)$ telles que $T \subseteq Ai$
- $T' = \varnothing$
- Pour chaque règle $e = (a,c) \in E$ faire

  $T' = T' \cup c$

Fin

Figure 5 : Algorithme de reformulation

Par exemple, supposons qu'un utilisateur pose une requête $T = \{sax, xquery\}$ et que l'on dispose de la règle d'association *sax, xquery* ➔ *xsl*. L'algorithme de reformulation va trouver l'ensemble $T'=\{xsl\}$. Ainsi, le SRI proposera automatiquement à l'utilisateur, en plus de l'ensemble d'unités élémentaires *U(T)*, l'ensemble *U(T').*

6. Conclusion

Dans cet article, nous avons proposé une approche de reformulation automatique de requêtes basée sur une technique d'extraction de connaissances. Nous avons intégré une base de connaissances composée de règles d'association dans le SRI que nous avons développé pour les documents XML. La base de règles est construite à partir des requêtes des utilisateurs et représente une connaissance qui évolue en fonction des termes utilisés dans ces requêtes.

Nous exploitons cette base de connaissances dynamique pour proposer une méthode de reformulation automatique de requêtes. Cela permet alors au moteur de recherche de trouver un deuxième ensemble de sous-documents XML permettant de diminuer le silence du SRI.

Une implémentation de l'approche est en cours dans un environnement Java avec des outils comme Tomcat, MySQL et Saxon.

Nos travaux en cours portent sur l'amélioration des performances de l'algorithme de reformulation. En effet, la taille de la base de règles peut être très importante, ce qui

peut augmenter le temps de réponse du SRI. Par ailleurs, la mise à jour du document de transactions peut également augmenter ce temps de réponse.

Une perspective de ce travail consiste à exploiter la base de connaissances pour offrir un service d'aide à la recherche pour les utilisateurs non-experts. Il s'agira de construire une base de connaissances à partir de requêtes d'experts d'un domaine et de l'utiliser pour assister d'autres utilisateurs.

Remerciements : Nous remercions nos étudiants Emma Valérie Traoré et Abderrahmane Ould Ahmed qui ont démarré l'implémentation de l'approche dans le cadre de leur projet opérationnel de DESS d'informatique.

7. Bibliographie

[AGA 00] Agarwal R. C., Aggarwal C. C and Prasad V. V. V, *Depth first generation of long patterns*. In Proceedings of the ACM SIGKDD Conference, 2000.

[AGA 01] Agarwal R. C., Aggarwal C. C and Prasad V. V. V, *A tree projection algorithm for generation of frequent itemsets*, volume 61, pages 350-371, 2001.

[AGR 93] Agrawal R., Imielinski T. and Swami A.. *Mining Association rules between sets of items in large databases*. In P. Buneman and S. Jajodia, editors, SIGMOD93, pages 207-216, Washington, D.C., USA, May, 1993.

[AGR 94] Agrawal R. and Srikant R.. *Fast algorithms for mining association rules in large databases*. In J. B. Bocca, M. Jarke and C. Zaniolo, editors, Proccedings of 20th International Conference on Very Large Data Bases, pages 487-499, Santiago, Chile, September 12-15 1994.

[BAE 99] Baeza-Yates R., Ribbiro-Neto B., *Modern Information retrieval,* Addison Wesley, ACM, Press New York, ISBN 0-201-39829-X, 1999.

[BIL 00] Billhardt H., Borrajo D., Maojo V., *Using terms co-occurrence data for document indexing and retrieval research,* p. 105-117, April 2000.

[CRO 04] Crouch C., Mahajan A., Bellamkonda A., *Flexible XML Retrieval Based on the Vector Space Model*, INEX 2004 Workshop Proceedings, Germany, December 2004, p. 292,302.

[DUM 94] Dumais S., *Latent semantic indexing* : TREC-3 report, In proceeding of the Third Text Retrieval conference, p.219-230, Gaithersburg, Maryland, novembre 1994.

[HAN 00] Han J., Pei J., Yin Y., *Mining frequent patterns without candidate generation*, In W. Chen, J. Naughton, and P. A. Bernstein, editors, 2000 ACM SIGMOD Intl. Conference on Management of Data, pages 1-12. ACM Press, 05-2000.

[HAN 02] Hang C., Ji-Rong W., Jian-Yun N., Wei-Ying M., *Query Expansion by Mining User Logs*, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, vol 15, N°4, July/Agust 2003.

[HAR 92] Harman D., *Relevance feedback revisited,* In proceedings of ACM SIGIR Conference on research and development in information retrieval, pp. 1-10, 1992.

[HLA 06] Hlaoua L., *Reformulation de requêtes par structure en RI dans les documents XML*, CORIA'06, Lyon, Mars 2006.

[IHA 04] Ihadjadene M, *Les systèmes de recherche d'informations : modèles conceptuels,* Lavoisier, 2004.

[LAT 03] Latiri C. Ch., Yahia S. Ben, Chevallet J.P. and Jaoua A., *Query expansion using fuzzy association rules between terms*, in JIM'2003 conference Journées de l'Informatique Messine, Metz , France, September 3-6, 2003.

[LO 05] Lo M., Hocine A., *ISYWEB : an XML-based Architecture for web information systems*, IEEE SITIS Conference, Yaounde, Novembre 2005.

[MAS 04] Mass Y., Mandelbrod M., *Relevance Feedback for XML Retrieval* , INEX 2004 Workshop Proceedings, Germany, December 2004, p. 303,310.

[MIH 04] Mihajlovic V., Ramirez G., De Vribes A., Hibmstra D., Blok H., *TIJAH at INEX 2004 Modeling Phrases and Relevance Feedback*, INEX 2004 Workshop Proceedings, Germany, December 2004, p. 276,291.

[PEI 00] Pei J., Han J. and Mao R., *Closet : an efficient algorithm for mining frequent closed itemsets.* In ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, pages 21-30, 2000.

[SAL 71] Salton G., *The SMART Retrieval System,* Experiments in automatic Document Processing, Prentice Hall, 1971.

[SAL 83] Salton G., McGill M.J., *Introduction to Modern Information Retrieval,* MCGraw Hill, 1983.

[SCH 05] Schenkel R., Thbobald M., *Relevance Feedback for Structural Query Expansion*, INEX 2005 Workshop Pre-Proceedings, Germany, November 2005, p. 260,272.

[SCO 90] Scott D., Susan T.D., Richard H., *Indexing by Latent Semantic Analysis,* Journal of the American Society of information Science, 1990.

[SMA 02] Smadhi S., Lo M., Hocine A., *Repository de documents XML : Modélisation et recherche d'informations pertinentes*, actes du congrès 7th Maghrebian Conference on Software Engineering and Artificial Intelligence (MCSEAI), Annaba, Algérie, Mai 2002.

[WAN 04] Wan J. W. W., Dobbie G., *Mining association rules from XML Data using XQuery*, ACM International Conference Proceeding Series; Vol. 54, 2004.

[WON 85] [9] Wong S.K.M., Ziarko W., Raghavan V.V., *Generalization vector space model in infomation retrieval,* Proceeding of the 8th Annual International ACM SIGIR conference on research and development in information retrieval, p. 18-25, 1985.

[YUN 05] Yun Z. Gruenwald L., *Query Expansion Using Web Access Log Files*, DEXA 2005.

[ZAK 00] Zaki M. J., *Generating non redundant association rules*. In Knowledge Discovery and Data Mining, pages 34-43, 2000.