A la Recherche de nœuds informatifs dans des corpus de documents XML

Ou pourquoi on a toujours besoin de plus petit que soi...

Karen Sauvagnat, Mohand Boughanem

IRIT - SIG
118 route de Narbonne
31 062 Toulouse Cedex 4
{sauvagna, bougha}@irit.fr

RÉSUMÉ *Un des principaux challenge de la Recherche d'Information dans des documents XML est le traitement des requêtes composées de simples mots-clés. L'utilisateur exprimant de telles requêtes ne donne en effet aucune indication au système sur la granularité de l'information qu'il désire. De quel type doit-être cette information ? Les documents XML pouvant être considérés comme des arbres, chercher les parties de documents pertinentes à une requête revient à chercher des sous-arbres pertinents. Ceci soulève les problématiques suivantes : Comment calculer la pertinence de ces sous-arbres ? Toute l'information structurelle contenue dans les documents est-elle utile et doit-elle être indexée ? Afin de répondre à ces problématiques, nous présentons ici une suite d'expérimentations reposant sur un modèle de propagation de la pertinence et utilisant le système XFIRM. Les performances de notre système sont évaluées dans le cadre de la campagne d'évaluation INEX et les premiers résultats montrent de bonnes performances par rapport aux soumissions officielles. De plus, les expérimentations effectuées laissent apparaître que les nœuds de petite taille, quoique non informatifs, semblent jouer un rôle non négligeable dans le calcul de la pertinence de leurs nœuds ancêtres.*

ABSTRACT. *One of the key challenge in XML retrieval is to answer queries composed of simple keywords terms. When expressing such queries, the user does not give any indication to the system concerning the granularity of the information he/she is expecting to have. What is the type of this information ? As XML documents can be considered as trees, searching relevant document parts can be considered as searching relevant sub-trees. This raises some open issues : how to compute the relevance of these sub-trees ? Is all the structural information contained in XML documents usefull ? What should we index ? In this paper, we present some experiments which aim at answering these open issues. We use a relevance propagation method based on the XFIRM system. We evaluate the performances of the system thanks to the INEX evaluation campaign and first results show a relatively high precision of our proposal comparing to INEX official results. Moreover, experiments show that small nodes, even if they are not informative, have some importance in the evaluation of their ancestors relevance.*

MOTS-CLÉS : *XML, propagation de la pertinence, recherche orientée contenu, unité d'indexation.*

KEYWORDS *XML, relevance propagation, content-oriented search, indexing unit*

1. Introduction

En recherche d'information traditionnelle, la granularité des réponses renvoyées aux utilisateurs est généralement restreinte au document tout entier. Cependant, un document possède souvent des contenus hétérogènes, et l'utilisateur doit alors aller chercher l'information pertinente à sa requête au milieu des autres thèmes abordés par le document. Le format XML (*eXtensible Markup Language*) permet d'ajouter de l'information structurelle au contenu des documents, et donc de traiter l'information avec une autre granularité que le document tout entier.

Lorsqu'un utilisateur désire interroger un corpus de documents XML, il peut formuler différents types de requêtes, selon sa connaissance de la collection ou bien selon la précision de son besoin en information. On distingue deux types principaux de requêtes : *(i)* les requêtes composées de simples mots-clés (aussi appelée requêtes sur le contenu seul, ou bien encore requêtes CO (*Content Only*)) et *(ii)* les requêtes composées de conditions de contenu et de structure (aussi appelée requêtes CAS (*Content-and-Structure*)). Les premières sont de loin les plus simples pour l'utilisateur, qui laisse le Système de Recherche d'Information (SRI) décider du type de l'information à renvoyer. Le second type de requête suppose par contre que l'utilisateur a au moins une connaissance partielle de la structure des documents qu'il interroge, puisqu'il formule des conditions sur la structure des documents. Lors du traitement des CAS, ces conditions de structure sont utilisées comme des indications pour trouver les parties de documents les plus pertinentes. Pour les requêtes CO, de telles conditions n'existent pas et d'autres méthodes doivent être utilisées.

Le challenge principal est d'identifier les parties de documents les plus pertinentes à une requête donnée. L'idée de travailler avec des parties de documents a été largement explorée dans le contexte de la recherche de passages (« *passage retrieval* ») (Salton *et al*., 1993). Ces travaux sont difficilement adaptables à la recherche dans des documents structurés puisque dans ces derniers, l'information est découpé selon des éléments de structure. En effet, les documents XML peuvent être représentés sous forme *d'arbre*, encore appelé arbre DOM (*Document Object Model*). Chaque *nœud feuille* contient du texte, et les autres *nœuds* (ou *éléments*) forment la structure du document et sont le point de départ de sous-arbres. Dans ce contexte, l'unité d'information renvoyée par un SRI doit donc être un nœud XML existant (c'est à dire un *sous-arbre*).

La pertinence d'un nœud vis à vis d'une requête est évaluée selon les deux notions suivantes : l'*exhaustivité* et la *spécificité* (Chiaramella *et al*., 1996) (Lalmas, 1997). On dit qu'un nœud est exhaustif à une requête s'il contient toutes les informations requises et qu'il est spécifique si tout son contenu concerne la requête. Chercher les nœuds les plus spécifiques et exhaustifs pour une requête CO revient alors à chercher les sous-arbres de taille minimale pertinents à la requête. Cette recherche soulève de nombreuses questions: toute l'information structurelle contenue dans les documents est-elle utile et doit-elle être indexée ? En d'autres termes, doit-on indexer tous les sous-arbres d'un document ? Comment calculer la pertinence de ces sous-arbres à une requête ? Nous nous intéressons ici à ces

diverses problématiques, que nous explorons en utilisant une méthode de propagation de la pertinence. Cette approche est basée sur le système *XFIRM (XML Flexible Information Retrieval Model*), présenté dans (Sauvagnat, 2004). La section suivante présente plus en détail les différentes problématiques liées aux requêtes orientées contenu et donne un bref aperçu des approches proposées dans la littérature. La section 3 présente les bases de notre modèle. Nos expérimentations ont été menées sur les outils fournis par la campagne d'évaluation INEX (section 4) et sont présentées dans la section 5.

2. Etat de l'art et contribution

Dans cette section, nous présentons un rapide état de l'art des approches proposées dans la littérature pour répondre aux questions inhérentes à la recherche orientée contenu. Cette recherche revenant à trouver les sous-arbres de taille minimale pertinents à la requête, la principale problématique soulevée est de savoir comment calculer la pertinence de ces sous-arbres. Le calcul de ce poids dépend des informations conservées dans l'index, c'est à dire du choix de l'unité d'indexation minimale utilisée. Quel est le rôle de l'information structurelle contenue dans les documents dans le calcul du poids de pertinence des sous-arbres ? Toute cette information est-elle nécessaire ? En d'autres termes, doit-on la conserver dans sa globalité dans l'index ?

2.1. Choix de l'unité d'indexation

Certaines approches considèrent que déterminer les unités d'indexation revient à déterminer quel est le type et la taille minimale des nœuds qui pourront être renvoyés par le système. En effet, tous les nœuds d'un document donné ne peuvent pas être considéré comme des réponses possibles. Par exemple, un nœud *titre* ne peut pas être pertinent à une requête parce que même s'il contient les termes de la requêtes, il n'apporte aucune information à l'utilisateur. De manière similaire, un utilisateur préférera retrouver dans une liste de résultats un nœud *paragraphe* contenant un nœud *titre* et un nœud *corps* qu'un nœud *corps* simple. La façon dont les nœuds de l'index sont choisis dépend de la DTD des documents, et ce choix peut être fait manuellement ou automatiquement.

Une approche opposée est de considérer que tous les nœuds de l'arbre du document doivent être présents dans l'index, car ils sont tous porteurs d'une certaine quantité d'information, que ce soit au niveau structurel ou au niveau contenu. Dans ce cas là, la réponse à une requête donnée doit être construite de manière dynamique: tous les nœuds ne peuvent pas être considérés comme informatifs (c'est à dire qu'ils sont de trop petite taille pour apporter de l'information), et ne doivent donc pas être renvoyés par le SRI.

2.2. Calcul de la pertinence des nœuds

Les différentes approches proposées pour le traitement des requêtes peuvent être divisées en trois sous-groupes dans (Abolhassani *et al.*, 2004). Les méthodes du premier groupe considèrent que le texte complet de chaque nœud de l'index est un document atomique. En d'autres termes, ces *approches indexent tous les sous-arbres* (jugés potentiellement pertinents) des documents. Comme les documents XML possèdent une structure hiérarchique, les nœuds de l'index sont imbriqués les uns dans les autres. Le score de pertinence d'un nœud donné est alors calculé à partir de statistiques sur les termes qu'il contient. Les sous-arbres sont ensuite triés selon leur score de pertinence. On trouvera des exemples de ces approches dans (Abolhassani *et al.*, 2004) et (Sigurbjörnsson *et al*., 2003).

Dans le second groupe, on trouve les approches *dépendantes du contexte*, pour lesquelles le score de pertinence d'un nœud donné dépend des statistiques sur les nœuds du même type. Une approche utilisant les réseaux bayésiens est proposée dans (Piwowarski *et al.*, 2002). Grabs et Scheck (Grabs *et al*., 2002) utilisent aussi cette idée pour des requêtes CAS: le poids d'un nœud de type donné (une *section* par exemple) dépend des statistiques sur les nœuds du même type. Dans (Mass *et al.*, 2003), Y. Mass et M. Mandelbrod proposent l'utilisation d'index différents pour chaque type de balises : par exemple, un index pour les nœuds de type *section*, un index pour les nœuds *paragraphe*, un index pour les nœuds *abstract*,… Les requêtes sont évaluées sur chacun de ces index et les résultats sont ensuite fusionnés.

Les approches traitant des *unités disjointes* forment le dernier groupe. Dans ces approches, le document XML est décomposé en unités disjointes, de telle façon que le texte de chaque nœud de l'index est l'union d'une ou plus de ces parties disjointes. Durant la recherche, le poids des nœuds plus haut dans la hiérarchie est calculé grâce à l'agrégation des poids des nœuds d'indexation. Une approche utilisant les modèles de langage est proposé dans (Ogilvie *et al.*, 2003) : le modèle de langage d'un nœud dans la hiérarchie est calculé comme la somme des modèles de langage des nœuds qu'il contient. Une méthode de propagation est proposée par Fuhr et al. dans (Gövert *et al*., 2002). Le poids de pertinence des nœuds est calculé grâce à la propagation des poids des nœuds les plus spécifiques dans l'arbre du document. Ces poids sont cependant diminués par multiplication par un facteur d'augmentation.

2.3. Contribution

Les expérimentations présentées dans cet article reposent sur le modèle XFIRM, décrit plus en détails dans la section suivante. Ce modèle est basé sur une méthode de propagation de la pertinence, et permet, grâce à la flexibilité apportée par son modèle de représentation, de tester de plusieurs modèles de recherche. De nombreuses méthodes de propagation de la pertinence ont été proposées dans la littérature (Fuhr et al., 2001), (Gövert *et al*., 2002) , (Anh *et al.*, 2002), (Grabs *et al.*, 2002), (Sauvagnat *et al*., 2004). Le but de cet article est d'évaluer les différentes problématiques présentées ci-dessus afin d'obtenir des performances optimales.

Dans ce qui suit, nous nous proposons donc :

1. D'évaluer l'impact de la taille des unités d'indexation sur les performances du système. Notre modèle indexe des unités disjointes, mais le problème de la taille de ces unités reste entier : à quel point les performances du système en sont-elles affectées ?

2. De tester plusieurs modèles de recherche, c'est à dire plusieurs formules de calculs du poids de pertinence des nœuds (que ce soit au niveau des nœuds feuilles ou bien au niveau des nœuds point de départ de sous-arbres).

3. 3. Traitement des requêtes orientées contenu avec le système XFIRM

Le modèle XFIRM (Sauvagnat, 2004) est basé sur un modèle de données générique permettant l'implémentation de nombreux modèles de RI et le traitement de collections hétérogènes (c'est à dire contenant des documents ne suivant pas la même DTD). Nous considérons qu'un document structuré ds_i est un arbre, composé de nœuds simples n_{ij}, de nœuds feuilles nf_{ij} et d'attributs a_{ij}. On trouvera un exemple d'arbre XML représentant un article sur la figure 1. L'ensemble des nœuds simples n_{ij}, et l'ensemble des nœuds feuilles nf_{ij} peuvent être définis en suivant strictement la représentation en arbre du document ou bien en transformant l'arbre original pour ne garder que les nœuds considérés comme intéressants. Quelle que soit la méthode utilisée, les nœuds feuilles sont porteurs de contenu, alors que les autres nœuds donnent simplement des indications de structure. Dans la suite de l'article, les expérimentations effectuées l'ont été sur deux index différents, selon que l'on ait gardé la structure entière des documents ou bien que l'on ait sélectionné seulement certains nœuds.

Afin de parcourir facilement l'arbre du document et de retrouver rapidement les relations ancêtres-descendants, le modèle XFIRM utilise une représentation des nœuds et des attributs basée sur XPath Accelerator (Grust, 2002). On trouvera une description détaillée de l'index dans (Sauvagnat , 2004).

Le traitement des requêtes CO est effectué comme présenté ci-dessous : une première étape consiste à évaluer la similarité des nœuds feuilles de l'index à la requête (on parle alors de *calcul des poids des nœuds feuilles*) et une seconde étape consiste à rechercher les sous-arbres pertinents. La pertinence des sous-arbres est évaluée en propageant le poids des feuilles dans l'arbre du document.

3.1. Evaluation du poids des nœuds de l'index

Soit $q=t_1,\ldots,t_n$ une requête CO. Les poids des nœuds feuilles identifiés dans l'arbre du document sont calculés grâce à la fonction de similarité $RSV_m(q,nf)$ (*Retrieval Status Value*), où m est le modèle de RI considéré.

$$RSV_m(q, nf) = \sum_{i=1}^{n} w_i^q * w_i^{nf} \qquad [1]$$

où w_i^q et w_i^{nf} sont respectivement le poids du terme *i* dans la requête *q* et le nœud feuille *nf*, le calcul de ces poids dépendant du modèle *m* considéré.

Nous avons testé plusieurs fonctions, présentées dans la section 5.

3.2. Propagation de la pertinence

Une valeur de pertinence est ensuite calculée pour chaque nœud de l'arbre de document, en utilisant les poids des nœuds feuilles qu'il contient. Les termes apparaissant près de la racine d'un sous-arbre paraissent plus porteurs d'information pour le nœud associé que ceux situés plus bas dans le sous-arbre. Il semble ainsi intuitif que plus grande est la distance entre un nœud et son ancêtre, moins il contribue à sa pertinence. Nous modélisons cette intuition par l'utilisation dans la fonction de propagation du paramètre $dist(n,nf_k)$, qui représente la distance entre le nœud *n* et un de ses nœuds feuille nf_k dans l'arbre du document, c'est à dire le nombre d'arcs séparant les 2 nœuds. La valeur de pertinence p_n d'un nœud est alors calculée selon la formule suivante :

$$p_n = \sum_{k=1..n} \alpha^{dist(n,nf_k)-1} * RSV(q,nf_k) \qquad [2]$$

où les nf_k sont les nœuds feuilles descendants de *n* et *N* est leur nombre total.

Illustrons cette propagation avec le document de la figure 1 et la requête « *Google* ».

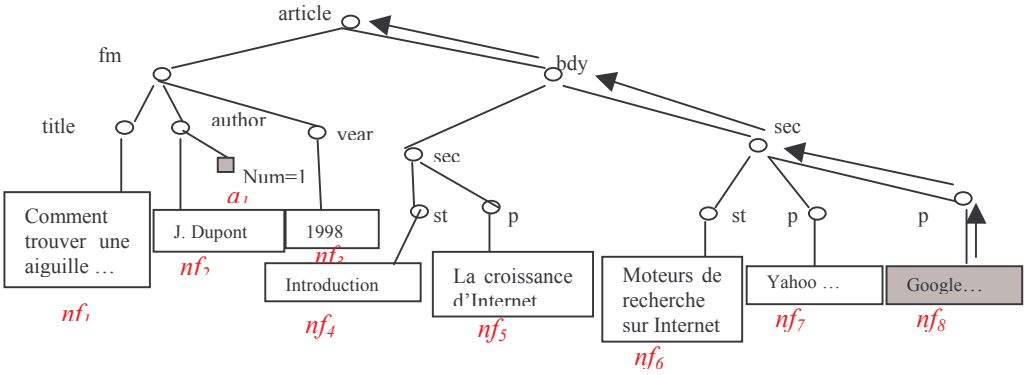

Figure 1: *Exemple de propagation de la pertinence dans un arbre XML*

Seul le nœud feuille nf_8 a un score de similarité non nul avec la requête. La pertinence de son nœud parent p est alors égale à $\alpha^0 * RSV(q,nf_8)$, c'est à dire à son propre score. De la même façon, tous les nœuds parents (et non ancêtres) de nœud feuilles auront une pertinence égale au score de similarité de leur nœud feuille descendant. Nous avons ensuite :

$p_{sec} = \alpha^1 * RSV(q,nf_6) + \alpha^1 * RSV(q,nf_7) + \alpha^1 * RSV(q,nf_8) =^{\circ} \alpha * RSV(q,nf_8)$

$p_{bdy}=\alpha^2*RSV(q,nf_4)+ \alpha^2*RSV(q,nf_5)+ \alpha^2*RSV(q,nf_6)+ \alpha^2*RSV(q,nf_7)+ \alpha^2*RSV(q,nf_8)=\alpha^2*RSV(q,nf_8)$

…

Plusieurs valeurs de α ont été testées dans les expérimentations présentées ci-dessous. Les nœuds sont ensuite renvoyés à l'utilisateur par ordre décroissant de pertinence à la requête.

Avant de décrire plus en détail notre protocole expérimental, la section suivante présente la collection de test que nous avons utilisé, à savoir la collection et les requêtes associées à la campagne d'évaluation INEX (*Initiative for the Evaluation of XML retrieval*) 2003.

4. La tâche CO de la campagne d'évaluation INEX

4.1. Collection et requêtes

Afin d'évaluer la performance des divers SRI pour la recherche d'information dans des documents structurés, la campagne d'évaluation INEX met à disposition des participants une collection de test, des tâches de recherche composées de requêtes de type divers ainsi que les jugements de pertinence associés. La collection de test est composé de 12135 articles provenant de 21 revues IEEE Computer Society parues de 1995 à 2002. Les articles sont de longueurs variées et totalisent 8 millions de nœuds. En moyenne, un article contient 1532 nœuds.

Les requêtes qui composent la tâche de recherche Content-Only (CO) ont pour but de retrouver des parties de documents pertinentes sans que l'utilisateur ne donne d'information sur la granularité de l'information à renvoyer. Lors de la campagne d'évaluation 2003, la tâche était composée de 36 « topics » décrivant 36 tâches de recherches différentes. Dans nos expérimentations, seul le champ *Title* (composés de simples mots-clés) des topics est utilisé pour le traitement des requêtes.

Les jugements de pertinence pour chaque requête sont effectués par les différents participants. Deux dimensions sont utilisées pour définir la pertinence : l'*exhaustivité* (e) et la *spécificité* (s). Pour chacune des 2 dimensions, une échelle de pertinence est utilisée, allant de 0 (pas exhaustif ou pas spécifique) à 3 (très exhaustif ou très spécifique).

4.2. Mesures de pertinence

Les mesures d'évaluation utilisées durant la campagne d'évaluation 2003 sont basées sur les traditionnels taux de rappel et taux de précision. Pour obtenir les courbes rappel/précision, les 2 dimensions de pertinence (*exhaustivité et spécificité*) sont agrégées en une seule valeur. Deux types de fonction d'agrégation sont utilisées:

- une agrégation "stricte" pour évaluer si un SRI est capable de retrouver des éléments très spécifiques et très exhaustifs

$$f_{strict}(e,s) = \begin{cases} 1 \text{ si } e = 3 \text{ et } s = 3 \\ 0 \text{ sinon} \end{cases} \quad [3]$$

- une agrégation "généralisée" pour évaluer les éléments selon leur degré de pertinence

$$f_{généralisée}(e,s) = \begin{cases} 1 \text{ si } e = 3 \text{ et } s = 3 \\ 0{,}75 \text{ si } (e,s) \in \{(2,3),(3,\{2,1\})\} \\ 0{,}5 \text{ si } (e,s) \in \{(1,3),(2,\{2,1\})\} \\ 0{,}25 \text{ si } (e,s) \in \{(1,2),(1,1)\} \\ 0 \text{ si } (e,s) = (0,0) \end{cases} \quad [4]$$

En 2004, d'autres fonctions d'agrégation ont été introduites. L'équation [4] accorde une préférence à la notion d'exhaustivité, attribuant de bons scores à des éléments exhaustifs mais pas forcément spécifiques. Ces éléments sont généralement de grande taille (comme des articles entier par exemple), ce qui implique que de bons résultats peuvent être obtenus en ne renvoyant que des documents et non des parties de documents. Afin de résoudre ce problème, une fonction d'agrégation "généralisée" orientée spécificité a été définie (Kazai *et al.*, 2004). Parallèlement, deux classes de fonctions d'agrégation ont été définies: on parle maintenant de fonctions orientées spécificité et de fonctions orientées exhaustivité. Les fonctions orientées spécificité (équations [5]) considèrent uniquement les éléments ayant le plus haut degré de spécificité, tandis que les fonctions orientées exhaustivité (équations [6]) ne considèrent que les éléments ayant le plus haut degré d'exhaustivité (Kazai, 2003).

$$f_{s3\_e321}(e,s) = \begin{cases} 1 \text{ si } e \in \{3,2,1\} \text{ et } s = 3 \\ 0 \text{ sinon} \end{cases} \qquad f_{s3\_e32}(e,s) = \begin{cases} 1 \text{ si } e \in \{3,2\} \text{ et } s = 3 \\ 0 \text{ sinon} \end{cases} \quad [5]$$

$$f_{e3\_s321}(e,s) = \begin{cases} 1 \text{ si } s \in \{3,2,1\} \text{ et } e = 3 \\ 0 \text{ sinon} \end{cases} \qquad f_{e3\_s32}(e,s) = \begin{cases} 1 \text{ si } s \in \{3,2\} \text{ et } e = 3 \\ 0 \text{ sinon} \end{cases} \quad [6]$$

Toutes ces mesures sont ensuite combinées pour calculer une précision moyenne, qui a été utilisée pour établir les classements officiels des participants lors de la campagne 2004 (alors qu'en 2003 seules les équations [3] et [4] ont été utilisées).

Bien que les expérimentations décrites ci-dessous aient été effectuées sur les requêtes de la campagne d'évaluation 2003[1], nous évaluerons notre système en présentant la moyenne des fonctions d'agrégation définies ci-dessus, considérées comme plus complètes pour l'évaluation des SRI (Kazai 2003). Nous indiquerons aussi les résultats obtenus pour les fonctions d'agrégation stricte et généralisées ([3] et [4]), afin de pouvoir les comparer avec les résultats officiels d'INEX 2003.

[1] Les jugements de pertinence pour les requêtes de 2004 n'étaient pas encore disponibles au début de nos expérimentations

5. Expérimentations et résultats

Les expérimentations que nous présentons ici ont pour but d'évaluer l'impact des unités d'indexation choisies et des formules utilisées pour le calcul de la similarité des nœuds feuilles à la requête et pour le calcul de la pertinence des sous-arbres des documents.

5.1. Impact de l'unité d'indexation

Comme nous l'avons vu plus haut, le choix de l'unité d'indexation minimale est l'une des premières problématiques soulevée par les requêtes orientées contenu. Il est couramment répandu dans la littérature que ce choix implique la définition de l'unité d'information minimale qui pourra être retournée à l'utilisateur. Deux points de vues s'affrontent. Le premier prétend qu'indexer tous les nœuds présente peu d'intérêt, puisque dans le cadre d'une recherche à partir de simples mots-clés, des nœuds de type titre par exemple ne doivent pas être renvoyés par le SRI à l'utilisateur car ils ne sont pas porteurs d'information. Dans ce cas-là, afin de ne pas perdre l'information textuelle portée par ces nœuds, cette dernière est propagée jusqu'au premier nœud faisant partie des nœuds sélectionnés pour faire partie de l'index (Gövert *et al.*, 2002). Un autre point de vue serait au contraire d'indexer tous les nœuds feuilles, car cela a le double avantage d'automatiser complètement le processus d'indexation mais aussi de permettre la réutilisation de l'index pour des requêtes composées de conditions de structure, aussi spécifiques soient-elles. C'est cette vision que nous avons utilisée dans nos précédentes expérimentations avec le modèle XFIRM sur des requêtes CAS (Sauvagnat *et al.*, 2004).

Afin de confronter ces deux approches et de vérifier la faisabilité de la seconde dans le cadre de requêtes CO, deux index de la collection INEX ont été créés :

- Dans le premier, certaines balises sont éliminées de l'index et le texte de leurs éventuels nœuds feuilles descendants est propagé jusqu'au premier nœud faisant partie de la liste des nœuds « indexables ». Ceci est en fait équivalent à simplifier la structure de l'arbre du document. Le choix des types de nœuds à supprimer de l'index est fait automatiquement, en utilisant des statistiques sur la collection : les types de nœuds comptant en moyenne moins de 2 termes (une fois les mots vides supprimés) sont écartés de l'index. Cette condition sur le nombre de termes peut paraître faible, mais elle diminue de plus de 25% le nombre de nœuds de l'index par rapport à la seconde solution proposée. Les types de nœuds supprimés sont essentiellement ceux utilisés pour la présentation des documents (*italique, gras*, …).
- Dans le second index, toute la structure des documents est conservée.

Dans la suite des expérimentations, nous noterons ces index respectivement *IS (Index Simplifié)* et *IC (Index Complet)*.

5.2. Impact des formules utilisées pour le calcul de la pertinence des nœuds

Nous présentons ici trois séries d'expérimentations effectuées sur les deux index présentés ci-dessus. Ces expérimentations ont pour but d'évaluer l'impact de la formule utilisée pour le calcul du poids des nœuds feuilles (équation [1]) ainsi que diverses valeurs du paramètre α utilisé dans l'équation [2] (et ce afin d'estimer l'impact du paramètre *distance* dans la formule de propagation). Les formules évaluées pour le calcul de la similarité des nœuds feuilles à la requête sont dérivées de formules utilisées dans le cadre de la RI traditionnelle. Ces dernières sont transformées afin de s'adapter à la nouvelle granularité des informations. Elles utilisent ou non la taille des nœuds feuilles pour calculer leur similarité à la requête.

5.2.1. Utilisation d'une formule simple ne prenant pas en compte la taille des nœuds feuille

Une des premières fonctions testée pour le calcul du poids des nœuds feuilles est une adaptation de la mesure tf*idf , couramment utilisée en RI. Afin de s'adapter à la nouvelle granularité (on ne parle plus de documents mais de nœuds feuilles), nous définissons ainsi la notion d'*ief (Inverse Element Frequency)* :

$$ief_i = \log(\frac{N}{n+1}) + 1 \qquad [7]$$

Où n est le nombre de nœuds feuilles contenant le terme i et N est le nombre total de nœuds feuilles.

L'équation [1] s'écrit alors :

$$RSV_m(q, nf) = \sum_{i=1}^{n} w_i^q * w_i^{nf} = \sum_{i=1}^{n} (tf_i^q * ief_i) * (tf_i^{nf} * ief_i) \qquad [8]$$

où tf_i^q et tf_i^{nf} sont respectivement la fréquence du terme i dans la requête q et dans le nœud feuille nf.

Le tableau 1 présente les précisions moyennes obtenues par notre système en faisant varier les différentes valeurs de α dans l'équation [2]. Les précisions moyennes considérées concernent toutes les fonctions d'agrégation présentées dans la section 4 (*Avg*), ainsi que la fonction d'agrégation Stricte (*s* : équation [3]) et la fonction d'agrégation Généralisée (*g* : équation [4]), utilisées pour le classement officiel des résultats de la campagne d'évaluation 2003.

	α=0.6			**α=0.8**			**α=1**		
	Avg	***s***	***g***	***Avg***	***s***	***g***	***Avg***	***s***	***g***
IS	0.1154	0.1292	**0.1002**	0.1121	0.1228	0.0991	0.1066	0.1226	0 .0940
IC	**0.1163**	**0.1322**	0.0982	0.1137	0.1258	0.0989	0.1097	0.1235	0.0966

Tableau 1 : *Précisions moyennes obtenues avec la formule tf-ief en faitsant varier le paramètre α sur les deux index considérés.*

5.2.2. Utilisation de la taille des nœuds feuilles dans la formule de calcul de similarité

La seconde formule que nous avons adaptée est celle utilisée dans le moteur de recherche plein-texte Mercure (Boughanem *et al*., 2001). Contrairement à la formule précédente, cette formule tient compte de la taille du nœud feuille considéré . L'équation [1] devient :

$$RSV(q,nf) = \sum_{i=1}^{n} tf_i^{\,q} * \frac{tf_i^{\,nf} * (h_1 + h_2 * ief_i)}{h_3 + h_4 * \frac{l}{\Delta l} + h_5 * tf_i^{\,nf}} \qquad [9]$$

où tf_i et ief_i sont définis comme précédemment, l est le nombre de termes dans le nœud feuille nf, Δl est la taille moyenne des nœuds feuille de la collection et h_1, h_2, h_3, h_4, h_5 sont des constantes fixées expérimentalement à 0.2, 0.8, 0.2, 0.7, et 1.

Les résultats obtenus sont présentés dans le tableau ci-dessous.

	α=0.6			**α=0.8**			**α=1**		
	Avg	***s***	***g***	***Avg***	***s***	***g***	***Avg***	***s***	***g***
IS	**0.0873**	**0.1047**	0.0761	0.0869	0.1047	**0.0763**	0.0851	0.1033	0.0746
IC	0.0858	0.1070	0.0709	0.0870	0.1081	0.0735	0.0851	0.1044	0.0730

Tableau 2 : *Précisions moyennes obtenues avec la formule de Mercure en faisant varier le paramètre α sur les deux index considérés.*

La dernière formule évaluée pour l'équation [1] est une adaptation de la formule du célèbre formule du BM25 (Sparck-Jones *et al*., 2000). Cette formule tient aussi compte de la taille des nœuds feuilles pour l'évaluation de leur pertinence :

$$RSV(q,nf) = \sum_{i=1}^{n} \log\left(\frac{N - n + 0.5}{n + 0.5}\right) * \frac{(k_1 + 1) f_i}{K + f_i} \qquad [10]$$

où N est le nombre total de nœuds feuille de la collection, n est le nombre de nœuds feuilles contenant le terme i, tf_i est la fréquence du terme i dans le nœud feuille nf, $K = k_1*((1-b)+b*l) / \Delta l$, avec $k_1=1.2$ et $b=0.75$, l le nombre de termes dans nf et Δl le nombre moyen de termes dans les nœuds feuilles de la collection.

Les résultats obtenus sur les deux index et avec différentes valeurs de α sont présentés dans le tableau 3.

	α=0.6			**α=0.8**			**α=1**		
	Avg	***s***	***g***	***Avg***	***s***	***g***	***Avg***	***s***	***g***
IS	**0.0918**	0.1103	0.0781	0.0916	**0.1107**	**0.0782**	0.0885	0.1084	0.0760
IC	0.0858	0.1067	0.0708	0.0865	0.1077	0.0731	0.0837	0.1033	0.0717

Tableau 3 : *Précisions moyennes obtenues avec la formule du BM25 en faisant varier le paramètre α sur les deux index considérés.*

5.2.3. Conclusion

Les résultats obtenus ci-dessus laissent apparaître les points suivants :

- La fonction la plus simple (*tf-ief*) permet d'obtenir des résultats significativement meilleurs que les deux autres formules, quel que soit l'index considéré. L'utilisation de la taille des nœuds feuilles dans les formules de *Mercure* et du *BM25* semble trop privilégier les nœuds de petite taille, ce qui ne devrait pas être le cas (ces nœuds ne sont en effet pas porteurs d'information): ces formules, utiles dans le cas de recherche sur des documents entiers, ne paraissent donc pas appropriées pour des recherches sur des éléments (dans le cadre de notre méthode de propagation de la pertinence).
- La formule *tf-ief* est la seule a fonctionner mieux sur l'index complet (IC) que sur l'index simplifié (IS). Cette remarque rejoint ainsi la précédente : l'IS contenant moins de nœuds de petites tailles que l'IC, ces derniers ne peuvent pas être renvoyés par le système et donc les performances sont meilleures.
- Les différentes valeurs de α testées laissent apparaître l'importance du paramètre distance dans la formule de propagation (équation [3]). Le cas où α =1 correspond à effectuer une simple somme des poids des nœuds feuilles pour évaluer la pertinence d'un nœud donné et entraîne une baisse des performances. De manière générale, quel que soit l'index considéré (IS ou IC) et quelle que soit la formule de calcul des poids utilisées, les meilleurs résultats sont obtenus avec α =0.6 (ce qui laisse penser que le paramètre distance a plus d'importance dans le cadre de requêtes CO que dans le cadre de requêtes CAS, où les meilleurs résultats étaient obtenus pour α =0.9 (Sauvagnat et al., 2004)).

Notre modèle obtient donc des résultats optimaux en utilisant une formule simple de calcul du poids des nœuds feuilles (*tf-ief*) et une formule de propagation donnant de l'importance au paramètre de distance séparant les nœuds. Ces résultats, comparés au résultats officiels d'INEX 2003, nous auraient classé premier pour la mesure utilisant la fonction d'agrégation stricte (cf. Tableau 4) et dans les 10 premiers pour la fonction d'évaluation généralisée.

Rang	*Précision moyenne*	*Organisation*	*Identifiant du run*
	0.1322		***Xfirm : tf-ief / α=0.6***
1	0.1214	U. of Amsterdam	UamsI03-CO-lambda=0.20
2	0.1144	U. of Amsterdam	UamsI03-CO-lambda=0.5
3	0.1102	U. of Amsterdam	UamsI03-CO-lambda=0.9
4	0.1001	Universitt Duisburg-Essen	factor0.2
5	0.0952	IBM, Haifa Research lab	CO-TDB-With-No-Clustering

Tableau 4 : *Classement de notre système parmi les résultats officiels de la campagne d'évaluation INEX 2003 pour une fonction d'agrégation stricte*

L'index utilisé dans le cas de ces paramètres optimaux ne semble pas jouer un

rôle primordial, même si on constate des performances légèrement meilleures lorsque la recherche est faite sur l'index le plus complet (IC).

Cependant, quelque soit l'index utilisé, certains nœuds très petits peuvent être renvoyés par le système alors qu'ils ne sont pas informatifs (les nœuds titre par exemple, qui en moyenne font plus de deux mots et font donc encore partie de l'IS). Ces nœuds peuvent en effet avoir un score de pertinence élevé, car la pertinence que nous avons défini renverrait un nœud contenant les seuls termes de la requête comme réponse idéale. Afin de résoudre ce problème, nous introduisons dans la section suivante la notion d'informativité.

5.3. Le problème des nœuds de petite taille

Afin de traiter les nœuds de petite taille, nous calculons pour chaque nœud un score d'informativité, qui sera fonction de la pertinence de ce nœud à la requête, mais aussi de sa taille. Les nœuds ne sont alors plus renvoyés à l'utilisateur par ordre décroissant de pertinence mais par ordre décroissant d'informativité.

5.3.1. Une solution naïve

Une première solution simple est de mettre un seuil sur le nombre de termes que doit contenir un nœud pour pouvoir être renvoyé par le système. La formule [2] est alors redéfinie comme suit : *Soit un nœud n et* $nf_i \in [1..N]$ *l'ensemble de ses nœuds feuilles descendants ayant un poids non nul. Soit* l_i *la taille du nœud feuille* nf_i *(c'est à dire le nombre de termes qu'il contient) et L la somme des tailles des* nf_i*. Si L est plus petit qu'une certain seuil x, alors le nœud n est considéré comme non informatif.*

$$i_n = \begin{cases} p_n & \text{si } L > x \\ 0 & \text{sinon} \end{cases} \quad \text{Avec} \quad L = \sum_{i=1..N} l_i \ , \ \forall i / \mathrm{RSV}(q, nf_i) > 0 \qquad [11]$$

Dans nos expérimentations nous fixons *x=25*, cette valeur correspondant de manière intuitive au nombre de mots minimum que doit contenir un nœud pour être porteur d'information. Le tableau 6 montre les résultats obtenus pour les deux index avec l'équation [8] pour le calcul du poids des nœuds feuilles (tf-ief) et α =0.6 dans la fonction de propagation (formule [11]) .

	L=0			**L=25**			**Gain**		
	Avg	***s***	***g***	***Avg***	***s***	***g***	***Avg***	***s***	***g***
IS	0.1154	0.1292	**0.1002**	0.1072	0.1205	0.0921	-7.8%	-6.8%	-8.1%
IC	**0.1163**	**0.1322**	0.0982	0.0983	0.1123	0.0806	-15,5%	-15,1%	-18%

Tableau 5 : *Gain obtenu par introduction d'un seuil sur la taille*

On observe, de manière surprenante, une perte de performance lorsque le seuil L est utilisé. Des résultats similaires, non présentés ici pour cause de manque de place,

sont obtenus avec des valeurs plus petites de *x* (*5* et *10*). La perte de précision moyenne est cependant moins importante sur l'IS. Ceci peut être expliqué par le fait que l'IS possède moins de nœuds de petite taille et que donc moins de nœuds sont supprimés par le seuil. Les résultats sur l'IC restent surprenant et peuvent s'expliquer par le fait que des nœuds de (très) petite taille (comme des nœuds titre ou sous-titre par exemple) ont été jugés pertinents par certains participants d'INEX, qui ont considéré que même s'ils n'apportent pas d'information à l'utilisateur, leur similarité à la requête est grande.

Du point de vue de l'utilisateur pourtant, de tels nœuds devraient être moins bien classés par le SRI. Ceci n'implique cependant pas qu'ils ne sont d'aucune utilité. De manière intuitive, on peut penser que le concepteur d'un document les utilise souvent pour faire ressortir des informations importantes. Ils peuvent ainsi donner des indications précieuses sur la pertinence de leurs nœuds ancêtres. Les expérimentations présentées ci-dessous cherchent à vérifier cette affirmation.

5.3.2. Une solution prenant en compte l'information véhiculée par les nœuds de petite taille

Afin d'évaluer l'impact des poids des nœuds de petite taille dans notre modèle de propagation de la pertinence, nous introduisons les règles ci-dessous. Soit l_k la taille du nœud feuille nf_k et Δl la taille moyenne d'un nœud feuille.

- Si un nœud feuille nf_k est de petite taille (c'est à dire de taille inférieure à la moyenne) l'informativité i_{par} de son nœud parent *par* doit être faible :

$$Si\ lk < \Delta l\ et\ dist(n, nf_k)=1\ alors\ \ i_{par} = \frac{l_k}{\Delta l} * p_{par} \qquad [12]$$

avec p_{par} calculé d'après [2].

- Mais son score de similarité à la requête doit augmenter le score d'informativité de ses nœuds ancêtres *anc* :

$$i_{anc} = \sum_{k=1..n} \alpha^{dist(anc,nf_k)-1} * \log(\Delta l / l_k) * RSV(q, nf_k) \qquad [13]$$

De manière synthétique, le score d'informativité d'un nœud *n* est défini comme suit:

$$i_n = \sum_{k=1..N} \alpha^{dist(n,nf_k)-1} * \beta * RSV(q, nf_k) \quad \text{avec} \quad \beta = \begin{cases} l_k / \Delta l & \text{si } dist(n, nf_k) = 1 \text{ et } l_k < \Delta l \\ \log(\Delta l / l_k) & \text{si } dist(n, nf_k) > 1 \text{ et } l_k < \Delta l \\ 1 & \text{sinon} \end{cases} \qquad [14]$$

Les résultats obtenus avec ces nouvelles formules (qui ont été ajustées par expérimentations) sont décrits dans la tableau 6. Nous obtenons une amélioration des performances sur l'IC de l'ordre de 3%, alors que pour l'IS, les précisions moyennes diminuent. Ceci semble montrer que les nœuds de petites tailles sont utiles pour le calcul des poids de leurs nœuds ancêtres et que par conséquent il est préférable de les conserver dans l'index.

	Formules [8] et [2]			Formules [8] et [14]			Gain		
	Avg	***s***	***g***	***Avg***	***s***	***g***	***Avg***	***s***	***g***
IS	0.1154	0.1292	0.1002	0.1130	0.1259	0.0990	-2.1%	-2.5%	-1.2%
IC	0.1163	0.1322	0.0982	**0.1198**	**0.1356**	**0.1021**	**+3.0%**	**+2.6%**	**+4.0%**

Tableau 6 : *Gain obtenu dans les précisions moyennes par introduction de règles sur la taille des nœuds pendant le calcul des poids de pertinence*

D'une manière générale, les expérimentations effectuées laissent apparaître qu'il est plus intéressant de considérer la taille des nœuds dans le calcul de leur informativité que dans le calcul de leur similarité à la requête. Ces résultats demandent cependant à être confirmés sur d'autres requêtes et d'autres jugements de pertinence : les jugements utilisés ici ne sont en effet pas homogènes en ce qui concerne l'évaluation de la pertinence des nœuds de petite taille (doit-on juger la similarité à la requête ou la quantité d'information apportée?). De même, d'autres mesures d'évaluation doivent être prises en compte afin de gérer le problème de l'imbrication des nœuds résultats (Kazai *et al.*, 2004).

6. Conclusion et perspectives

Dans cet article, nous nous sommes intéressés au traitement des requêtes orientées contenu dans des corpus de documents XML. Afin de répondre à de telles requêtes, nous utilisons un modèle basé sur la propagation de la pertinence et reposant sur le système XFIRM. Nous avons cherché à évaluer l'impact dans la recherche du choix des unités d'indexation ainsi que l'impact des formules utilisées pour calculer la pertinence des sous-arbres (notamment l'utilisation lors de la propagation du paramètre modélisant la distance séparant les nœuds). Les premiers résultats montrent les bonnes performances de notre modèle par rapport aux résultats officiels de la campagne d'évaluation INEX, et ce en utilisant une formule de calcul de similarité simple, ne prenant pas en compte la taille des nœuds.

Cependant, le score de pertinence défini ne suffit pas à identifier les sous-arbres répondant de manière optimale à la requête, puisque la requête elle-même est considérée comme réponse idéale. Ainsi, des nœuds de petites tailles peuvent ainsi être renvoyés à l'utilisateur alors qu'ils ne sont pas informatifs. Nous introduisons alors la notion de score d'informativité, dans le calcul duquel les nœuds de petite taille, quoique eux-même non informatifs, ont une importance prépondérante. Les performances obtenues doivent être vérifiées sur les requêtes d' INEX 2004.

L'utilisation de la taille des nœuds semble ainsi jouer un rôle important dans le recherche des nœuds répondant à la requête de l'utilisateur. Le concepteur d'un document utilise des nœuds de petite taille pour faire ressortir des informations importantes : la sémantique des balises utilisées devrait aussi nous permettre d'augmenter les performances de notre modèle.

7. Bibliographie

Abolhassani M., Fuhr N., *Applying the divergence from Randomness approach for content-only search in XML documents*, Proceedings of ECIR 2004, Sunderland, p. 409-419.

Anh, V.N. Moffat, A. : *Compression and IR approach to XML retrieval*. In Proceedings of INEX 2002, Dagstuhl, Germany, 2002.

Boughanem, M., Chrisment, C., Tmar, M. *: Mercure and MercureFiltre applied for Web and Filtering Tasks at TREC 10*. In Proceedings of TREC 10, Maryland, USA, Nov. 2001.

Chiaramella Y., Mulhem P., Fourel F., *A Model for Multimedia Information Retrieval*, 1996, Technical report, FERMI ESPRIT BRA 8134, University of Glasgow.

Fuhr, N., Grossjohann, K., *XIRQL : a query language for information retrieval in XML documents*. In Proceedings of SIGIR 2001 : 172-180.

Grabs T., Scheck H.-J., *Flexible information retrieval from XML with PowerDB XML*, Proceedings in the First INEX Workshop, December 2002, p. 26-32.

Gövert N., Abolhassani M., Fuhr N., Grossjohann K., *Content-oriented {XML} retrieval with HyReX*, Proceedings of the first INEX Workshop, Dagstuhl, Germany, 2002.

Grust, T, "*Accelerating XPath Location Steps*". Proceedings of the 2002 ACM SIGMOD Conference on Management of Data, Madison, Wisconsin, USA, ACM Press, 2002.

Kazai G., *Report of the INEX 2003 Metrics working group*, Proceedings of INEX 2003, Dagstuhl, Germany, December 2003.

Kazai G., Lalmas M., de Vries A.P., *The overlap problem in Content-oriented XML retrieval evaluation*, Proceedings of SIGIR 2004, Sheffield, England, July 2004, p. 72-79.

Lalmas M., *Dempster-Shafer's theory of evidence applied to structured documents: modeling uncertainty*, Proceedings of SIGIR'97, Philadelphia, USA}, 1997,p. 110-118.

Mass, Y., Mandelbrod, M. , *Retrieving the most relevant XML component*, Proceedings of INEX 2003, Germany, December 2003, p. 53-58.

Ogilvie P., Callan J., *Using Language Models for Flat Text Queries in XML Retrieval*, Proceedings of INEX 2003 Workshop, Dagstuhl, Germany, December 2003, p. 12-18.

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

Salton G., Allan J., Buckley C., *Approaches to passage retrieval in full text information*, Proceedings of SIGIR'93, Pittsburgh, PA, 1993.

Sparck-Jones K., Walker S., Robertson S.E. : *A probabilistic model of information retrieval/ development and comparative experiments. Parts 1&2*. IPM, 36(6) :779-840, 2000.

Sauvagnat K., *XFIRM, un modèle flexible de Recherche d'Information pour le stockage et l'interrogation de documents XML*, Actes de CORIA'04 , Toulouse, 2004, p. 121-142.

Sauvagnat K., Boughanem M., Chrisment C. *: Searching XML documents using relevance propagation*. In Proceedings of SPIRE 04, Padova, Italie, octobre 2004.

Sigurbjörnsson B., Kamps J., de~Rijke M., *An element-based approach to XML retrieval*, Proceedings of INEX 2003 workshop, Dagstuhl, Germany, december 2003.