Research of Query Optimization Technology Based on XML Database

Xia Meiyun,Xin Wensheng
Information engineering institute of Henan Province Jiaozhuo university
2230600@sina.com.cn

***Abstract*-for the existing shortcomings in current XML document query optimization technology, this paper presents an efficient query optimization method based on view. The method can quickly and efficiently find an effective view, thus to greatly improve the speed of query and better achieve the XML document query optimization. And then it designs a XML document processing system, and through experimental results verifies that the proposed query optimization technique is feasible and effective.**

Keywords-XML; query optimization; database; view

I. INTRODUCTION

With the continuous expansion of XML applications, more and more data is represented and stored by using XML standards. From the database point of view, the data in these many XML documents can be collected and analyzed. How to search the contents of these documents is becoming increasingly important. Because of the characteristics of Web data, the data XML documents described must be irregular and incomplete, that is, the so-called "semi-structured data" [1] , which is different from our common relational data model or object-oriented data model, so how to effectively query XML documents is a hot research problem.

II. XML DATABASE TECHNOLOGY

A. XML technology

XML (Extensible Markup Language) because of its support for interactive information systems and data integration, data retrieval precision, long life and other unique advantages received general attention of information technology community. XML is a set of norms the W3C created, which is derived from the SGML, and SGML is a language to create markup languages, a meta-markup language. XML combines the rich features of SGML and HTML ease of use, and retains the SGML scalable features, which makes it fundamentally different from HTML. XML redefined the parameters and internal values of SGML, making programming simplified, easy to spread and interact on Web. XML mainly has the following important features: scalability, flexibility, self-descriptive, and concise.

B. XML database categories

XML database is a collection of XML documents, and these documents are long-lasting and operational. Currently there are three types of XML databases:

1) XML Enabled Database (XEDB) is a database supported for XML. It is to expand the XML data processing functions based on the original database system, to make it adapt to XML data storage and query needs.

2) Native XML Database (NXDB) is a pure XML database. It is to naturally process XML data, make the XML documents as the basic logic storage unit, and specially design the adapted data model and approach against the XML data storage and query features, and need not to perform any conversion work while accessing to XML data, thus can provide better performance [3] .

3) Hybrid XML Database (HXDB) is hybrid XML database, which can be regarded as XML-supported database or pure XML database according to application requirements.

C. XML database key technologies

1) XML database storage technology

The XML document storage ways in XML database are mainly three: in text format to store the entire document, model-based approach and the hybrid mode. The three ways have their own characteristics, in general, to return to the text format data, the appropriate use of text-based systems; to return to DOM, the use of model-based system is faster.

2) XML database index technology

Index is an important data structure in database, which is an important method to improve query efficiency. To establish an effective index structure against XML data can fast, directly and effectively improve the efficiency of search queries, while reducing the time or cost.

III. VIEW-BASED XML QUERY OPTIMIZATION IMPROVEMENT

A. XML database query structure

Different pure XML databases have different architecture, but at least to include storage management, index management, query processing, data import, transaction management, concurrency control modules and other basic function modules, a common pure XML database architecture as shown in figure 1.

978-1-4244-8102-6/10/$26.00 © 2010 IEEE

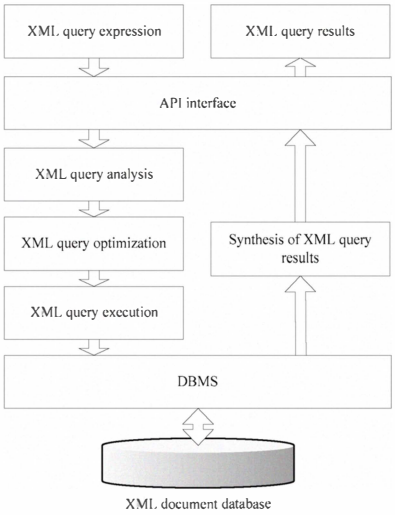

Figure 1. XML database architecture

From the figure it can be seem that the XML query processing in pure XML database mainly has four main steps: query parsing, logic optimization, physical optimization, and query execution.

1) Query parsing

It is to convert the query expression to some kind of internal expression to be used to access the data within XML documents, thus to facilitate machine processing, and pave the road for the next optimization process.

2) Logic optimization

It uses model information, specifications, and internal expression. At this stage, the system does not take into account the value of actual data and data storage situation. The same query can be converted into different equivalent expressions.

Judging from the level, logic optimization can be divided into two levels: syntactic and semantic level. Syntax-level optimization is to independently analyze the query expression branch or the logic inclusion relations between nodes without relying on any other information, to remove the redundant part; semantic-level optimization is to search the redundancy branch or node in query expressions through the mode information the database provided [2], or semantic inclusion, structure inclusion and other integrity constraints.

3) Physical optimization

Logic optimization result is one or more query trees, how to determine the execution order of different query fragments is the core of XML physical optimization. Physical optimization is to use cost model and statistical information to calculate the implementation price of different expressions and different algorithms, thus to choose the query plan with the lowest price.

4) Query execution

It is to access to data and get the query results according to the implementation strategies and algorithms the physical optimization identified.

B. View-based XML query optimization improvement

View-based query rewriting technique is an important technology of query optimization, which is a fundamental problem of database research. It is closely related to the query optimization, data warehousing, information integration, semantic caching and other issues, has more and more attention. Currently Internet has a flood of semi-structured data, and generates a large number of semi-structured views in information integration process, so how to use semi-structured view to rewrite the user queries, reduce the response time becomes a hot issue.

1) The basic idea of original view-based query rewriting

The basic idea of using the original view-based query rewriting to achieve query optimization is to directly use the existing view query results to reduce the number of database visits, thus to improve the query efficiency. The mainly work to achieve the query rewriting is: to select effective views from the large of views; to find the candidate mapping from the view mapping relationship; query rewriting and so on. But this method is to obtain the correct query rewriting program through complete traverse the candidate rewriting program space, but there is no cut of the program in the index space, so the specific implementation of this algorithm is less efficient.

2) The basic idea of improved view-based query rewriting

This paper introduces some of the strategies of query rewriting bucket algorithm in relational database based on the original query rewriting, that is, to use the link between the sub-objectives in the inquiry and the characteristics of semi-structured data query itself, to limit the generated rewriting space from the query and view aspects, thus to obtain minimal candidate query rewriting plan [7].

The view-based XML query rewriting presented in this paper is divided into two steps, in which, the first is to generate all possible candidate programs for query rewriting, and there is exponential relationship between the number and the number of sub-query, sub-view. The second step is to verify all the generated candidate query rewriting programs to find the right program, because the view and query mapping in the first step candidate query rewriting program, this step of query rewriting program test can be completed in polynomial time.

3) Implementation of improved query rewriting

a) Regular path expression feature value extraction

This process can be divided into two parts, the first is conversion process, to convert the regular path expression into the determination of the finite state machine with the smallest number of states; the second is the extraction process, to extract the corresponding feature value from the determined finite state machine. There is an algorithm to achieve this process, and this algorithm uses breadth-first approach, MDAF state transition function as the point basis of the modes in breadth-first traversal, the initial state as a starting point for search, to completely cover the MDAF map, thus to get the feature value.

b) Establishment of view index structure

To find a useful view for a certain query rewriting from a lot of views can well improve the query efficiency. At present, databases commonly use index technologies to establish an index structure for the view, and then through its search to achieve an effective view search. The main steps to build an index are: first determine whether the view is a valid view for the query rewriting, it needs to examine what kind of conditions to satisfy between the query and view; second, design a graph-like structure with the feature value of each view as node, the containment relationship between them as the connection, to establish an index for the view, and then use it to quickly find a group of view collection can be used for query rewriting.

c) Effective view selection algorithm

Its basic idea is to use the view index structure to quickly eliminate the unwanted views, get a set of candidate views, and then carry out a consistency test between the path contained in each candidate view and the path to be found, thus to find the effective views [6]. This can fast obtain a group of candidate views, and get the effective view only with comparison between the candidate view and query, which can greatly reduce the number of comparison to increase efficiency.

IV. XML QUERY PROCESSING SYSTEM

A. System design

In order to improve the efficiency of traditional XML query processors, this paper designs a query processing system based on view technology, to execute XML query optimization treatment and query calculation. This system consists of three modules: the query optimization module for document query, effective view management module and query calculation module, the overall framework as shown in figure 2.

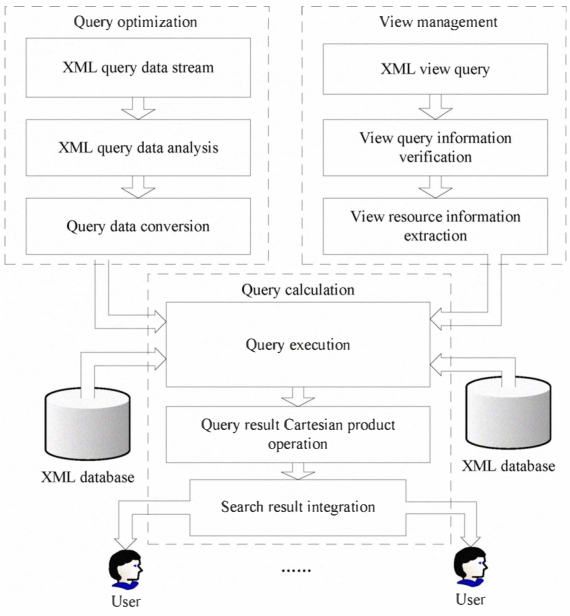

Figure 2. XML document query processing system overall framework

1) XPath query optimization module: the main function is lexical, syntax analysis for query XPath expressions users submitted, to achieve the flow query of XML documents.

2) Effective view management module: the main function is to store and manage the existing views to quickly, efficiently extract effective view information.

3) Query calculation module: the main function is to perform queries, that is, queries of the view in the memory or the data stream of XML document, and carry out Cartesian product operation for the query results [3], thus to get the final results of user queries.

B. Experimental performance analysis

Through the above XML document processing system analysis it can be seen that this system is mainly to achieve XML document query optimization. Therefore, we will make the above system as test system, and through the experimental results and analysis to prove that the system this paper designed and its used query optimization techniques plays an important role in the improvement of XML document query efficiency.

The data set used in this experiment is XMark, check set from 50 to 400, the XML document stream is 5MB, the maximum depth of document tree is 20, and the maximum recursion depth of it is 9. The query tool uses the automatically generated XPath tool, and set the maximum depth of each query is 6, the largest branch node number is 4. In the experimental data statistical process, each group of query will be run for 10 times, and the average processing time is taken. The table 1 is the experimental results of query processing for the same XML document with different query conditions.

TABLE I. THE TIME OF SAME XML DOCUMENTS' MULTI-INQUIRIES

Types	Query number					
	50	100	150	200	300	400
Optimized	12.88	21.98	28.32	56.12	69.25	98.65
Non-optimized	16.01	28.39	36.87	71.36	92.58	121.18

As can be seen from the table, in the system using the proposed optimization techniques, the average query response time compared to not using optimization techniques, has been significantly increased, and with the growing number of inquiries, query efficiency improvement will become increasingly evident. This verifies that the proposed XML query optimization technology has a very significant effect to reduce the query response time and improve the query efficiency.

V. CONCLUSION

XML in recent years became the standard of data representation in scientific and business applications, and data exchange of Web applications [4]. As the massive growth of Web data, in order to extract the potential of XML, XML query optimization is the technical aspect must to be solved. This paper presents an efficient query optimization method based on view. The method can quickly and

efficiently find an effective view, thus to greatly improve the speed of query and better achieve the XML document query optimization. And then it designs a XML document processing system, and experimental results show that the proposed query optimization technique has a very significant effect to reduce the query response time and improve the query efficiency, thus has high availability.

REFERENCES

[1]. Milo Tova, Suciu Dan, Vianu Vietor, Type checking for XML transformers Journal of Computer and System Sciences [J], 2003, 66(l): 66-97.

[2]. C Y Chan, P Felber, M Garofalakis et al. Efficient filtering of XML documents with XPath Expression. California: International Conf. Data Engineering, 2002.

[3]. Rachel Pottinger and Alon Levy, A scalable algorithm for answering queries using views, In Proc. of VLDB Journal, 2001.

[4]. W3C DOM Working Group, Document Object Model (DOM) Level 3 Core Specification. W3C, 2003: 7-11.

[5]. H. Jiang, W. Wang, H. Lu, J. X. Yu. Holistic twig joins on indexed XML documents [C]. In: Proceedings of the Conference on Very Large Data Bases.2003:273-284.

[6]. B. Ludascher, P. Mukhopadhayn, and Y Papakonstantinou, A transducer-based XML query rocessor, In Proceedings of 28th International Conference on Very Large Data Bases, Hongkong: Angust 2002.

[7]. M.Brantner, S.Helmer, C.-C. Kanne, and G. Moerkotte. Full-fledged Algebraic XPath Processing in Natix [C], In Proc.21st Int. Conf. on Data Engineering, 2005: 705-716.