ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Data Mining For XML Query-Answering Support (2015)

Pataballa Jayasree
M.Tech Student,
Dept. of Computer Science and Technology,
Andhra University.

Vizianand Row
Assistant Professor
Dept. of Computer Science and Technology,
Andhra University.

Abstract:
While keyword query empowers ordinary users to search vast amount of data, the ambiguity of keyword query makes it difficult to effectively answer keyword queries, especially for short and vague keyword queries. To address this challenging problem, in this paper we propose an approach that automatically diversifies XML keyword search based on its different contexts in the XML data. Given a short and vague keyword query and XML data to be searched, we firstly derive keyword search candidates of the query by a simple feature selection model. And then, we design an effective XML keyword search diversification model to measure the quality of each candidate. After that, two efficient algorithms are proposed to incrementally compute top-k qualified query candidates as the diversified search intentions. Two selection criteria are targeted: the k selected query candidates are most relevant to the given query while they have to cover maximal number of distinct results. At last, a comprehensive evaluation on real and synthetic datasets demonstrates the effectiveness of our proposed diversification model and the efficiency of our algorithms.

Introduction

XML has become a popular format for storing and sharing data across heterogeneous platforms. The XML format is neutral, flexible and interoperable [5]. It is widely used in applications as it can allow applications to have communication though they are built in different platforms. The XML documents are plenty in enterprises and the data retrieval can be done in two ways. The first approach is that user gives keywords and the program searches for relevant documents. The second approach is give XML queries that are answered. The first approach is done using conventional information retrieval technique [4] that works on the search process based on the given search word. With respect to query answering, it is not easy to process such request. To make this searching easy this paper presents data mining for XML query answering support. XML documents are validated by either DTD or schema. However, schema presence is not mandatory to process XML file [3].

This paper presents data mining framework for XML query answering support. The XML documents essence is extracted and kept in another XML file in the form of TARs. With the help of this XML query answering becomes easy.

Problem Statement:

Even if the system processing time is acceptable by accelerating the keyword query evaluation with efficient algorithms the unclear and repeated search intentions in the large set of retrieved results will make users frustrated.

- A large number of structured XML queries may be generated and evaluated;
- There is no guarantee that the structured queries to be evaluated can find matched results due to the structural constraints;
- Similar, the process of constructing structured queries has to rely on the metadata information in XML data.

Existing System

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

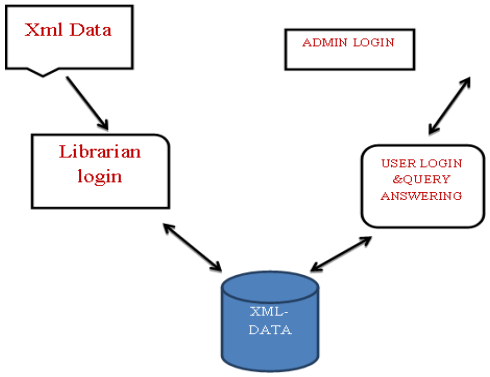
ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

as answer to a query may be too big to convey interpretable knowledge.

Proposed system

In this work we describe an approach based on Tree-based Association Rules (TARs) mind rules, which provide approximate, in tensional information on both the structure and the contents of XML documents, and can be stored in XML format as well.

SYSTEM ARCHITECTURE:



Proposed Framework

The proposed XML query answering support framework is as shown in fig. 1. The purpose of this framework is to perform data mining on XML and obtain intentional knowledge. The intentional knowledge is also in the form of XML. This is nothing but rules with supports and confidence. In other words the result of data mining is TARs (Tree-based Association Rules).

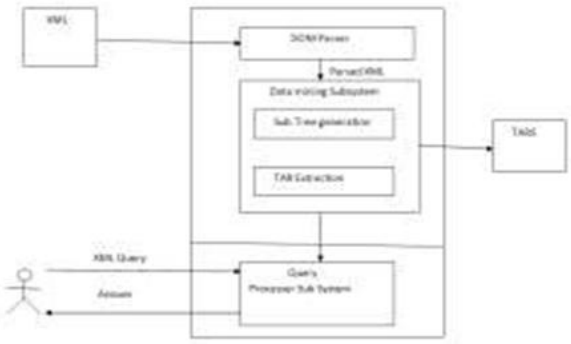


Fig. 1 – Proposed XML query answering support framework

As can be seen in fig. 1, the framework is to have data mining for XML query answering support. When XML file is given as input, DOM parser will parse it for wellformedness and validness. If the given XML document is valid, it is parsed and loaded into a DOM object which can be navigated easily. The parsed XML file is given to data mining sub system which is responsible for sub tree generation and also TAR extraction. The generated TARs are used by Query Processor Sub System. This module takes XML query from end user and makes use of mined knowledge to answer the query quickly.

Modules Description:

1. Admin Module
2. User Module
3. View Module
4. Download Module

Module Description:

Module 1: Admin Module

Admin will upload new connection form based on regulations in various states. Admin will be able upload various details regarding user bills like a new connection to a new user, amount paid or payable by user. In case of payment various details regarding payment will be entered and separate username and password will be provided to users in large.

Module 2: User Module

User will be able to view his bill details on any date may be after a month or after months or years and also he can to view the our bill details in a various ways for instance, The year wise bills, Month wise bills, totally paid to bill in EB. This will reduce the cost of transaction. If user thinks that his password is insecure, he has option to change it. He also can view the registration details and allowed to change or edit and save it.

Module 3: View Module

Admin has three ways to view the user bill details, the 3 ways are

i) SPJ

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

ii) PIVOT
iii) CASE
i) SPJ: While using SPJ the viewing and processing time of user bills is reduced.
ii) PIVOT: This is used to draw the user details in a customized table. This table will elaborate us on the various bill details regarding the user on monthly basis.
iii) CASE: using CASE query we can customize the present table and column based on the conditions. This will help us to reduce enormous amount of space used by various user bill details. It can be viewed in two difference ways namely Horizontal and Vertical.
In case of vertical the number of rows will be reduced to such an extent it is needed and column will remain the same on other hand the Horizontal will reduce rows as same as vertical and will also increase the columnar format

Module 4: Download Module

User will be able to download the various details regarding bills. If he/she is a new user, he/she can download the new connection form, subscription details etc. then he/she can download his /her previous bill details in hands so as to ensure it.

Tar Extraction

Extracting TARs through data mining is a process with two steps. In the first step frequent subtrees that satisfy given support are mined are mined. In the second step interesting rules that have confidence above given threshold are calculated from the frequent subtrees. Finding frequent sub trees is described in [1], [2], [6], [7], [8], [9]. Algorithm 1 finds frequent sub trees and calculates interesting rules.

Algorithm 1 Get-Interesting-Rules $(D, minsupp, minconf)$

```
1: // frequent subtrees
2: F_S = FindFrequentSubtrees (D, minsupp)
3: ruleSet = ∅
4: for all s ∈ F_S do
5:   // rules computed from s
6:   tempSet = Compute-Rules(s, minconf)
7:   // all rules
8:   ruleSet = ruleSet ∪ tempSet
9: end for
10: return ruleSet
```

Function 2 Compute-Rules $(s, minconf)$

```
1:  ruleSet = ∅; blackList = ∅
2:  for all c_s, subtrees of s do
3:    if c_s is not a subtree of any element in blackList then
4:      conf = supp(s) / supp(c_s)
5:      if conf ≥ minconf then
6:        newRule = ⟨c_s, s, conf, supp(s)⟩
7:        ruleSet = ruleSet ∪ {newRule}
8:      else
9:        blackList = blackList ∪ c_s
10:     end if
11:   end if
12: end for
13: return ruleSet
```

The rules obtained from algorithm 1 are written to an XML file. Then indexing is made. Afterwards when XML queries are made, the proposed system uses index and TARs and quickly answers the query.

Experiments And Results

Environment

The environment used to develop the prototype application includes JSE (Java Standard Edition) 6.0, Net Beans IDE that run in Windows 7 OS. A PC with 2 GB RAM and 2.9x GHz processor is used. The Java SWING API is used to build graphical user interface while IO and JAXP (Java API for XML Parsing) are used for implementing functionality. The main application GUI is as shown in fig.

Fig. 2 – The GUI of the prototype application

As can be seen the GUI has provision to choose an XML file as input. It also allows choosing a file for storing extracted rules. A text area is provided to show the XML file content. View Tree button shows the XML file with graphical tree representation. Generate Large ItemSet button generates frequent sub treesthat will be used for further processing. On clicking the GenerateRuleFile button, it extracts TARs from the

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

given XML file and finally extracted rules are saved into the given TAR file. The QueryAnswering button invokes a form where user can enter queries. The query interface is shown in fig. 3.

HOME Page:

User Login:

Registration page:

Admin Login:

About Us Page:

OutPut Page:

Conclusion

In this paper we presented a framework for extracting TARs from given XML file so as to support XML queries. Towards this end, the aim of this paper is to mine frequent association rules and store the mined content in XML format; use the TARs to support query answering or to gain information from XML databases. A prototype application is built to test the efficiency of the proposed framework. The application takes XML filem as input and generates TARs and then finally index file that helps in query processing. The experimental results revealed that the proposed application is useful and can be used in real time applications.

References

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rulesin large databases. In Proc. of the 20th Int. Conf. on Very Large DataBases, pages 487–499. Morgan Kaufmann Publishers Inc., 1994.

[2] T. Asai, H. Arimura, T. Uno, and S. Nakano. Discovering frequentsubstructures in large unordered trees. In Technical Report DOITR216, Department of Informatics, Kyushu University. http://www.i.kyushuu.ac.jp/doitr/trcs216.pdf, 2003.

[3] D. Barbosa, L. Mignet, and P. Veltri. Studying the xml web: Gatheringstatistics from an xml sample. World Wide Web, 8(4):413–438, 2005.

[4] Gary Marchionini. Exploratory search: from finding to understanding.Communications of the ACM, 49(4):41–46, 2006.

[5] World Wide Web Consortium. Extensible Markup Language (XML) 1.0,1998. http://www.w3C.org/TR/REC-xml/.

[6] K. Wang and H. Liu. Discovering typical structures of documents: aroad map approach. In Proc. of the 21st Int. Conf. on Research andDevelopment in Information Retrieval, pages 146–154, 1998.

[7] Y. Xiao, J. F. Yao, Z. Li, and M. H. Dunham. Efficient data mining formaximal frequent subtrees. In Proc. of the 3rd IEEE Int. Conf. on DataMining, page 379. IEEE Computer Society, 2003.

[8] X. Yan and J. Han. Closegraph: mining closed frequent graph patterns.In Proc. of the 9th ACM Int. Conf. on Knowledge Discovery and DataMining, pages 286–295. ACM Press, 2003.

[9] M. J. Zaki. Efficiently mining frequent trees in a forest: algorithms and applications. IEEE Transactions on Knowledge and Data Engineering,17(8):1021–1035, 2005.Mirjana