Coûts de distance d'édition pour la Recherche d'Information XML

Cyril Laitang — Karen Pinel-Sauvagnat — Mohand Boughanem

Institut de Recherche en Informatique de Toulouse, Equipe SIG-RFI, France
{ cyril.laitang, karen.sauvagnat, mohand.boughanem }@irit.fr

RESUME. *La recherche d'information structurée (RIS) sur documents de type XML permet de retourner des parties de documents répondant plus précisément aux besoins des utilisateurs. Ces derniers, parfois exprimés sous forme de requêtes structurées, peuvent tout comme les documents être représentés sous forme d'arbres. Notre approche utilise ces représentations arborescentes et mesure la pertinence des éléments XML au travers de la distance d'édition. Cette dernière se définit comme la somme des coûts minimaux d'opérations de suppression, d'insertion et de substitution permettant de rendre les arbres isomorphes. Attribuer un coût à ces opérations a donc une conséquence directe sur la qualité de l'appariement. C'est ce problème que nous nous proposons d'étudier dans cet article. Nous avons évalué notre approche au travers de la tâche SSCAS d'INEX 2005 et sûr la tâche DATACENTRIC d'INEX 2010. Les résultats que nous obtenons montrent son intérêt.*

ABSTRACT. *Structured information retrieval (SIR) on XML documents allows to retrieve focused parts of documents that match the user needs. These needs can be expressed throught content and structured queries, that as well as XML documents can be represented as trees. Our approach uses these trees through tree edit distance to estimate the relevance of XML elements. Tree edit distance is the minimum set of insert, delete, and replace operations to turn one tree to another. The effectiveness of tree edit distance strongly relies on these costs. In this paper we will study the estimation of these costs in the context of SIR. Our model was evaluated over the SSCAS INEX's 2005 task as well as the INEX's 2010 Datacentric track and our first results show the interest of such an approach.*

MOTS-CLES : *Recherche d'information structurée, graphes, XML, distance d'édition, DTD.*

KEYWORDS: *Structured information retrieval, graphs, XML, Tree Edit distance, DTD.*

Ce travail a bénéficié d'une aide de l'Agence Nationale de la Recherche portant la référence ANR-08-CORD-009.

1. Introduction

La *recherche d'information structurée* (RIS) sur documents de type XML (*eXtended Markup Language*) cherche à évaluer le degré de pertinence de parties de textes structurées par des balises. Pour ce faire, elle se base sur les contraintes exprimées au travers de requêtes pouvant allier à la fois contenu et structure. On parle de requêtes CAS pour *Content And Structure*.
Dans de précédents travaux (Laitang *et al.,* 2011), nous avons proposé une approche de RIS combinant scores de contenu et de structure. La principale innovation de cette approche portait sur l'utilisation de la distance d'édition permettant le calcul du degré d'isomorphisme entre deux arbres formés respectivement par la structure des documents et les contraintes structurelles des requêtes (Figure 1).

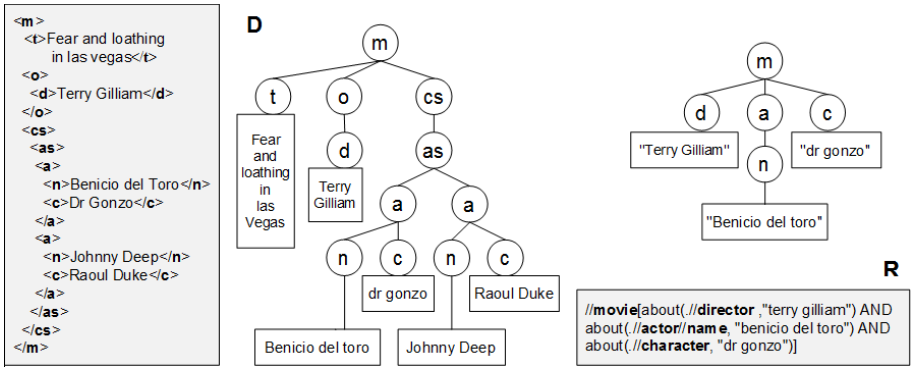

Figure 1. *Représentation sous forme d'arbres d'un document XML D et d'une requête CAS R dans laquelle nous recherchons un "movie" ayant pour "director" "terry gilliam", pour "actor" "benicio del toro" et dont un des "character" est "dr gonzo". Le texte est contenu dans les nœuds feuilles, "movie" est le noeud racine de D et de R et "director", "actor", "name" et "character" (abrégés ici en "d", "a", "n", et "c") sont appelés nœuds internes.*

Si l'intérêt principal de la distance d'édition est d'obtenir une mesure fine des dissemblances structurelles entre deux arbres, les coûts de substitution et de suppression que nous avions appliqués restaient empiriques, et ne permettaient pas de prendre en compte de manière satisfaisante la distance entre la sémantique des balises. Dans cet article, nous proposons une méthode de calcul de ces coûts basée sur la DTD.

La suite de cet article est organisée comme suit. La section 2 présente l'état de l'art sur l'évaluation des distances entre éléments dans les documents XML et sur l'établissement de coûts dans la distance d'édition ; la section 3 présente succinctement notre modèle général avant de se focaliser sur les coûts ; enfin la section 4 présente les résultats obtenus sur les collections INEX 2005 et 2010.

2. Évaluation de la similarité entre les nœuds

La spécificité de notre modèle est d'allier un modèle de RIS à la théorie des graphes et plus particulièrement à la distance d'édition entre arbres. De ce fait nous nous intéresserons ici à ces deux domaines spécifiques. Dans un premier temps nous étudierons les différentes mesures de similarités structurelles proposées par les approches de l'état de l'art en RIS, avant de présenter les approches d'évaluation des coûts pour la distance d'édition.

2.1. *Gestion des contraintes structurelles dans les documents XML pour la RIS*

Nous identifions trois grandes familles d'approches pour l'évaluation de la pertinence des éléments[1] dans le cadre d'une recherche structurée (Trotman, 2009) : la classification des éléments selon des ensembles d'équivalence, l'augmentation des scores des résultats par la structure et la prise en compte des distances dans la structure.

La première famille d'approches cherche à regrouper les éléments équivalents (ayant des balises sémantiques identiques). Les approches les plus simples font ce travail manuellement. C'est ce que font par exemple (Mass *et al.,* 2005). Ce type de méthode est coûteux en temps et demande la participation d'un expert sur le corpus pour l'évaluation. Dans des corpus hétérogènes (Doan *et al.,* 2005) utilisent les DTDs pour identifier des ensembles d'éléments similaires mais syntaxiquement différents. D'autres approches utilisent la structure pour augmenter le score des éléments retournés. Ainsi (Ganguly *et al.,* 2011) utilisent conjointement la relaxation de termes dans les requêtes et un modèle de langue. La structure sert alors plutot à l'identification de la catégorie sémantique dans laquelle se place le document. On retrouve cette idée dans (Ramírez, 2011). A l'inverse (Da C. Hummel *et al.,* 2011) recréent la structure de la requête à partir du contenu. Un certain nombre de travaux utilisent quand à eux les distances entre les nœuds à des fins de propagation ou de calcul. Ainsi dans son approche (Hubert, 2005) utilise cette technique couplée à une réduction structurelle. Un élément répondant à la contrainte verra son score augmenté ou réduit en fonction de sa correspondance et de sa distance. De manière similaire (Sauvagnat *et al.,* 2006) propagent la mesure de pertinence du contenu au travers de la structure. Une approche davantage centrée sur la structure peut être trouvée dans (Ben Aouicha *et al.,* 2010) dans laquelle l'auteur créé des arcs virtuels pondérés entre tous les nœuds.

2.2. *Évaluation des isomorphismes par la distance d'édition*

Dans cette section nous rappelons brièvement le contexte de la distance d'édition entre arbres avant de présenter un certain nombre de travaux ayant eu pour thème l'évaluation des coûts de cette famille d'algorithmes d'appariements.

---

1. Un élément XML est un texte structuré par une ou plusieurs balises (comme dans la figure 1). Dans la suite de cet article, on utilisera indifféremment le vocabulaire nœud ou élément XML.

2.2.1. *Distance d'édition*

La distance d'édition sur arbres est une extension de la distance d'édition sur chaînes de caractère de (Levenshtein, 1966) formalisée par (Tai, 1979). C'est un algorithme généraliste permettant de mesurer le degré d'isomorphisme entre deux arbres qui se définit comme le coût minimal d'opérations (suppression, ajout, substitution) pour transformer un arbre en un autre. De nombreux travaux ont porté sur la réduction de sa complexité et les articles de (Bille, 2005) et (Dulucq *et al.,* 2003) permettent d'en obtenir un aperçu exhaustif.

Algorithmiquement, soient deux forêts (ensemble d'arbres) F et G, Γ_F et Γ_G leurs fils les plus à droite, $T(\Gamma_F)$ l'arbre enraciné en Γ_F et les fonctions de coût $c_{del}()$ et $c_{match}()$ respectivement de suppression (ou ajout) et de substitution. La distance d'édition $d(F,G)$ est évaluée récursivement à la manière d'un parcours pré-ordre par la formule suivante :

$$
\begin{array}{l}
d(F,\emptyset) = d(F-\Gamma_F,\emptyset) + c_{del}(\Gamma_f) \\
d(\emptyset,G) = d(\emptyset,G-\Gamma_G) + c_{del}(\Gamma_G) \\
d(F,G) = min \left\{ \begin{array}{l}
d(F-\Gamma_F,G) + c_{del}(\Gamma_F) \\
d(F,G-\Gamma_G) + c_{del}(\Gamma_G) \\
\quad d(T(\Gamma_F)-\Gamma_F, T(\Gamma_G)-\Gamma_G) \\
\quad +d(F-T(\Gamma_F), G-T(\Gamma_G)) + c_{match}(\Gamma_F,\Gamma_G)
\end{array} \right.
\end{array}
\qquad [1]
$$

2.2.2. *Coûts dans la distance d'édition*

Dans la distance d'édition entre arbres, la majorité des approches fixent les coûts de suppression à 1. La substitution d'un nœud par un autre est quant à elle fixée à 0 lorsqu'ils sont identiques et à 1 dans le cas contraire (Bille, 2005). Dans le cas d'une évaluation plus fine elle est souvent établie empiriquement à l'observation des résultats et en fonction du domaine. Les travaux portant sur l'évaluation des coûts de distance d'édition sont donc principalement axés autours d'approches d'apprentissage dites stochastiques car non déterministes.

Comme nous venons de le voir la distance d'édition sur les arbres est une généralisation de la distance d'édition entre chaînes de caractères. Il est donc naturel de trouver un certain nombre de travaux portants sur l'estimation des coûts dans ce domaine. Ainsi l'utilisation de transducteurs stochastiques est proposée par (Ristad *et al.,* 1998) et (Oncina *et al.,* 2006) pour l'apprentissage des coûts.

Concernant les arbres, dans leur article (Neuhaus *et al.,* 2007) proposent une approche d'apprentissage automatique des coûts basée sur l'estimation de la probabilité de distribution des mesures de distance d'édition. Dans (Bernard *et al.,* 2006) les auteurs proposent deux méthodes différentes. La première basée sur la distribution des jointures entre paires d'arbres et la seconde au travers d'une distribution conditionnelle (basée sur des contraintes). Enfin, une dernière approche (Mehdad, 2009) pro-

pose d'appliquer un algorithme d'optimisation par essaim particulaire (*particul swarm optimisation*) à l'évaluation des coûts.

Nous venons de le voir, lorsque les coûts ne sont pas fixés empiriquement leur évaluation se fait au travers de processus d'apprentissage. Si ces approches apparaissent pertinentes dans leur domaine nous proposons ici une nouvelle approche n'utilisant pas de techniques d'apprentissage.

3. Modèle de RIS par la distance d'édition

Notre modèle allie calcul d'un score de contenu et d'un score de structure pour chaque nœud sélectionné comme réponse à la requête. Dans cette section, nous rappelons les étapes de notre modèle avant de nous intéresser plus particulièrement à notre proposition de coûts pour les opérations de distance d'édition.

3.1. *Calcul du score de contenu*

Dans un document XML le contenu textuel est situé dans les nœuds feuilles. Notre première étape est l'évaluation de la pertinence de ces nœuds par rapport à l'ensemble des conditions de contenu de la requête au moyen d'un $tf \times idf$ (*fréquence du terme* $\times$ *fréquence inverse du document*) tel que proposé par (Jones, 1973). Ceci nous permet d'obtenir un score $p(x)$ pour chaque nœud feuille x de la collection. Nous cherchons ensuite tous les nœuds ancêtres des nœuds feuilles pertinents ($p(x) > 0$). Pour chacun de ces ancêtres $a \in Anc(x)$,nous calculons un score $c(n)$ au travers de la formule de propagation suivante :

$$c(n) = \begin{cases} \underbrace{\frac{p(n)}{\mid feuilles(n) \mid}}_{(i)} + \underbrace{\frac{p(a_1) - p(n)}{\mid feuilles(a_1) \mid}}_{(ii)} + \underbrace{\frac{c(a_1) - \frac{p(a_1)}{|feuilles(a_1)|}}{\mid fils(a_1) \mid}}_{(iii)} & si \text{ n} \neq racine \\ \underbrace{\frac{p(n)}{\mid feuilles(n) \mid}} & sinon \end{cases} \quad [2]$$

Elle se compose de trois parties que sont (i) *le score intermédiaire de contenu* obtenu par la somme $p(n) = \sum_{x \in feuilles(n)} p(x)$ des scores des nœuds feuilles de l'arbre enraciné en n divisé par $\mid feuilles(n) \mid$ le nombre de feuilles ; (ii) *le score de voisinage* obtenu au travers du nœud père a_1. $p(a_1)$ est le *score intermédiaire de contenu* de a_1 et $\mid feuilles(a_1) \mid$ le nombre de nœuds feuilles parmi les descendants de a_1. Cette partie nous permet de transmettre un partie du score des nœuds frères au

travers du père. (iii) *Le score des nœuds ancêtres* est obtenu au travers du score de contenu $c(a_1)$ du père a_1 moins le *score intermédiaire*. Ce score permet de transmettre une faible partie de la pertinence générale du document.

Cette fonction permet d'équilibrer les scores entre nœuds racine et nœuds feuilles et de retransmettre une partie du score global du document à tous les nœuds. Elle assure enfin la propagation entre nœuds frères ce qu'un modèle uniquement basé sur la distance hiérarchique entre les nœuds ne permet pas.

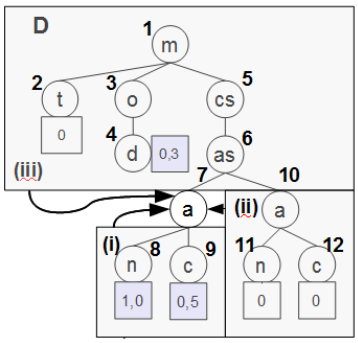

Figure 2. *Application de la formule de propagation dans le calcul du score de contenu du nœud n_7.*

La figure 2 illustre la transmission des score de contenu du noeud n_7. Par application de la formule [2], aprés avoir calculé le score de contenu du noeud père $n_6 = 0.38$ nous obtenons :

$$c(n_7) = \begin{cases} \frac{p(n_7)}{|feuilles(n_7)|} + \frac{p(n_6)-p(n_7)}{|feuilles(n_6)|} + \frac{c(n_6)-\frac{p(n_6)}{|feuilles(n_6)|}}{|fils(n_6)|} \\ \frac{1.5}{2} + \frac{1.5-1.5}{4} + \frac{0.38-\frac{1.5}{4}}{2} \end{cases} \quad [3]$$

3.2. *Calcul du score de structure*

Notre principal apport porte sur la découverte de coûts permettant l'augmentation de la qualité de l'évaluation de pertinence des éléments retournés par notre système de RIS. Nous rappelons d'abord brièvement le fonctionnement de notre algorithme de mesure de similarité structurelle avant d'aborder notre nouvelle méthode d'évaluation des coûts pour la distance d'édition appliquée à la recherche d'information.

3.2.1. *Calcul du score de structure par la distance d'édition*

Notre algorithme de distance d'édition est basé sur une stratégie de décomposition optimale telle que définie par (Dulucq *et al.,* 2003) obtenue au travers des *chemins lourds* de (Klein, 1998). Algorithmiquement, soient deux forêts F et G respectivement formées depuis un document et une requête, p_F et p_G les positions courantes

dans O_F et O_G les *chemins optimaux*[2] de F et de G, la fonction $O.get(p)$ permettant de retourner le nœud correspondant à la position dans le chemin O et $next(p_F)$ et $next(p_G)$ les fonctions permettant de retourner respectivement les premiers nœuds de la chaine du chemin optimal de F et de G n'appartenant pas au sous arbre enraciné par O_F.get(p_F) et O_G.get(p_G) :

Algorithm 1: Distance d'édition par chemins optimaux

```
p_F, p_G = 1;
d(F, G, p_F, p_G) begin
    if F = ∅ then
        if G = ∅ then
            return 0;
        else
            return d(∅, G - O_G.get(p_G)), p_F, p_G++) + c_del (O_G.get(p_G));
        end
    end
    if G = ∅ then
        return d(F - O_F.get(p_F)), ∅, p_F++, p_G) + c_del (O_F.get(p_F));
    end
    a = d(F - O_F.get(p_F), G, p_F++, p_G) + c_del (O_F.get(p_F));
    b = d(F, G - O_F.get(p_F), p_F, p_G++) + c_del (O_G.get(p_G));
    c = d(T(O_F.get(p_F)) - O_F.get(p_F), T(O_G.get(p_G)) - O_G.get(p_G), p_F++,
    p_G++) + d(F - T(O_F.get(p_F)), G - T(O_G.get(p_G)), next(p_F), next(p_G)) +
    c_match (O_F.get(p_F), O_G.get(p_G));
    return min(a, b, c);
end
```

La forêt F correspond aux sous-arbres enracinés depuis les nœuds feuilles pertinents évalués dans la partie contenu (section 3.1) et enracinés par chaque ancêtre à partir du premier nœud partageant un label avec la requête. G est quant à elle formée par la requête.

La figure 3 illustre la correspondance entre les nœuds dans le chemin optimal et les nœuds présent dans l'arbre document obtenu au travers des deux fonctions p_F++ et $next(p_F)$. On constate que $next(p_F)$ correspond à un "saut" vers la racine du premier sous arbre formé par un nœud non présent dans le sous arbre enraciné par le nœud courant.

3.2.2. *Coûts des opérations de distance d'édition*

Le choix de nos coûts répond à un certains nombre de contraintes propres à notre domaine. Premièrement la distance d'édition est traditionnellement utilisée pour comparer deux arbres dont le degré de dissemblance (que ce soit au niveau des arcs ou des nœuds) est relativement faible. En ce sens, si l'impact du coût de substitution est primordiale, celui du coût de suppression l'est moins. Dans notre modèle les écarts de

2. Les chemins optimaux sont composés de la suite des nœuds sur lesquels appliquer la récursion afin de minimiser le nombre de sous-arbres en mémoire.

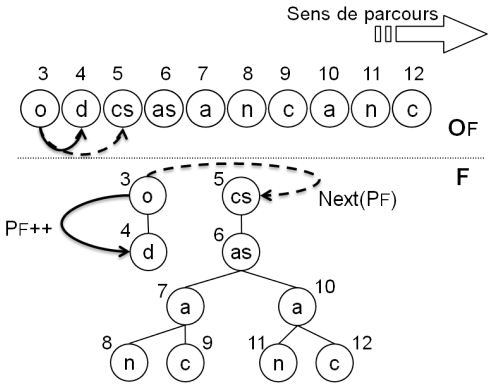

Figure 3. *Utilisation du chemin optimal pour le calcul de la distance d'édition. Exemple de déplacement.*

cardinalités entre arbres formés par les contraintes structurelles des requêtes CAS et les sous-arbres documents peuvent être élevés. Ceci s'explique par la taille réduite des requêtes (quelques éléments) face à celle des documents. Par conséquent le nombre de suppressions sera plus important et son coût aura un impact important sur l'apport de notre part de structure. Parallèlement les coûts de substitution doivent être plus élevés que les coûts de suppression car cette dernière est équivalente théoriquement à une suppression suivie d'une insertion. La troisième contrainte porte sur l'homogénéité des coûts. De la même façon que l'on prend en compte la représentation d'un terme au niveau du corpus il est nécessaire d'évaluer la structure par rapport à la collection. C'est pourquoi les coûts doivent être représentatifs des relations au niveau de la collection. La dernière contrainte porte sur la complexité générale de l'approche. Dans un système de RIS le temps et la charge de calcul doivent être réduits au maximum.

Cette analyse nous oriente vers l'utilisation de la DTD (*Document Type Definition*) pour mesurer ces coûts. La DTD regroupe l'ensemble des règles d'imbrication de balises dans les documents de la collection. Mesurer la distance entre ces éléments doit permettre d'évaluer de manière efficace leur proximité sémantique.
Notre approche se décompose en trois étapes que sont la conversion de la DTD en graphe, l'évaluation des distances et la mise en place dans les coûts.

La première étape consiste en la traduction de la DTD des documents en un graphe dans lequel les nœuds seront les balises et les arcs les liens entre ces balises. Ce graphe sera non-orienté pour deux raisons. La première est que nous considérons que la distance entre ascendant et descendant est équivalente avec celle entre descendant et ascendant. La seconde est que nous souhaitons qu'il soit fortement connexe ce qui n'est pas garanti dans un graphe orienté. Un exemple de conversion est donné dans la figure 4. Il est à noter que la structure ainsi obtenue peut contenir des cycles. Dans le cas de notre exemple, à savoir la collection Datacentric d'IMDB, deux DTDs sont disponibles. Une pour "person" et une pour "movie". Nous obtenons ainsi trois graphes : un

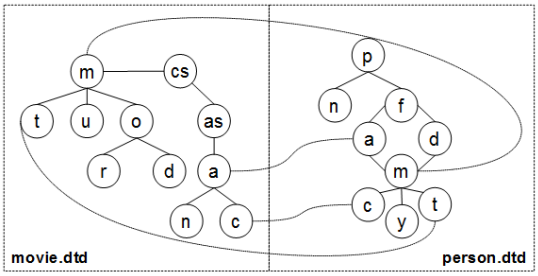

Figure 4. *Représentation sous forme de graphe des relations pour les deux DTDs de la collection INEX 2010 Datacentric. Les labels équivalents entre les deux sont liés pour représenter un graphe global fusionné.*

pour chaque DTD et un fusionné pour les requêtes dont les contraintes structurelles ne sont pas suffisamment précises pour discriminer l'une ou l'autre.

Pour calculer le coût de substitution $c_{match}(n_1, n_2)$ d'un nœud n_1 par un nœud n_2 auxquels sont respectivement associés les labels l_1 et l_2 nous recherchons le chemin le plus court dans le graphe de la DTD entre ces deux labels. Ce chemin est obtenu au travers d'un algorithme de Floyd-Warshall (Floyd, 1962). Le choix de la découverte des chemins les plus courts nous permet de surmonter le problème des cycles dans le graphe. Nous divisons ce résultat par le plus long des chemins entre l_1 et tous les autres labels de la DTD obtenus par l'application de cet algorithme. Le fait de moyenner la distance du chemin ainsi obtenu par le plus long des chemins courts au départ de ce label permet d'ajuster le coût en fonction de la longueur des chemins rencontrés. L'intuition est que plus un nœud possédera un degré faible et plus le chemin maximal sera grand puisqu'il sera d'autant plus difficile de l'atteindre. Nous réduisons ainsi le coût et donc l'importance des nœuds les plus isolés.

Si nous reprenons la figure 4 et choisissons de substituer m par d en nous limitant au graphe formé par la DTD de "movie" alors le chemin le plus court pour aller de m vers d est 2. De la même manière, le plus grand des chemins courts à partir de m est $m \rightarrow c$ de longueur 4. Notre coût sera donc de $2/4$. Formellement, pour notre fonction de calcul du chemin le plus court noté $sp()$:

$$c_{match}(n_1, n_2) = \frac{sp(l_1, l_2)}{max(sp(l_1, l_x))} \forall x \in DTD \quad [4]$$

Afin de calculer le coût de suppression d'un nœud nous procédons de manière similaire à la substitution. La différence principale est que nous effectuons la mesure de distance de ce nœud à tous les labels de la requête et que nous sélectionnons celui dont le score est le plus élevé. En procédant ainsi nous nous assurons de respecter la contrainte de $c_{match} > c_{del}$. Formellement :

$$c_{del}(n_1) = max(\frac{sp(l_1, l_y)}{max(sp(l_1, l_x))}) \forall x \in DTD; \forall y \in R \quad [5]$$

La figure 5 illustre le calcul du coût de suppression d'un nœud au label o en fonction de la DTD. La distance la plus grande parmi les chemins les plus courts d'un nœud de label o à un nœud de label d, a, n, m ou p est de 5. Le chemin le plus long que l'on peut parcourir de o à une autre balise est $o \rightarrow c$ soit 5. Notre coût de suppression vaudra donc 1.

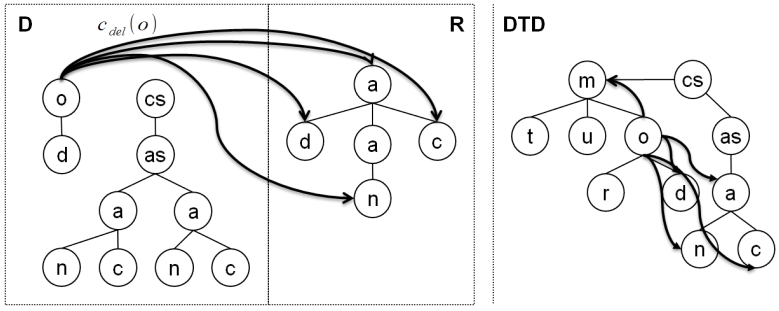

Figure 5. *Calcul du coût de suppression entre un nœud document et un nœud requête et calcul de son coût au travers des distance extraites de la DTD.*

3.2.3. *Combinaison du score de structure*

La combinaison du score de structure est obtenue par la formule suivante :

$$s(n) = \frac{\sum_{a \in Anc(n)} (1 - \frac{d(T(a),R)}{|T(a)|})}{\mid Anc(n) \mid} \quad [6]$$

Pour $Anc(n)$: l'ensemble des ancêtres de n ; a : un nœud sur l'ensemble des ancêtres de n tel que $a \in Anc(n)$; $T(a)$: le sous arbre enraciné en a ; $d(T(a), R)$: la distance d'édition entre le sous arbre $T(a)$ et la requête R.

3.3. *Combinaison finale des scores*

Le score $score(n)$ de chaque nœud candidat n est finalement évalué par la combinaison des scores normalisés $\in [0, 1]$. Formellement, pour un paramètre $\lambda \in [0, 1]$:

$$score(n) = \lambda \times c(n) + (1 - \lambda) \times s(n). \quad [7]$$

Enfin les éléments correspondant aux éléments cibles de la requête (ceux que l'on cherche à retourner) sont filtrés et renvoyés.

4. Évaluations et expérimentations

4.1. *Collections*

Nous avons évalué notre approche pour les différentes mesures de coûts sur les collections INEX 2005 et 2010 et comparé nos résultats aux participants officiels.

4.1.1. *INEX 2005*

INEX (*Initiative for the Evaluation of XML Retrieval*) est la campagne d'évaluation de référence pour la RIS sur documents XML. Afin d'évaluer notre approche nous avont tout d'abord utilisé la collection de 2005 composée de 16000 articles scientifiques au format XML issus de IEEE Computer Society.

Deux grand types de tâches sont disponibles. La tâche basée sur les requêtes *Content Only* (CO) et la tâche basée sur les requêtes *Content And Structure* (CAS). Notre choix s'est porté sur la tâche CAS de 2005 dans la mesure où la prise en compte des contraintes structurelles dans l'évaluation ne fut pas reconduite les années suivantes. La tâche CAS comporte elle-même quatre sous-tâches différenciées par leur niveau d'exigence quant à la position et l'environnement de l'élément retourné. Nous utilisons la sous-tâche SSCAS stricte sur l'élément cible et sur son environnement (le reste des contraintes de structures spécifiées dans la requête). Cette sous-tâche comporte 8 requêtes.

L'évaluation des sous-tâches CAS se fait au travers de deux mesures (Kazai *et al.,* 2006) : *Non-interpolated mean average effort-precision* (MAeP) soit la moyenne de l'effort précision à chaque rang et *Normalized cumulated gain* (nxCG). Ce dernier est le gain cumulé par rapport à un seuil :

$$nxCG(i) = \frac{xCG(i)}{xCI(i)} \quad [8]$$

Pour un rang i, par rapport au vecteur de classement idéal xCI et à la somme des score au rang i xCG.
Enfin, nous appliquons une quantification "stricte" qui considère comme pertinents uniquement les éléments étant très spécifiques et très exhaustifs.

4.1.2. *INEX 2010*

Depuis 2010 la tâche *Datacentric* basée sur la base de données du site IMDB fournit une collection de 4 418 102 documents fortement structurés. Elle se compose de 1 594 513 documents portant sur les films, 1 872 492 sur les acteurs et 951 097 sur les métiers annexes tels que producteurs, compositeurs. Cette collection comporte deux DTDs, une pour les personnes et une pour les films. La tâche comporte 28 requêtes.

L'évaluation de la tache *Datacentric* est effectuée au travers de deux mesures que sont *MAiP* et *MAgP T2I* (Trotman *et al.,* 2010).
- La *MAiP* (Mean average interpolated precision) est calculée sur un certain nombre de points de rappel (eg : 0.00, 0.01, etc..). Elle est utilisée pour la mesure du degré de pertinence dans une recherche ciblée.
- La *MAgp T2I* permet de mesurer l'exhaustivité des résultats obtenus. La mesure elle-même est une généralisation de la précision et du rappel. Le T2I est une mesure de tolérance à la non pertinence parmi les passages retournés [3]. Cette mesure est utilisée dans le cadre d'une recherche pertinente dans le contexte.

3. Lors de la formation des Qrels, les passages pertinents sont surlignés dans chaque document.

4.2. *Résultats*

Nous avons évalué nos approches sur les deux collections avec deux versions de nos coûts. Ces versions sont :
- *Référence* : Elle représente notre ancienne mesure de coûts. La substitution est égale à 0 pour des balises identiques et à 1 dans le cas contraire. De même les coûts de suppression sont fixés à 0.5 pour des éléments présents dans la requête et à 1 dans le cas contraire.
- *Nouveaux coûts* : Elle représente notre nouvelle mesure de coûts de suppression et de substitution présentée dans la section 3.2.2.

Les performances globales sont mesurées en faisant varier le paramètre λ de l'équation [7] de 0 pour le maximum de structure à 1.0 pour une solution tout contenu.

4.2.1. *INEX 2005*

La MAeP est la mesure de référence permettant de classer les participants à la tache d'evaluation d'INEX 2005. La figure 6 présente les résultats obtenus pour différentes variation de λ. Nous observons une augmentation de la moyenne pour une part légèrement moins importante de contenu par rapport à la structure ($\lambda = 0.6$) que pour notre solution initiale ($\lambda = 0.5$). Il semble donc que nous puissions obtenir une mesure des coûts relativement fiable sans qu'il soit nécessaire de fixer les coûts empiriquement.

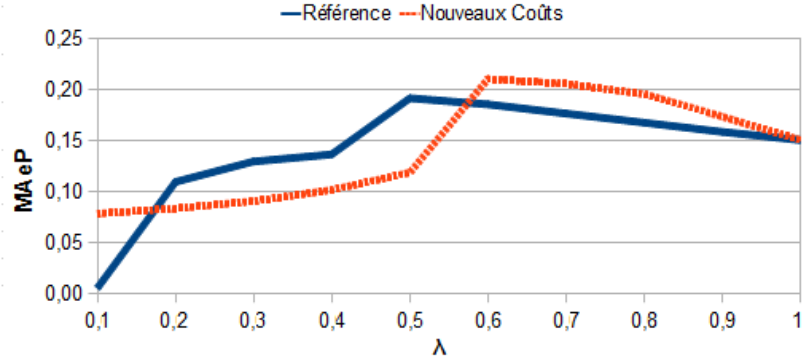

Figure 6. *Evolution de la MAeP selon le paramètre λ.*

Le tableau 1 nous permet de placer notre meilleur "run" (ici $\lambda = 0.6$) par rapport aux meilleurs participants de l'époque : l'institut Max Planck avec son système TopX (Theobald *et al.,* 2005) centré sur une approche base de donnée, IBM Haifa Research Lab qui propose une adaptation du modèle vectoriel (Mass *et al.,* 2004) et l'University of Klagenfurt qui utilise elle aussi un modèle vectoriel (Hassler *et al.,* 2005). Nous observons que nos résultats pour les mesures *nxCG* sont sensiblement les mêmes que les meilleurs de l'époque ; en moyenne 5% en dessous du premier et 13% au dessus du second. Par rapport à notre solution initiale nous restons dans le même ordre de

Runs	MAeP	nxCG10	nxCG25	nxCG50
Nouveaux Coûts	**0.21**	0.4	0.354	0.366
Référence	0.191	0.425	0.420	0.390
MaxPlanck	0.1334	**0.45**	0.3956	0.3894
IBMHaifa	0.1023	0.225	**0.4278**	**0.4067**
Unisys	0.1001	0.325	0.32	0.3489

Tableau 1. *Évolution des mesures pour* $\lambda = 0.6$ *par rapport aux participants pour la tâche SSCAS en quantisation stricte.*

résultats. Nos meilleurs résultats portent toujours sur la *MAeP* que nous améliorons encore de 10 % par rapport à notre solution initiale. Ceci nous permet d'obtenir une amélioration de +60% au dessus du premier run officiel d'INEX et de +105% par rapport second.

4.2.2. *INEX 2010*

Les figures 7 et 8 présentent les écarts de résultats entre nos deux versions pour les mesures *MAiP* et *MAgP*. Nous observons un écart entre nos deux versions de $+130\%$ sur la *MAgP* et $+50\%$ sur la *MAiP* en faveur de l'utilisation de la DTD.

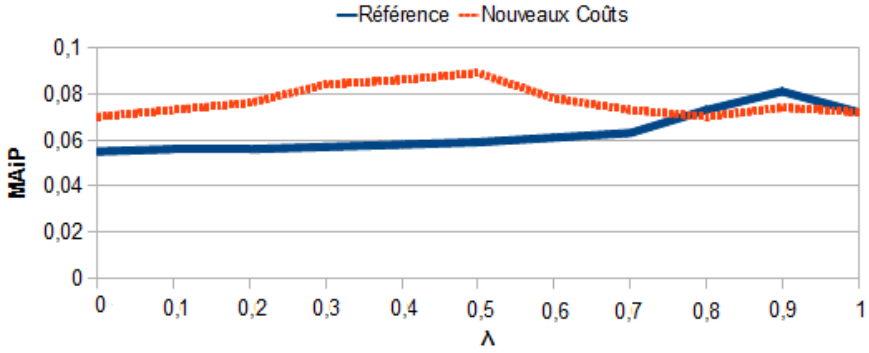

Figure 7. *Évolution de la mesure MAiP selon le paramètre* λ.

Le tableau 2 présente nos résultats par rapport aux meilleurs résultats des participants à la tâche Datacentric pour la mesure $MAgP$. Les résultats de l'université d'Otago sont obtenus au travers d'un BM-25 entrainé sur INEX 2009 (Jia *et al.,* 2010) et sur un modèle de langue mesurant les divergences des résultats par rapport à une distribution aléatoire (Amati *et al.,* 2002). Le run de l'université de Kasetsart (Wichaiwong *et al.,* 2011) quant à lui utilise une version améliorée du BM25F. Enfin les résultats UPF sont ceux de l'université Pompeu Fabra (Ramírez, 2011) pour lesquels les expérimentations portent sur une technique d'indexation séparée des documents (par films, par acteurs, etc.).

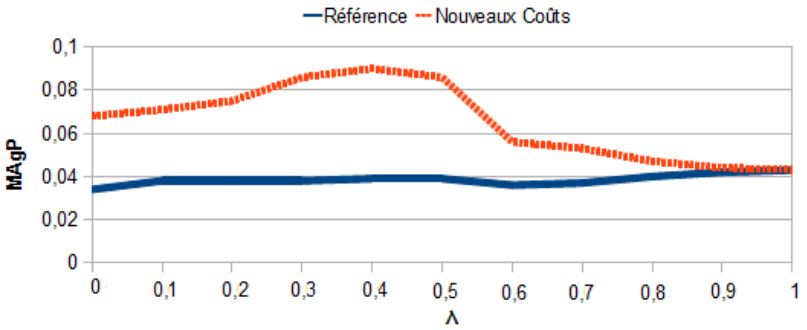

Figure 8. *Évolution de la mesure MAgP selon le paramètre* λ.

Runs	MAgP
OTAGO-2010-DC-BM25	0.2491
UPFL15TMI	0.2458
UPFL15TMImov	0.2433
Kasetsart	0.1811
OTAGO-2010-DC-DIVERGENCE	0.1561
Nouveaux coûts	0.089
Référence	0.039

Tableau 2. *Présentation de nos résultats sur la mesure MAgP avec* $\lambda = 0.5$ *par rapport à quelques uns des participants de la tache Datacentric 2010.*

Comme nous pouvons le constater nos résultats restent en deçà des autres participants d'INEX 2010. Ainsi nous nous plaçons en sixième position sur la mesure *MAgP*. Cela est probablement dû à notre technique de score de contenu qui concatène les contraintes au lieu de les utiliser de manière distincte. Ainsi nous prévoyons de revoir notre manière de considérer ces contraintes afin d'obtenir une base plus forte pour notre solution d'amélioration par la structure.

5. Conclusion et perspectives

Nous avons dans cet article présenté une nouvelle approche de calcul des coûts de distance d'édition basée sur la DTD qui vient renforcer notre précédent modèle. L'amélioration des résultats sur les deux collections nous permet de déterminer que la DTD peut servir à évaluer les différences structurelles entre les balises de l'ensemble des documents du corpus pour un coût relativement faible.

Nos futurs travaux s'orienteront autour de quatres axes. Nous souhaitons améliorer l'extraction des sous-arbres pour le calcul de la structure. Nous projetons également

de renforcer les scores de la partie contenu en traitant chaque condition de manière séparée. Nous envisageons la possibilité de pondérer les arcs du graphe de la DTD selon le type d'imbrication qu'ils représentent. Enfin nous souhaitons étudier l'influence des types de requêtes et de collection afin d'étudier leurs impacts sur les résultats de nos solutions.

6. Bibliographie

Amati G., Van Rijsbergen C. J., « Probabilistic models of information retrieval based on measuring the divergence from randomness », *ACM Trans. Inf. Syst.*, vol. 20, p. 357-389, October, 2002.

Ben Aouicha M., Tmar M., Boughanem M., « Flexible document-query matching based on a probabilistic content and structure score combination », *Symposium on Applied Computing (SAC), Sierre, Switzerland*, ACM, mars, 2010.

Bernard M., Habrard A., Sebban M., « Learning Stochastic Tree Edit Distance », *ECML*, p. 42-53, 2006.

Bille P., « A survey on tree edit distance and related problems », *Theoritical computer science*, vol. 337, n 1-3, p. 217-239, 2005.

Da C. Hummel F., Da Silva A. S., Moro M. M., Laender A. H. F., « Automatically generating structured queries in XML keyword search », *Proceeding of the Initiative for the Evaluation of XML Retrieval*, INEXÍ0, Springer-Verlag, p. 194-205, 2011.

Doan A., Halevy A. Y., « Semantic Integration Research in the Database Community : A Brief Survey », *AI Magazine*, vol. 26, p. 83-94, 2005.

Dulucq S., Touzet H., « Analysis of tree edit distance algorithms », *Proceedings of the 14th annual symposium of combinatorial pattern matching*, p. 83-95, 2003.

Floyd R. W., « Algorithm 97 : Shortest path », *Commun. ACM*, vol. 5, p. 345-, June, 1962.

Ganguly D., Leveling J., Jones G., Palchowdhury S., Pal S., Mitra M., « DCU and ISI@INEX 2010 : Adhoc and Data-Centric Tracks », *in* , S. Geva, , J. Kamps, , R. Schenkel, , A. Trotman (eds), *Comparative Evaluation of Focused Retrieval*, vol. 6932 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, p. 182-193, 2011.

Hassler M., Bouchachia A., « Searching XML Documents - Preliminary Work », *Proceedings of the Initiative for the Evaluation of XML Retrieval*, p. 119-133, 2005.

Hubert G., « XML Retrieval Based on Direct Contribution of Query Components », *Proceedings of the Initiative for the Evaluation of XML Retrieval*, p. 172-186, 2005.

Jia X., Alexander D., Wood V., Trotman A., « University of Otago at INEX 2010 », *Proceedings of the Initiative for the Evaluation of XML Retrieval*, p. 250-268, 2010.

Jones K. S., « Index term weighting », *Information Storage and Retrieval*, vol. 9, n 11, p. 619-633, 1973.

Kazai G., Lalmas M., « INEX 2005 evaluation measures », *Proceedings of the Initiative for the Evaluation of XML Retrieval*, Springer Verlag, p. 16-29, 2006.

Klein P. N., « Computing the Edit-Distance between Unrooted Ordered Trees », *Proceedings of the 6th Annual European Symposium on Algorithms*, ESA '98, Springer-Verlag, London, UK, p. 91-102, 1998.

Laitang C., Pinel-Sauvagnat K., « Utilisation de la théorie des graphes et de la distance d'édition pour la recherche d'information sur documents XML », *in* , G. Pasi, , P. Bellot (eds), *CORIA*, Éditions Universitaires d'Avignon, p. 349-364, 2011.

Levenshtein V., « Binary Codes Capable of Correcting Deletions, Insertions and Reversals », *Soviet Physics Doklady*, vol. 10, p. 707, 1966.

Mass Y., Mandelbrod M., « Component Ranking and Automatic Query Refinement for XML Retrieval », *Proceedings of the Initiative for the Evaluation of XML Retrieval*, p. 73-84, 2004.

Mass Y., Mandelbrod M., « Using the INEX Environment as a Test Bed for Various User Models for XML Retrieval », *Proceedings of the Initiative for the Evaluation of XML Retrieval*, p. 187-195, 2005.

Mehdad Y., « Automatic cost estimation for tree edit distance using particle swarm optimization », *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, ACLShort '09, p. 289-292, 2009.

Neuhaus M., Bunke H., « Automatic learning of cost functions for graph edit distance », *Information Science*, vol. 177, n 1, p. 239-247, 2007.

Oncina J., Sebban M., « Learning stochastic edit distance : Application in handwritten character recognition », *Pattern Recogn.*, vol. 39, p. 1575-1587, September, 2006.

Ramírez G., « UPF at INEX 2010 : Towards Query-Type Based Focused Retrieval », *in* , S. Geva, , J. Kamps, , R. Schenkel, , A. Trotman (eds), *Proceeding of the Initiative for the Evaluation of XML Retrieval*, vol. 6932 of *Lecture Notes in Computer Science*, Springer, p. 206-218, 2011.

Ristad E. S., Yianilos P. N., « Learning String Edit Distance », *IEEE Transactions on Pattern Recognition and Machine Intelligence*, vol. 20, n 5, p. 522-532, May, 1998.

Sauvagnat K., Boughanem M., Chrisment C., « Answering content-and-structure-based queries on XML documents using relevance propagation », *Information Systems, Special Issue SPIRE 2004*, vol. 31, p. 621-635, janvier, 2006.

Tai K.-C., « The Tree-to-Tree Correction Problem », *J. ACM*, vol. 26, p. 422-433, July, 1979.

Theobald M., Schenkel R., Weikum G., « Topx XXL », *Proceedings of the Initiative for the Evaluation of XML Retrieval*, p. 201-214, 2005.

Trotman A., « Processing Structural Constraints », *in* , L. Liu, , M. T. Özsu (eds), *Encyclopedia of Database Systems*, Springer US, p. 2191-2195, 2009.

Trotman A., Wang Q., « Overview of the INEX 2010 Data Centric Track », *in* , S. Geva, , J. Kamps, , R. Schenkel, , A. Trotman (eds), *INEX*, vol. 6932 of *Lecture Notes in Computer Science*, Springer, p. 171-181, 2010.

Wichaiwong T., Jaruskulchai C., « XML Retrieval More Efficient Using Double Scoring Scheme », *in* , S. Geva, , J. Kamps, , R. Schenkel, , A. Trotman (eds), *Comparative Evaluation of Focused Retrieval*, vol. 6932 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, p. 351-362, 2011.