Optimization of Query Processing in XML Document using TAR and Path Based Indexing

D.Karthiga, S.Gunasekaran

Computer Science and Engineering, V.S.B Engineering College Karur, Tamilnadu.

Abstract

The increasing amount of XML datasets available to casual users increases the necessity of investigating techniques to extract knowledge from these data. Data mining is widely applied in the database research area in order to extract frequent values from both structured and semi structured datasets. 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. Documents are often so large that the data set returned as answer to a query may be too big to convey required knowledge. Thus an approach to mine the data using Tree-based association rules from XML documents. TAR rules will provide information on both the structure and the content of XML documents. Moreover, they can be stored in XML format to be queried later on. The mined knowledge is approximate, intensional knowledge used to provide the quick, approximate answers to queries and also the information about structural regularities that can be used as data guides for document querying. An association rule is useful for discovering interesting relationship hidden in the datasets. The indexing mechanism improves the speed of data retrieval operations. Thus an approach called Path Based Indexing is used to retrieve the Query in an efficient way.

Keywords

XML, approximate query-answering, Data Mining, TAR, Path Based Indexing.

1. Introduction

XML [12] serves dual functionalities as markup language and data format. It separates presentation and data thus offering independency and flexibility for content association. Due to this nature of flexibility, data interchanged between two very different
systems can use XML as the data format. XML tree-like structure is intuitive, human readable, and easy to understand. With the help of XML schema or DTD, the type and attributes of each tag usable for certain XML document can be well defined. An XML query language defines more comprehensible and structurized construct for conducting operation on an XML document or various XML documents. For processing the query, an XML query engine or processor translates the syntaxes and executing the operations hinted by the query. Output is returned after process and processing time is projected to be minimum thus alluding efficient processing. However, as non binary format, performing query over XML data pertaining to arbitrary applications is still an intriguing issue.

Data mining traditionally is used to find trends in data stored in a database. One data mining technique that has proved popular is association rule mining[10], which finds associations between items in a database. In recent years, the successful development of eXtensible Markup Language (XML) as a standard to represent semi structured data and the fast growing amount of available XML data sets a pressing need for languages and tools to manage collections of XML documents, as well as to mine interesting information out of them. With developments like Xyleme which is a huge warehouse integrating XML data from the Web, it is essential that direct techniques for mining XML data are developed. The query language XQuery[11] was proposed by theW3C in order to provide a flexible way to extract XML data and provide the necessary interaction between the web world and database world. XQuery is expected to become the standard query language for extracting XML data from XML documents. Therefore, if we can mine XML data using XQuery, then we can integrate the data mining technique into XML native databases. So, we are interested to know whether XQuery is expressive enough to mine XML data. One data mining technique that has proved popular is association rule mining . It finds associations between items in a database. In the past, most effort was put to design query processor to support declaration in query languages. These days, the issue has shifted to relational XML storage and integration with data management system. The rest of this paper is arranged as follows. We initially review the evolving path of XML query languages. Then, we provide different approaches for xml query processing by extracting the ideas and comparing the proposals. Finally, we provide possible direction for future xml database and sum up our conclusion. XQuery had been a moving target for some time before it was established as W3C recommendation in 2007. A big part of XQuery semantics adopts Quilt's. XQuery uses XPath[13] for path expressions and FLWOR structure for describing the whole query. As a recommended standard, a lot of researches nowadays discuss the method of optimizing XQuery translation and processing by a

Manuscript received October 5, 2013
Manuscript revised October 20, 2013

query processor and integrating XQuery into a full-fledged XML database management system. Pros: clear semantics, integration with XPath.
The query on XML often contains regular path expressions. This means the expression of a query contains wild cards like *, $, and so on. DataGuide can decrease query processing time by the path index which gives a dynamic outline of the structure of semistructured data. But this is only applicable to the query with single regular expression and not applicable to the query with complex regular expression having various ones. The 2-Index overcomes the weak point of previous work, but the size of indexes may be the square of the data nodes. T-Index partially solved the problems in 2-Index, but still could not overcome the problem where indexes can not cover all possible paths. The query processing techniques using signature also have been largely studied. In relational database, the signature technique is used to select matched tuples by select condition. The signature is used in object-oriented database to reduce page I/O when evaluate the path expression. However, regular path expressions can not be processed by these methods. Loreand eXcelon are the representative databases that deal with semi structured data or XML. These keep the structure of graph for atypical data and store the nodes of the graph as objects. As the query is processed by visiting nodes of a tree, the reduction of the node fetching is the key point for query optimization.

1.1. XQuery

XQuery, an XML Query Language which were invented by the World Wide Web Consortium (W3C), offers an effective and standardized way to query any kind of XML information. There are many systems that support XQuery queries. However, the XStreamCast system is designed for the XQuery processing of continuously data streams. It processes XML fragments which are transmitted by the server. XStreamCast is a push-based continuously streamed XML query processing system. It supports multiple servers and clients. As it is push-based, the servers broadcast streamed XML data, which is in the form of XML fragments, to the clients concurrently while the clients tune-in to the streamed XML fragments and evaluate XML queries against that data. The information of XML data can be attained from a weather report, stock market, relational databases, or text documents. The clients can be networked devices, such as cell phones, PDAs, laptops et c., as long as they are able to connect to the network and provide some storage for XML data. Once the clients get the XML data transmitted by the servers, they analyze the query that was submitted by the user first. That is, the user query is parsed and converted appropriately to an evaluation query depending on its syntaxes. The XML fragments are analyzed and only the ones useful to the query are stored in the client's memory. The query is, then applied to the stored XML data. This thesis presents an efficient way for processing XQuery given by the user by using an XQuery processing algorithm, which gets abstract syntax trees as input and converts them into suitable forms for processing. 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. In fact, it is not mandatory for an XML document to have a defined schema: 50 percent of the documents on the web do not possess one. 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. The database-compatible data types supported by XML Schema provide a way to specify a hierarchical model. However, there are no explicit constructs for defining classes and properties in XML Schema, therefore ambiguities may arise when mapping an XML-based data model to a semantic model. In this paper we focus on a subset of XPath queries, which contain child axis "/", descendant axis "//" and name tests. This type of query is commonly used as a building block for more complex XPath and other XML query languages. A query can be represented as a linear path, therefore is named as path query. For example Fig.1. query /incidents/incident//ballistic item//bullet. In this representation, a node is created for each tag and the return tag is indicated by a box. For the incoming edge of a node, a single line represents a child axis, and double lines represent a descendant axis. To process a path query efficiently on an encoded XML stream, we first encode the query as follows: From the header of the en- coded XML stream, we get the dictionary that maps an XML tag to an integer. Then we replace every tag in the XPath query with its corresponding integer, and keep the value predicate as it is. For example, a query=Incidents=incident =country[text() = "Italy"] is encoded as =1=2=3[text() ="Italy"]. Together with intrinsically unstructured documents, there is a significant portion of XML documents which have only an implicit structure, that is,

their structure has not been declared in advance, for example via a DTD or an XML- Schema.

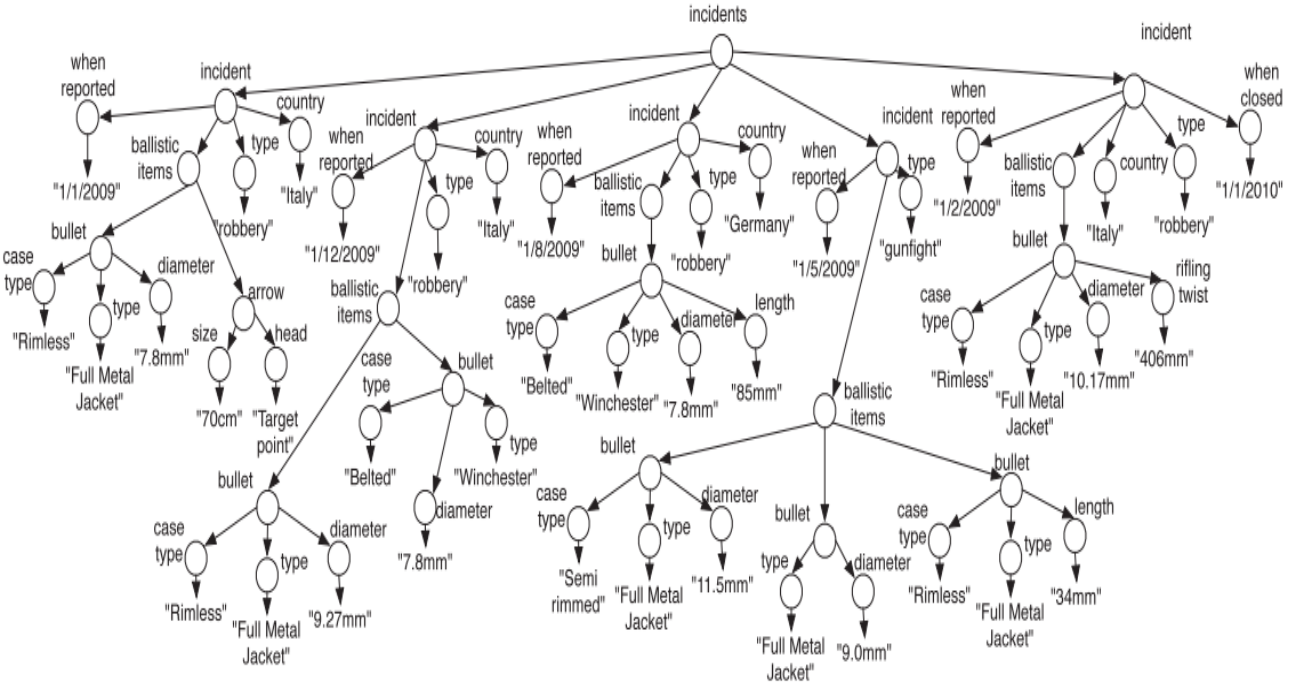

Fig.1.Sample Xml File

Querying such documents is quite difficult for users for two main reasons:

1) They are not able to specify a reasonably likely structure in the query conditions.

2) They are very often confused by the large amount of information available.

This limitation of XML is a crucial problem, which did not emerge in the past years in the context of traditional (relational) database management systems, and thus must be addressed in order to provide access to these data to a wider set of users. The application of data mining techniques to extract useful knowledge from XML has received a lot of attention in the recent years due to the wide availability of these datasets. In particular, the process of mining association rules to provide summarized representations of XML documents has been investigated in many proposals and in particular either by using languages (e.g. XQuery) and techniques developed in the XML context, or by implementing graph/tree-based algorithms. By mining frequent patterns from XML documents, we provide the users with partial, and often approximate, information both on the document structure and on its content. Such patterns can be useful for the users to obtain information and implicit knowledge on the documents and to be more effective in query formulation. Moreover, this information is also useful for the system, which is provided with discovered information, like hidden integrity constraints, which can be used for semantic optimization.

Association rules describe the co-occurrence of data items in a large amount of collected data and are usually represented as implications in the form {X,Y} where x and y are two arbitrary sets of data items, such that X\Y. The quality of an association rule is usually measured by means of support and confidence. Support corresponds to the frequency of the set{X,Y} in the dataset, while confidence corresponds to the conditional probability of ending Y, having found And is given by sup(X $\Rightarrow$ Y)=sup(X). In this work we extend the notion of association rule originally introduced in the context of relational databases, in order to adapt it to hierarchical nature of XML documents. In particular, we consider the element only Info set content model, which allows an XML nonterminal tag to include only other elements and/or attributes, while the text is conned to terminal elements. Furthermore, with- out loss of generality, we do not consider some features of the Infoset that are not relevant to the present work, such as namespaces, the ordering label, the referencing formalism through ID-IDREF attributes, URIs, and Links.

Association analysis is a useful data mining technique exploited in multiple application domains. One of the best

known is the business field where the discovering of purchase patterns or associations between products that clients tend to buy together is used for developing an effective marketing. The attributes used in this domain are mainly categorical data, which simplifies the procedure of mining the rules. In the last years the application areas involving other types of attributes have increased significantly. Some examples of recent applications are finding patterns in biological databases, extraction of knowledge from software engineering metrics or obtaining user's profiles for web system personalization . Associative models have been even used in classification problems as the base of some efficient classifiers . Numerous methods for association rule mining have been proposed, however many of them discover too many rules, which represent weak associations and uninteresting patterns. The improvement of association rules algorithms is the subject of many works in the literature. Most of the research efforts have been oriented to simplify the rule set, to generate strong and interesting patterns as well as to improve the algorithm performance. When attributes used for inducing the rules take continuous values, these three objectives can be achieved by means of an efficient data discretization procedure such as the proposed in this paper. In a set of transactions D the strength of an association rule in the form "If X then Y" is mainly quantified by the following factors: x Confidence or predictability. A rule has confidence c if c% of the transactions in D that contain X also contain Y. A rule is said to hold on a dataset D if the confidence of the rule is greater than a user-specified threshold. x Support or prevalence. The rule has support s in D if s% of the transactions in D contain both X and Y.

1.2. Association rules

In particular, the idea of mining association rules to provide summarized representations of XML documents has been investigated in many proposals either by using languages (e.g., XQuery [11]) and techniques developed in the XML context, or by implementing graph- or tree-based algorithms. In this paper, we introduce a proposal for mining and storing Tree-Based Association Rules (TARs) as a means to represent intensional knowledge in native XML. Intuitively, a TAR represents intensional knowledge in the form {S B,SH} , where S B is the body tree and S H the head tree of the rule and S B is a subtree of SH . The rule {S B,S H} states that, if the tree SB appears in an XML document D ,it is likely that the "wider" (or "more detailed"), tree S H also appears in D. Graphically, we render the nodes of the body of a rule by means of black circles, and the nodes of the head by empty circles.

It allows to obtain and store implicit knowledge of the documents, useful in many respects:

a) When a user faces a data set for the first time, she/he does not know its features and frequent patterns provide a way to quickly understand what is contained in the data set;

b) Besides intrinsically unstructured documents, there is a significant portion of XML documents which have some structure, but only implicitly, that is, their structure has not been declared via a DTD or an XML Schema. Since most work on XML query languages has focused on documents having a known structure, querying the above-mentioned documents is quite difficult be- cause users have to guess the structure to specify the query conditions correctly. TARs represent a data guide that helps users to be more effective in query formulation;

c) It supports query optimization de- sign, first of all because recurrent structures can be used for physical query optimization, to support the construction of indexes and the design of efficient access methods for frequent queries, and also because frequent patterns allow to discover hidden integrity constraints, that can be used for semantic optimization;

d) For privacy reasons, a document answer might expose a controlled set of TARs instead of the original document, as a summarized view that masks sensitive details .

TARs can be queried to obtain fast, although approximate, answers. This is particularly useful not only when quick answers are needed but also when the original documents are unavailable. In fact, once extracted, TARs can be stored in a (smaller) document and be accessed independently of the data set they were extracted from.

Summarizing, TARs are extracted for two main purposes: 1) to get a concise idea—the gist—of both the structure and the content of an XML document, and 2) to use them for intensional query-answering , that is, allowing the user to query the extracted TARs rather than the original document. In this paper, we concentrate mainly on the second task. We have applied our techniques in the Odyssey EU Project, 1 whose objective is to develop a platform for automated sharing, management, processing, analysis and use of ballistic, and crime scene information across Europe. Frequent patterns, in the form of TARs, provide summaries of these integrated data sets shared by different EU Police Organizations. By querying such summaries, investigators obtain initial knowledge about specific entities in the vast data set(s), and are able to devise more specific queries for deeper investigation. An important side effect of using such a technique is that only the most promising specific queries are issued toward the integrated data, dramatically reducing time and cost. This paper provides a method for deriving intensional knowledge from XML documents in the form of TARs, and then storing these TARs as an alternative, synthetic data set to be queried for providing quick and summarized

answers. Our procedure is characterized by the following key aspects:

1. It works directly on the XML documents, without transforming the data into any 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.
3. It stores association rules in XML format.
4. It translates the queries on the original data set into queries on the TARs set.

The aim of our proposal is to provide a way to use intensional knowledge as a substitute of the original document during querying and not to improve the execution time of the queries over the original XML data set, like in [5]. Accordingly, the paper's contributions are

- An improved version of the TARs extraction algorithm introduced in [1], which was based on Path Join [2]. The new version uses the better performing 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.
- Automatic user-query transformation into "equivalent" queries over the mined intensional knowledge. The notion of equivalence in this setting is given in Section
- As a formal corroboration of the accuracy of the process, the proof that our intensional-answering process is sound and complete up to a frequency threshold.

2. Tree Based Association Rules

Association rules describe the co-occurrence of data items in a large amount of collected data and are represented as implications of the form X =Y , where X and Y are two arbitrary sets of data items, such thatX \ Y ;. The quality of an association rule is measured by means of support and confidence. Support corresponds to the frequency of the set X ,Y in the data set, while confidence corresponds to the conditional probability of finding Y , having found X and is given by support. In this paper, we extend the notion of association rule introduced in the context of relational databases to adapt it to the hierarchical nature of XML documents. Following the Infoset conventions, we represent an XML document as a tree (N,E,r) where N is the set of nodes, r is the root of the tree, E is the set of edges, L is the label function which returns the tag of nodes (with L the domain of all tags) and the content function which returns the content of nodes (with C the domain of all contents). We consider the element-only Infoset content model [8], where XML nonterminal tags include only other elements and/or attributes, while the text is confined to terminal elements. We are interested in finding relationships among subtrees 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.

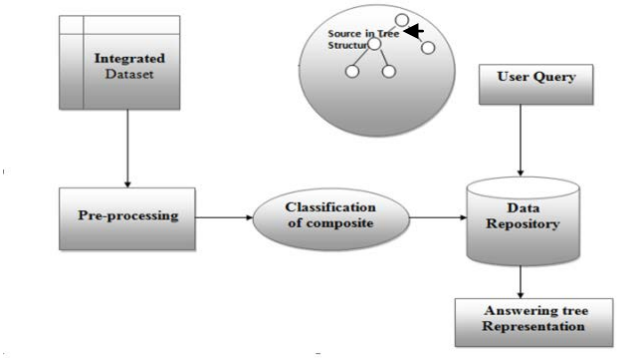

Fig.2.Working of an XML Dataset.

3. Path Based Indexing

The application TAR is to view the each and every attribute in a tree representation manner. The indexing is a mechanism in which the value of index is stored and processed. This paper includes the range value of the each attributes of the TAR. The indexing is a good mechanism where the value of the tree can be stored for the easy and quick retrieval of the query is possible which is given by the user. In the existing work the indexing is the mechanism maintained by TAR itself which provide approximate answering to the query. But, because of using this Path Based Indexing [2] mechanism the approximation level is decreased and the intensional answering to the query is increasing accordingly.

4. Related work

4.1 Efficiently mining frequent trees in a forest: algorithms and applications [8]

Mining frequent trees is very useful in domains like bioinformatics, Web mining, mining semistructured data, etc. We formulate the problem of mining (embedded) subtrees in a forest of rooted, labeled, and ordered trees. We present TREEMINER, a novel algorithm to discover all frequent subtrees in a forest, using a new data structure called scope-list. We contrast TREEMINER with a pattern matching tree mining algorithm (PATTERNMATCHER), and we also compare it with TREEMINERD, which

counts only distinct occurrences of a pattern. We conduct detailed experiments to test the performance and scalability of these methods. We also use tree mining to analyze RNA structure and phylogenetics data sets from bioinformatics domain.

4.2 Mining tree-based association rules from XML documents [4]

The increasing amount of XML datasets available to casual users increases the necessity of investigating techniques to extract knowledge from these data. Data mining is widely applied in the database research area in order to extract frequent correlations of values from both structured and semi- structured datasets. In this work we describe an approach to mine Tree-based association rules from XML documents. Such rules provide information on both the structure and the content of XML documents; moreover, they can be stored in XML format to be queried later on. The mined knowledge is approximate, intensional knowledge used to provide: (i) quick, approximate answers to queries and (ii) information about structural regularities that can be used as dataguides for document querying. A prototype of the proposed system is also brie y described.

4.3 Querying XML Documents by Using Association Rules [5]

In the last years XML is becoming a standardized means for representing semi structured data, i.e. data having a structure which is not regular and fixed in advance. In this context, documents may have "similar" content but different structure and thus in this work we propose an approximate way to query XML data by means of association rules.

4.4 Extracting Association Rules from XML Documents using Query [9]

Data mining is generally considered the extraction and analysis of information from databases. With the rapid growth of XML data available online, mining XML data from the web is becoming increasingly important. In support of this trend, several encouraging attempts at developing methods for mining XML data have been proposed. However, efficiency and simplicity are still a barrier for further development. In this paper, we show that any XML document can be mined for association rules using only the query language XQuery without any pre-processing or post-processing.

4.5Answering XML Queries Using Indexes [2]

The problem of answering XML queries using indexes is to find efficient methods for accelerating the XML query with pre-designed index structures over the XML database. This problem received increasing interests and has been lucubrated in recent years. Regular path expression is the core of the XML query languages e.g., XPath and XQuery. Most of the state-of-the-art XML indexes, therefore, hammer at how to efficiently answer the path- based XML queries. This paper surveys various approaches to indexing XML data proposed in the literature. We give a step by step analysis to show the evolution of index structures for XML path information, based on tree structures or more commonly, directed labeled graphs. For each approach, we first present the specific issue it aims to tackle, and then the proposed solution presented. Furthermore, construction, physical data storage and maintenance costs, are analyzed.

5. Identified problems

- Input file is not directly work as XML document.
- The extraction and mining of one specified attribute is not specified.
- Intensional Answering is not present.
- TAR will give approximate answering to the query.
- Query optimization is slow in the existing system.

6. Proposed Work

- Every input file taken in the form of XML document
- A single attribute can be specified and for those attribute the TAR construction can be viewed. Thus, not only the approximate answering can be specified but also the related answers can be viewed through tree based representation.
- We can get the intensional answering to the given query by more than the approximate level of the TAR construction. As the use of Path Based Indexing [2] mechanism, the approximation of the query level is increased.
- In the proposed system, the optimization is increased when compared to the Existing system.
- The relationship between two attributes is maintained by the document builder factory in the XML document.

7. Results

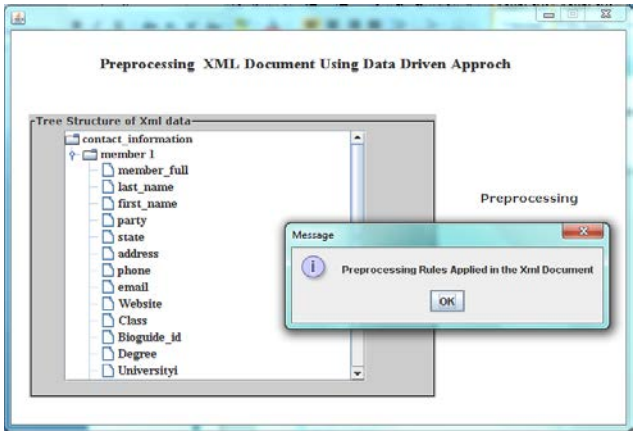

Fig.3.Preprocessing process Applied

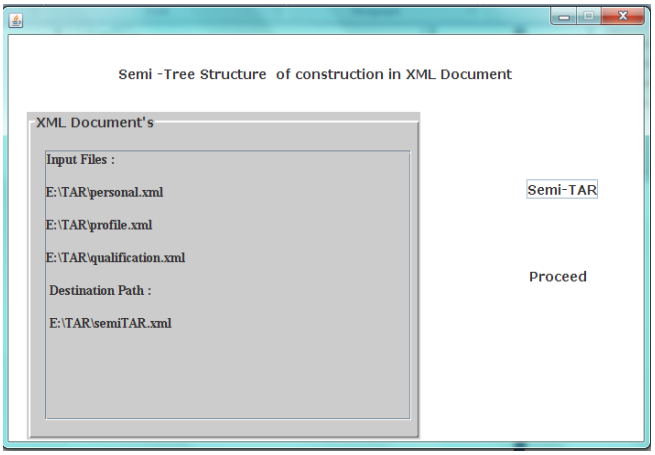

Fig.4.Forming Semi structured document

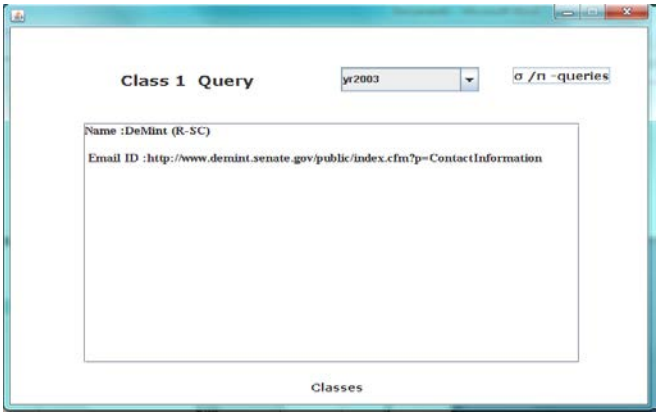

Fig.5.Answering Query

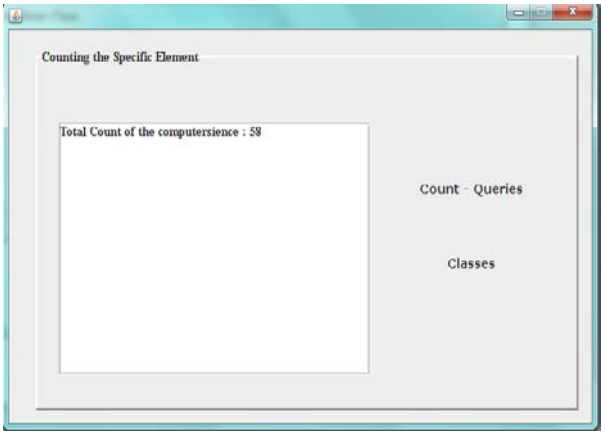

Fig.6.Counting the Elements

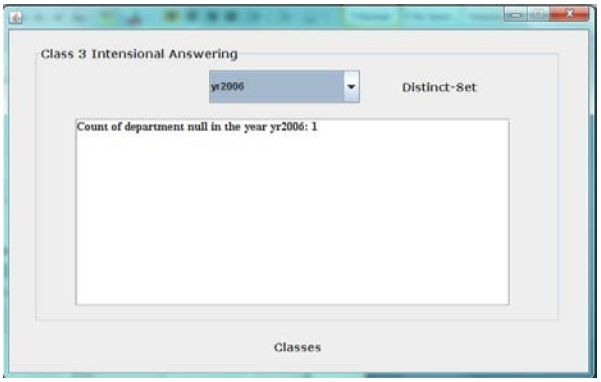

Fig.7.Intensional Answering to a Query

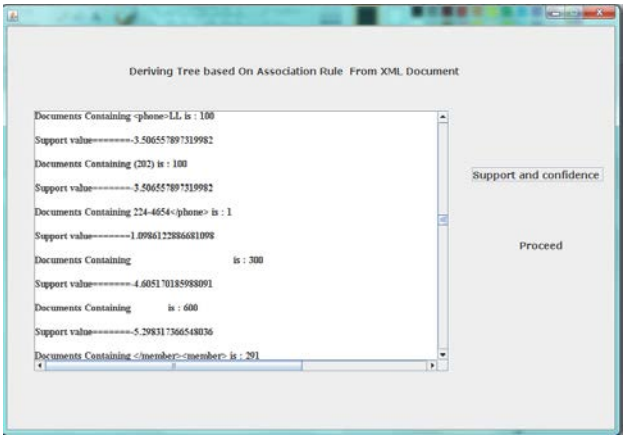

Fig.8.Calculation of Support and Confidence

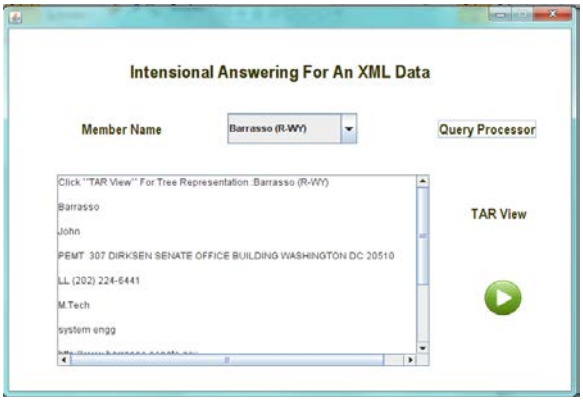

Fig.9.Intensional Answering in TAR View

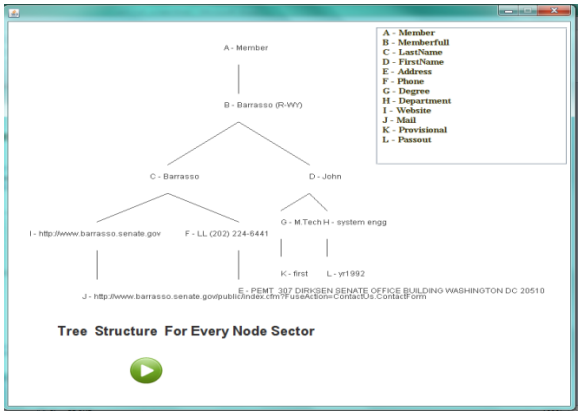

Fig.10.TAR View

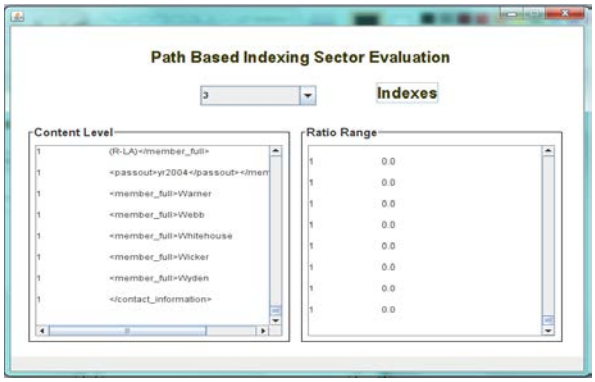

Fig.11.Path Based Indexing

8. Future Direction

In future, the application of data mining techniques to extract useful knowledge from XML datasets has received a lot of attention in the recent years due to the wide availability of datasets. The classification rule which is one of the data mining techniques that can be applied in the semi structured document .Thus by adding the classification rule the complexity of the semi structured document increases and the intensional answering to the query will be increased accordingly.

9. Conclusions

Mined all frequent association rules without imposing any restriction on the structure and the content of the rules. Thus proposed an algorithm that extends Path Based Indexing and allows us to extract frequent tree-based association rules from XML documents. The main goals we have achieved are: 1) Mined frequent association rules without imposing any restriction on the structure and the content of the rules; 2) Stored mined information in XML format; as a consequence, 3) It can effectively use the extracted knowledge to gain information, by using query languages for XML, about the original datasets where the mining algorithm has been applied. The intensional information in TARs provides a valid support in several cases. It allows obtaining and storing implicit knowledge of the documents.

References

[1] M. Mazuran, E. Quintarelli, and L. Tanca, “Data Mining from XML Documents,” technical report, PolitecnicoMilano,http://ieee.org/quintare/Papers/MQT09-RR.pdf, 2012.

[2] K. Wong, J.X. Yu, and N. Tang, “Answering XML Queries Using Path-Based Indexes: A Survey,” World Wide Web, vol. 9, no. 3, pp. 277-299, 2006.

[3] 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.

[4] M. Mazuran, E. Quintarelli, and L. Tanca, “Mining Tree-Based Frequent Patterns from XML,” Proc. Eighth Int’l Conf. Flexible Query Answering Systems, pp. 287-299, 2009.

[5] 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.

[6] A. Termier, M. Rousset, and M. Sebag, “Dryade: A New Approachfor Discovering Closed Frequent Trees in Heterogeneous TreeDatabases,”Proc. IEEE Fourth Int’l Conf. Data Mining,pp. 543-546,2004.

[7] E. Baralis, P. Garza, E. Quintarelli, and L. Tanca, “Answering XMLQueries by Means of Data Summaries,”ACM Trans. InformationSystems, vol. 25, no. 3, p. 10, 2007.

[8] M.J. Zaki, “Efficiently Mining Frequent Trees in a Forest:Algorithms and Applications,”IEEE Trans. Knowledge and DataEng.,vol. 17, no. 8, pp. 1021-1035, Aug. 2005.

[9] 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.
[10] D. 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.
[11] World Wide Web Consortium. XQuery 1.0: An XML query language,2007. http://www.w3C.org/TR/xquery/
[12] World Wide Web Consortium. Extensible Markup Language (XML) 1.0,1998. http://www.w3C.org/TR/REC-xml/.
[13] World Wide Web Consortium. XQuery 1.0: An XML query language,2007. http://www.w3C.org/TR/xpath/