"To Optimize XML Query Processing using Compression Technique"

Lalita Dhekwar
Computer engineering department
Nagpur institute of technology,Nagpur
Lalita_dhekwar@rediffmail.com

Prof. Jagdish Pimple
Computer engineering department
Nagpur Institute of Technology. Nagpur

Abstract- XML is a standard for exchanging and presenting information on the Web because XML makes data flexible in representation and easily portable as well. However, XML data is also recognized as verbose since it heavily increase the size of the data due to the repeated tags and structures. The data verbosity problem gives rise to many challenges of conventional query processing and data exchange. The XML increase the overhead of bandwidth-and memory-limited devices.XML compression and optimization are one of the solutions of the verbosity problems of XML.Although many effective XML compressors, such as XMill, have been proposed to solve the data size problem but it does not address the problem of running queries on compressed XML data. Other compressors have been proposed to query compressed XML data..

Keywords- *: XML, data compression, query processing, Web applications.*

*****

I. INTRODUCTION

Extensible Markup Language (XML) is becoming the standard format for electronic data storage and exchange. With the increasing popularity of XML data format, it can also widely use in the business applications as a database or storing the metadata information as well. XML defines a set of rules for converting the simple data into the standard format. The main characteristics of XML which makes XML more popular in data communication are

• Self-Descriptors: It is possible to understand an XML document without external context.

• Evolvable and Extensible: XML formats can be designed to evolve gracefully overtime, whereas retaining backwards and forwards compatibility.

• Isomorphic: XML formats are constant in the all the environment so in the hybrid environment the XML format is used for data communication.

• Human-Readable: XML, as a text format, is easy for people to read and understand with common, non-proprietary tools (i.e., a text editor), and is simple to modify with those same tools. This encourages the adoption of XML-based formats and aids greatly in debugging and ad hoc uses of them.

• Simplicity: XML format is very simple to use with few encodings. It supports the Unicode format for supporting the world language characters.

XML is often referred to as self-describing data because it is designed in a way that the schema is repeated for each record in the document. On one hand, this self-describing feature provides XML with immense flexibility but on the other hand, it also introduces the main problem of verbosity of XML documents which results in huge document sizes. This huge size lead to the fact that the amount of information has to be transmitted, processed, stored, and queried is often larger than that of other data formats. This can be a serious problem in many occasions, since data has to be transmitted quickly and stored compactly. Large XML documents not only consume transmission time, but also consume large amounts of storage space.

The problem can be addressed if XML compression techniques are used to reduce the space requirements of XML. There are two types of compressors on the basis of XML awareness: General text based compressor and XML concise compressor.

The general text based compressors are XML-Blind, treats XML documents as usual plain text documents and applies the traditional text compression techniques. gZip, WinRAR, 7Zip are belongs to this group. 7Zip compressor by default uses the 7Z format and that format by default uses the LZMA method to compress the data. LZMA is an improved version of LZ77 algorithm which improves the data compression ratio.

XML concise compressors are aware about the XML structure so that they can take an advantage of XML structure to compress the data to increase the compression ratio. The XMill is one of the types of XML compressor which eliminates the redundant data by identifying the similarities between the semantically related data. The XMill also uses the gZip library to compress the XML string type of data. To improve the compression ratio of XMill

compressor this work added 7Zip library in addition with gZip to compress the XML string data.

II. RELATED WORKS

A very large number of XML compression techniques have been proposed in the recent years. This section gives the overview the few of the most promising techniques namely gZip, XMill, XGrind, Xpress, and XComp.

2.1 GZip

gZip is the most widely used commercial compressor available. Since XML is represented as a text file, an obvious choice for a compression tool would be a general-purpose compression tool such as gZip. gZip was developed by Jean-loup Gailly and Mark Adler and uses a combination of the LZ77 algorithm and Huffman coding [2].
The benefits of using such a technique would be:

- This tool is widely available in both open-source and commercial implementations

• gZip provides better compression rate (40-50%) and freedom from patented algorithms

• Using gZip requires no knowledge of the document-structure.

• gZip is built into http and web-servers as a standard feature.

However, the main disadvantage with using gZip to compress XML files is:

• Compression of elements/attributes may be limited due to the long-range dependencies between elements and between attributes.

This means that generic compression algorithms are limited because they do not leverage document semantics [2].

2.2 XMill

XMill was developed by Hartmut Liefke and Dan Suciu in the summer of 1999. XMill was designed at AT&T Labs Research in New Jersey, USA. XMill is based on a regrouping strategy that takes advantage of the XML elements. XMill groups XML text strings with respect to their meaning and exploits similarities between those text strings for compression. Hence, XMill typically achieves much better compression rates than conventional compressors such as gZip [2] [3].
XMill applies three principles to compress XML data [2][3]:

• **Separate structure from data**: The structure, consisting of XML tags and attributes, is compressed separately from the data, which consists of a sequence of data items (strings) representing elements contents and attribute values.

• **Group related data items**: Data items are grouped in to containers and each container is compressed separately. For example, all<name> data items form one container, while all <phone> items form a second container. This is a generalization of column-wise compression in relational databases.

• **Apply semantic compressors**: Some data items are text or numbers, while others may be DNA sequences. XMill applies specialized compressors (semantic compressors) to different containers.

Benifits

XMill achieves better compression rate compared to gzip (by a factor of 2, for data-like XML documents) without sacrificing speed. This owes to the fact than it separates structure from content. This makes it a clear winner for applications like data archiving since these applications require lesser disk space. At the same time, it reduces network bandwidth. XMill is moderately faster than gzip in XML publishing. However, relative advantage of XMill depends on the application it is used [2].

Disadvantages

• Compressed output of XMill is not queryable.

• If the size of the input document is less than 20KB, XMill will not exhibit any significant advantage over gzip.

• It requires user assistance to achieve the best compression [2] [3].

2.3 XGrind

XGrind is a compression tool for XML documents developed by Pankaj M. Tolani, and Jayant R. Haritsa of the Indian Institute of Science in July 2002. XGrind supports querying the compressed document. At the same time, this tool retains the structure of the original XML document too. This facilitates reuse of standard XML techniques for processing the compressed document. These features make it very important in applications hosted on resource-limited computing devices like Palm Tops [2].
The two main techniques involved in this compression are:

• **Meta-Data Compression**: This is similar to that of XMill.

• **Enum-type attributed value compression**: This is captured in the DTD itself. This is accomplished by XGrind by examining the DTD of the document and encoding values using a simple encoding scheme to represent an enumerated domain of values.

Benefits

• Considerable improvements in query response time

• Disk bandwidth is effectively increased as increased information density

• Memory usage is more while performing the XML compression.

Disadvantages

• XGrind does not support several operations like non-equality selections. In addition, it cannot perform any join, aggregation, and nested queries or construct operations [2].

2.4 Xpress

Jun-Ki Min, Myung-Jae Park and Chin-Wan Chung of Korea Advanced Institute of Electrical Engineering & Computer Science (KAIST) developed XPRESS and the paper was presented at SIGMOD 2003. XPRESS is a compressor that supports direct and efficient evaluations of queries on compressed XML data. XPRESS adopts a novel encoding method, called reverse arithmetic encoding. XPRESS achieves significant improvements on query performance for compressed XML data and reasonable compression ratios [2].

The primary characteristics of XPRESS:

• **Reverse Arithmetic Encoding**: Unlike existing compressors that represent each tag by a unique identifier (making it inefficient to handle path expressions), XPRESS adopts reverse arithmetic encoding that encodes a label path as a distinct interval in [0.0,1.0]

• **Automatic Type Inference**: A Type Inference Engine is devised here thus avoiding any human intervention in determining the types of data values.

• **Apply Diverse Encoding Methods to Different Types**: Depending on the inferred data type received from the type inference engine, it can apply proper encoding methods to these data values. Hence, it can achieve a high compression ratio and minimal partial decompression overhead as mentioned above.

• **Semi-adaptive Approach**: XPRESS employs a semi-adaptive approach. Unlike XGrind, the statistics is not changed during compression. At the same time, encoding rules are independent to the location.

Homomorphic Compression: This preserves the structure of the input XML document. Thus, XML segmentation that satisfies given query conditions is efficiently extracted [2].

2.5 XComp

An alternative XML-specific compression tool presented in this paper is XComp that was developed as a thesis paper by Weimin Li in the University of Waterloo. Similar to XMill, XComp takes advantage of the XML's self-describing feature to separate structure from data based on the semantics.

The two main principles employed in XComp are

1. Separating document structure from data
2. Grouping content data based on semantics.

Three basic components of XML structure, namely tags, attributes, and data items play major role in the development of XComp. This feature is used mainly in order to use specific methods to process structure so that the compression ratio could be improved significantly. Related data will be compressed together after reorganization

III. PROBLEM STATEMENT

The xml standard contains following problems or issues

- There is lot of “redundant” in xml document, including white space, and element and attribute names. there is lot of "redundant" data in xml documents, including white space, and element and attribute names.
- XML is self-describing and document size is larger than other formats. this will affect query processing. xml is self-describing and document size is larger than other formats. this will affect query processing. xml is self-describing and document size is larger than other formats. this will affect query processing.
- As a self-describing format, xml brings flexibility, but compromises efficiency as a self-describing format, xml brings flexibility, but compromises efficiency.
- Most xml documents are stored in file systems, so we need an efficient way to store file-based xml. most xml documents are stored in file systems, so we need an efficient way to store file-based xml.
- Web services use the xml for data communication due to this large volume of xml data to be sent over networks. reducing data size helps conserve network bandwidth web services use the xml for data communication due to this large volume of xml data to be sent over networks. reducing data size helps conserve network bandwidth.

- This work is an attempt to resolve all or few of the above xml issues.

IV. OBJECTIVES

The main aim of this work is to optimization of existing methods of XML compression. This work is an attempt to address all or few of the following defined problems:

• Optimize the XML data transmission network bandwidth.

• Optimize the XML file memory storage size.

• Optimize the memory utilization while performing the compression.

• Optimize the manual approach in compression technique.

• Improve the compression ratio with respect to already implemented few compression techniques.

V. PROPOSED SYSTEM ARCHITECTURE

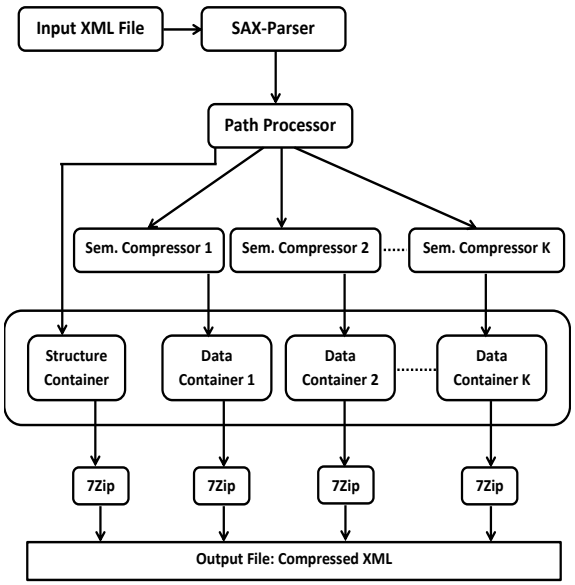

Figure 1: Proposed System Architecture for Compression

- The XML file is parsed by a SAX8 parser that sends tokens to the path processor. Every XML token (tag, attribute, or data value) is assigned to a container. Tags and attributes, forming the XML structure, are sent to the structure container. Data values are sent to various data containers, according to the container expressions, and containers are compressed independently.
- The core of XMill is the path processor that determines how to map data values to containers. The user can control this mapping by providing a series of container expressions on the command line. For each XML data value the path processor checks its path against each container expression, and determines either that the value has to be stored in an existing container, or creates a new container for that value.
- Containers are kept in a main memory window of fixed size (the default is 8MB). When the window is filled, all containers are compressed by 7Zip LZMA algorithm, stored on disk and the compression resumes. In effect this splits the input file into independently compressed blocks.
- The decompressor is simpler, and its architecture is also similar to compressor architecture but data flow from bottom to top instead top to bottom and it is the contrast of compression process. After loading and decompressing the containers, the decompressor parses the structure container, invokes the corresponding semantic decompressor for the data items and generates the output.

VI. PRAPOSED WORK

The XMill compressor achieves better compression ratio with good compression time, so XMill compressor has been chosen for the optimization process. XMill compress data on the basis of three principles: separate the XML structure, group related data item and apply sematic compressors. Once these steps are performed then data container and structure container are compress by using the gZip library. The compression ratio of XMill compressor can be improved by adding the 7Zip library in it. So this work provided one more options in XMill tool for compressing the XML data by using 7Zip library. By default, this library uses the LZMA algorithm to compress data. LZMA algorithm uses up to 4GB length dictionary for eliminating the duplicate string due to this compression ratio improved as compared to LZ77 algorithm but long size of dictionary makes this algorithm slower. LZ77 algorithm depends on the greedy approach for parsing whereas LZMA depends on the Look-a-head approach which makes the compression process more effective than the gZip

REFERENCE

[1] Smitha S. Nair, "XML Compression Techniques: A Survey," Department of Computer Science, Universsity of Iowa, USA,.

[2] H. Liefke and D. Suciu, “XMill: An Efficient Compressor for XML Data,” Proc. of ACM SIGMOD Intl. Conf. on Management of Data, May 2000.

[3] Pankaj M. Tolani, Jayant R. Haritsa, “XGRIND: A query-friendly XML compressor,” in: ICDE ’02: Proceedings of the 18th International Conference on Data Engineering, IEEE Computer Society, Washington, DC, USA, 2002, p. 225.

[4] Jun-Ki Min, Myung-Jae Park, Chin-Wan Chung, “XPRESS: A queriable compression for XML data,” in: Proceedings of the ACM SIGMOD International

Conference on Management of Data, ACM, 2003, pp. 122–133.

[5] J. Cheney, “Compressing XML with Multiplexed Hierarchical PPM Models,” in: DCC ’01: Proceedings of the Data Compression Conference (DCC ’01), IEEE Computer Society, Washington, DC, USA, 2001, p. 163.

[6] Przemysław Skibiński, Szymon Grabowski and Jakub Swacha, "EFFECTIVE ASYMMETRIC XML COMPRESSION," Publication in Software—Practice and Experience, 2007.

[7] Sherif Sakr, "XML compression techniques: A survey and comparison," Elsevier, Information and Software Technology, 75 (2009) 303–322, 2009.

[8] Wikimedia Foundation, Inc, “Lempel-Zip-Markov chain algorithm,” 5 June2014,.

[9] Wilfred Ng, Lam Wai Yeung and James Cheng, "Comparative Analysis of XML Compression Technologies," World Wide Web: Internet and Web Information Systems, 9, 5–33,Springer Science + Business Media, Inc. Manufactured in The Netherlands, DOI: 10.1007/s11280-005-1435-2, 2005.

[10] H. Liefke and D. Suciu, “An Extensible Compressor for XML Data,” Proc. of ACM SIGMOD Intl. Conf. on Management of Data, 2000.),