Méthode Hybride de Comparaison Structurelle et Sémantique des Documents XML

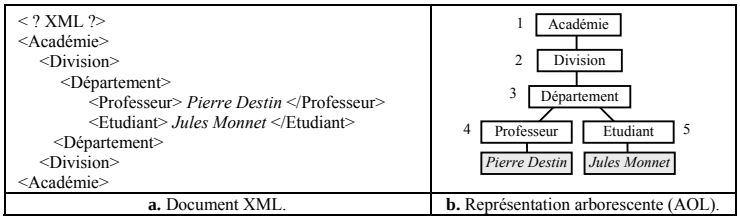
Joe Tekli[1], Richard Chbeir[1] and Kokou Yetongnon[1]

[1] LE2I Laboratory UMR-CNRS, University of Bourgogne
21078 Dijon Cedex France
{joe.tekli, richard.chbeir, kokou.yetongnon}@ u-bourgogne.fr

Mots clés: XML, données semi structurées, similarité structurelle, distance d'édition entre arbres, similarité sémantique, modèle d'espace vectoriel.

1 Introduction

Depuis son lancement par W3C à la fin des années 90, le standard XML s'est établi comme un moyen insigne pour la représentation et l'échange efficaces des données. Les informations destinées à être diffusées sur le Web sont désormais représentées avec le format XML afin de garantir leur interopérabilité. Un document XML est constitué d'un ensemble d'éléments atomiques et complexes (i.e., contenant d'autres éléments) hiérarchiquement structurés, dotés d'attributs atomiques, représentant ainsi des informations riches en structure et sémantique dans une seule entité (Figure 1). L'usage de XML s'articule autour du stockage et de la recherche d'information, la communication entre bases de données, ainsi que l'interaction entre les services Web.



a. Document XML. **b.** Représentation arborescente (AOL).

Fig. 1. Exemple de document XML.

Suite à l'utilisation accrue et sans précèdent de XML sur le Web, la comparaison de documents XML devient capitale dans les domaines des Bases de Données (BD) et Recherche d'Information (RI). Les applications de la comparaison XML sont variées telles que le contrôle de versions (identification et gestion des changements entre différentes versions d'un document XML) [5], [6], [7], l'intégration de données (identification de documents XML similaires, provenant de sources de données différentes, les intégrant afin de permettre à l'utilisateur d'accéder à des informations plus complètes) [11], [12], la classification et le groupage de documents XML publiés sur le Web contre un ensemble prédéfini de grammaires XML [1] (similairement aux schémas des BD traditionnelles, les grammaires XML sont nécessaires pour la protection, l'indexation, la recherche et l'extraction d'informations des documents correspondants) [3], [18], ainsi que la recherche de données XML (identification et ordonnancement des résultats selon leurs similarités aux requêtes, afin de récupérer les meilleurs résultats possibles) [20], [26].

Un nombre d'algorithmes permettant la comparaison des données semi-structurées, en particuliers les documents XML, ont été proposés dans la littérature. La plupart utilisent les techniques de programmation dynamique afin de calculer la distance d'édition entre deux structures arborescentes, les documents XML étant représentés comme Arbres Ordonnées Labellisés (AOL, cf. Figure 1.b). D'autre part, certaines approches proposent des extensions de méthodes conventionnelles de RI (en particulier le modèle vectoriel utilisé dans la plupart des approches) [2], [10]. Dans cette étude, nous limitons notre présentation au premier groupe de méthodes, approches basées sur le concept de distance d'édition, visant la comparaison de données XML rigoureusement structurées et produisant des résultats généralement plus précis (exploités surtout dans les applications de contrôle de versions, classification/groupage et recherche XML moyennant des requêtes structurelles complexes). En outre, les méthodes basées sur des concepts du domaine de RI visent souvent des données XML moins structurées (où on trouve beaucoup plus de texte libre au niveau des contenus des éléments/attributs atomiques) et produisent normalement des résultats moins précis (utiles pour la recherche XML moyennant des requêtes textuelles simples, par exemple les requêtes en mots clés). Notons que les valeurs textuelles des éléments/attributs XML atomiques ne sont pas considérées dans notre présente étude, mais uniquement les labels des éléments/attributs (cf. Section 2).

Dans le contexte de comparaison des données XML, deux problèmes majeurs se présentent : la *similarité structurelle* et la *similarité sémantique*.

[1] Une grammaire XML (DTD ou Schéma XML [9]) définit les éléments, leurs attributs, leurs dispositions structurelles, ainsi que les règles auxquelles ils obéissent dans les documents XML correspondants.

D'un point de vue structurel (i.e., considérant les relations parent/enfant ainsi que l'ordonnancement entre éléments XML frères, identifiés par leurs labels), une étude rigoureuse des approches existantes de similarité XML [6], [8], [18] nous a conduit à identifier certains cas où le résultat de la comparaison est inexact. Ces résultats montrent des similarités structurelles non détectées au niveau des sous-arbres XML, comme nous allons montrer dans les exemples de motivation (Section 2.1). En outre, nous avons remarqué que la plupart des méthodes de comparaison XML existantes visent exclusivement les propriétés structurelles des documents XML, ne tenant aucun compte de leurs caractéristiques sémantiques (i.e., la signification sémantique des labels des éléments/attributs XML, comme par exemple la similarité entre *Professeur* et *Conférencier* dans Figure 3, Section 2.2). Cependant, l'évaluation de la similarité sémantique entre les documents publiés sur le Web est d'importance primordiale afin d'améliorer le résultat de la recherche : identification et ordonnancement des résultats en fonction de leurs degrés de similarité sémantique [14].

L'importance de la similarité sémantique dans les mécanismes de recherches de données, ainsi que la prolifération des documents XML sur le Web, nous ont incité à examiner le sujet de similarité XML dans ses deux facettes *structurelle* et *sémantique*. Notre objectif est le développement d'une approche de similarité XML hybride paramétrée permettant i) la détection des similarités structurelles XML (une étude détaillée est développé dans [22], [23]), ii) l'identification des similarités sémantiques entre documents XML (un travail préliminaire est développé dans [21]) iii) ainsi que le réglage de la comparaison XML selon le contexte et l'application, donnant plus d'importance à l'une des facettes *structurelle* ou *sémantique*, afin de produire des résultats de comparaison plus précis. En résumé, nous étendons des approches existantes, en particulier celles développées dans [6], [18], afin de considérer les différentes similarités structurelles entre sous-arbres XML. Nous étendons la similarité structurelle, en combinant le model d'espace vectoriel en RI [15] et une méthode d'évaluation de la similarité sémantique entre termes/expressions [13], afin de considérer les similarités sémantiques entres sous-arbres XML, en fonction d'une base de connaissance de référence (réseau sémantique).

Dans ce qui suit, nous focalisons notre présentation autour de la motivation (Section 2) et les idées principales du travail (Section 3). Les algorithmes, les résultats expérimentaux et l'état de l'art sont détaillés dans [24].

2 Motivation

Cette section met en valeur l'importance de la détection des similarités structurelles et sémantiques en comparant les documents XML.

2.1 Similarité Structurelle

Les documents XML peuvent inclure plusieurs éléments optionnels et répétitifs [18]. Tels éléments induisent des répétitions de sous-arbres XML identiques/similaires, dans le même document. Considérons par exemple les arbres XML *A*, *B* et *C* (représentés sous formes d'AOL, où les labels des nœuds désignent les noms des éléments/attributs XML correspondants). On se rend compte facilement que l'arbre XML *A* est plus similaire à *B* qu'à *C*, le sous-arbre constitué des nœuds *b*, *c* et *d* dans *A*, figurant deux fois dans *B* (B_1 et B_2) et une seule fois dans *C* (C_1). Un autre exemple est celui des arbres XML *A*, *D* et *E*. Ce cas est différent de son précédent puisque les sous-arbres qui se répètent ne sont pas identiques, mais sont similaires au sous-arbre source (D_2 est similaire à A_1, mais pas identique). Telles similarités structurelles, en particulier la répétition de sous-arbres similaires (non identiques), restent non détectées avec les approches de comparaison XML existantes, en particulier les méthodes dans [6], [8], [18]. D'autres similarités structurelles non détectées sont discutées dans [24], notamment les similarités entres sous-arbres se trouvant à des niveaux structurels différents, ainsi que les répétitions de nœuds feuilles.

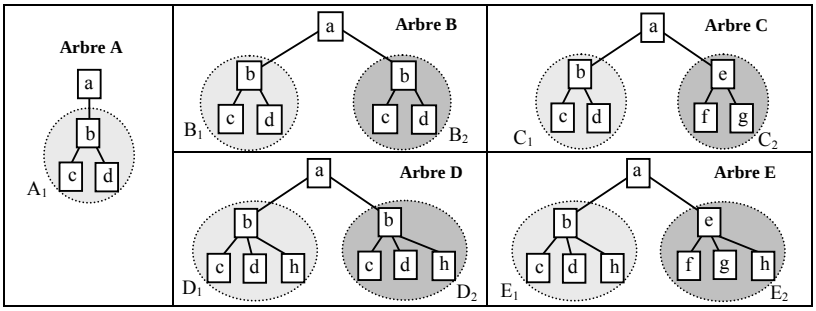


Fig. 2. Représentations arborescentes d'un ensemble de documents XML factices.

2.2 Similarité Sémantique

Afin d'insister sur le besoin de prendre en considération la signification sémantique en comparant les documents XML, nous considérons les exemples dans la Figure 3. L'usage des méthodes traditionnels de mesure de similarité structurelle (méthodes généralement basées sur le concept de distance d'édition, e.g., [6], [8], [18]), fournit une même valeur de similarité en comparant l'arbre XML *X* aux arbres *Y* et *Z*. Pourtant, en dépit de leurs similitudes structurelles, on peut évidement reconnaître que l'arbre XML *X* partage plus de caractéristiques sémantiques avec l'arbre *Y* qu'avec *Z*. En particulier, les pairs de mots *Académie*/*Collège* et *Professeur*/*Conférencier*, des documents

X/*Y*, semblent sémantiquement plus similaires que les pairs *Académie*/*Usine* et *Professeur*/*Superviseur*, extraites des arbres XML *X* et *Z* (en fonction d'une base de connaissance générique comme WordNet[1], décrivant des concepts rencontrés dans le langage courant).

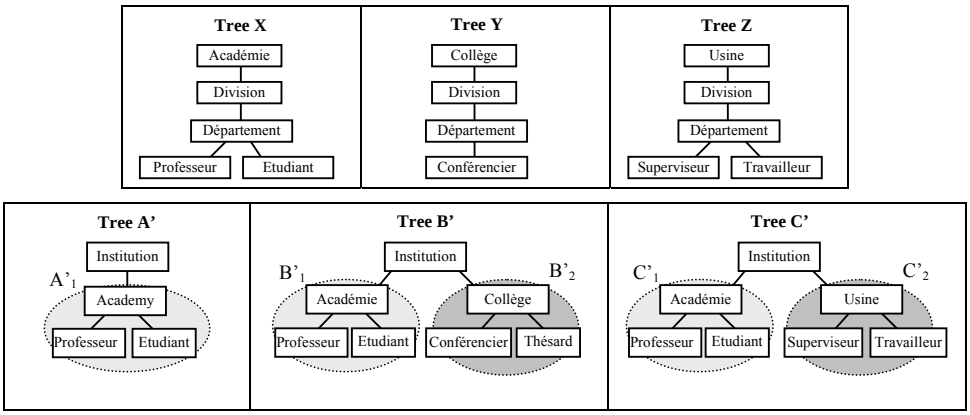


Fig. 3. Représentations arborescences de documents XML.

Un exemple (relativement plus compliqué) de similarités sémantiques non détectées, est celui des arbres XML *A'*, *B'* et *C'*. Il est comparable à ceux de la Figure 2, suggérant la nécessité de détecter les similarités entre sous-arbres XML dans le processus de comparaison. Malgré que *A'*/*B'* et *A'*/*C'* sont structurellement indiscernables, on peut voir que l'arbre XML *A'* est sémantiquement plus similaire à *B'*, qu'à *C'*. Le sous-arbre A'_1, composé des nœuds *Académie*, *Professeur* et *Etudiant* est sémantiquement similaire à B'_2 (composé des nœuds *Collège*, *Conférencier* et *Thésard*), tandis qu'il est sémantiquement différent de C'_2 (composé des nœuds *Usine*, *Superviseur* et *Travailleur*). D'autres exemples de similarités sémantiques non détectées sont détaillées dans [24].

3 Méthode Hybride de Similarité XML

Notre méthode de comparaison de documents XML est constituée de quatre principaux modules : i) *Struct-CBS* (*Structural Commonality Between Sub-trees*) pour l'identification de la similarité structurelle entre sous-arbres, ii) *Sem-RBS* (*Semantic Resemblance Between Sub-trees*) pour l'identification de la similarité sémantique entre sous-arbres,, iii) *TOC* (*Tree edit Operations Costs*) pour le calcul des coût des opérations d'édition, iv) et *TED* (*Tree Edit Distance*) calculant la distance d'édition entre deux structures arborescentes. En résumé, le module *TOC* exploite *Struct-CBS* et *Sem-RBS* afin d'évaluer les similarités structurelles et sémantiques entre les sous-arbres des documents XML comparés, leurs résultats constituant les coûts des opérations d'édition de sous-arbres. Ces coûts sont employés dans *TED*, un algorithme calculant la distance d'édition entre deux arbres XML. Ainsi, les entrées de notre méthode de similarité XML sont:

- Les documents XML à comparer (représentés sous formes d'AOLs),
- Un paramètre α permettant à l'utilisateur d'assigner plus d'importance à la similarité structurelle ou sémantique, selon ses besoins en comparaison,
- Une base de connaissance *BC* de référence à employer dans l'évaluation de la similarité sémantique.

Par conséquent, notre méthode quantifie la similarité entre deux documents (arbres) XML, produisant une valeur de similarité normalisée comprise dans l'intervalle *[0, 1]*. L'architecture globale est présentée dans la Figure 4.

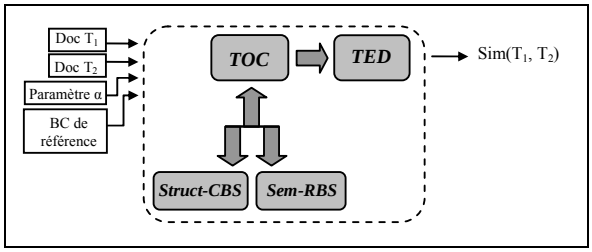


Fig. 4. Architecture globale de notre méthode de comparaison XML.

[1] WordNet est un système de référence lexicale accessible via le Web, développé à l'université de Princeton NJ au Etats-Unis, essayant de modéliser la connaissance lexicale d'une personne parlant l'Anglais comme langue native. WordNet organise les mots, verbes, adjectives et adverbes dans des groupes de synonymes dits *synsets*, chacun représentant un concept lexical de base [16] (http://www.cogsi.princeton.edu/cgi-bin/webwn).

3.1 Définitions de base

Définition 1 - Arbre Ordonné Labellisé (AOL) : C'est un arbre avec un noeud racine unique, dans lequel les nœuds sont ordonnés et labellisés (cf. Figure 1). Dans la suite, le terme *arbre* dénotera AOL ●

Définition 2 – Distance d'édition entre arbres : La distance d'édition entre deux arbres A et B est le coût minimal de tous les scripts d'édition permettant de transformer A en B : $Dist(A, B) = Min\{Cost_{ES}\}$. Ainsi, le problème de comparaison de deux arbres A et B, identification de leur similarité structurelle, revient à calculer leur distance d'édition [25] (généralement, $Sim(A, B) = 1 / 1+Dist(A, B)$) ●

Définition 3 - Script d'édition: Il s'agit d'une séquence d'opérations d'édition op_1, op_2, …, op_k. Appliqué à un arbre T, l'arbre résultant T' est obtenu en exécutant les opérations d'édition englobée dans le script d'édition ES, selon leur ordre d'apparition dans le script. En associant un coût, $Cost_{Op}$, à chacune des opérations dans ES, le coût global de ES sera égal à la somme des coût des ses opérations : $Cost_{ES} = \sum_{i=1}^{|ES|} Cost_{Op_i}$ ●

L'algorithme *TED* de distance d'édition utilisé dans notre approche emploie cinq opérations d'édition : insertion de nœud feuille (*Insert*), suppression de nœud feuille (*Delete*), mise à jour de nœud (*Update*, modifiant le label du nœud concerné), insertion d'arbre (*InsertTree*) et suppression d'arbre (*DeleteTree*). Les coûts des opérations d'insertion/suppression d'arbres seront calculés selon les similarités structurelles/sémantiques entre les arbres XML correspondants, les opérations appliquées sur des nœuds individuels ayant des coûts unitaires (comme avec les méthodes classiques de distance d'édition).

En plus des caractéristiques structurelles des documents XML, les noms des éléments/attributs portent des significations sémantiques (Section 2.2), jusque-là non traitées par la plupart des approches de comparaison XML existantes, en particulier celles basées sur le concept de distance d'édition. Cependant, la détection et l'analyse des significations sémantiques entre concepts constituent un axe d'étude central dans les domaines de traitement du langage naturel (TLN) et de recherche d'information (RI). Dans ce contexte, les concepts sont généralement organisés dans des structures dites : bases de connaissances (thésaurus, taxonomies et/ou ontologies) :

Définition 4 – Base de connaissance: Elle revient généralement à un réseau sémantique, construit d'un ensemble de concepts représentant des groupes d'objets (des mots/expressions synonymes dans le cas de WordNet [16]), et un ensemble de liens interconnectant les concepts, soulignant des relations sémantiques (*IsA*, *HasA*, *PartOf*, *HasPart*, …) ●

Plusieurs méthodes ont été proposées afin de déterminer la similarité sémantique entre les concepts d'une base de connaissances, en particulier, une méthode efficace avancée par Lin [13], que nous exploitons dans notre approche.

3.2 'Communalité' Structurelle entre Sous-arbres (*Struct-CBS*)

Les similarités structurelles entre sous-arbres (Section 2.1) XML sont généralement non détectées avec les méthodes de comparaison XML existantes. D'après nos recherches, les auteurs dans [18] sont les premiers à étudier le sujet, introduisant des opérations d'insertion/suppression d'arbres (*InsertTree* et *DeleteTree*). Selon [18], un arbre A peut être insérer/supprimer dans T s'il existe un arbre équivalent à A, déjà contenu dans T. Ainsi, la méthode développé dans [18] détecte les similarités entre les arbres XML A/B dans la Figure 2, transformant A en B en une seule opération d'édition (insertion de B_2 dans A, B_2 étant identique à A_1) tandis que la transformation de l'arbre A en C nécessite toujours 3 opérations d'insertions consécutives (insertion des nœuds e, f et g). Cependant, certaines similarités structurelles restent non détectées lorsque les sous-arbres à insérer/supprimer ne sont pas identiques ou inclus dans l'arbre source/destination respectivement. Un exemple typique est celui des arbres XML A, D et E. Comme D_2 n'est pas contenu dans A, il sera inséré via 4 opérations d'édition (insertion des nœuds individuels b, c, d, et h) au lieu d'une seule (insertion du sous-arbre D_2), sans détecter le fait qu'une partie de D_2 (nœuds b, c et d) est identique à A_1. Ainsi, on obtient $Dist(A, D)=Dist(A, E)$ ($Sim(A, D)=Sim(A, E)$, avec $Sim=1/(1+Dist)$), ne considérant pas des similarités structurelles entre A et D.

Afin de détecter les divers genres de similarités structurelles entre sous-arbres, et par conséquent entre documents XML, nous introduisons la notion de *communalité structurelle* entre sous-arbres.

Definition 5 – Communalité structurelle entre sous-arbres : Etant donné deux sous-arbres A et B, la *communalité structurelle* entre A et B, $StructCom(A, B)$, désigne l'ensemble de nœuds $N = \{n_1, …, n_p\}$ tel que $\forall\, n_i \in N$, n_i apparaît dans A et B avec le même label, la même profondeur et le même ordonnancement ●

En d'autres termes, $StructCom(A, B)$ identifie les correspondances structurelles entre les différents sous-arbres de A et B. Elle est, par la suite, normalisée par les cardinalités des sous-arbres correspondants, afin d'obtenir des valeurs comprises entre $[0, 1]$:

– $\frac{|\,StructCom(SbT_i, SbT_j)|}{Max(|SbT_i|, |SbT_j|)} = 0$ Lorsque les sous-arbres XML sont complètement dissimilaires, $|StructCom(SbT_i, SbT_j)| = 0$

– $\frac{|\,StructCom(SbT_i, SbT_j)|}{Max(|SbT_i|, |SbT_j|)} = 1$ Lorsque les sous-arbres sont identiques, $|StructCom(SbT_i, SbT_j)| = |SbT_i| = |SbT_j|$

Par exemple, dans la Figure 2, *StructCom(A_1, B_2) = 3*, la *communalité structurelle normalisée* étant égale à *1* (soulignant le fait que A_1 et B_2 sont identiques). En outre, *StructCom(A_1, D_2) = 3*, la *communalité structurelle normalisée* entre A_1 et D_2 étant égale à ¾.

La *communalité structurelle* entre sous-arbres est calculée via un algorithme dédié, intitulé *Struct-CBS*, basé sur le concept de distance d'édition. Il est développé dans [24].

3.3 Ressemblance Sémantique entre Sous-arbres (*Sem-RBS*)

Dans la section 2.2, nous avons brièvement motivé l'importance de considérer la signification sémantique des labels des nœuds XML dans le processus de comparaison, que nous identifions par *ressemblance sémantique*. Plusieurs méthodes de détection de la similarité sémantique entre pairs de mots/expressions, en fonction d'une base de connaissance de référence (Définition 4), ont été proposées dans la littérature. Cependant, à l'exception de quelques approches théoriques et extrêmement complexes (voir [24] pour une présentation et discussion détaillées de l'état de l'art), la similarité sémantique entre deux groupes de mots/expressions (e.g., les labels des nœuds de deux sous-arbres XML) n'est toujours pas quantifiée avec les approches existantes.

Afin de détecter les *ressemblances sémantiques* entre sous-arbres XML, nous combinons le modèle d'espace vectoriel traditionnel, développé dans le domaine de RI [15], avec une méthode classique d'évaluation de la similarité sémantique entre mots/expression [13]. En comparant deux sous-arbres SbT_i et SbT_j, chaque sous-arbre sera représenté par un vecteur dédié, V_i et V_j respectivement, avec des poids soulignant les similarités sémantiques entre leurs labels.

Définition 6 – Espace Vectoriel de sous-arbres XML : Etant donné deux sous-arbres XML SbT_i et SbT_j, les vecteurs V_i et V_j sont construits dans un espace dont chaque dimension représente une unité d'indexation distincte. Chaque unité d'indexation souligne un label l_r différent, parmi les labels des nœuds des sous-arbres SbT_i et SbT_j. Le poids d'un vecteur V_i, au niveau de la dimension l_r, noté $w_{Vi}(l_r)$, souligne le poids sémantique de l_r dans le sous-arbre SbT_i ●

Définition 7 – Poids sémantique : Le poids sémantique d'un nœud v_r, de label l_r, dans le sous-arbre SbT_i, de vecteur V_i, est composé de deux facteurs : le facteur de similarité nœud/vecteur *Sim(v_r, V_i, BC)* et le facteur de profondeur *D-factor(v_r)*, tel que $w_{Vi}(l_r) = Sim(v_r, V_i, SN) \times D\text{-}factor(v_r) \in [0, 1]$.

– *Sim(v_r, V_i, SN)* quantifie la similarité sémantique entre le label l_r du nœud v_r et le vecteur V_i. C'est le maximum de la similarité sémantique entre l_r et tous les nœuds de SbT_i, en fonction d'une base de connaissance *BC* de référence. Formellement, $Sim(v_r, V_i, BC) = \underset{v \in Vi}{Max}(Sim(v_r, v, BC)) \in [0, 1]$. Dans ce contexte, une mesure de similarité sémantique classique est exploitée afin de calculer la similarité entre pairs de labels, *Sim(v_r, v, BC)* (Nous employons celle développée dans [13] grâce à son efficacité. Autres méthodes auraient pu être employées aussi).

– *D-factor* souligne l'influence sémantique de la profondeur (*Depth*) d'un nœud sur le document XML correspondant. En général, l'on considère que les informations placées près de la racine du document (arbre) XML sont plus importantes et significatives que celles placées plus loin dans la hiérarchie [3], [26]. Ainsi, l'influence sémantique des labels des nœuds plus élevés dans la hiérarchie de l'arbre XML est plus importante que celles des nœuds plus bas. Ceci est mathématiquement concrétisé comme suit (formule adaptée de [26]):

$$D\text{-}factor(p.l) = \frac{1}{1 + p.d} \in [0, 1]$$ où *p.l* et *p.d* désignent respectivement le label et la profondeur du noeud *p*● **(1)**

Par conséquent, après avoir transformé les sous-arbres XML en des vecteurs de poids sémantiques, la ressemblance sémantique entre deux sous-arbres est évaluée avec une mesure de similarité classique entre vecteurs : *produit scalaire*, *cosinus*, *mesure de Jaccard*, … [4]. Dans notre approche, nous adoptons *cosinus*, mesure largement exploitée dans le domaine de RI [4], [19]:

$$Sem\text{-}RBS(SbT_i, SbT_j, BC) = \cos(V_i, V_j) = \frac{\sum_{r=1}^{n} w_{SbTi}(l_r) \times w_{SbTj}(l_r)}{\sqrt{\sum_{r=1}^{n} w_{SbTi}(l_r)^2 \times \sum_{r=1}^{n} w_{SbTj}(l_r)^2}} \in [0, 1] \quad \textbf{(2)}$$

La ressemblance sémantique *Sem-RBS* entre sous-arbres XML est évaluée avec un algorithme dédié, détaillé dans [24].

Comparons, par exemple, les sous-arbres *A'1*, *B'2* et *C'2* de la figure 3. Les vecteurs correspondants sont présentés dans la Figure 5. Ainsi, *Sem-RBS(A'1, B'2) = 0.9752 > Sem-RBS(A'1, C'2) = 0.5461*, soulignant le fait que *A'1* et *B'2* sont sémantiquement plus similaires que *A'1* et *C'2* (selon une *BC* générique extraite de WordNet, les exemples étant initialement développés en Anglais).

	Académie	Professeur	Etudiant	Collège	Conférencier	Thésard
$V_{A'1}$	1	0.5	0.5	0.7970	0.3838	0.4202
$V_{B'2}$	0.7970	0.3838	0.4202	1	0.5	0.5

a. Vecteurs correspondants aux sous-arbres *A'1* et *B'2*.

	Académie	Professeur	Etudiant	Usine	Superviseur	Travailleur
$V_{A'1}$	1	0.5	0.5	0.2662	0.1804	0.1804
$V_{C'2}$	0.2662	0.1804	0.1804	1	0.5	0.5

b. Vecteurs correspondant aux sous-arbres *A'1* et *C'2*.

Fig. 5. Espaces vectoriels et vecteurs permettant le calcul de la ressemblance sémantique entre *A'1*/*B'2* et *A'1*/*C'2*.

3.4 Coûts des opérations d'édition (*TOC*)

Le module *TOC* permet de calculer les coûts des opérations d'insertion/suppression d'arbres, tenant compte des *communalités structurelles* et *ressemblances sémantiques* entre les sous-arbres XML. En employant les résultats de *Struct-CBS* et *Sem-RBS*, le module *TOC* identifie :

- Les similarités entre chaque pair de sous-arbres *(SbT_i, SbT_j)* dans les arbres XML source et destination T_1 et T_2 respectivement, calculant les coûts des opérations d'insertion/suppression d'arbres en fonction.
- Les similarités entre chaque sous-arbre de l'arbre source T_1, et l'arbre XML destination T_2 dans sa totalité, modifiant les coûts des opérations de suppression d'arbre en fonction.
- Les similarités entre chaque sous-arbre de l'arbre destination T_2, et l'arbre XML source T_1 dans sa totalité, modifiant les coûts des opérations d'insertion d'arbre en fonction.

Selon *TOC*, les coûts des opérations d'insertion/suppression d'arbres varient comme suit:

$$\text{Cost}_{\text{InsTree/DelTree}}(\text{SbT}_i) = \sum_{\text{Tous les noeuds } x \text{ de SbT}_i} \text{Cost}_{\text{Ins/Del}}(x) \times \frac{1}{1 + \alpha\,\text{Struct-CBS}(\text{SbT}_i, \text{SbT}_j) + (1-\alpha)\,\text{Sem-RBS}(\text{SbT}_i, \text{SbT}_j)} \quad \textbf{(3)}$$

Tel que $\alpha \in [0, 1]$ est un paramètre fourni par l'utilisateur.

Le coût maximal d'une opération d'insertion/suppression d'arbre (obtenue lorsque le sous-arbre concerné SbT_i ne partage aucune *communalité structurelle* ou *ressemblance sémantique* avec les arbres source/destination respectivement) est la somme des coûts unitaire[1] (=1) des opérations d'insertion/suppression de chacun de ses nœuds individuels. Le coût minimal d'une opération d'insertion/suppression d'arbre (obtenue lorsque le sous-arbre concerné SbT_i est structurellement et sémantiquement identique avec, au moins, un des sous-arbres XML source/destination respectivement) est égal à la moitié de son coût maximal. Le coût minimal est défini de telle manière afin de garantir que le coût d'insertion/suppression d'un sous-arbre soit toujours plus grand ou égal à celui de l'insertion/suppression d'un nœud feuille (la preuve est développée dans [24]). En fait, *TOC* est basé sur l'intuition que les opérations manipulant des arbres sont plus coûteuses que celles manipulant des nœuds individuels

Ainsi, avec *TOC*, nous calculons les coûts des opérations d'insertion/suppression de sous-arbres selon les *communalités structurelles* et *ressemblances sémantiques* entres les sous-arbres concernés. En plus, *TOC* permet à l'utilisateur d'assigner plus d'importance aux similarités structurelles ou sémantiques en variant la valeur du paramètre $\alpha \in [0, 1]$:

- Pour $\alpha = 1$, *TOC* considère les *communalités structurelles* en calculant les coûts des opérations d'édition (via *Struct-CBS*).
- Pour $\alpha = 0$, les *ressemblances sémantiques* sont uniquement considérées dans le calcul des coûts des opérations (via *Sem-RBS*).

Cette faculté permet à l'utilisateur d'adapter la méthode de comparaison XML selon le scénario en place ainsi que ses besoins et sa perception de la similarité XML, insistant sur l'aspect structurel ou (inclusif) sémantique des documents XML comparés. L'algorithme *TOC* est développé dans [22], [23].

[1] Avec la majorité des approches de distance d'édition classiques, on assigne des coûts identiques unitaires aux opérations d'insertion/suppression de noeuds [5], [18].

3.5 Distance d'édition entre arbres XML

L'algorithme de distance d'édition *TED*, employé dans notre étude, est une adaptation d'un algorithme initialement développé par Nierman et Jagadish dans [18]. En plus des coûts des opérations d'insertion/suppression variant en fonction des similarités structurelles/sémantiques entre sous-arbres XML, *TED* exploite les coûts des opérations de mise à jour (*Update*, voir Section 3.1). Avec l'opération de mise à jour, *TED* compare les racines des sous-arbres traités au niveau du processus récursif (au départ, les racines des arbres XML sont traités).

Avec les approches classiques de distance d'édition, le coût de l'opération de mise à jour souligne l'égalité/différence entre les nœuds XML concernés :

- Coût minimal lorsque les nœuds comparés sont de labels identiques, $Cost_{Upd}(a, b) = 0$ quand $a.l = b.l$.
- Coût maximal unitaire autrement, i.e., $Cost_{Upd}(a, b) = 1$ quand $a.l \neq b.l$

Cependant, afin de considérer les similarités sémantiques entre les labels des nœuds (et non pas seulement l'égalité/différence), nous développons le modèle de coût de l'opération de mise à jour comme suit :

$$\text{Cost}_{\text{Upd}}(\text{a}, \text{b}) = \begin{bmatrix} (1 - (1-\alpha)\,(Sim(\text{a.l}, \text{b.l}, \text{BC})) \times \dfrac{\alpha + D\text{-}factor(\text{a.d})}{1 + \alpha\ D\text{-}factor(\text{a.d})} & Si\ \text{a.l} \neq \text{b.l} \\ 0 \quad \text{autrement} \end{bmatrix} \text{ avec } \alpha \in [0, 1] \qquad \textbf{(4)}$$

Le paramètre α est le même employé dans *TOC* afin d'assigner plus d'importance aux similarités structurelles ou sémantiques en comparant les sous-arbres XML :

- Pour $\alpha = 1$, on considère uniquement l'égalité/différence entre labels en calculant le coût de l'opération d'édition (coût égal à *0* ou *1* exclusivement), similairement aux approches de distance d'édition classiques.
- Pour $\alpha = 0$, la similarité sémantique entre les labels des noeuds affectés par l'opération de mise à jour, ainsi que la profondeur des nœuds correspondant (notons que les nœuds traités par l'opération de mise à jour sont toujours du même niveau, $a.d = b.d$), seront considérées en assignant le coût de l'opération. Dans ce cas, le coût de l'opération de mise à jour sera compris dans l'intervalle *[0, 1]*.

Considérons par exemple les documents *X*, *Y* et *Z* de la figure 3. Avec, $\alpha = 0$, les opérations de mise à jour appliquées aux racines des arbres auront les coûts suivants :

- $\text{Cost}_{\text{Upd}}(\text{X}.Racine, \text{Y}.Racine) = 1 - (\text{Sim}(Academie, Collège, BC) \times 1) = 0.2030$
- $\text{Cost}_{\text{Upd}}(\text{X}.Racine, \text{Z}.Racine) = 1 - (\text{Sim}(Academie, Usine, BC) \times 1) = 0.7337$

Il est clair que l'opération de transformation du label *Académie* en *Collège* est moins coûteuse que celle de la transformation du label *Académie* en *Usine*, identifiant le fait que les labels *Académie* et *Collège* sont sémantiquement plus similaires que les labels *Académie* et *Usine* (selon une base de connaissance *BC* générique extraite de WordNet, les exemples étant initialement développés en Anglais).

Notons que similairement à *Sem-RBS* (cfr., Section 3.2), les valeurs de similarité sémantique entre labels sont calculés avec la méthode de similarité sémantique développé dans [13], que nous adoptons dans notre étude (autres méthodes auraient pu être employées), en utilisant une base de connaissance *BC* de référence comme WordNet [16].

Tableau 1 présente les valeurs de similarités obtenues, lors de l'application de notre approche de similarité XML, aux divers exemples de comparaison traités dans le papier.

Tab. 1. Valeurs de distance/similarité obtenues en comparant les documents (arbres) XML des Figures 2 et 3 (exemples de motivation).

		Notre approche		**N. & J. [18]**	**Dalamagas *et al.* [8]**	**Chawathe [6]**
		Distance	Similarité			
A/B	(α=1)	1.5	0.4	*Détectée*	*Non détectées*	*Non détectées*
A/C	(α=1)	3	0.25			
A/D	(α=1)	3.2856	0.2333	*Non détectées*		
A/E	(α=1)	5	0.1667			
X/Y	(α=0)	0.9904	0.5024			
X/Z	(α=0)	1.9399	0.3401			
A'/B'	(α=0)	1.5189	0.3970			
A'/C'	(α=0)	1.9403	0.3401			

Les résultas montrent que notre méthode est capable de détecter des similarités structurelles et sémantiques entre documents XML, non traités par les approches existantes (les dernières produisent des valeurs de similarités identiques en dépit de la présence de similarités structurelles/sémantiques – les valeurs de similarités pour Nierman and Jagadish [18], Dalamagas *et al.* [8], et Chawathe [6] sont omises dans le Tableau 1 pour simplifier la présentation), à l'exception des similarités structurelles entre *A/B* détectées par la méthode de Nierman and Jagadish [18] (cfr., Section 3.2). Des expérimentations rigoureuses sur des documents XML réels et synthétiques ont été effectuées afin de tester l'efficacité de notre approche en comparaison avec les méthodes existantes de similarité XML. Les résultats sont détaillés dans [24] et soulignent la supériorité de notre approche (en termes de *précision* et *rappel*).

4 Conclusion

Dans cette étude, nous proposons une approche hybride de comparaison des documents XML. Notre méthode combine les calculs de distance d'édition entre structures arborescentes et l'évaluation de la similarité sémantique entre mots/expressions, afin de détecter les similarités structurelles et sémantiques entre documents XML. Notre méthode permet à l'utilisateur d'adapter la mesure de similarité selon ses besoins en comparaison, en attribuant plus de poids aux caractéristiques structurelles ou sémantiques des documents.

Nous avons montré l'applicabilité de notre méthode dans un contexte de RI générique (employant des fragments adaptés de WordNet comme références sémantiques). Cependant, l'approche peut être exploitée dans des contextes plus spécifiques, comme par exemple la comparaison de séquences de protéines ou la comparaison de gènes (décrits sous formes de documents XML, ou de données semi-structurées) [1], la comparaison de documents multimédia MPEG-7 [17], … Dans telles applications, l'obtention de résultats de comparaison pertinents nécessite l'emploie de bases de connaissances spécifiques, complètes, et adaptées aux domaines d'applications considérés.

Nous sommes actuellement en train d'étendre notre méthode afin de considérer, non seulement les labels des nœuds des documents XML, mais aussi leurs contenus en information. En plus, nous souhaitons étendre notre méthode afin de comparer les grammaires XML (DTDs ou Schémas XML [9]), besoin centrale dans les applications de groupage (*clustering*) et d'intégration de données (permettant la génération de vues globales et unifiées des données, l'utilisateur pouvant ainsi accéder efficacement à des informations plus complètes). Tenir compte de la sémantique, dans les comparaisons de documents et grammaires XML, demeure un sujet d'étude prometteur.

References

[1] Adak S., Batra V.S., Bhardwaj D.N., Kamesam P.V., Kankar P., Kurhekar M.P., Srivastrava B., 2002. A System for Knowledge Management in Bioinformatics, In *Proceedings of CIKM'02*, Virginia USA, pp. 638-641.

[2] Amer-Yahia S., Lakshmanan L.K.S. and Pandit S. 2004. FleXPath: Flexible Structure and Full-Text Querying for XML. In *Proceedings of ACM SIGMOD*, pp. 83-94.

[3] Bertino E., Guerrini G., Mesiti M. 2004. A Matching Algorithm for Measuring the Structural Similarity between an XML Documents and a DTD and its Applications, *Elsevier Computer Science, 29*, pp.23-46.

[4] Boughanem M. 2006. Introduction to Information Retrieval, In *Proceedings of EARIA 06* (*Ecole d'Automne en Recherche d'Information et Application*), Chapter 1.

[5] Chawathe S., Rajaraman A., Garcia-Molina H., and Widom J. 1996.Change Detection in Hierarchically Structured Information. In *Proc. of the ACM SIGMOD'96*, Canada.

[6] Chawathe S. 1999. Comparing Hierarchical Data in External Memory. *VLDB'99*, pp. 90-101.

[7] Cobéna G., Abiteboul S. and Marian A. 2002. Detecting Changes in XML Documents. In *Proc. of the IEEE Int. Conf. on Data Engineering*, pp. 41-52.

[8] Dalamagas, T., Cheng, T., Winkel, K., and Sellis, T. 2006. A methodology for clustering XML documents by structure. *Information Systems, 31, 3,* pp.187-228.

[9] Fallside D., Walmsley P., XML Schema part 0: Primer Second Edition W3C, October 2004, http://www.w3.org/TR/xmlschema-0/

[10] Fuhr N. and Großjohann K. 2001. XIRQL: A Query Language for Information Retrieval. In: *Proceedings of ACM-SIGIR*, New Orleans, pp. 172-180.

[11] Guha S., Jagadish H.V., Koudas N., Srivastava D. and Yu T. 2002. Approximate XML Joins. In *Proceedings of ACM SIGMOD'02*, pp. 287-298.

[12] Liang W. and Yokota H. LAX: An Efficient Approximate XML Join Based on Clustered Leaf Nodes for XML Data Integration. In *Proceedings of BNCOD'05*, Springer LNCS 3567, (2005) pp 82-97.

[13] Lin D. 1998. An Information-Theoretic Definition of Similarity. *In Proceedings of the 15th Int. Conf. on Machine Learning*, pp. 296-304, Morgan Kaufmann Pub. Inc.

[14] Maguitman A. G., Menczer F., Roinestad H. and Vespignani A. 2005. Algorithmic Detection of Semantic Similarity. In *Proceedings of the 14th International WWW Conference*, 107-116.

[15] McGill M. J. 1983. *Introduction to Modern Information Retrieval*. McGraw-Hill, New York.

[16] Miller G. 1990. WordNet: An On-Line Lexical Database. *International Journal of Lexicography*, 3(4).

[17] Moving Pictures Experts Group, MPEG-7, http://www.chiariglione.org/mpeg/standards/mpeg-7/mpeg-7.htm (visited on the 17th of September 2008).

[18] Nierman A. and Jagadish H. V. 2002. Evaluating structural similarity in XML documents. In *Proceedings of the 5th SIGMOD Workshop on The Web and Databases.*

[19] Salton G. and Buckley C. 1988. Term-weighting approaches in automatic text retrieval. *Inf. Process. Manag. 24*, 5, 513 -523.

[20] Schlieder T. 2001. Similarity Search in XML Data Using Cost-based Query Transformations. In *Proceedings of 4th SIGMOD Workshop on The Web and Databases*.

[21] Tekli J., Chbeir R. and Yetongnon K. 2006. Semantic and Structure Based XML Similarity: An Integrated Approach, In *Proceedings of the 13th International Conference on Management of Data (COMAD'06)*, Delhi, India, pp. 32-43.

[22] Tekli J., Chbeir R. and Yetongnon K. 2007. Efficient XML Structural Similarity Detection using Sub-tree Commonalities. In *Proceedings of the 22nd Brazilian Symposium on Databases (SBBD'07)*, *ACM SIGMOD DiSC*, Joao Pessoa, Brazil, 116-130.

[23] Tekli J., Chbeir R., Yetongnon K., 2007. A Fine-grained XML Structural Comparison Approach. In *Proceedings of The 26th International Conference on Conceptual Modeling (ER'07)*, Auckland, New Zealand, (LNCS 4801), pp. 582-598.

[24] Tekli J., Chbeir R. and Yetongnon K., An XML Document Comparison Framework. To appear in *Information Systems Journal*, 2008.

[25] Zhang K. and Shasha D. 1989. Simple Fast Algorithms for the Editing Distance Between Trees and Related Problems. *SIAM J. of Computing, 18, 6*, pp. 1245-1262.

[26] Zhang Z., Li R., Cao S. and Zhu Y. 2003. Similarity Metric in XML Documents. *Knowledge Management and Experience Management Workshop*.

Joe TEKLI a obtenu son diplôme d'Ingénieur en Télécommunications de l'Université des Pères Antonins, Liban, en 2005 (classé majeur de promo, avec mention Très Bien). Bénéficiant d'une Bourse de l'Agence Universitaire de la Francophonie (AUF), il a obtenu son Master Recherche en Informatique et Image de l'Université de Bourgogne, en 2006 (classé majeur de promo, avec mention Très Bien). Il parfait actuellement sa Thèse en Informatique au Laboratoire LE2I-CNRS Université de Bourgogne, bénéficiant d'une allocation de recherche attribuée par le Ministère de l'Education et de la Recherche. Ses travaux de recherche se concentrent autour de la comparaison de documents/grammaires XML et leurs applications, ainsi que l'intégration de données RSS, et la fragmentation des données multimédia. Il est membre de la section française de ACM SIGAPP et membre organisateur de la conférence SITIS. Ses travaux ont fait l'objet de publications dans des conférences et journaux internationaux (à titre indicatif, *Elsevier CS Review*, *ER*, *WISE*, *COMAD*).

Richard CHBEIR a obtenu son doctorat en informatique à l'INSA de Lyon en 2001. Il a rejoint le laboratoire LE2I-CNRS de l'Université de Bourgogne à Dijon en Septembre 2002. Il s'intéresse aux problèmes de recherche d'information multimédia et XML, et aux modèles de contrôle d'accès complexes. Il a plusieurs publications à son actif dans des journaux (IEEE Transactions on SMC, Journal of Methods of Information in medicine, JAS, etc.) et conférences internationaux (ACM SAC, Visual, IEEE CIT, FLAIRS, PDCS, etc.), et est impliqué comme membre de comité de programme dans plusieurs manifestations scientifiques (IEEE Multimedia, ACM SAC, EuroPar, SBBD, etc.).

Kokou YETONGNON est Professeur à l'Université de Bourgogne, et chef de l'équipe de Gestion Distribuée d'Informations au sein du laboratoire LE2I UMR-CNRS.