Recherche approchée d'information dans une base de documents semi-structurés

Fusion d'informations structurées et textuelles

Eugen Popovici – Gildas Ménier – Pierre-François Marteau

VALORIA
Université de Bretagne Sud
Rue Yves Mainguy, 56 000 Vannes
{eugen.popovici, gildas.menier, pierre-francois.marteau}@univ-ubs.fr

RÉSUMÉ. *Nous proposons des algorithmes dédiés à l'indexation et à la recherche approximative d'information dans les bases de données hétérogènes semi-structurées XML. Le modèle d'indexation proposé est adapté à la recherche de contenu textuel dans les contextes XML définis par les structures d'arbres. Les mécanismes de recherche approchée mis en œuvre s'appuient sur une distance de Levenshtein modifiée et des heuristiques de fusion d'information. Une implémentation exploitant simultanément l'information structurée, i.e. l'arborescence des éléments XML, et le contenu des documents indexés est décrite. Les performances obtenues dans le cadre de la campagne d'évaluation INEX 2005 sont présentées et analysées.*

ABSTRACT. *We propose specific data structures designed to the indexing and retrieval of information elements in heterogeneous XML data bases. The indexing scheme is well suited to the management of various contextual searches, expressed either at a structural level or at an information content level. The approximate search mechanisms are based on a modified Levenshtein editing distance and information fusion heuristics. The implementation described highlights the mixing of structured information presented as field/value instances and free text elements. The retrieval performances of the proposed approach are evaluated within the INEX 2005 evaluation campaign.*

MOTS-CLÉS : *XML, Base de données hétérogènes, recherche et extraction d'information, fusion d'information, distance d'édition de Levenshtein, opérateurs de recherche, INEX 2005.*

KEYWORDS: *XML, Heterogeneous data base, information retrieval, information fusion, Levenshtein edition distance, heuristic based operators, INEX 2005.*

1. Introduction

La société de l'information engendre un nombre considérable de documents hétérogènes, tant en forme qu'en nature. Face à cette hétérogénéité, l'émergence d'XML traduit un effort croissant qui vise à normaliser la présentation et l'archivage de l'information. Dans nombre de domaines de spécialité des structures de documents standardisés (*Document Type Definition*) et des langages d'interrogation *centrés données* (*XPath*, *XQuery*) sont proposées pour améliorer le partage, l'échange d'information et l'interopérabilité des applications. Cet effort de normalisation, s'il permet d'espérer une amélioration d'accès aux données disponibles en ligne ne résout pas cependant l'ensemble des difficultés concernant la qualité de l'information accessible. Pour répondre à ce besoin, les approches *centrées documents* (Fuhr *et al.*, 2004) (Carmel *et al.*, 2003) (Theobald *et al.*, 2002) (Schlieder, 2002) (Sauvagnat *et al.*, 2004) visent le développement d'algorithmes de recherche d'information semi-structurée basés sur une exploitation des contextes structurels d'occurrence des éléments informationnels et sur des heuristiques de classement des réponses.

W3C propose *XQuery 1.0 and XPath 2.0 Full-Text* (Amer-Yahia *et al.*, 2005) s'appuyant sur les approches « centrées données » et sur des techniques des classements du contenu textuel de documents. Cependant, les contraintes structurelles pour la portée de la recherche d'information textuelle doivent être complètement spécifiées et leur interprétation reste toujours stricte. Du fait de l'hétérogénéité des structures et des contenus des documents disponibles au format XML, il n'est pas envisageable d'imaginer que l'utilisateur puisse disposer d'une connaissance suffisante sur la structure des documents accessibles pour orienter sa recherche d'information.

Différentes approches qui traitent de manière approximative les contraintes structurelles ont été proposées pour rentabiliser l'effort de structuration dans le cadre de la recherche d'information XML. Ces approches sont basées sur des relations sémantiques (Theobald *et al.*, 2002) ou sur différents degrés de relaxation concernant l'organisation des informations structurelles. Parmi ces dernières on peut citer des techniques basées sur des correspondances d'arbres (Schlieder, 2002) (Amer-Yahia *et al.*, 2004) ou des chemins (Carmel *et al.*, 2003).

Dans cet article, nous proposons plus précisément d'évaluer l'amélioration de l'accès à l'information apportée par des mécanismes d'alignement approximatif des éléments structurels (Menier *et al.*, 2002) (Marteau *et al.*, 2003). Nous décrivons également un modèle de recherche d'information susceptible de s'appuyer sur les éléments structurants des documents semi-structurés pour affiner l'analyse et la recherche des contenus non structurés. Les résultats concernant la pertinence de l'approche obtenus dans le cadre de INEX 2005 (*INitiative for the Evaluation of XML Retrieval*) sont finalement présentés et analysés.

2. Contexte XML

Dans la structure de l'arbre codant un document XML, chaque nœud n peut être rattaché à la racine de l'arbre n_0 par l'intermédiaire du chemin $p(n)$. Ce chemin est une séquence ordonnée de nœuds d'éléments XML n_i éventuellement associes à un ensemble non ordonné de couples <attribut, valeur> $A(n_i)$. Il constitue le contexte d'occurrence du nœud n (le nœud extrémité du chemin $p(n)$) dans le document considéré. Dans le cas où un nœud[1] de l'arbre peut être décomposé en sous éléments textuels $\{M_i\}$ (mot, lemme ou chaîne de caractères), chaque sous élément M_i est considéré comme un nœud terminal ne possédant pas d'attribut. Dans ce cas, on considèrera que le contexte d'occurrence de M_i est le chemin suivant :

$$p(n)=<n_0, A(n_0)> <n_1, A(n_1)> \ldots <n, A(n_n)> <M_i, \emptyset> .$$

3. Paradigme de recherche d'information

Plutôt que l'exploitation de principes d'alignement approximatif d'arbres (Schlieder, 2002) (Amer-Yahia *et al.*, 2004) de haute complexité algorithmique on recherche plutôt ici des principes de recherche de bas niveau capables d'aligner approximativement des sous chemins de type $p(n)$ contenus dans les arbres associés aux documents indexés avec un ou des sous chemin $p(r)$ contenus dans la spécification d'une requête. Nous recherchons à définir un langage de requête plus évolué, qui intègre des expressions de recherche élémentaires $\{p(n)\}$ et des expressions plus complexes, constituées à partir de l'assemblage d'expressions élémentaires.

3.1. Alignement approximatif des sous-structures p(n)

Soit R une requête élémentaire exprimée sous la forme d'un mot M_i associée à un chemin p^R pour lequel les couples <attributs, valeur> sont remplacés par des conditions ou contraintes Cd qui portent sur les valeurs d'attributs.

$$p^R(n)=<n_0, Cd(n_0)> <n_1, Cd(n_1)> \ldots <n, Cd(n_n)> <M_i, \emptyset>$$

Nous proposons d'évaluer la pertinence des réponses pour une requête élémentaire R par la similarité entre le chemin p^R représentant la requête et l'ensemble des chemins indexés associé aux réponses de la manière suivante :

$$\sigma(p^R, p_i^{R,D}) = 1 / (1+\delta_L (p^R, \{ p_i^{R,D}\}))$$

où δ_L s'apparente à une pseudo distance d'édition de type Levenshtein (Levenshtein, 1966) et $\{p_i^{R,D}\}$ représente l'ensemble des chemins issus de la racine de l'arbre T^D

[1] Feuille avec du contenu textuel ou nœud avec du contenu mixte.

associé au document D et dont le nœud extrémité correspond de manière stricte ou approximative au nœud extrémité du chemin p^R correspondant à la requête.

La complexité d'un tel algorithme est : $O(l(p^R)\cdot prf(T^D)\cdot|\{p_i^{R,D}\}|)$, avec $|\{p_i^{R,D}\}|$ le cardinal de l'ensemble des chemins $\{p_i^{R,D}\}$, $l(p^R)$ la longueur du chemin p^R et $prf(T^D)$ la profondeur de l'arbre T^D.

3.2. Distance d'édition entre deux chemins p^R et $p_i^{R,D}$: $\delta_L(p^R, p_i^{R,D})$

Soit p^R le chemin correspondant à la requête élémentaire R et $\{p_i^{R,D}\}$ l'ensemble des chemins retenus pour la recherche de similarité. Nous proposons une pseudo-distance[2] d'édition (Wagner *et al.*, 1974) en exploitant une matrice de coût paramétrable pour calculer la similarité entre un chemin $p_i^{R,D}$ et le chemin correspondant à la requête élémentaire p^R (Ménier *et al.*, 2002). Cette distance détermine la transformation de coût minimal qui permet de transformer $p_i^{R,D}$ en p^R.

Les opérations élémentaires d'édition exploitées par cette distance et la fonction de coût c qui leur est associée sont définies de la manière suivante :

– **La substitution** : un nœud n^R du chemin p^R est remplacé par (ou aligné sur) un nœud n du chemin $p_i^{R,D}$ avec un coût élémentaire $c(n^R,n)$ que l'on considère pouvoir varier entre 0 (correspondance parfaite) et $c^{max} = 1$ (aucune correspondance).

– **L'élimination** : un nœud n du chemin $p_i^{R,D}$ est éliminé avec un coût $c(n,\varepsilon)$ que l'on considère pouvoir varier entre 0 et $c^{max} = 1$.

– **L'insertion** : un nœud n est inséré dans le chemin $p_i^{R,D}$ avec un coût $c(\varepsilon,n)$ que l'on considère pouvoir varier entre 0 et $c^{max} = 1$.

Soit l'ensemble $\mathcal{T}_{p_i^{R,D} \to p^R}$ des transformations décomposables en une séquence de transformations élémentaires (insertion, élimination ou substitution) et permettant de transformer le chemin $p_i^{R,D}$ en p^R. A toute transformation τ de l'ensemble $\mathcal{T}_{p_i^{R,D} \to p^R}$ on associe un coût global noté $C(\tau)$ assimilé à la somme des coûts des transformations élémentaires qui composent τ. L'algorithme de Wagner&Fisher (Wagner *et al.*, 1974) permet de déterminer parmi les transformations de l'ensemble $\mathcal{T}_{p_i^{R,D} \to p^R}$ la transformation de coût minimum notée τ^* (transformation optimale) avec une complexité $O(l(p_i^{R,D})\cdot l(p^R))$ comme précisé précédemment.

Soit C_i^* le coût minimum d'alignement du chemin $p_i^{R,D}$ sur le chemin p^R :

$$\delta_L(p^R, p_i^{R,D}) = C_i^* = \underset{\tau \in \mathcal{T}_{p_i^{R,D} \to p^R}}{Min} C(\tau) = C(\tau^*) .$$

Pour p^R et $p_i^{R,D}$ donnés, la similarité $\sigma(p^R, p_i^{R,D}) \to 0$ quand le nombre des noeuds différents augmente. Naturellement, pour une concordance totale (mêmes éléments XML avec conditions satisfaites sur les attributs) on aura un coût minimum $C_i^* = 0$ et une similarité maximale $\sigma(p^R, p_i^{R,D}) = 1$.

[2] On parle de pseudo distance dans la mesure où la propriété de symétrie n'est pas assurée quelle que soit la fonction de coût c envisagée.

3.3. Requêtes complexes

Des requêtes complexes sont construites à partir de requêtes élémentaires et d'opérateurs de composition booléens ou relevant d'heuristiques spécifiques : à ce jour, nous proposons huit opérateurs pour la construction des requêtes hors prise en compte des attributs. Les opérateurs pour la spécification des contraintes portant sur les attributs sont définis et détaillés dans (Marteau *et al.*, 2003). Les opérateurs utilisés pour la construction du langage de requête sont les suivants :

– les opérateurs booléens ou ensemblistes n-aires {*or*, *and*} qui permettent de rechercher une liste d'arguments dans un même contexte XML,

– l'opérateur ensembliste binaire {*without*} qui permet d'éliminer de la liste des résultats en cours, les résultats d'une autre requête,

– l'opérateur permettant de rechercher une séquence d'arguments {*seq*} : par exemple, (seq message * erreur) recherchera les éléments contenant les mots 'message' suivi par un mot quelconque (*), suivi par le mot 'erreur',

– l'opérateurs de recherche en contexte : {*in*} qui permet de rechercher une liste d'arguments respectivement dans un contexte XML déterminé,

– l'opérateur de recherche structurelle approchée {*in+*} permettant de fusionner par le biais d'une fonction d'agrégation linéaire pondérée les valeurs de similarité obtenues sur les informations structurelles et sur le contenu,

– l'opérateur {*same+*} permettant de pondérer les arguments de l'opérateur en fonction d'une heuristique prédéfinie, de type TFIDF (*Term Frequency by Inverse Document Frequency*) (Salton *et al.*, 1988) par exemple..

– l'opérateur ensembliste binaire {*filter*} permettant de filtrer un ensemble d'éléments résultats à partir d'un autre ensemble de résultats tenant compte de la pertinence associée à leur contexte global d'occurrence.

Le système analyse une requête R et produit un ensemble de réponses pondérées par un degré de pertinence. Soit $r(R) = \{ (e_i, v_i) \}$ l'ensemble réponse produit, avec e_i un élément de réponse et v_i une valeur numérique prise dans $[0..1]$ indiquant sa pertinence par rapport à la requête.

Une requête complexe est en fait un arbre de requêtes, dans lequel l'ensemble final est fabriqué par fusion (ou filtrage) successive des ensembles crées par les requêtes filles R_n. Une requête fille peut être elle-même une requête complexe, ou bien une requête simple qui ne met en œuvre qu'un mot associé à un contexte d'occurrence recherché p^R ou un mot seul (M_i). Ces combinaisons ensemblistes font intervenir des opérateurs de modification de pertinence :

$r(or (R_0,\ldots R_n)) = \{ (e_i, v_i) \}$ avec $v_i = \underset{k}{Max}(v_k)$ tq $(e_i, v_k) \in \bigcup_j^n r(R_j)$;

$r(and (R_0,\ldots R_n)) = \{ (e_i, v_i) \}$ avec $v_i = \underset{k}{Min}(v_k)$ tq $(e_i, v_k) \in \bigcup_j^n r(R_j)$;

$r(without (R_0, R_1)) = \{ (e_i, v_i) \}$ avec $v_i = Max(0, v_i^0 - v_i^1)$ tq

$(e_i, v_i^0) \in r(R_0)$ et $(e_i, v_i^1) \in r(R_1)$;

r(seq (M₀,…Mₙ)) = { (eᵢ,vᵢ) }, $v_i = 1$ si la séquence $M_0..M_n$ existe dans l'élément e_i, *0* si non ;

r(in (path , R₀,…Rₙ)) = { (eᵢ,vᵢ) } avec $v_i = Min\{ \underset{k}{Min}\ (v_k), \Delta(path, p(e_i))\}$ tq $(e_i, v_k) \in \bigcup_j^n r(R_j)$; ou *p(e)* le chemin de l'élément *e* ; et Δ *(path, p(e))=1* ssi *path==p(e)*, *0* si non ;

(in+ (path, R₀,…Rₙ)) = { (eᵢ,vᵢ) } avec $v_i = \beta.\Delta(path, p(e_i)) + (1-\beta).\underset{k}{Min}(v_k)$ tq $(e_i, v_k) \in \bigcup_j^n r(R_j)$, ou Δ *(path, p(e))* = σ *(path, p(e))* la similarité structurelle entre les deux chemins (section 3.1) et $\beta \in [0..1]$ un facteur permettant de pondérer l'importance des informations sur la structure des documents vis-à-vis de leur contenu ;

r(same+ (R₀,…Rₙ)) = { (eᵢ,vᵢ) } avec $v_i = \tau \cdot \sum_k \lambda_k . v_k$ tq $(e_i, v_k) \in \bigcup_j^n r(R_j)$;

ou λ_k est un facteur de pondération qui caractérise le pouvoir discriminant de l'élément de réponse (e_i, v_k) dans la collection :

$\lambda_k = 1 - log((1+ N^{D(e_i,v_k)}) / (1+ N^D))$; ou $N^{D(e_i,v_k)}$ le nombre de documents contenant l'élément de réponse (e_i, v_k) ; N^D le nombre total de documents ; et τ une constante de normalisation $\tau = 1 / \Sigma_k(\lambda_k)$;

Soit e^D un élément de réponse descendant du document *D*.

r(filter (R₀, R₁)) = { (eᵢ^D, vᵢ) } avec $v_i = (v_i^1 + \underset{j}{Max}(v_j^0))/2$ tq $\forall_D \forall_i (e_i^D, v_i^1) \in r(R_1)$ et $\forall_j (e_j^D, v_j^0) \in r(R_0)$.

4. Évaluations

L'approche décrite ci-dessus a été récemment évaluée lors de la campagne INEX 2005 dans le cadre de la tâche *VVCAS* (Popovici *et al.*, 2005). Afin d'évaluer la performance des différents SRI la collection de test *inex-1.8* est utilisée pour répondre à un jeu de requêtes portant sur la structure et le contenu de documents (*CAS*). Une requête *CAS* exprime deux types de contraintes structurelles : celles qui spécifient les éléments XML dans lesquels il s'agit de rechercher l'information (les éléments supports) et celles qui spécifient les éléments XML qu'il faut retourner (les éléments cibles). Pour la tâche *VVCAS* les contraintes structurelles sur les éléments supports et les éléments cibles sont interprétées comme vagues.

4.1. Les données expérimentales

La collection de test *inex-1.8* contient 16819 articles en format XML provenant de 24 revues de *IEEE Computer Society* publiées entre 1995-2004. Cette base totalise un volume de 725 Mo dans sa forme canonique. La collection est considérée sans attributs ou avec des valeurs d'attributs sans intérêt pratique pour les utilisateurs (Trotman *et al.*, 2004 – « Une note sur les attributs »). Elle contient 141 noms d'éléments XML différents[3] composant 7948 contextes XML uniques en ignorant les attributs et les valeurs des attributs. La longueur maximale d'un contexte est de 20 nœuds, avec une longueur moyenne de 8 nœuds.

4.2. Le jeu de requêtes

Le jeu de requêtes *INEX 2005 CAS* est formé de 47 requêtes contenant des références explicites sur la structure de documents. Par exemple, la requête « 275 » (Figure 1) recherche des *section*s sur la découverte des patterns fréquents dans des *articles* contenant un *résumé* sur la fouille des données.

Parmi ces requêtes, 30 sont des requêtes subordonnées de la forme //A[B][4], 6 sont des requêtes indépendantes de la forme //A[B] et 11 sont des requêtes complexes de la forme //A[B]//C[D][5] constituées des requêtes subordonnées. Seulement 17 requêtes – les requêtes complexes et les requêtes indépendantes – sont utilisées pour l'évaluation dans la campagne INEX 2005.

```
//article[about(.//abs, "data mining")]//sec[about(., "frequent itemsets")]
```

Figure 1. *Un exemple de requête complexe sur la structure et le contenu (CAS topic 275) du jeu de requêtes INEX 2005 CAS exprimée en NEXI (Trotman et al., 2004).*

4.3. Mesures de pertinence

Les jugements de pertinence pour chaque requête sont effectués par les différents participants selon deux dimensions : *l'exhaustivité* et *la spécificité*. Un élément est exhaustif s'il contient toutes les informations requises par la requête ; un élément est

[3] Les statistiques sont calculées du point de vue du système d'indexation. C'est-à-dire, en utilisant des classes de nom d'éléments XML équivalentes pour les *Paragraphs*, *Sections*, *Lists*, et *Headings* comme défini en (Trotman *et al.*, 2004). De la même façon, les contextes XML associés aux éléments XML vides ne comptent pas dans les statistiques.
[4] //A[B] = retourne des éléments A à propos de B (Sigurbjörnsson *et al.*, 2005).
[5] //A[B]//C[D] = retourne des éléments C descendants de A, où A est à propos de B et C est à propos de D (Sigurbjörnsson *et al.*, 2005).

spécifique si tout son contenu concerne la requête. Ces deux dimensions sont agrégées en utilisant deux types de fonction d'agrégation : *aggrégation stricte* (uniquement les réponses « parfaites » - complètement exhaustives et complètement spécifiques sont prises en compte) et *aggrégation généralisée* (qui récompense aussi des réponses partiellement pertinentes).

Les mesures officielles utilisées dans INEX 2005 sont : le gain cumulé étendu normalisé *(nxCG)* et la courbe effort-précision/gain-rappel *(ep/gr).*

Pour un rang *i* donné, la valeur *nxCG[i]* reflète le gain relatif que l'utilisateur a accumulé jusqu'à ce rang par rapport au gain qu'il aurait pu obtenir si le système avait produit un classement optimal de réponses. *MAnxCG[i]* représente la moyenne des valeurs obtenues pour nxCG de 1 à *i*.

L'effort-précision (*ep*) à un niveau donné de gain-rappel (*gr*) indique l'effort nécessaire (estimé en nombre de réponses visitées) à un utilisateur parcourant les réponses retournées par un système, relatif à l'effort nécessaire pour accumuler le même niveau de gain-rappel utilisant un classement de réponses idéal. *MAep* représente la moyenne non interpolée des valeurs obtenues pour *ep* à des niveaux « naturels » de *gr* (i.e. à chaque réponse pertinente retournée) .

Les détails sur les métriques utilisées pour l'évaluations dans le cadre de INEX 2005 sont disponibles dans (Kazai *et al.*, 2005).

4.4. Le processus d'indexation

La base de documents XML est transformée dans sa forme canonique[6]. Les mots fréquents et jugés non pertinents (les mots vides) sont éliminés. Au moment de l'indexation les mots sont *stemmés* (i.e. réduits à leur pseudo-radicaux). On indexe seulement les termes *alphanumériques* comme défini en (Trotman *et al.*, 2004). Les nombres, les attributs, les valeurs des attributs et les éléments XML vides ne sont pas indexés. Le modèle d'indexation est basé sur des listes inverses adaptées pour la prise en compte des contextes XML *p(n)* et sur un schéma d'étiquetage des nœuds permettant de résoudre les relations d'héritage dans l'arbre. Ce modèle est implémenté utilisant une combinaison de structures de données de type BTree et RECNO de Berkeley DB[7]. Le temps d'indexation (sur un Pentium IV- 2,4 GHz, avec 1,5 Go de mémoire RAM) pour un volume de 725 Mo de données est d'environ 60 minutes et respecte une évolution quasi linéaire. La taille de l'index obtenu représente 1,28 fois le volume des données initiales.

[6] Canonical XML Processor, http://www.elcel.com/products/xmlcanon.html
[7] http://www.sleepycat.com

4.5. Pondérations du schéma de similarité structurelle pour INEX 2005

Dans le langage officiel d'INEX 2005 – NEXI (Trotman *et al.*, 2004), le chemin associé a une requête est défini comme une séquence de nœuds parmi lesquels la seule relation acceptée est la relation de descendance. Par conséquence le chemin p^R est modélisé comme une *sous séquence* de contextes indexés $p_i^{R,D}$, où une sous séquence ne doit pas nécessairement être constituée de nœuds contigus. Pour s'adapter à cette approche, on permet la suppression des nœuds dans les contextes indexés sans pénalisation en affectant le coût d'élimination d'un nœud dans la distance d'édition à 0 : $c(n,\varepsilon) = 0$, $c(\varepsilon,n)=\xi$, $c(n^R,n)= \xi$ si $n^R \neq n$ où 0 si $n^R=n$.

La similarité structurelle basée sur le schéma de pondération ci-dessus modélise un utilisateur ayant une connaissance précise mais incomplète sur le nom des éléments XML indexés et sur leurs relations de descendance. Elle prend en compte : i) l'ordre d'occurrence des éléments correspondant entre la requête p^R et le contexte indexé $p_i^{R,D}$ ii) et le nombre des éléments XML sans correspondant spécifié dans la requête p^R.

4.6. Stratégies de recherche

Nous évaluons deux stratégies pour la recherche du contenu textuel : *COCAS_SAMEPLUS* (utilisant l'opérateur *same+*) *et COCAS_SEQ (*qui emploie une recherche séquentielle stricte (*seq*) à l'intérieur de l'opérateur *same+*) ; ainsi que l'effet de la prise en compte des informations structurelles sur les deux stratégies. (*VVCAS_SAMEPLUS* et *VVCAS_SEQ*).

Une recherche séquentielle stricte (*seq*) retourne uniquement des éléments contenant tous les mots recherchés en séquence avec des indices consécutifs dans le texte (ignorant les termes fréquents). Une recherche basé sur l'opérateur *same+* va retourner aussi des éléments contenant seulement un sous-ensemble des termes recherchés. Les résultats sont classés en prenant en compte : i) le nombre de termes différents recherchés trouvés à l'intérieur de l'élément ii) le caractère discriminant (i.e. IDF) des termes recherchés dans la collection. Cette stratégie n'impose pas de contraintes sur l'ordre d'occurrence de termes recherchés.

Les deux stratégies (*VVCAS_SAMEPLUS* et *VVCAS_SEQ*) prennent en compte les informations explicitées sur les contextes XML par le biais de l'opérateur *in+* (section 3.3) avec $\beta = 0,5$. Pour *COCAS_SEQ* et *COCAS_SAMEPLUS* ces informations sont complètement ignorées

Chaque stratégie est basée sur un jeu de requêtes différent obtenu d'une manière automatique à partir du jeu de requêtes *INEX 2005 CAS* exprimé dans le langage NEXI (Trotman *et al.*, 2004). Selon les différentes interprétations, les transformations utilisent un ensemble spécifique d'opérateurs pour la description des

requêtes complexes. Un exemple de traduction pour la requête complexe 275 (Figure 1.) selon les différentes stratégies de recherche est présente dans le tableau 1.

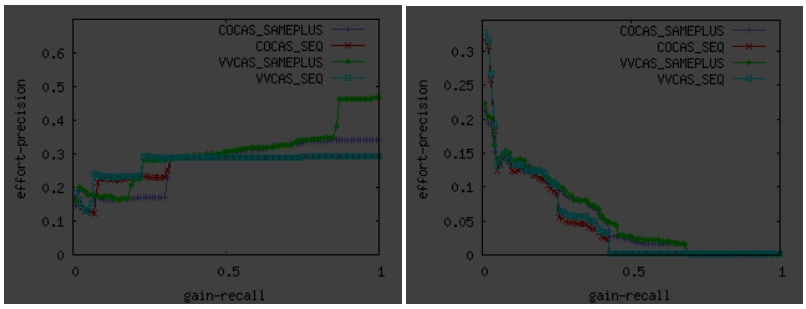
VVCAS_SEQ	`(FILTER (IN+ [/article/abs/] (SAME+ (SEQ data mining))) (IN+ [/article/sec/] (SAME+ (SEQ frequent itemsets))))`
COCAS_SEQ	`(FILTER (SAME+ (SEQ data mining)) (SAME+ (SEQ frequent itemsets)))`
VVCAS _SAMEPLUS	`(FILTER (IN+ [/article/abs/] (SAME+ data mining)) (IN+ [/article/sec/] (SAME+ frequent itemsets)))`
COCAS _SAMEPLUS	`(FILTER (SAME+ data mining) (SAME+ frequent itemsets))`

Tableau 1. *Traduction de la requête complexe 275 utilisant des opérateurs du langage récursif (section 3.3) selon la stratégie de recherche adoptée.*

4.7. Résultats expérimentaux

Dans la suite, nous présentons des résultats portant sur la qualité des réponses retournées par le système selon les différentes stratégies de recherche décrites, comparés aux résultats officiels pour la tâche *VVCAS* de INEX 2005.

Les métriques rapportées sont les mesures officielles utilisée dans INEX 2005 : le gain cumulé étendu normalisé *(nxCG)* et la courbe effort-précision/gain-rappel *(ep/gr)* (Figure 2, Tableau 2) ; calculées avec les paramètres *overlap=off*[8] et *quant.=strict*[9].



a) *b)*

Figure 2. *a) nxCG (overlap=off, quant.=strict) ; b) ep/gr (overlap=off, quant.=strict) pour les différentes stratégies de recherche et les résultats officiels de la campagne d'évaluation INEX 2005, tâche VVCAS.*

[8] les réponses entrelacées sont autorisées.
[9] seulement les éléments complètement exhaustives et spécifiques sont prises en compte.

	nxCG				ep/gr				
	@10	@25	@50	MAnxCG@1500	0,01/@15	0,02/@30	0,03/@45	0,1/@150	MAep
COCAS_SAMEPLUS	0.1222	0.1933	0.1822	0.273	0.2126	0.1945	0.1936	0.1338	0.0525
COCAS_SEQ	0.1444	0.16	0.1273	0.2647	0.3161	0.3084	0.2578	0.124	0.0448
VVCAS_SAMEPLUS	0.1444	**0.2022**	**0.1889**	**0.304**	0.2235	0.206	0.2013	**0.1389**	**0.0558**
VVCAS_SEQ	**0.1667**	0.1689	0.1339	0.2724	**0.327**	**0.3169**	**0.2673**	0.133	0.0487

Tableau 2. *nxCG[i] pour un rang i donné ; L'effort-précision (ep) à un niveau donné de gain-rappel ; calculées avec (Overlap=off, Quant=strict) ;*

En analysant les résultats obtenus comparés avec les résultats officiels de INEX 2005 on en déduit : i) que le système a une bonne précision pour des valeurs de rappel peu élevées. En particulier, les meilleures valeurs rapportées pour *nxCG@{10,25,50} (overlap=off, quant.=strict)* (Tableau 2) peuvent nous classer (non officiellement) parmi les deux meilleurs (sur 28) soumissions pour la tâche VVCAS (Figure 2.a). ; ii) La meilleure performance globale est obtenu par la stratégie *VVCAS_SAMEPLUS* sur la mesure *ep/gr(overlap=off, quant=strict)* avec *MAep = 0,0558*. Cette performance est équivalente à une cinquième place non officielle (Figure 2.b). ; iii) les informations sur la structure des documents et conjointement la distance d'édition proposée sur les chemins XML augmente la pertinence des réponses retournées par le système. Le gain obtenu sur la mesure *MAep* est de 6,28% pour *VVCAS_SAMEPLUS* et de 8,7% pour VVCAS_SEQ (Tableau 2). et iv) la stratégie de recherche séquentielle approximative conduit à une perte de précision (*ep*) d'environ 32% pour les premiers 15 résultats retournés en comparaison avec l'approche stricte, mais cette stratégie augmente la qualité globale des résultats (le gain sur *MAep* est de 14,57% pour VVCAS_SAMEPLUS et de 17,18% pour COCAS_SAMEPLUS*)*. Ce comportement est vérifié par les valeurs rapportées pour *nxCG* (Tableau 2).

5. Conclusion

Nous avons présenté et évalué des algorithmes de recherche approximative pour la recherche des informations enfouies dans des bases de documents XML. La similarité entre une requête et la structure d'un document est évaluée grâce à l'utilisation d'une distance de Levenshtein modifiée qui intègre des heuristiques et pondérations spécifiques permettant de s'affranchir d'une connaissance complète ou rigoureuse de la structure des documents exploités.

Les résultats obtenus dans la campagne d'évaluation INEX 2005 pour la tâche VVCAS ont été présentés et analysés. Les résultats sont encourageants, particulièrement pour les premières réponses retournées par le système. Ce fait est important car les premières réponses ont une grande probabilité d'être consultées par les utilisateurs.

6. References

Amer-Yahia S., Botev C., Buxton S., Case P., Doerre J., McBeath D., Rys M., Shanmugasundaram J., « XQuery 1.0 and XPath 2.0 Full-Text W3C Working Draft », 3 November 2005

Amer-Yahia S., Lakshmanan L., Pandit S., « FleXPath: Flexible Structure and Full-Text Querying for XML », SIGMOD 2004.

Carmel D., Maarek Y. S., Mandelbrod M., Mass Y. and Soffer A., « Searching XML documents via XML fragments », *SIGIR 2003*, Toronto, Canada pp. 151-158.

Fuhr N., Groβjohann K., XIRQL: « An XML query language based on information retrieval concepts », *TOIS*, v.22 n.2, p.313-356, April 2004.

INEX, INitiative for the Evaluation of XML Retrieval, http://inex.is.informatik.uni-duisburg.de/2005/news.html, April 2005 - December 2005.

Kazai G., « Lalmas M., INEX 2005 Evaluation Metrics », *Pre-proceedings of INEX 2005 Workshop*, 2005, p. 401-406.

Levenshtein A., « Binary Codes Capable of Correcting Deletions, Insertions and Reversals », *Sov.Phy. Dohl*. Vol.10, P.707-710, 1966

Marteau P.F., Ménier G., « Alignement approximatif d'arbres pour la recherche d'information en contexte dan les données XML hétérogènes », *Technique et sciences informatiques*. Volume 22 – n°7-8/2003, p. 1011-1034.

Ménier G., Marteau P.F., « Information retrieval in heterogeneous XML knowledge bases », *IPMU* , IEEE, 1-5 July, Annecy, France, 2002.

Salton G. and Buckeley C., « Term-weighting approaches in automatic text retrieval », *Information Processing and Management*, 24, 513-523, 1988.

Popovici E., Ménier G., Marteau P.F., « SIRIUS: A Lightweight XML Indexing and Approximate Search System at INEX 2005 », *Pre-proceedings of the INEX 2005 Workshop*, 2005, p. 234-248.

Sauvagnat K., Boughanem M., Chrisment C., « Searching XML documents using relevance propagation »,. *SPIRE 04*, Padoue, Italie, 6- 8 octobre 2004, p. 242-254

Schlieder T., « Schema-Driven Evaluation of Approximate Tree-Pattern Queries », EDBT 2002.

Theobald A., Weikum G., « The Index-Based XXL Search Engine for Querying XML Data with Relevance Ranking », EDBT 2002.

Trotman A., Sigurbjörnsson B., « Narrowed Extended XPath I (NEXI) », *Proceedings of the INEX 2004 Workshop*, 2004, p. 16-40.

Wagner R., Fisher M., « The String-to-String Correction Problem »,. *Journal of the Association for Computing Machinery*, Vol.12, No.1, p.168-173, 1974