MQX: Multi-Query Processing Engine for Compressed XML Data

Xiaoling Wang, Aoying Zhou, Juzhen He
Department of Computer Science and Engineering
Fudan University
China, 200433
wxling@fudan.edu.cn

Wilfred Ng
Department of Computer Science and Engineering
Hong Kong University of Science and Technology
HongKong
wilfred@cse.ust.hk

ABSTRACT

In this demonstration, we present an XML query engine MQX, which is developed for processing multiple subscribed XPath queries over compressed XML documents. MQX is equipped with efficient mechanisms for query rewriting, query organization and query optimization. To our knowledge, this is the first prototype to address the problem of efficiently processing multiple XML queries in a co-operative framework based on efficient XML-conscious compression technique. To demonstrate the novel features of MQX, we build a co-operative network and implement a content-based subscription system. The most distinguishing feature of our prototype is that multi-query over compressed XML data can be processed as a whole, resulting in faster data dissemination. Another important feature is that the engine reduces the bandwidth consumption greatly compared to the traditional XML query engines SAXON.

1. INTRODUCTION

In this demonstration, we present a prototype called MQX which is capable of processing multiple subscribed XPath query over compressed XML data.

First, unlike relational data and distributed SQL, XML data has a complex structure in which the XPath query may consist of double slash ("//"), wildcard ("*") and branch prediction ("[ ]"), which is fundamentally different from SQL expressions.

Second, XML data are verbose due to their repeated tags and structures. Most XML documents are compressed in order to reduce the storage size on the server and during publication. Previous work has studied the techniques for efficient evaluation path expressions on original XML document [3] or single-query processing on compressed data [2,5] based on the different XML compression methods. However, it is still not clear how to exploit the existing XML compression techniques to enhance XML subscribe/dissemination applications [4]. It is not uncommon to have the network jam-packed when the workload of the server is heavy. Compressing documents is promising to alleviate the problem of heavy loading in these applications.

Therefore, the problem motivates us to develop MQX, which provides valuable strategies to disseminate compressed results to clients over the network. The system is able to execute users' subscribed queries over compressed XML data stored in the server.

The remaining part of this paper is organized as follows. We briefly explain the architecture of MQX in Section 2. In Section 3 we discuss the research underlying MQX. Finally, Section 4 presents a demonstration plan.

2. THE CO-OPERATIVE NETWORK

To demonstrate the features of MQX, we build a co-operative network and implement a content-based subscription system. Assume that there are some co-operation relationships among clients where the server keeps a large number of compressed XML documents, and clients request to obtain information or news from the server in a cooperative way, i.e., each client can send results to others. To prevent the server from becoming a bottleneck, we adopt a distributed approach where all clients participate in the dissemination process.

The procedure is described in the following steps.

1. The XML documents compressed by an interval encoding technique [2] and dictionary encoder. The compressed documents are stored in the server.

2. The clients subscribe XPath queries to server.

3. The server rewrites each subscribed XPath query into query units which can be evaluated directly over the compressed documents and reorganizes these units together. By analyzing the containment relationship among queries, the submitted queries are organized into a novel Structural-Query-Index Tree (SQIT)[1] which supports path sharing in multi-query processing. The results of query can be located in the compressed document by using the query index. SQIT will be detailed in Section 4.

5. After query evaluation, the server sends the result fragments and corresponding sub-tree extracted from SQIT to its first-level children clients according to the SQIT.

6. Each client Ci receives a result fragment <Fi, Si>, where Si is used as the sub-index to help Ci to obtain results for its children from the fragment of Fi. The location of results for each child query has been recorded in the sub-index Si, thus Ci can locate them directly and sends new result fragments and related sub-index to corresponding clients.

7. This procedure halts when all leaf clients of SQIT receive their query results.

3. MQX ENGINE

Figure 1 illustrates the structure of MQX engine, which consists of four main components described below.

The first one is the XML Compression Tool (abbreviated to XCT) component, which facilitates data compression and storage.

The second one is the multi-query processor, called MQEngine. The MQEngine is the kernel of our system, which supports query

rewriting, multi-query organization and optimization. Each query is rewritten into the processing units and organized into SQIT[1].

The third component is the Dissemination Manager (DM) which sends the results to specific clients in the first-level of SQIT. The server delivers the compressed XML fragments to clients with query nodes as child of root in the SQIT. Then, each client sends the related results contained in its fragment to its children nodes in SQIT.

The fourth component is the interface unit for the clients, which provides a user-friendly interface to show the structure of the SQIT and the result of the current node.

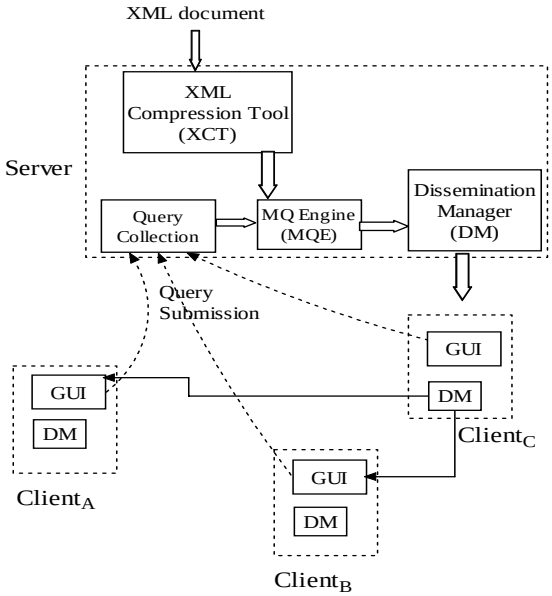

Figure 1 A Simplifed Architecture of MQX

4. DEMOSTRATION PLAN

In this demonstration, we plan to set up a network with at least four laptops to illustrate the MQX applications. One acts as server, other three are clients. In order to simulate the real-life scenario, we also build another experimental network that simulates thousands of (virtual) clients, which interact with the server and laptops via a load of subscribed queries.

We demonstrate our system using the XMark dataset [7] and NITF (new industry text format) dataset [6]. XMark contains about 40 original queries. The queries for NITF are created by YFilter path generator, where the size of XML document ranges from 1KB to 50MB. We have four specific objectives as follows.

1. We show the efficiency of this co-operative environment, where no decompression is involved in query processing.

2. We present the dissemination process and clarify how MQX distributes the computation for query processing and how the computation load of the server is reduced.

3. We show the efficiency of the MQEngine by comparing it with SAXON [8]. We compare the query processing time between MQEngine and SAXON by testing complex XPath queries over the XMark documents with size ranging from 1MB to 10MB. The performance of our approach is shown in Figure 2. Our approach outperforms SAXON when the document size increases.

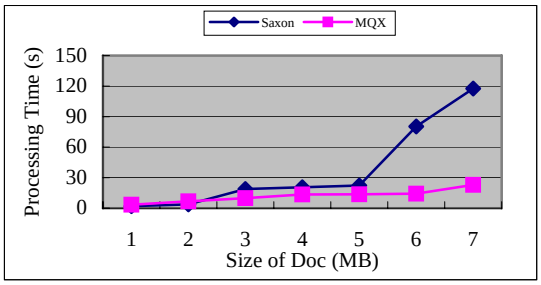

Figure 2 SQIT vs. SAXON

4. We demonstrate the benefit of bandwidth consumption reduction by comparing the XML fragment results between our approach and common processing strategies which return original XML fragments. As Figure 3 shows, the gain of bandwidth consumption reduction is very large, which proves that our approach saves the bandwidth significantly.

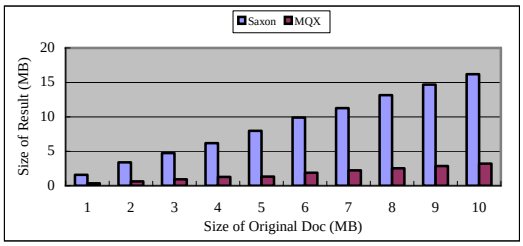

Figure 3 Benefits of Result Size

5. REFERENCES

[1] J. He, W. Ng, X. Wang and A. Zhou. An Efficient Co-operative Framework for Multi-Query Processing over Compressed XML Data. International Conference of Database Systems for Advanced Applications. DASFAA 2006

[2] J. Min, M. Park, Chin-Wan Chung. XPRESS: A Queriable Compression for XML Data. In Proc. Of ACM SIGMOD, 2003

[3] X. Dong, A.Y. Halevy, I. Tatarinov. Containment of Nested XML Queries, In Proc. of the 30th VLDB, 2004

[4] Y. Diao, S. Rizvi, M. J. Franklin. Towards an Internet-Scale XML Dissemination Service. In Proc. of the 30th VLDB, 2004

[5] P. Buneman, M. Grohe, C. Koch. Path Queries on Compressed XML. In Proc. of the 29th VLDB, 2003

[6] NITF. http://www.nitf.org/index.php

[7] XMark benchmark. http://www.xml-benchmark.org/

[8] SAXON. http://saxon.sourceforge.net