# Frequent Pattern Mining for XML Query-Answering Support

**Alfiya Iqbal Ahmed Shaikh, Sanchika Bajpai**

***Abstract- Extracting information from semi structured documents is difficult task. It is more crucial as there is a huge amount of digital information on the Internet is growing rapidly. Sometimes, documents are often so large that the data set returned as answer to a query may be large to even convey interpretable knowledge. This paper describes an approach which takes RSS feeds as input for which Tree-Based Association Rules (TARs): mined rules are used. It provides more approximate and intentional information on both the structure and the contents of Extensible Markup Language (XML) documents which can then be stored in XML format as well. This generated mined knowledge is later used to provide: 1) The gist of the structure and the content of the XML document and 2) Quick and more approximate answers to queries. This paper focuses on the second feature. In this paper we show a novel approach for finding frequent patterns in XML documents.***

***Keywords:- XML, document, RSS, TARs, approximate.***

## I. INTRODUCTION

The data over the internet is not structured. It is thus not very easy to store and parse the stored data using databases. The database research field has concentrated on the Extensible Markup Language (XML) [3] 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.XML is used to represent huge amount of data without any absolute schema and structure. To retrieve information from XML document two techniques are used keyword search and query retrieval. Keyword search is used when we have to match exact desired word. Query retrieval is used whenever document is following certain schema but its availability of documents with schema is partially fulfilled. So when we [2] search the desired query over document when we are unknown of the schema it fails. Unstructured document causes excess of information to be included in answer which is not essential and formulation of query becomes difficult. If at all your formulation of query goes wrong the resultant system will thus fail to give exact expected answer. Mining of XML documents is quite different from structured data mining and text mining. The structure of an XML document is indicated by element tags and their nesting. It allows the representation of semi-structured and hierarchal data containing the values of individual items and the relationships between data items. Mining of contents along with structure provides new means into the process of knowledge discovery.

**Manuscript Received on July 2014.**

**Alfiya Iqbal Ahmed Shaikh**, Student of Master of Engineering, Department of Computer Engineering, Bhivrabai Sawant Institute of Technology & Research, Affiliated to Pune University, Wagholi, Pune, Maharashtra, India.

**Sanchika Bajpai**, Asst. Prof., Department of Computer Engineering, Bhivrabai Sawant Institute of Technology & Research, Affiliated to Pune University, Wagholi, Pune, Maharashtra, India.

As for query-answering, since query languages for semi-structured data rely on the document structure to convey its semantics, in order for query formulation to be effective users need to know this structure in advance, which is often not the case. It is not obligatory for an XML document to have a defined schema: 50 percent of the documents on web do not have any defined schema [5]. When users specify queries without knowing the document structure, they may fail to retrieve information which was there, but under a different structure. This limitation is a crucial problem which did not emerge in the context of relational database management systems. 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. As a consequence, when accessing for the first time a large data set, gaining some general information about its main structural and semantic characteristics helps investigation on more specific details. This paper addresses the need of getting the gist of the documents before querying it, both in terms of content and structure. Discover a recurrent pattern inside XML documents provides high-quality knowledge about the document content: frequent patterns are in fact intensional information about the data contained in the document itself; they specify the document in terms of a set of properties rather than by means of data [6]. As opposed to the detailed and precise information conveyed by the data, this information is partial and often approximate, but synthetic, and concerns both the document structure and its content. The basic concept is the need of 'getting the gist' of the document before querying it, both in terms of content and structure. Discovering recurrent patterns inside XML documents provides high-quality knowledge about the document content: frequent patterns are in fact intensional information about the data contained in the document itself; they specify the document in terms of a set of properties rather than by means of data.

## II. LITERATURE SURVEY

The problem of association rule mining was initially proposed and many implementations of the algorithms were developed in the database literature. Some methods use XQuery to extract association rules from simple XML documents. They propose a set of functions, written in XQuery, which implement the Apriori algorithm. Wan and Dobbie [2] show that their approach performs well on simple XML documents but it is very difficult to apply to complex XML documents with an irregular structure.

This limitation is overcome by where Braga et al, [1], introduced a proposal to enrich XQuery with data mining and knowledge discovery capabilities, by introducing XMINE RULE, an operator for mining association rules for native XML documents. They formalize the syntax and semantics for the operator and propose some examples of complex association rules. However, XMINE is based on the MINE RULE operator, which works on relational data only. This means that, after a step of pruning of unnecessary information, the XML document is translated into the relational format. Moreover, in many techniques, the designer is forced to specify the structure of the rule to be extracted and then to mine it, if possible. This means that the designer has to specify what should be contained in the body and head of the rule, i.e., the designer has to know the structure of the XML document in advance, and this is an unreasonable requirement when the document does not have a DTD. A document type definition (DTD) is a set of *markup declarations* that define a *document type* for an SGML-family markup language (SGML, XML and HTML). A DTD uses a terse formal syntax that declares precisely which elements and references may appear where in the document of the particular type, and what the elements' contents and attributes are. A DTD can also declare entities that may be used in the *instance* document, [6]. A Document Type Definition (DTD) defines the legal building blocks of an XML document. It defines the document structure with a list of legal elements and attributes. A DTD can be declared inline inside an XML document, or as an external reference. Another limitation of these approaches is that the extracted rules have a fixed root, thus once the root node of the rules to mine has been fixed, only its descendants are analyzed. Our idea is to take a more general approach to the problem of extracting association rules, i.e., to mine all frequent rules, without any Apriori knowledge of the XML data set.[7]. The same idea was presented by J. Paik and H.Y. Youn introducing HoPS, [8], for extracting association rules in a set of XML documents called as XML association rules. Such rules are called XML association rules and are implications of the form X → Y, where X and Y are fragments of an XML document. The two trees X and Y have to be disjoint; moreover, both X and Y are embedded sub-trees of the XML documents which means that they do not always represent the actual structure of the data. Another proposal had a limitation that it does not consider the possibility to mine general association rules within a single XML data set; achieving this feature is one of our goals. The idea of using association rules as summarized representations of XML documents was also introduced where the XML summary is based on the extraction of rules both on the structure (schema patterns) and on content (instance patterns) of XML data sets. Similarly our idea is to mine starting from frequent subtrees of the tree-based representation of the XML documents. This helps in effectively answering user queries.[3]. In [4], Wan and Dobbie use XQuery [2] to extract association rules from simple XML documents. They propose a set of functions, written in XQuery, which implement the Apriori algorithm [1].The existing system uses CMTreeMiner algorithm and expat library system for XML document parsing the C++ implementation for both ordered and unordered sub-tree extraction. DRYADEPARENT is currently the fastest tree mining algorithm and CMTreeMiner is the second with respect to efficiency. However, DRYADEPARENT extracts embedded sub-trees which are trees that maintain the ancestor relationship between nodes but do not distinguish, among the <ancestor, descendant> pairs, the <parent, child> ones. Moreover, both [1] and [4] force the designer to specify the structure of the rule to be extracted and then to mine it, if possible.

## III. IMPLEMENTATION DETAILS

In particular, the idea of mining association rules [1] to provide summarized representations of XML documents has been investigated in many proposals either by using languages (e.g., XQuery [2]) and techniques developed in the XML context, or by implementing graph- or tree-based algorithms. This paper proposes to develop an application to mine frequently occurring subtrees from the generated XML data set by RSS Feed links of different newspaper links. Then storing them in another XML document. These rules are extracted only if support and confidence of nodes is greater than the provided threshold values. Then apply our Frequent Pattern Miner algorithm for XML. The idea of using association rules as summarized representations of XML documents was also introduced in [4] where the XML summary is based on the extraction of rules both on the structure (schema patterns) and on content (instance patterns) of XML data sets. The limitations of this approach are: 1) the root of the rule is established a-priori and 2) the patterns, used to describe general properties of the schema applying to all instances, are not mined, but derived as an abstraction of similar instance patterns and are less precise and reliable.

### *A. Fundamental Concepts*

The idea of the project basically revolves around the concept of Tree base Association rule. Mining tree-based association rules is a process composed of two steps:

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

Association rule [1] is an hint of the form A U B, where the rule A and B are subset of the set C of element in a set of transactions D and A ł B= Ø. A rule X Y states that the transaction T that has the elements in A are likely to contain also the elements in B. Association rules are distinguished by two survey: the support, which gives the percentage of transactions present in D that contain both items A and B (A U B); the confidence, which calculates the percentage of transactions in D containing the element A that also contain the elements B (support (A U B)/support (B)). In XML context, both D and C are group of trees. In this work TAR is generated using Natural Language Processing. The project works for RSS news links of multiple newspapers. RSS stands for "Really Simple Syndication". It is a way to easily distribute a list of headlines, update notices, and sometimes content to a wide number of people. It is used by computer programs that organize those headlines and notices for easy reading.

RSS works by having the website author maintain a list of notifications on their website in a standard way. This list of notifications is called an "RSS Feed". People who are interested in finding out the latest headlines or changes can check this list. Special computer programs called "RSS aggregators" have been developed that automatically access the RSS feeds of websites you care about on your behalf and organize the results for you.

Producing an RSS feed is very simple and hundreds of thousands of websites now provide this feature, including major news organizations like the New York Times, the BBC, and Reuters, as well as many weblogs. RSS provides very basic information to do its notification. It is made up of a list of items presented in order from newest to oldest. Each item usually consists of a simple title describing the item along with a more complete description and a link to a web page with the actual information being described. Sometimes this description is the full information you want to read (such as the content of a weblog post) and sometimes it is just a summary.

### *B. Architecture Details*

This paper introduces a proposal for mining and storing Tree-Based Association Rules (TARs) as a means to represent intensional knowledge as an alternative, synthetic data set would be queried for providing quick and summarized answers. For calculating the TAR we make use of Natural Language Processing. The algorithm for finding TAR is given below :

**Algorithm:** Extracting Tree Based Association Rules from XML Files

***Input:*** *RSS Feed Files* $R_n$, *Similarity Threshold* $T_h$, *Support S, Confidence C.*

***Output:*** *XML Tree Based Association Rules*

1. Read all $R_n$ by using XML Reader.
2. Extract Inner Element **<*Title*>** Tags within all $R_n$ and insert them into a new XML File $x_f$.
3. Initialize NodeList ***Lst*** to contain Visited $T_{sub}$ in $x_f$.
4. Initialize ***ItemSet***, a Key Value Pair structure to contain Element ***e*** and its support $f_s$.
5. For Each Subtree $T_{sub}$ in ***Xf***
6. Extract <Description> Tag $I_{desc}$ from $T_{sub}$
7. If(Count(Lst)==0 )
8. Then Add $I_{desc}$ into ***Lst***
9. Else For Each element $e_i$ in ***Lst***
10. SimilarityScore = Compare($I_{desc}$, $e_i$)
11. If(SimilarityScore >= $T_h$)
12. Then If ***ItemSet*** contains $e_i$
13. Then Increment support of $e_i$ by 1 from ***ItemSet***
14. Else Add $e_i$ to ***ItemSet*** and increment support of $e_i$ by 2
15. Else Add $e_i$ to ***ItemSet*** and increment support of $e_i$ by 1
16. Add $I_{desc}$ into ***Lst***
17. Using ItemSet as reference data structure containing Item $e_i$ and its support $f_s$ , desired support C and confidence C, generate frequent ItemSet **F.**
18. For Each ItemSet $I_{fr}$ in ***F***
19. Insert $I_{fr}$ in XML Document $X_{TAR}$
20. Return **Xtar**

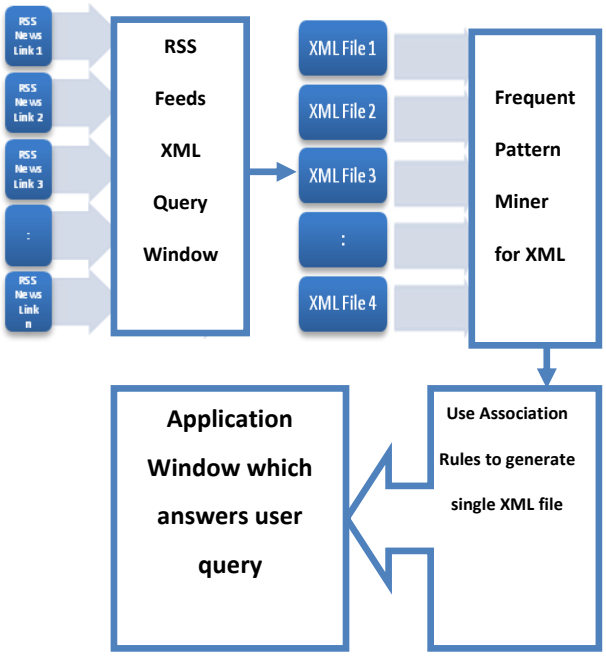


**Fig. 1 Operation Flow for Frequent Pattern Miner for XML**

The above figure shows the operation flow of Frequent Pattern Miner for XML as given in algorithm procedure. The algorithm is characterized and works by the following key procedure:

1. First we provide the RSS Feed Links of different newspapers as input documents.
2. We then generate the FinalXML file for links.
3. From FinalXML read all <Title> tags.
4. Read all subtrees in each <Title> tag.
5. In first run add subtree into Itemset of first <title>
6. Read subtree of next <Title> tag.
7. Compare every subtree with all elements of existing Itemset.
8. Set similarity threshold. If this is satisfied by comparison result, then initialize that subtree with same naming convention in existing Itemset.
9. Generate new Itemset for every <Title>.
10. Apply Frequent Pattern Miner to compute frequent Itemset.
11. This generated information is the result to user query as a new XML document.

## IV. CONCLUSION

**Table 1. Comparative Analysis of Implemented Approach with existing Document summarization and Information Retrieval Methods**

| Methods | IR Ranking | Use of Dictionary | Type of Input Information | Generate Frequent Patterns? | Query Indepen-dent | Applicable on Web Pages? | Applicable for Smart phone Devices? |
|---|---|---|---|---|---|---|---|
| Automatic Document Summarization By Sentence Extraction | Statistical (tf-Idf) | No | Structured Only | No. Generates Summary | No | No | No |
| Query-Specific Document Summarization | Statistical (Okapi Equation) | No | Structured Only | No. Generates Summary | No | No | No |
| A Language Independent Algorithm for Single and Multiple Document Summarization | Natural Language Processing | Yes | Structured Only | No. Generates Summary | Yes | No | No |
| OCELOT | Natural Language Processing | Yes | Structured and Non Structured Data (XML) | No. Generates Summary | Yes | Yes | No |
| **Implemented Approach** | **Natural Language Processing** | **Yes** | **Structured and Non Structured Data (XML)** | **Yes** | **Yes** | **Yes** | **Yes** |

The main goals of the paper are to achieve:

1. Mine all frequent association rules restriction on the structure and the content of the rules;
2. Store mined information in XML format;
3. Use the extracted knowledge to gain information about the original data sets;
4. Update mined TARs when the original XML data sets change.

Using the concept of TAR in conjunction with Natural Language Processing is very effective.

## REFERENCES


1. Braga, A. Campi, S. Ceri, M. Klemettinen, and P. Lanzi, "Discovering Interesting Information in XML Data with Association Rules," Proc. ACM Symp. Applied Computing, pp. 450-454, 2003.
2. World Wide Web Consortium, XQuery 1.0:An XML Query Language , http://www.w3C.org/TR/xquery, 2007.
3. World Wide Web Consortium, Extensible Markup Language (XML) 1.0,http://www.w3C.org/TR/REC
4. J.W.W. Wan and G. Dobbie, "Extracting Association Rules from XML Documents Using XQuery," Proc. Fifth ACM Int'l Workshop Web Information and Data Management, pp. 94-97, 2003.
5. D. Barbosa, L. Mignet, and P. Veltri, "Studying the XML Web: Gathering Statistics from an XML Sample," World Wide Web, vol. 8, no. 4, pp. 413- 438, 2005.
6. R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules in Large Databases," Proc. 20th Int'l Conf. Very Large Data Bases, pp. 478-499,1994.
7. C. Combi, B. Oliboni, and R. Rossato, "Querying XML Documents by Using Association Rules," Proc. 16th Int'l Conf. Database and Expert Systems Applications, pp. 1020-1024, 2005.
8. J. Paik, H.Y. Youn, and U.M. Kim, "A New Method for Mining Association Rules from a Collection of XML Documents," Proc. Int'l Conf. Computational Science and Its Applications, pp. 936-945, 2005
9. Yogesh R.Rochlani , Prof. A.R. Itkikar, "Integrating Heterogeneous Data Sources Using XML Mediator", ijcsn, vol 1, issue 3, 2012.
10. Arundhati Birari1, Prof. Ranjit Gawande2, "Mining Tree-Based Association Rules for XML Query Answering", ijettcs vol 2,issue 3 , 2013
11. V. Kasthuri Muthu, A. Sameera T. "Mining Algorithm for XML Query Answering Support", ijcset, vol4, 2013