International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-2 Issue-1, October 2012

Extracting Information from Semistructured Xml Using Tars

R.Sree Lekshmi, B. Sasi kumar

Abstract - Extracting information from semi structured documents is a very hard task, and is going to become more and more critical as the amount of digital information available on the internet grows. Indeed, documents are often so large that the dataset returned as answer to a query may be too big to convey interpretable knowledge. This work describe an approach based on Tree-based Association Rules (TARs) mined rules, which provide approximate, intentional information on both the structure and the contents of XML documents. This mined knowledge is used to provide: structure and the content of the XML document and quick, approximate answers to queries.

Keywords—XML, approximate query-answering, data mining, intensional information, succinct answers.

I. INTRODUCTION

In recent years the database research field has concentrated on XML (extensible Markup Language [2]) as a flexible hierarchical model suitable to represent huge amounts of data with no absolute and fixed schema, and a possibly irregular and incomplete structure. There are two main approaches to XML document access: *keyword-based search* and *query-answering*. The first one comes from the tradition of information retrieval [3], where most searches are performed on the textual content of the document. For *query-answering*, since query languages for semi structured data rely on document structure to convey its semantics, in order for query formulation to be effective and users need to know this structure in advance.

In fact, it is not mandatory for an XML document to have a defined schema: 50% of the documents on the web do not possess one [5].Frequent, dramatic outcomes of this situation are either the *information overload* problem, where too much data are included in the answer because the set of keywords specified for the search captures too many meanings, or the *information deprivation* problem, where either the use of inappropriate keywords, or the wrong formulation of the query, prevent the user from receiving the correct answer.

The goal of this paper is to provide a method for mining intentional knowledge from XML datasets expressed by means of association rules. We introduce a proposal for mining, and also storing tree-based association rules for two main purposes:

1) To get a concise view of both the structure and the content of XML documents, and 2) to use them for intentional query answering. Our mining procedure is characterized by the following key aspects: 1) it works directly on the XML documents, without transforming the data into relational or any other intermediate format, 2) it looks for general association rules, without the need to impose what should be contained in the antecedent and consequent of the rule, and 3) it stores association rules in XML format. The aim of our proposal is to provide a way to use intentional knowledge as a substitute of the original document during querying and to improve the execution time of the queries over the original XML dataset, like in [7]. Accordingly, the paper's contributions are: an improved version of the TARs extraction algorithm introduced in [8], which was based on PathJoin [9]. The new version uses the better-performing CMTreeMiner [8] to mine frequent subtrees from XML documents. Approach validation by means of experimental results, considering both the previous and the current algorithm and showing the improvements.

II. TREE BASED ASSOCIATION RULES

Association rules [1] describe the co-occurrence of data items in a large amount of collected data and are represented as implications of the form $X \Rightarrow Y$, where X and Y are two arbitrary sets of data items, such that $X \cap Y = \emptyset$. The quality of an association rule is measured by means of support and confidence. Support corresponds to the frequency of the set $X \cup Y$ in the dataset, while confidence corresponds to the conditional probability of finding Y, having found X and is given by $supp(X \cup Y)/supp(X)$.

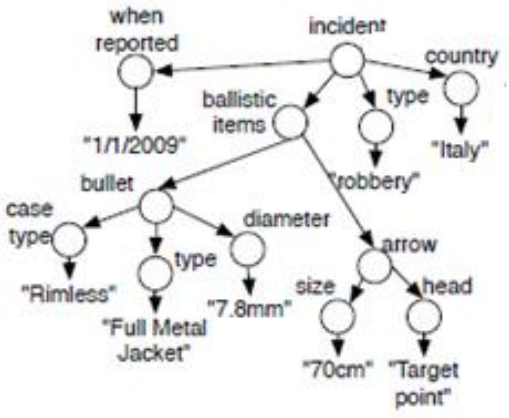

Fig.1. Sample XML file:"incidents.xml"

We are interested in finding relationships among sub trees of XML documents. Thus, since both textual content of leaf elements and values of attributes convey "content", we do not distinguish between them. As a consequence, for the sake of readability, we do not report the edge label and the node type label in the figures. Attributes and elements are characterized by empty circles, whereas the textual content of elements, or the value of attributes, is reported under the outgoing edge of the element or attribute it refers to (see Figure 1).

Manuscript published on 30 October 2012.
***** Correspondence Author (s)
R. Sree Lekshmi, Dept. of Computer Science & Engineering. The Rajas Engineering College, Vadakangulam, India.
B. Sasi kumar, Professor, Dept of Computer Science The Rajas Engineering College, Vadakangulam. India.

We represent an XML document as a tree (N, E, r, L, c) where N is the set of nodes, $r \in N$ is the root of the tree, E is the set of edges, $l: N \rightarrow L$ is the label function which returns the tag of nodes (with L the domain of all tags) and $c: N \rightarrow c \cup \{\perp\}$ is the content function which returns the content of nodes (with C the domain of all contents).

Given two trees T=(N_T, E_T , R_T , L_T, C_T) and S=(N_S, E_S, R_S, L_S, C_S), S is an induced sub tree of T if and only if there exists a mapping $\theta : N_S \rightarrow N_T$ such that for each node $ni \in N_S$, $L_T(ni) = L_S(nj)$ and $C_T(ni) = C_S(nj)$,where $\theta(ni) = nj$,and for each edge $e = (n1, n2) \in E_S$, $(\theta(n1), \theta(n2)) \in E_T$. Moreover, S is a rooted subtree of T if and only if S is an induced subtree of T and $rS = rT$ .Given a tree $T = (N_T, E_T, R_T, L_T, C_T)$ a subtree of T, $t = (N_t, E_t, R_t, L_t, C_t)$and a user-fixed support threshold s_{min} :(i) t is frequent if its support is greater or at least equal to s_{min} (ii) t is maximal if it is frequent and none of its proper super trees is frequent; (iii) t is closed if none of its proper supertrees has support greater than that of t. Figure 1shows an example of an XML document (Figure 2a), its tree-based representation (Figure 2b), two induced subtrees (Figure 2c) and a rooted subtree (Figure2d)

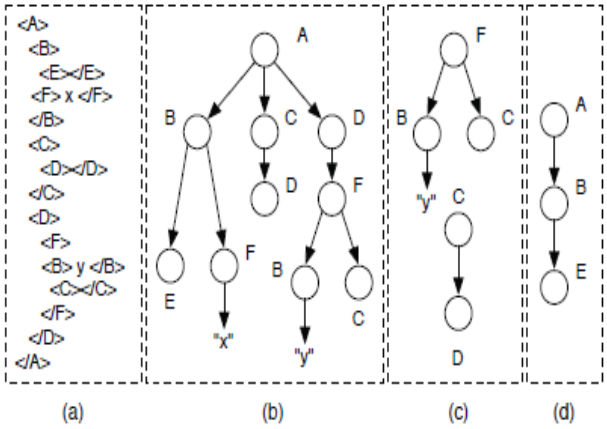

Fig.2. (a) An example of XML document, (b) Its tree-based representation, (c) Two induced sub trees, and (d) a rooted sub tree only

III. TAR EXTRACTION AND PROCESSING

Tree-based rules are extracted from an XML document. Moreover, we explain how they are stored and used to respond to the types of queries. Association rules describe the co-occurrence of data items in a large amount of collected data the quality of an association rule is usually measured by means of support and confidence. Given function count(S,D) denoting the number of occurrences of a sub tree S in the tree D and function cardinality(D) denoting the number of nodes of D, it is possible to define formally the two measures as:

$$support(S_B \Rightarrow S_H) = \frac{count(S_H, D)}{cardinality(D)}$$

$$confidence(S_B \Rightarrow S_H) = \frac{count(S_H, D)}{count(S_B, D)}$$

Given an XML document it is possible to extract two types of tree-based association rules:

iTAR: instance TARs are association rules providing information both on the structure and values contained in a target XML document.

sTAR: structure TARs are association rules on the structure of the XML document.

Figure3 shows a sample XML document and some sTARs. Rules (1) and (3) are rooted sTARs; rule (2) is an extended sTAR. Rule (1) states that, if there is a node labeled *A* in the document, with a 86% probability that node has a child labeled *B*. Rule (2) states that, if there is a node labeled *B*, with a 75% probability its parent is labeled *A*. Finally, Rule (3) states that, if a node *A* is the grandparent of a node *C* (notice the empty node, parent of node *C*), with a 75% probability the child of *A* and parent of *C*, is labeled *B*. By observing sTARs users can guess the structure of an XML document, and thus use this approximate schema to formulate a query when no DTD or schema is available.

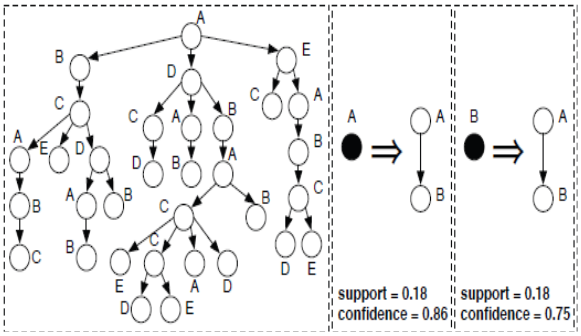

Fig.3. Graphical representation of STARs

A. ASSOCIATION RULE EXTRACTION

Mining tree-based association rules is a process composed of two steps:

1. Mining frequent sub trees from the XML document;
2. Computing interesting rules from the previously mined frequent sub trees.

The inputs of the algorithm are the set of frequent subtrees F_S, and the minimal threshold for the confidence of the rules, minconf.

Algorithm1Get-Interesting-Rules (F_s, Minconf)

1: rules Set =∅ ;
2: for all s∈ F_s do
3: tempest = Compute-Rules(s, minconf)
4: rules Set = $ruleSet \cup tempSet$
5: end for
6: return rule Set

Depending on the amount of frequent sub trees and their cardinality, the amount of generated rules may be very high. The explosion of the number of mined association rules.

Algorithm 2 Compute-Rules (s, minconf)

1: rule Set =∅ ;
2: for all cs sub trees of s do
3: conf = supp(s) / supp (c_s)
4: if conf ≥ minconf then
5: new Rule =< cs; s; conf, supp(s)
6: rule Set = rule Set ∪ newRuleg
7: end if
8: end for
9: return ruleSet

The number of frequent item sets occurs also in the relational context and optimization of the basic algorithm has been proposed in [2]. Such optimization will be adapted to our XML context, for mining tree-based association rules.

In fact, given a frequent sub tree S, all rules derived from that tree have the same support and different confidences.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

Since the confidence of a rule $S_B \rightarrow S_H$ can be computed as support (S_H)/support(S_B), the support of S_B influences the confidence of rules having the same body tree; the higher the support of the body tree, the lower is the confidence of the rule.

B. XML MINING

The problem of computing interesting rules from frequent sub trees can be compared with the problem of extracting classical association rules from large sets of elements; initially in [2] Algorithm1 shows how tree-based association rules are mined. The inputs of the algorithm are the set of frequent sub trees, FS, and the minimal threshold for the confidence of the rules, minconf.

TAR mining is a process composed of two steps:

(i) Mining frequent sub trees, that is, sub trees with a support above a user defined threshold, from the XML document.

(ii) Computing interesting rules, that is, rules with a confidence above a user defined threshold, from the frequent sub trees.

C . INTENTIONAL QUERY ANSWERING

ITARs can be used to provide intentional information about the actual data contained in the mined XML documents, when these documents are no available or reachable any more, or when the user prefers to obtain a faster possibly partial answer. Of course, this choice is worthwhile only if intentional queries processing is faster than extensional query processing. Therefore we introduce indices on tree-based association rules in order to improve the access to mined trees and in general the process of intentional query.

IV. CONCLUSION

The main goals we have achieved in this work are: 1) mine all frequent association rules without imposing any a-priori restriction on the structure and the content of the rules; 2) store mined information in XML Format; 3) use extracted knowledge to gain information about the original datasets.

REFERENCES

1. Data mining for XML query-answering support Mirjana Mazuran, Elisa Quintarelli,and Letizia Tanca “IEEE Transaction on knowledge and Data Engineering”
2. World Wide Web Consortium. Extensible Markup Language (XML) 1.0,\1998.http://www.w3C.org/ xml/.
3. GaryMarchionini.Exploratorsearch: from finding to uderstanding. Communications of the ACM, 49(4):41–46, 2006.
4. R.Agrawal and R.Srikant. Fast algorithms for mining association rules in large databases. In Proc. of the 20th Int. Conf. on Very Large Data Bases, pages 487–499. Morgan Kaufmann Publishers Inc., 1994.
5. D. Barbosa, L. Mignet, and P. Veltri. Studying the xml web: Gathering statistics from an xml sample. World Wide Web, 8(4):413–438, 2005.
6. T. Asai, K. Abe, S. Kawasoe, H. Arimura, H. Sakamoto, and S. Arikawa.Efficient substructure discovery from large semi-structured data. In Proc.of the SIAM Int. Conf. on Data Mining, 2002.
7. K. Wong, J. X. Yu, and N. Tang. Answering xml queries using pathbased indexes: A survey. World Wide Web, 9(3):277–299, 2006.
8. Y. Xiao, J. F. Yao, Z. Li, and M. H. Dunham. Efficient data mining for maximal frequent subtrees. In Proc. of the 3rd IEEE Int. Conf. on Data Mining, page 379. IEEE Computer Society, 2003.
9. Y. Chi, Y. Yang, Y. Xia, and R. R. Muntz. Cmtreeminer: Mining both closed and maximal frequent subtrees. In Proc. of the 8th Pacific-Asia Conf. on Knowledge Discovery and Data Mining, pages 63–73, 2004.
10. M. Mazuran, E. Quintarelli, and L. Tanca. Mining tree-based frequent patterns from xml. In Proc. of the 8th Int. Conf. on Flexible Query Answering Systems.

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)