IOSR Journal of Computer Engineering (IOSRJCE)
ISSN: 2278-0661, ISBN: 2278-8727Volume 8, Issue 4 (Jan. - Feb. 2013), PP 41-45
www.iosrjournals.org

Query Evaluation for XML Databases using Partial Decompression

Vijay Gulhane[1], M.S.Ali[2]
[1](*Department of computer Science and Engg.,Sipna College Of Engineering and Tech/ Amravati University India)*
[2](*PRM&M College Of engineering and Management/Amravati University , India)*

Abstract:*This research work demonstrates the Extraction, compression and query processing of XML documents for Adaptive Compression Techniques and Efficient Query Evaluation. Proposed here are the algorithms for xml compression and Efficient Query Evaluation as - Feasible XML compression using data compression algorithm. Qurey Processor using Sax parsing and Interfaces. It is shown that using the proposed techniques for xml data compression will pave a way for better compression and improve the compression ratio and performance of compressor system*

Keywords: *XML compression, partial decompression, SAX parser, compressor Systems, Query processor, Ziv-Lempel algorithm.*

I. INTRODUCTION

The proposed research aims at the investigation of efficient Partial query processing techniques on compreesed xml dataset in database systems. Mainly our focusing on query proceeding time and memory space.There are a number of XML compression techniques available today which were developed, and tested over the last few years. The problem with XML is that it is text-based, and verbose by its design (the XML standard explicitly states that terseness in XML markup is of minimal importance"). As a result, the amount of information that has to be transmitted, processed and stored is often substantially larger in comparison to other data formats. This can be a serious problem in many occasions, since the 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.[H. Liefke and D. Suciu. (2000)]. This section enumerates a few of the most promising techniques namely XGrind, XQC and The evaluation of these compressors considers three main aspects: compression speed, compression ratio, [Al-Hamadani, Baydaa (2011)]

A very large number of XML compressors have been proposed in the literature of recent years. These XML compressors can be classified with respect to two main characteristics. The first classification is based on their awareness of the structure of the XML documents. The main objective of this research work is to initiate an enquiry XML compression in real-time database systems (RTDBS). Xml documents are identified as the major resources that need to be compressed and effectively partial decompression even at the compressed database level with support from underlying operating systems. The main criteria in assessing any XML compressor and decompressor, the success ratio in terms of the CR1 and CS(compression ratio and compressed size) .

II. PRINCIPLES OF THE PROPOSED XML COMPRESSION SCHEME

Several compression techniques have been adopted in these tools, and with varying results. Unfortunately, it is not easy to judge which compressor is the best, or which is the worst. Such a ranking would be necessarily unfair, since each of the compressors that we are aware of differs from the others in some way, and is suited for different purposes. For example, some of the compressors allow transparent access to the compressed data via the SAX or DOM interfaces (XMLZip or Millau, for instance), but achieve poor compression results. On the other hand, there are XMill and XMLPPM that compress very well, but there is no way to parse the compressed data unless it is decompressed first. Also, some compressors work incrementally and allow on-line processing of the data (XMLPPM), while some are offline by their design (XMill). There is no clear winner |no \ideal" XML compressor|, and one has to formulate his requirements first before he picks and applies the compressor that seems to be the most suitable.

We believe that there are still many paths to be explored in the field of XML data compression. In this work, we present a syntactical compression scheme that is based on probabilistic modeling of XML structure. It does not need the DTD since it infers all the necessary information directly from the input XML data. Moreover, it works incrementally, making on-line compression and decompression possible. Transparent parsing of the compressed data using the SAX interface is possible.

III. ALGORITHM AND CORPUS SET

3.1Lempel-Ziv Codes

Lempel-Ziv coding represents a departure from the classic view of a code as a mapping from a fixed set of source messages (letters, symbols or words) to a fixed set of codewords. We coin the term free-parse to characterize this type of code, in which the set of source messages and the codewords to which they are mapped are defined as the algorithm executes. While all adaptive methods create a set of codewords dynamically, defined-word schemes have a fixed set of source messages, defined by context (eg., in text file processing the source messages might be single letters; in Pascal source file processing the source messages might be tokens). Lempel-Ziv coding defines the set of source messages as it parses the ensemble.

The Lempel-Ziv algorithm consists of a rule for parsing strings of symbols from a finite alphabet into substrings, or words, whose lengths do not exceed a prescribed integer L(1); and a coding scheme which maps these substrings sequentially into uniquely decipherable codewords of fixed length L(2) [Ziv and Lempel 1977]. The strings are selected so that they have very nearly equal probability of occurrence. As a result, frequently-occurring symbols are grouped into longer strings while infrequent symbols appear in short strings. This strategy is effective at exploiting redundancy due to symbol frequency, character repetition, and high-usage patterns. Figure 3.1 shows a small Lempel-Ziv code table. Low-frequency letters such as Z are assigned individually to fixed-length codewords (in this case, 12 bit binary numbers represented in base ten for readability). Frequently-occurring symbols, such as blank (represented by _) and zero, appear in long strings. Effective compression is achieved when a long string is replaced by a single 12-bit code.

Symbol string	Code
A	1
T	2
AN	3
TH	4
THE	5
AND	6
AD	7
_	8
__	9
___	10
0	11
00	12
000	13
0000	14
Z	15
###	4095

Fig 3.1 -- A Lempel-Ziv code table.

The Lempel-Ziv method is an incremental parsing strategy in which the coding process is interlaced with a learning process for varying source characteristics [Ziv and Lempel 1977]. The Lempel-Ziv algorithm parses the source ensemble into a collection of segments of gradually increasing length. At each encoding step, the longest prefix of the remaining source ensemble which matches an existing table entry (alpha) is parsed off, along with the character (c) following this prefix in the ensemble. The new source message, alpha c, is added to the code table. The new table entry is coded as (i,c) where i is the codeword for the existing table entry and c is the appended character. For example, the ensemble 010100010 is parsed into { 0, 1, 01, 00, 010 } and is coded as { (0,0), (0,1), (1,1), (1,0), (3,0) }. The table built for the message ensemble EXAMPLE is shown in Figure 3.1. The coded ensemble has the form: { (0,a), (1,space), (0,b), (3,b), (0,space), (0,c), (6,c), (6,space), (0,d), (9,d), (10,space), (0,e), (12,e), (13,e), (5,f), (0,f), (16,f), (17,f), (0,g), (19,g), (20,g), (20) }. The string table is represented in a more efficient manner than in Figure 5.1; the string is represented by its prefix codeword followed by the extension character, so that the table entries have fixed length. The Lempel-Ziv strategy is simple, but greedy. It simply parses off the longest recognized string each time rather than searching for the best way to parse the ensemble.

TABLES 3.1(CORPUS DETAILS)

File Name	Discription size	**Elements**	**attributes**	Max Depth	Avg-Depth
Mondial	1 MB	22423	47423	5	3.59274
Swissport	109 MB	2977031	2189859	5	3.55671
Treebank	82 MB	2437666	**1**	36	7.87279
Dblp	127 MB	3332130	404276	6	2.90228
Yahoo	24KB	342	**0**	5	3.76608
UWN	2 MB	66729	**6**	5	3.95243

Ebay	34.7 KB	156	0	5	3.7564
Ubid	19.8 KB	342	0	5	3.7661
321gnoe	24.9 KB	311	0	5	3.7653

OUR CORPUS

1.SwissProt

SWISS-PROT is a curated protein sequence database which strives to provide a high level of annotations (such as the description of the function of a protein, its domains structure, post-translational modifications, variants, etc.), a minimal level of redundancy and high level of integration with other databases.

2.Treebank (partially encrypted)

English sentences, tagged with parts of speech. The text nodes have been encrypted because they are copywritten text from the Wall Street Journal. Never the less, the deep recursive structure of this data makes it an interesting case for experiments.

3.Mondial

World geographic database integrated from the CIA World Factbook, the International Atlas, and the TERRA database among other sources.

4.DBLP Computer Science Bibliography

The DBLP server provides bibliographic information on major computer science journals and proceedings. DBLP stands for Digital Bibliography Library Projects. Shakespearea collection of the plays of William Shakespeare in XML [5].The first four datasets given above are regarded as data-centric as the XML documents have a very regular structure, whereas the last one is regarded as document-centric as the XML documents have a less regular structure:

IV. EXPERIMENTAL EVALUATION

Table 4.1 Auction Dataset

Auction Data KB		CS	CR1	CR2	T1	T2	T3	TA
Yhoo	24.8	21.7	7	12.5	0.708	0.733	0.725	0.722
Ebay	34.7	41.6	9.590778	-19.8847	0.843	0.84	0.84	0.841
Ubid	19.8	13.5	5.454545	31.81818	0.729	0.717	0.715	0.7203
321gnoe	24.9	22.5	7.228916	9.638554	0.729	0.722	0.72	0.7237

Table 4.2: University Courses

University Courses		CS	CR1	CR2	T1	T2	T3	TA
Reed	227	164	5.779736	27.7533	1.143	2.98	1.228	1.7837
Uwm	2.22	1.3	4.684685	41.44144	2.334	2.533	2.3	2.389
WSU	1.57	1.11	5.656051	29.29936	5.901	3.612	3.538	4.3503

Table 4.3: Transaction Processing Performance Council (TPC)

TPC-H		CS	CR1	CR2	T1	T2	T3	TA
Nation	4.47	3.87	6.926174	13.42282	0.0.0.27	0.0.0.17	0.0.0.46	0.0.0.30
customer	502	425	6.772908	15.33865	0.0.1.174	0.0.0.985	0.0.1.280	0.0.1.146
Supplier	28.5	24.6	6.905263	13.68421	0.0.0.95	0.0.0.64	0.0.0.98	0.0.0.85
Partsupp	2.13	1.82	6.835681	14.55399	0.0.4.947	0.0.4.69	0.0.3.965	0.0.4.53
Order	5.12	3.44	5.375	32.8125	0.0.12.716	0.0.7.191	0.0.8.466	0.0.9.45
Part	603	490	6.500829	18.73964	0.0.1.921	0.0.1.453	0.0.1.611	0.0.1.66

Table 6.5:DBLP datasets

Dataset		S	CR1	CR2	T1	T2	T3	TA
dblp1(KB)	1003	438	3.493519	56.33101	0.0.0.30	0.0.0.24	0.0.20	0.0.0.24
Dblp(MB)	131	114	6.961832	12.9771	0.0.10.656	0.0.9.166	0.0.7.135	0.0.8.98
Publication(KB)	5.06	4.99	7.889328	1.383399	0.0.0.69	0.0.0.52	0.0.0.18	0.0.0.46
Authers(KB)	67.3	64.1	7.619614	4.754829	0.0.0.166	0.0.0.154	0.0.0.142	0.0.0.154

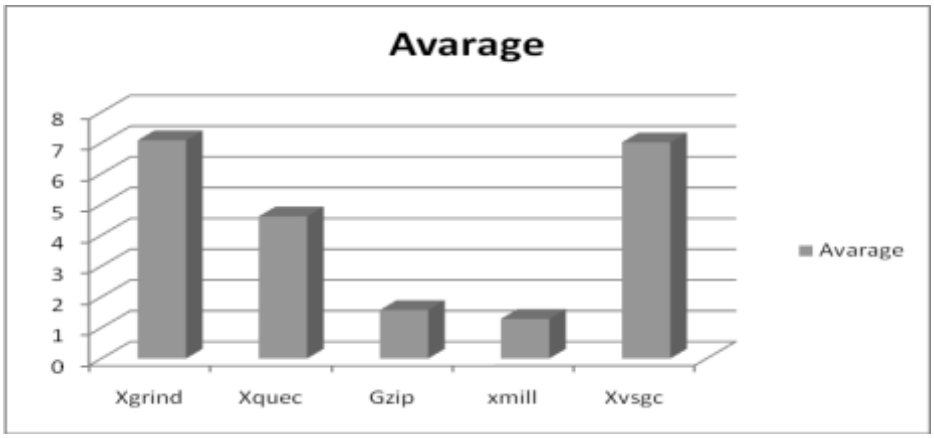

Figure:4.1 CR1 of XVSGC WITH OTHERS

V. CONCLUSION

To make the XML compressor adaptive and efficient, it is necessary to make the algorithm adaptive and optimal. We started this work with an objective to initiate an enquiry in existing XML compression techniques. The proposed work intent to achieve 'optimal', 'adaptive' and 'flexible', efficient ,'fast' query processing , in XML compression and efficient query evaluation by dividing it into following sub goals

-Developing a compression algorithm for XML database.

-Developing an algorithm and query processor, this will help for adaptive /efficient query evaluation and partial decompression. From the results, using the proposed technique of compression of XML databases and query evaluation, it can be concluded that: Compression ration achieved is better than XGrind and comparable with XCQ. Both XMill and XCQ consistently achieve a better compression ratio than gzip.

Compression time is lesser than a Quriable XCQ and XGrind but slightly more compression time than XMill. Decompression time by proposed approach is better than decompression time for XCQ and XGrind and comparable with XMill. A good Query response time and Query speedup factor (QRT and QSF) is achieved with respect to XGrind & XQuec. Finally it can be concluded that using the proposed technique of compression of XML databases and query evaluation over it ,will better improve the performance of XML database systems. During the current research work, several areas have been identified that could be further investigated. The major area of immediate research is an indexing scheme, to aid querying compressed XML databases and development of natural advance compressor. It is anticipated that this will enable further research.

REFERENCES

[1] James Cheng and Wilfred Ng *"XQzip: Querying Compressed XML Using Structural Indexing" E. Bertino et al. (Eds.): EDBT 2004, LNCS 2992, pp. 219–236, 2004. _c Springer-Verlag Berlin Heidelberg 2004*

[2] Andrei Arion, Angela Bonifati, Ioana Manolescu, Andrea Pugliese "*XQueC: A Query-Conscious Compressed XML Database" ACM Journal Name, Vol. , No. , 20, Pages 1–31.*

[3] JunKi Min MyungJae Park ChinWan Chung *"XPRESS: A Queriable Compression for XML Data" SIGMOD 2003, June 912, 2003, San Diego, CA. Copyright 2003 ACM 158113634X/ 03/06*

[4] Mustafa Atay, Yezhou Sun, Dapeng Liu, Shiyong Lu, Farshad Fotouhi *"MAPPING XML DATA TO RELATIONAL DATA: A DOM-BASED APPROACH" Department of Computer Science Wayne State University, Detroit, MI 48202*

[5] Sherif Sakr "XML compression techniques: *A survey and comparison " National ICT Australia (NICTA), 223 Anzac Parade, NSW 2052, Sydney, Australia Journal of Computer and System Sciences 75 (2009) 303–322*

[6] Pankaj M. Tolani Jayant R. Haritsa *XGRIND: A Query-friendly XML Compressor" Proceedings of the 18th International Conference on Data Engineering (ICDE.02) 1063-6382/02 $17.00 © 2002 IEEE*

[7] Wilfred Ng · Wai-Yeung Lam Peter T. Wood · Mark *Levene "XCQ: A queriable XML compression system" Knowl Inf Syst (2006) DOI 10.1007/s10115-006-0012-z*

[8] Wilfred Ng Lam Wai Yeung James Cheng *"Comparative Analysis of XML Compression Technologies" Department of Computer Science The Hong Kong University of Science and Technology Hong Kong*

[9] Weimin *Li "XCOMP: AN XML COMPRESSION TOOL" A thesis presented to the University of Waterloo. Waterloo, Ontario, Canada, 2003*

[10] Michael Ley *"DBLP — Some Lessons Learned" VLDB '09, August 2428, 2009, Lyon, France Copyright 2009 VLDB Endowment, ACM 0000000000000/ 00/00*.

[11] AlHamadani, Baydaa *"Retrieving Information from Compressed XML Documents According to Vague Queries" July, 2011 University of Huddersfield Repository http://eprints.hud.ac.uk/*

[12] Vojtech Toman *"Compression of XML Data" Master Thesis Prague, March 20, 2003 Faculty of Mathematics and Physics Charles University*

[13] Smitha S. Nair XML Compression Techniques*: A Survey Department of Computer Science, University of Iowa ,Iowa City, Iowa, USA*

[14] Wai Yeung, Lam Wilfred Ng, Peter T. Wood Mark Levene *XCQ: XML Compression and Querying System Hong Kong University of Science and Technology, Birkbeck College, University of London*

[15] Andrei Arion1, Angela Bonifati2, Gianni Costa2, Sandra D'Aguanno1,etel *"Efficient Query Evaluation over Compressed XML Data" E. Bertino et al. (Eds.): EDBT 2004, LNCS 2992, pp. 200–218, 2004. _c Springer-Verlag Berlin Heidelberg 2004*

[16] Gregory Leighton and Denilson Barbosa "Optimizing XML Compression (Extended Version)" arXiv:0905.4761v1 [cs.DB] 28 May 2009

[17] Michael Ley "DBLP XML Requests- Appendix to the paper "DBLP — Some Lessons Learned" (June 17, 2009) Michael Ley Universit ¨ at Trier, Informatik D–54286 Trier Germany

[18] Mustafa Atay, Yezhou Sun, Dapeng Liu, Shiyong Lu, Farshad Fotouhi "MAPPING XML DATA TO RELATIONAL DATA: A DOM-BASED APPROACH" arXiv : 1010 .1746 V1[CS.DB] 8 oct 2010 , http://arxiv.org/abs/1010.1746v1

[19] Debra A. Lelewer and Daniel S. Hirschberg *" Data Compression"*

[20] Huffman, D. A. 1952. *A Method for the Construction of Minimum-Redundancy Codes. Proc. IRE 40, 9 (Sept.), 1098-1101.*

[21] Abramson, N. 1963. *Information Theory and Coding. McGraw-Hill, New York.*

[22] Apostolico, A. and Fraenkel, A. S. 1985. Robust Transmission of Unbounded Strings Using Fibonacci Representations. Tech. Rep. CS85-14, Dept. of Appl. Math., The Weizmann Institute of Science, Rehovot, Sept.

[23] Abramson, N. 1963. *Information Theory and Coding. McGraw-Hill, New York.*

[24] Pasco, R. 1976. *Source Coding Algorithms for Fast Data Compression. Ph. D. dissertation, Dept. of Electrical Engineering, Stanford Univ., Stanford, Calif.*

[25] Rissanen, J. J. 1976. *Generalized Kraft Inequality and Arithmetic Coding. IBM J. Res. Dev. 20 (May), 198-203.*

[26] Rubin, F. 1979. *Arithmetic Stream Coding Using Fixed Precision Registers. IEEE Trans. Inform. Theory 25, 6 (Nov.), 672-675.*

[27] Ryabko, B. Y. 1987. *A Locally Adaptive Data Compression Scheme. Commun. ACM 16, 2 (Sept.), 792*

[28] Elias, P. 1987. *Interval and Recency Rank Source Coding: Two On-Line Adaptive Variable-Length Schemes. IEEE Trans. Inform. Theory 33, 1 (Jan.), 3-10.*

[29] Horspool, R. N. and Cormack, G. V. 1987. *A Locally Adaptive Data Compression Scheme. Commun. ACM 16, 2 (Sept.), 792-794.*

[30] Llewellyn, J. A. 1987. *Data Compression for a Source with Markov Characteristics. Computer J. 30, 2, 149-156.*

[31] Tanaka, H. 1987. *Data Structure of Huffman Codes and Its Application to Efficient Encoding and Decoding. IEEE Trans. Inform. Theory 33, 1 (Jan.), 154-156.*

[32] Vitter, J. S. 1987. *Design and Analysis of Dynamic Huffman Codes. J. ACM 34, 4 (Oct.), 825-845.*

[33] Witten, I. H., Neal, R. M., and Cleary, J. G. 1987. *Arithmetic Coding for Data Compression. Commun. ACM 30, 6 (June), 520-540.*

[34] Faller, N. 1973. *An Adaptive System for Data Compression. Record of the 7th Asilomar Conf. on Circuits, Systems and Computers (Pacific Grove, Ca., Nov.), 593-597.*

[35] Gallager, R. G. 1978. *Variations on a Theme by Huffman. IEEE Trans. Inform. Theory 24, 6 (Nov.), 668-674.*

[36] Knuth, D. E. 1985. *Dynamic Huffman Coding. J. Algorithms 6, 2 (June), 163-180.*

[38] Ziv, J., and Lempel, A. 1977. *A Universal Algorithm for Sequential Data Compression. IEEE Trans. Inform. Theory 23, 3 (May), 337-343.*