Fouille dans la structure de documents XML

Amandine Duffoux, Omar Boussaid,
Stéphane Lallich, Fadila Bentayeb

ERIC – Université Lumière Lyon 2
5 avenue Pierre Mendès-France – 69676 Bron Cedex – France
{ aduffoux | boussaid | lallich | bentayeb }@eric.univ-lyon2.fr

Résumé. La prolifération des documents XML appelle des techniques appropriées pour extraire et exploiter l'information contenue dans ces documents. On distingue deux approches de fouille : *XML Content Mining* portant sur le contenu et *XML Structure Mining* qui a trait à la structure des documents. Combiner ces deux approches est très intéressant. Les informations contenues dans la structure orientent la fouille sur le contenu. Nous présentons la première étape de cette démarche : une nouvelle méthode d'extraction des règles d'association à partir de la structure des documents XML qui permet de gérer les aspects hiérarchiques de ces documents tout en améliorant les mécanismes d'extraction grâce à la création d'une structure spéciale représentant la hiérarchie des balises rencontrées.

Mots-Clés : document XML, structure, règles d'association.

1 Introduction

La norme XML (eXtensible Markup Language) s'impose comme le nouveau standard de transport des données. Son succès est dû à sa capacité de décrire toutes sortes de données à travers les concepts de DTD (*Document Type Definition*) ou de schémas XML qui sont de véritables grammaires. Il devient donc essentiel de mettre en place des techniques appropriées pour extraire et exploiter les informations contenues dans ces documents. La fouille de données dans les documents XML [Garofalakis *et al.*, 1999] se divise en fouille sur le contenu (*XML Content Mining*) et en fouille à partir de la structure (*XML Structure Mining*). La fouille sur le contenu utilise habituellement les techniques de *text mining*. Certains auteurs [Braga *et al.*, 2002] se sont notamment intéressés à l'extraction de règles d'association. Néanmoins, ces travaux ont très peu pris en compte, voir pas du tout, l'aspect hiérarchique existant entre les balises. L'intérêt de la fouille à partir de la structure des documents XML (structure intra et inter-documents) [Nayak *et al.*, 2002] porte sur l'information que véhicule l'organisation hiérarchique des balises. Certains travaux [Moh *et al.*, 2000] ont porté sur l'extraction d'une DTD à partir d'un ensemble de documents XML de même structure. Ces deux approches, jusqu'à présent séparées, sont complémentaires. Il nous paraît donc judicieux d'orienter la fouille sur le contenu grâce aux connaissances extraites à partir de la structure. Pour cela, nous souhaitons dégager les liens existants entre les balises (qu'elles soient ou non imbriquées) et utiliser ces résultats pour étudier leur contenu.

Dans cet article, nous présentons la première étape de cette démarche, l'extraction de règles d'association à partir de la structure d'un ensemble de documents XML. Les règles d'association ont prouvé leur efficacité dans la découverte de relations intéressantes parmi une grande masse de données [Agrawal *et al.*, 1993]. La méthode que nous proposons permet non seulement de gérer les aspects hiérarchiques des documents

XML mais aussi d'améliorer les mécanismes d'extraction de ces règles. Or, l'extraction des règles d'association à partir de documents XML pose des problèmes liés au format des données et à la gestion de la hiérarchie entre les balises. Pour palier ces problèmes, nous proposons un pré-formatage des données et la création d'une DTD minimale représentant la hiérarchie de l'ensemble des balises rencontrées. Nous présentons ensuite l'algorithme *Apriori* et l'algorithme classique d'extraction de règles d'association adaptés à nos structures de données. Enfin, nous donnons les résultats obtenus sur un corpus de documents XML avec le prototype développé. Ces résultats valident notre approche puisque les temps de traitement et la qualité des règles extraites sont améliorés par rapport à un système n'utilisant pas de DTD minimale.

2 Spécificités des documents XML

Nous constatons un certain nombre de problèmes spécifiques liés aux documents XML. Contrairement à un contexte relationnel, le format de ces documents n'est pas adapté aux algorithmes "classiques" de recherche d'*itemsets* fréquents et d'extraction de règles d'association. Ainsi, pour pouvoir appliquer ces algorithmes, il faut extraire les informations sur les balises et structurer ces informations pour les rendre exploitables. Ce format spécifique impose de mettre en place des outils appropriés de gestion de la hiérarchie, afin de tirer pleinement profit de l'information supplémentaire contenue dans l'organisation hiérarchique des balises.

D'autre part, une même balise peut être renseignée plusieurs fois dans un même document et à des niveaux hiérarchiques différents. Il faut donc gérer ces balises sans introduire de redondance dans les traitements et sans perdre d'information sur la structure hiérarchique des balises.

3 Fouille à partir de la structure des documents XML

Notre démarche consiste à pré-formater les données XML afin de les rendre exploitables par les algorithmes classiques de fouille et mettre en place des structures adéquates pour la gestion de la hiérarchie entre les balises. Nous appliquons l'algorithme *Apriori* adapté à nos strucutres pour la recherche d'*itemsets* fréquents. Enfin, les règles d'association sont extraites et les résultats sont présentés sous forme de documents XML.

3.1 Pré-formatage des données

La première tâche du pré-formatage consiste à extraire les balises. Ceci est effectué à l'aide du parseur événementiel SAX qui déclenche un traitement à chaque début ou fin d'un document ou d'un élément (balise). Au début d'un document, le parseur collecte des informations sur son *nom*, son *chemin* (en local ou sur internet) et le *nombre de balises* qu'il contient. Au début d'un élément, il collecte son *nom* et son *nombre d'occurrences* dans l'ensemble des documents (une balise est comptabilisée une seule fois par document) ainsi que *son (ses) parent(s)* et *son (ses) enfant(s)* dans l'arbre du document. A la fin du document, il enclenche l'archivage de celui-ci dans une liste chaînée. De même, à la fin d'un élément, la balise est archivée dans une autre liste chaînée. Si elle existe déjà dans cette liste, ses informations sont alors complétées par la nouvelle occurrence. Ainsi, si une balise apparaît plusieurs fois, elle ne sera archivée qu'une seule fois mais l'information sur sa hiérarchie est conservée. Soient les documents "sampl1.xml" et "sampl2.xml" et les résultats obtenus pour la balise jour et le document "sampl2.xml" :

<Personne> <Nom> Dupont </Nom> <Prénom> Marie </Prénom> <Naissance> <Jour> 12 </Jour> <Mois> décembre </Mois> <Année> 1930 </Année> </Naissance> </Personne>	<Personne> <Nom> Martin </Nom> <Naissance> <Jour> 29 </Jour> <Mois> février </Mois> <Année> 1970 </Année> </Naissance> <Décès> <Jour> 4 </Jour> <Mois> mars </Mois> <Année> 2002 </Année> </Décès> </Personne>	Nom Balise = "Jour" Nb d'occurences = 2 *(comptée une fois par document)* Parents = {Naissance, Décès} Enfants = ensemble vide Nom Document= "Sampl2.xml Chemin = "C ://Exemple Nombre de balises = 7

Lors de l'analyse des documents par le parseur SAX, une matrice booléenne est constituée indiquant si une balise est renseignée ou non dans un document. Cette matrice permettra d'accéder rapidement aux balises composant un document ou aux documents comprenant une balise donnée.

3.2 Gestion de la hiérarchie

Afin de gérer les liens hiérarchiques existants entre les balises, nous mettons en place des structures de données spécifiques dont la principale est la *DTD minimale*. Une DTD est une grammaire contenant la description de la structure des documents XML qui lui sont attachés. Dans notre cas, nous n'avons qu'un sous-ensemble des documents XML respectant un même schéma. Nous créons, à partir de ce sous-ensemble, une DTD contenant uniquement les balises rencontrées en ne prenant en compte que leurs caractéristiques effectives. Ainsi, nous obtenons la DTD minimale pour l'ensemble des documents analysés. La construction de cette DTD se fait en plusieurs étapes.

Nous créons d'abord un **arbre de l'ensemble des balises**. L'arbre est une représentation logique du document XML. Il sert à décrire la hiérarchie de l'ensemble des balises lui appartenant. Lors de l'analyse d'un document XML par le parseur, un arbre temporaire associé au document est créé selon l'algorithme suivant :

```
Création d'une pile vide
Création d'un arbre contenant uniquement une racine vide
Tant qu'il reste des balises dans le document faire
        Si Balise ouvrante rencontrée alors
                Si la racine de l'arbre est vide alors ajouter cette balise à la racine
                Sinon ajout de cette balise dans l'arbre sous le noeud représentant
                                la balise placée en tête de la pile
                        ajout de cette balise au sommet de notre pile
                Fin si
        Sinon ('Balise fermante') suppression du sommet de la pile
        Fin si
Fin Tant que
```

Il sera fusionné dans un arbre général (Fig. 1), où chaque noeud représente une balise. La racine correspond à l'élément principal et les feuilles sont les éléments atomiques. Ainsi, une branche de l'arbre représente une imbrication de balises. Notons que la liste des balises ne contient qu'une seule occurrence de chaque balise alors que l'arbre contient toutes les balises pour assurer une lecture correcte de chaque document.

Ensuite, nous **reconstituons la DTD**. L'arbre général représente la structure hiérarchique de l'ensemble des balises rencontrées. La DTD indique en plus si une balise est conditionnelle, i.e. si elle n'est pas toujours renseignée, ou s'il existe une liste de balises. Ces deux conditions sont représentée respectivement par *?* et *+*. Afin de reconstituer la DTD minimale, nous allons utiliser l'arbre général et les informations

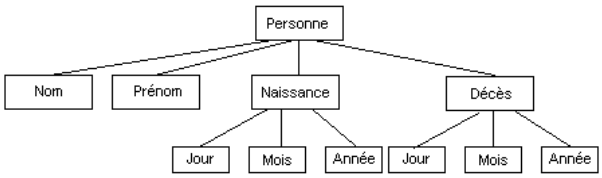

FIG. 1 – *Structure de l'arbre général des documents Sampl1 et Sampl2*

concernant la présence et la fréquence des balises dans chaque document. Notons que cette DTD minimale ne contient que des éléments simples, suffisants pour l'utilisation qui en est faite. Dans notre exemple, la DTD minimale extraite sera la suivante :

```
<! ELEMENT Personne (Nom, Prénom?, Naissance, Décès?)>
<! ELEMENT Nom(#PCDATA)>
<! ELEMENT Prénom(#PCDATA) >
<! ELEMENT Naissance (Jour, Mois, Année)>
        <! ELEMENT Jour(#PCDATA)>
        <! ELEMENT Mois(#PCDATA)>
        <! ELEMENT Année(#PCDATA)>
<! ELEMENT Décès (Jour, Mois, Année)>
```

3.3 Recherche des itemsets fréquents

En plus de l'adaptation de *Apriori* à nos structures de données, nous avons ajouté des traitements au niveau de la constitution des *itemsets* candidats afin de prendre en compte la structure hiérarchique des documents XML.

Lors de la constitution des 1-*itemsets* candidats, nous prenons l'ensemble des balises extraites lors de la phase de pré-formatage et nous supprimons celles qui sont renseignées dans l'ensemble des documents. Elles n'apportent aucune information supplémentaire dans notre cas. De plus, on ne sélectionne que les balises qui ne présentent pas de redondance par rapport à la hiérarchie grâce à la DTD minimale. Pour l'exemple Sampl1 et Sampl2, les balises *Jour*, *Mois* et *Année* sont obligatoirement renseignées si *Naissance* l'est. Elles n'apportent donc aucune information supplémentaire par rapport à cette dernière du point de vue de la structure. Nous supprimons donc les balises *Jour*, *Mois* et *Année* des 1-*itemsets candidats*.

Lors de la constitution de l'ensemble des k-*itemsets* candidats, nous utilisons également la DTD minimale pour supprimer les redondances. Dans notre exemple, *Prénom* n'est pas forcément renseignée si *Personne* l'est, elle n'a donc pas été supprimée lors de la constitution des 1-*itemsets* candidats. Or, si un *itemset* contient à la fois *Prénom* et *Personne*, il est redondant. Dans ce cas, nous supprimons l'item *Personne* de l'*itemset* et supprimons, si nécessaire, les doublons.

3.4 Extraction des règles d'association

Nous effectuons l'extraction de deux types de règles d'association. Les **règles d'association "classiques"** sont extraites grâce à l'algorithme "classique" d'extraction de règles d'association [Agrawal *et al.*, 1993] adapté à nos structures de données. Les **règles d'association "hiérarchiques"** permettent de représenter la fréquence des liens entre des balises directement imbriquées. Chaque balise admettant des balises

imbriquées est considérée comme antécédent d'une règle. Toutes les balises imbriquées dans celle-ci sont considérées comme conséquence de cette règle. Dans ces deux cas, les règles d'association extraites sont présentées **sous forme de documents XML**. En effet, grâce aux fonctionnalités offertes par ce format, il est simple de représenter la connaissance extraite. Les résultats en XML respectent la DTD suivante :

```
<!ELEMENT Rule(BodyList, HeadList, Support, Confidence)>
<!ELEMENT BodyList(Body+)>
<!ELEMENT Body(#PCDATA)>
<!ELEMENT HeadList(Head)>
<!ELEMENT Head(#PCDATA)>
<!ELEMENT Support(#PCDATA)>
<!ELEMENT Confidence(#PCDATA)>
```

4 Expérimentation

Afin de valider notre méthode, nous avons développé un prototype en Java. Nous avons successivement extrait les règles d'association sans et avec utilisation de la DTD minimale à partir d'un ensemble de 400 articles du dictionnaire national américain de médecine MEDLINE®, distribué en format XML[1].

Lors de la phase d'extraction sans recours à la DTD minimale, pour des valeurs de support et de confiance minimum respectivement de *5%* et *75%*, nous obtenons plus de *15 000* règles en *4.234 s* (temps CPU). De plus, ces règles présentent de fortes redondances par rapport à la hiérarchie. Lors de la phase d'extraction avec recours à la DTD minimale, pour les mêmes seuils de support et de confiance, nous obtenons moins de *5 000* règles, soit moins du tiers que précédemment. Nous obtenons nos règles en *2.659 s*, soit un gain de temps de 38%.

Les résultats obtenus sont encourageants puisque lors de l'extraction avec recours à la DTD minimale nous n'avons plus de redondances dans les règles extraites. Cette suppression a pour conséquence naturelle de diminuer le nombre de ces règles. Ainsi, la DTD minimale permet de réduire significativement les temps de traitement. De plus, notre méthode sera d'autant plus performante que le nombre de niveaux de hiérarchie sera important.

5 Conclusion et perspectives

Dans cet article, nous avons présenté une méthode originale d'extraction de règles d'association à partir de la structure de documents XML. Cette méthode permet de représenter les liens existants entre les balises d'un ensemble de documents XML de même structure. Nous avons développé un prototype qui confirme l'apport de notre méthode à la tâche d'extraction de règles d'association dans le cadre de la structure de documents XML. La DTD minimale permet de réduire de manière significative le nombre d'*itemsets* candidats et d'améliorer le temps de traitement de *Apriori*. Nous obtenons un ensemble restreint de règles pertinentes. Notre travail peut être utile pour la mise en place d'une plateforme de gestion des documents XML (réparation et création d'une DTD minimale pour des documents n'ayant pas un même schéma).

Cette extraction constitue une première étape d'une plus large démarche de *XML Mining*. Notre approche va permettre d'orienter la fouille sur le contenu, notamment celui des balises ayant un lien entre elles. Enfin, dans une démarche de représentation des

[1] http ://www.nlm.nih.gov/bsd/licensee/data_elements_doc.html

données hétérogènes (textes, images, vidéos, sons, ...) dans le format XML[Darmont *et al.*, 2002], le recours à cette méthode de fouille s'avère pertinent.

Références

[Agrawal *et al.*, 1993] R. Agrawal, T. Imielinand, et A. Swami. Mining association rules between sets of items in large databases. In P. Buneman et S. Jajodia editors, editors, *SIGMOD93*, pages 207–216, Washington DC,USA, mai 1993.

[Braga *et al.*, 2002] D. Braga, A. Campi, M. Klemettinen, et P.L. Lanzi. Mining association rules from xml data. In *Data Warehousing and Knowledge Discovery, 4th International Conference, DaWaK 2002*, volume 2454, pages 21–30, Aix-en-Provence, France, Septembre 2002. Springer.

[Darmont *et al.*, 2002] J. Darmont, O. Boussaid, F. Bentayeb, S. Rabaseda, et Y. Zellouf. *Multimedia Systems and Applications*, volume 22, chapter Web multiform data structuring for warehousing, pages 9–27. Kluwer Academic Publishers, 2002.

[Garofalakis *et al.*, 1999] M. Garofalakis, Rastogi, S. Seshredi, et K. Shim. Data mining and the web :past, present and future. In *2nd International ACM Workshop on Web Information and Data Management(WIDM)*, pages 43–47, Kansas city,USA, 1999.

[Moh *et al.*, 2000] C.H. Moh, E.P. Lim, et W.K. Ng. Dtd-miner : A tool for mining dtd from xml documents. In *Second International Workshop on Advance Issues of E-Commerce and Web-Based Information Systems (WECWIS 2000)*, page 144, Milpitas, California, june 2000.

[Nayak *et al.*, 2002] R. Nayak, R. Witt, et A. Tonev. Data mining and xml documents. In *Proceedings of the 2002 International Conference on Internet Computing*, Nevada, USA, June 2002.

Summary

XML is created to structure, store and send information. It plays an increasily important role in the exchange of data on the Web and elsewhere. A large number of software vendors have adopted this standard. Thus, it will be essential for discovering new knowledge from many XML documents likely to arise in the next few years. Then, there is a great need to apply data mining techniques to retreive and analyse vast amount of XML documents.

Two main and complementary approaches are used to mine XML documents. XML content mining and XML strcuture mining. Mining for XML content is especially mining for values. Mining for XML strcuture is essentially mining for schema. Since XML is semi-structured data, mining for XML strcuture is of interest.

In this paper, we propose a new method for XML strcuture mining using association rules. This approach prensents two main advantages. First, it takes into account the hierarchical aspect of the XML documents and secondly, it improves the mechanism of rule extracting.