Keyword Based Searching According to the Movie Names

Sunilkumar Reddy P¹, Govindarajulu¹, Srinivasulu Asadi^{2,*}

¹Dept of CS, College of Sciences S V University, Tirupati, AP-517 501, India

²Dept of IT, SVEC, Tirupati, J.N.T.University, Anantapur, A.P - 517102, India

*Corresponding Author: srinu_asadi@yahoo.com

Copyright © 2014 Horizon Research Publishing All rights reserved.

Abstract Keyword based queries are inherently ambiguous such that given a set of keywords the database search engine has only an uncertain guess about the user's informational need represented by the query. Possibly high complexity of the data makes providing intelligent search results effectively extremely challenging. Databases enable users to precisely express their informational needs using structured queries. However, database query construction is a laborious and error-prone process, which cannot be performed well by most end users. Keyword search alleviates the usability problem at the price of query expressiveness. As keyword search algorithms do not differentiate between the possible informational needs represented by a keyword query, users may not receive adequate results. This paper presents Extended Incremental Query Processing - a novel approach to bridge the gap between usability of keyword search and expressiveness of database queries. Extended Incremental Query Processing enables a user to start with an arbitrary keyword query and incrementally refine it into a structured query through an interactive interface. The enabling techniques of Extended Incremental Query Processing include: 1) A probabilistic framework for incremental query construction; 2) A probabilistic model to assess the possible informational needs represented by a keyword query; 3) An algorithm to obtain the optimal query construction process. This paper presents the detailed design of Extended Incremental Query Processing, and demonstrates its effectiveness and scalability through experiments over real-world data and a user study. Extracting information from semi structured documents is a very hard task. Documents are often so large that the data set returned as answer to a query may be too big to convey interpretable knowledge. In this, we describe an approach based on Tree-Based Association Rules (TARs): mined rules, which provide approximate, intentional information on both the structure and contents of XML documents. This mined knowledge is later used to provide: a concise idea—the gist—of both the structure and the content of the XML document .quick, approximate answers to queries.

Keywords Query Formulation, Search Process, Keyword Query, Multi-Agent System, Query Based Learning, Model Driven Approach and Entropy and Extended Incremental Query Processing

1. Introduction

With the growth of structured information available on the Web and in online databases, it becomes increasingly difficult for users to find the exact data they seek for. Structured queries are a powerful tool to exactly describe a user's informational need and retrieve the intended information from a database. However, formulation of a structured query is a challenging task for a naive user as it requires precise knowledge of the database schema and the query language. On the other hand, a keyword query is easy to put and it is more customary for the users.

Keyword search allows ordinary users to search for information without any expert knowledge of the database schema. However, keyword search lacks expressiveness to precisely describe a user's informational need, and may return irrelevant or incomplete results. To take advantage of both, i.e., expressiveness of structured queries and usability of keyword search, a query ranking approach has been introduced which translate a keyword query into a ranked list of structured queries, such that the user can select the query that best represents her informational need.

Such a query ranking approach has two limitations. First, each keyword can occur in nearly any textual attribute of a database, the number of possible query interpretations grows sharply with number of textual attributes and the size of the schema. Second, even a theoretically optimal ranking algorithm can, at best, rank the most common query interpretations highest, so the users with less frequent informational needs may not receive adequate results. To overcome these limitations we introduced a new system EIQP designed to fill the gap between usability of keyword search and expressiveness of database queries. EIQP provides a query construction interface that enables users to create their own structured queries in an interactive way based on the query construction options automatically suggested by the system. If the user accepts or rejects an option, the system can use this information to automatically reduce the interpretation space of the user's keyword query and suggest new options. The interaction process between the user and EIQP continues until the user finds the desired interpretation of the keyword query and retrieves the

corresponding search results. Initially, in order to generate query construction options, EIQP made use of the database schema and keyword occurrence statistics.

The enabling techniques of EIQP include:

- 1) A probabilistic framework for incremental query construction.
- 2) A probabilistic model to assess the possible informational needs represented by a keyword query.
- 3) An algorithm to obtain the optimal query construction process. (MQCP).

Using EIQP, a user can benefit from both, a conventional ranking interface and a more controllable query construction interface. The former allows the user to immediately identify the most common interpretation of her query. The latter enables the user to clarify her search intent step by step, which is especially helpful when the intended query interpretation does not receive a good rank. We use XAMPP tool in our paper, for performing various data processing techniques, there by getting the results most efficiently and accurately.

The goal of data mining is to extract or mine knowledge from large amounts of data. Consequently, data mining consists of more than collecting and managing data, it also includes analysis and prediction. The extensible Markup Language (XML) has become a standard language for data representation and exchange XML is a Standard, flexible syntax for data exchanging Regular, structured data. Mining of XML documents significantly differs from structured data mining and text mining. XML allows the representation of semi-structured and hierarchal data containing not only the values of individual items but also the relationships between data items. Due to the inherent flexibility of XML, in both structure and semantics, discovering knowledge from XML data is faced with new challenges as well as benefits. Mining of structure along with content provides new insights and means into the process of knowledge discovery. As for query-answering, since query languages for semi structured data rely the on 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. This limitation is a crucial problem which did not emerge in the context of relational database management systems. As a consequence, when accessing for the first time a large dataset, gaining some general information about its main structural and semantic characteristics helps investigation on more specific details.

2. Related Work

2.1. Keyword Search in Databases

The continually growing amount of structured information has little use without the availability of an efficient search function. The most relational databases provide users with a full-text search capability restricted to one table attribute and

enabling the execution of a simple query that affects only one data entry of table column. But due to the complexity of the database and its schema, each attempt to put an intricate query becomes a challenging task. One should still have knowledge of the database schema and use a structured query language (e.g. SQL) in order to satisfy his information need. In the age of internet and search engines ordinary users are more acquainted with a keyword search. This kind of search is very intuitive and convenient, as they only need to type some keywords in the search window and become a list with ranked suggestions for their request. But what happens behind the keyword search process? What is the magic behind the transformation of the simple keyword query into the relevant individual results?

The following aspects are important during this process:

1. Data modeling
2. Structural ambiguity of the results
3. Ranking strategies

In order to enable the database search, first the properties and peculiarities of database data and schema are reflected in a data model. In the second step, on the basis of a selected model the number of candidates as possible results is generated. Thereby there exists a vast variety on their structural presentation. As users are interested only in a limited number of relevant results, the disambiguation and ranking of the candidates is done.

2.2. Data Modeling

A lot of attempts have been made in order to support keyword search on relational databases. The first of them to mention are the Discover, DBXplorer. As the keywords can be present in different attributes and tables, these systems were the pioneers generating the matching rows obtained by joining several table. For this purpose a database is modeled as a graph with tuples as nodes and edges between the nodes are primary to foreign key relationships. But with the growing amount of information the size of the databases increases and the relationships between the tables become more complex

2.3. Structural Ambiguity of the Results

During the search step the suggestions that meet the users' information need are created. As modeling of the dataset and extraction algorithms used by the research community is very diverse, the structure of results also vary. EIQP query construction system uses query templates and during a search phase fills them with keywords. First the smallest templates are used to generate the partial interpretations. Next during the expansion step the more complex queries up to complete ones are created. With the help of such query hierarchy there is no need to generate the whole interpretation space. The appropriate options help to construct the query incrementally and ensure the scalability of the system.

2.4. Ranking Strategies

As to the analysis of the results, there exist two approaches. The traditional methods generate and explore all possibilities, whereas the newer strategy is to rank structures. Let us first have a look how the traditional methods proceed. At the beginning tuple trees are extracted as the possible answers for a keyword query. In the next step a ranking strategy is used in order to decide which results are the most relevant ones. The user can then explore the ranked list to identify the desired interpretations.

2.5. Faceted search in Databases

User-driven query disambiguation has been successfully applied in Information Retrieval in the context of faceted search. Faceted search engines, such as the product search engine of Google and the Clusty search engine, organize search results into meaningful groups, called facets, by applying some clustering or categorization algorithms. Users can easily shrink the scope of the search by focusing on a small number of facets. Several navigational techniques were proposed to support users in finding information in a hierarchy of faceted categories. The interface of EIQP is similar to a faceted interface, whereas each facet corresponds to a query construction option. SUITS, a faceted interface that enables users to interactively disambiguate keyword queries. However, SUITS lacks a theoretical foundation for verifying its effectiveness.

2.6. Incremental Query Construction

Nowadays in the age of web search machines ordinary users let himself be spoilt with the ease of use of a keyword query. Unfortunately, this simplicity doesn't mean high expressiveness and quality of the retrieved results. The less knowledge about the intent behind a query is provided, the more effort is needed to extract the satisfying information from database. On the contrary, databases are equipped with a powerful query language that allows asking even trickiest and unusual questions but is too complex and hard to understand for an ordinary user. Many effort have been made in order to combine the high expressiveness of a database query language and the simplicity of a keyword query. The recent approaches make use of the database structure and provide users with a possibility to refine their information need and actively participate in the information retrieval process.

3. Keyword Based Techniques

3.1. Techniques

- In the existing system, they propose a set of functions written in Xquery; perform well on simple XML documents.

- XQuery is the language for querying XML data. It is like SQL for databases. It is built on XPath expressions. It is supported by all major databases.
- After that there was a proposal based on XMINE RULE, an operator for mining association rules for native XML documents.
- 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.
- In another system, the designer must specify the structure of the rule to be extracted and then to mine it.
- XQuery expressions that can be used to retrieve useful information from the extracted sets of association rules. Such useful information can provide intentional answers to queries formulated as XQuery expressions.
- The overall querying process appears completely transparent to the user: if the actual dataset is not available, the intentional (approximate) answer will be automatically provided by querying the rule set.

3.2. Disadvantages of Existing System

- The limitation in XQuery is that it is difficult to apply to complex XML documents with an irregular structure.
- 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.
- In Xquery we cannot get the exact answers to queries containing aggregators.

ALGORITHM: GREEDY ALGORITHM USING MQCP

Input: HQ = Initial Query Hierarchy;

TQ = Top Level of HQ;

T:- Threshold;

Output: c= Final Structured Query;

Step 1: If |TQ| is below T then

If HQ can be expanded, then

Expand HQ;

Else if |TQ| is equal to 1, then

return c;

end if

end if;

Step 2: Initialize partial_query best_r :=null; float best_gain := +∞;

Step 3: for each R in HQ, do

if IG(TQ|R) is below the best_gain, then

best_gain := IG(TQ|R);

best_r := R;

end if;

end for;

Step 4: Present best_r to user;

Step 5: if best_r is accepted, then

```

Sub(best_r) := all queries subsumed by best_r;
    TQ := TQ ^ Sub(best_r);
    else if best_r is rejected, then
Sub(best_r) := all queries subsumed by best_r;
    TQ := TQ - Sub(best_r);
    end if;

```

Step 6: Go to Step 1.

4. Implementation Results

In our paper work, we have taken the real time example i.e., very useful for investigations. This includes storing the incidents in XML format that happen in day to day life. These incidents can be updated every day. If the investigator wants to get the required information from that, he can get the information in three formats using PHP.

4.1. Get the gist

Get the gist is nothing but after updating all the incidents if we want to get required information i.e., getting about particular incident. This is that how many incidents happen nothing but finding the support. And at the same time how much percentage of the incident that is happened nothing but finding confidence.

Get the gist means retrieving information from XML document using PHP (Hypertext Preprocessor). It can provide intentional information about XML document. Get the gist allows intentional information extraction from an XML document, given the support, confidence.

Here we can give input as XML document and processing it using PHP according to user requirements. First we have to load the XML document into web server after connecting to local host. Here we are using WAMP server in PHP environment to execute the programs. And used HTML for user interfaces.

4.2. Get the idea

Get the idea includes get the gist as well as getting the original document to compare the results. Get the gist means retrieving information from XML document using PHP(Hypertext Preprocessor). It can provide intentional information about XML document. Get the gist allows intentional information extraction from an XML document, given the support, confidence.

Here we can give input as XML document and processing it using PHP according to user requirements. First we have to load the XML document into web server after connecting to local host. Here we are using WAMP server in PHP environment to execute the programs. And used HTML for user interfaces. Get the idea allows to show the intentional information as well as the original document, to give users the possibility to compare the two kinds of information. From that original document the user can

verify the results with original documents.

4.3. Get the answers

Get the answers is nothing but we have the date tag in our XML document i.e., in XML we can specify the user defined tag. From that date tag we can the list of incidents that are happened with their details. This includes some date comparison functions to display the accurate results.

Get the answers allows to query the intentional knowledge and the original XML document. Users have to write an extensional query; when the query belongs to the classes we have analyzed it is translated and applied to the intentional knowledge.

The incidents document is having the elements like incident type, when it is reported, bullet case type and etc., here we can give XML document as input and the two dates from which the user wants the incidents that are happened. Here we can query the XML document by giving the two dates and getting the incidents that are happened between two dates. We can not only retrieve incident type and also when it is reported, where it is happened etc., all the data can be retrieved.

Here we used three classes of queries are to query the XML document. We used various functions to retrieve data from XML documents. Some of them are \$_POST method to get the data directly that is entered by user from textbox. This can be stored in one variable and processed by taking values from XML documents. To get values from XML documents there is a function get Elements By Tag Name() and passing tag name as parameter. This gives the values of that tag as output.

These tag values can be stored in array format. We used foreach loop to loop through arrays that is to compare with entered values. We used node values for getting tag values from the XML document. And used strtotime() function to convert given entered date in textbox in to given date format.

Figure 4.1. When two keywords are given as input and the Query Hierarchy.

Figure 4.2. Result for the user intended structured Query.

Figure 4.3. GUI for login into the application

Figure 4.4. GUI for getting answers

This will show the results after testing is completed. This consists of various screenshots of the paper which will briefly explain the execution of the paper.

5. Conclusion

The presented Extended Incremental Query Processing (EEIQP) - a novel system, which enables construction of structured queries from keywords. We

presented a conceptual framework for the incremental query construction as well as a probabilistic model, which enables consistent assessment of the probability of a query interpretation. We presented an algorithm for generating optimal query construction plans, which enables the user to obtain the intended structured query with a minimal number of interactions. Our experimental results and user study show that EEIQP is highly helpful when user intended structured queries cannot be found within the top-ranked results. This can be used in place of any keyword search. Here the user is given with certain options to find the desired structured queries by which the user can get the exact results from the data base. In our system user can get the exact result instead of searching the exact result in the bunch of the result which is happened in the normal keyword searches. In the ranking search the most common query interpretations highest, whereas the users with less frequent informational needs may not receive adequate results. Using EEIQP, a user can benefit from both, a conventional ranking interface and a more controllable query construction interface. The former allows the user to immediately identify the most common interpretation of her query. The latter enables the user to clarify her search intent step by step, which is especially helpful when the intended query interpretation does not receive a good rank. By, using this we can get the exact results which the user is expected to get. In this paper implements for getting information from XML documents using query answering. Basically retrieving information from XML documents is a hard task because XML store large amount of data. Querying such is not easy to implement. But this can be done using PHP. Required data can be extracted from XML documents using that scripting language. We collect the incidents that are happened in our day to day life. These data can be stored in XML format. As there is a large data set, it is difficult to investigate or finding information about the particular incident. We proposed the system that it can give the results about that event based on the queries given by user. Finally, we showed that there is easy way to retrieve information from XML documents. In this paper we had done the examples for one dynamic XML document. A possible extension of this work concerns for doing the results for any type of XML document.

REFERENCES

- [1] I. Androustopoulos, G.D. Ritchie, and P. Thanisch, "Natural Language Interfaces to Databases—An Introduction," J. Language Eng., vol. 1, no. 1, pp. 29-81, 1995.
- [2] S. Agrawal, S. Chaudhuri, and G. Das, "DBXplorer: A System for Keyword-Based Search over Relational Databases," Proc. Int'l Conf. Data Eng. (ICDE), 2002.
- [3] M. Al-Muhammed and D.W. Embley, "Ontology-Based Constraint Recognition for Free-Form Service Requests," Proc. Int'l Conf. Data Eng. (ICDE '07), 2007.

- [4] H. Bast, A. Chitea, F. Suchanek, and I. Weber, "ESTER: Efficient Search on Text, Entities, and Relations," Proc. 30th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR), 2007.
- [5] D. Braga, A. Campi, and S. Ceri, "XQBE (XQuery By Example): A Visual Interface to the Standard XML Query Language," ACM Trans. Database Systems, vol. 30, no. 2, pp. 398-443, 2005.
- [6] A. Blum, "Microsoft English Query 7.5: Automatic Extraction of Semantics from Relational Databases and OLAP Cubes," Proc. 25th Int'l Conf. Very Large Data Bases (VLDB), 1999.
- [7] V.T. Chakaravarthy, H. Gupta, P. Roy, and M. Mohania, "Efficiently Linking Text Documents with Relevant Structured Information," Proc. Int'l Conf. Very Large Data Bases (VLDB), 2006.
- [8] J. Chambers, W. Cleveland, B. Kleiner, and P. Tukey, Graphical Methods for Data Analysis. Wadsworth, 1983.
- [9] Clusty, <http://clusty.com/>, 2011.
- [10] E. Demidova, X. Zhou, and W. Nejdl, "EIQP: Incremental Query Construction, a Probabilistic Approach," Proc. 26th IEEE Int'l Conf. Data Eng. (ICDE), 2010.
- [11] E. Demidova, X. Zhou, G. Zenz, and W. Nejdl, "SUITS: Faceted User Interface for Constructing Structured Queries from Key- words," Proc. 14th Int'l Conf. Database Systems for Advanced Applications (DASFAA), 2009.
- [12] H. He, H. Wang, J. Yang, and P.S. Yu, "BLINKS: Ranked Keyword Searches on Graphs," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD), 2007.
- [13] M.A. Hearst, "Clustering versus Faceted Categories for Informa- tion Exploration," Comm. ACM, vol. 49, no. 4, pp. 59-61, Apr. 2006.
- [14] V. Hristidis, L. Gravano, and Y. Papakonstantinou, "Efficient IR- Style Keyword Search over Relational Databases," Proc. Int'l Conf. Very Large Data Bases (VLDB), 2003.
- [15] V. Hristidis and Y. Papakonstantinou, "DISCOVER: Keyword Search in Relational Databases," Proc. Int'l Conf. Very Large Data Bases (VLDB), 2002.
- [16] H.V. Jagadish, "Making Database Systems Usable," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD), 2007.
- [17] M. Jayapandian and H.V. Jagadish, "Expressive Query Specifica- tion through Form Customization," Proc. 11th Int'l Conf. Extending Database Technology: Advances in Database Technology (EDBT), 2008. [11] W.S. Cleveland, Visualizing Data. Hobart Press, 1993.
- [18] E. Kandogan, R. Krishnamurthy, S. Raghavan, S. Vaithyanathan, and H. Zhu, "Avatar Semantic Search: A Database Approach to Information Retrieval," Proc. ACM SIGMOD Int'l Conf. Manage- ment of Data (SIGMOD), 2006.
- [19] G. Koutrika, A. Simitsis, and Y.E. Ioannidis, "Explaining Structured Queries in Natural Language," Proc. Int'l Conf. Data Eng. (ICDE), 2010.
- [20] M. Ka 'ki, "Findex: Search Result Categories Help Users when Document Ranking Fails," Proc. SIGCHI Conf. Human Factors in Computing Systems, 2005.
- [21] Y. Li, H. Yang, and H.V. Jagadish, "Constructing a Generic Natural Language Interface for an XML Database," Proc. Int'l Conf. Extending Database Technology (EDBT), 2006.
- [22] F. Liu, C. Yu, W. Meng, and A. Chowdhury, "Effective Keyword Search in Relational Databases," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD), 2006.
- [22] Z. Liu and Y. Chen, "Identifying Meaningful Return Information for XML Keyword Search," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD), 2007.
- [23] Y. Luo, X. Lin, W. Wang, and X. Zhou, "SPARK: Top-k Keyword Query in Relational Databases," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD), 2007.
- [24] C.D. Manning, P. Raghavan, and H. Schu "tze, Introduction to Information Retrieval. Cambridge Univ. Press, 2008. [26] M. McCandless, E. Hatcher, and O. Gospodnetic, Lucene in Action, second ed. Manning, 2008.
- [25] F. Mesquita et al., "LABRADOR: Efficiently Publishing Relational Databases on the Web by Using Keyword-Based Query Inter- faces," Information Processing and Management: An Int'l J., vol. 43, no. 4, pp. 983-1004, 2007.
- [26] A. Nandi and H.V. Jagadish, "Assisted Querying Using Instant- Response Interfaces," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD), 2007.
- [27] G. Pass, A. Chowdhury, and C. Torgeson, "A Picture of Search," Proc. First Int'l Conf. Scalable Information Systems (InfoScale '06), 2006.
- [28] J.R. Quinlan, "Induction of Decision Trees," Machine Learning, vol. 1, no. 1, pp. 81-106, Mar. 1986.
- [29] S. Roy, H. Wang, G. Das, U. Nambiar, and M.K. Mohania, "Minimum Effort Driven Dynamic Faceted Search in Structured Databases," Proc. ACM Conf. Information and Knowledge Manage- ment (CIKM), 2008.
- [30] S. Tata and G.M. Lohman, "SQAK: Doing More with Keywords," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD), 2008.
- [31] T. Tran, P. Cimiano, S. Rudolph, and R. Studer, "Ontology-Based Interpretation of Keywords for Semantic Search," Proc. Sixth Int'l the Semantic Web and Second Asian Conf. Asian Semantic Web Conf. (ISWC), 2007.
- [32] P. Wu, Y. Sismanis, and B. Reinwald, "Towards Keyword-Driven Analytical Processing," Proc. ACM SIGMOD Int'l Conf. Manage- ment of Data (SIGMOD), 2007.
- [33] G. Zenz, X. Zhou, E. Minack, W. Siberski, and W. Nejdl, "From Keywords to Semantic Queries—Incremental Query Construction on the Semantic Web," J. Web Semantics, vol. 7, no. 3, pp. 166-176, 2009.
- [34] M.M. Zloof, "Query-by-Example: A Data Base Language," IBM Systems J., vol. 16, no. 4, pp. 324-343, 1977.
- [35] Q. Zhou, C. Wang, M. Xiong, H. Wang, and Y. Yu, "SPARK: Adapting Keyword Query to Semantic Search," Proc. Sixth Int'l the Semantic Web and Second Asian Conf. Asian Semantic Web Conf. (ISWC), 2007.
- [36] X. Zhou, G. Zenz, E. Demidova, and W. Nejdl, "SUITS: Constructing Structured Queries from Keywords," technical report, L3S Research Center, Hannover, Germany, <http://>

- www.l3s.de/~demidova/suits-TR.pdf, 2008.
- [37] Mirjana Mazuran, Elisa Quintarelli, and Letizia Tanca “Data mining for XML query Answering support” Vol. 24, No. 8, august 2012.
 - [38] 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.
 - [39] T. Asai, K. Abe, S. Kawasoe, H. Arimura, H. Sakamoto, and S. Arikawa, “Efficient Substructure Discovery from Large Semi-Structured Data,” Proc. SIAM Int’l Conf. Data Mining, 2002.
 - [40] E. Baralis, P. Garza, E. Quintarelli, and L. Tanca, “Answering XML Queries by Means of Data Summaries,” ACM Trans. Information Systems, vol. 25, no. 3, p. 10, 2007.
 - [41] 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.
 - [42] 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.
 - [43] Y. Chi, Y. Yang, Y. Xia, and R.R. Muntz, “CMTreMiner: Mining both Closed and Maximal Frequent Subtrees,” Proc. Eighth Pacific- Asia Conf. Knowledge Discovery and Data Mining, pp. 63-73, 2004.
 - [44] A. Evfimievski, R. Srikant, R. Agrawal, and J. Gehrke, “Privacy Preserving Mining of Association Rules,” Proc. Eighth ACM Int’l Conf. Knowledge Discovery and Data Mining, pp. 217-228, 2002.
 - [45] L. Feng, T.S. Dillon, H. Weigand, and E. Chang, “An XMLEnabled Association Rule Framework,” Proc. 14th Int’l Conf. Database and Expert Systems Applications, pp. 88-97, 2003.
 - [46] S. Gasparini and E. Quintarelli, “Intensional Query Answering to XQuery Expressions,” Proc. 16th Int’l Conf. Database and Expert Systems Applications, pp. 544-553, 2005.
 - [47] B. Goethals and M.J. Zaki, “Advances in Frequent Itemset Mining Implementations: Report on FIMI 03,” SIGKDD Explorations Newsletter, vol. 6, no. 1, pp. 109-117, 2004.
 - [48] G. Marchionini, “Exploratory Search: From Finding to Understanding,” Comm. ACM, vol. 49, no. 4, pp. 41-46, 2006.
 - [49] M. Mazuran, E. Quintarelli, and L. Tanca, “Mining Tree-Based Association Rules from XML Documents,” technical report, Politecnico di Milano, <http://home.dei.polimi.it/quintare/apers/MQT09-RR.pdf>, 2009.
 - [50] 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.
 - [51] K. Wang and H. Liu, “Discovering Typical Structures of Documents: A Road Map Approach,” Proc. 21st Int’l Conf. Research and Development in Information Retrieval, pp. 146-154, 1998.
 - [52] K. Wang and H. Liu, “Discovering Structural Association of Semi structured Data,” IEEE Trans. Knowledge and Data Eng., vol. 12, no. 3, pp. 353-371, May/June 2000.
 - [53] Y. Xiao, J.F. Yao, Z. Li, and M.H. Dunham, “Efficient Data Mining for Maximal Frequent Subtrees,” Proc. IEEE Third Int’l Conf. Data Mining, pp. 379-386, 2003.
 - [54] X. Yan and J. Han, “CloseGraph: Mining Closed Frequent Graph Patterns,” Proc. Ninth ACM Int’l Conf. Knowledge Discovery and Data Mining, pp. 286-295, 2003.