iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 6 **Issue:** XII **Month of publication:** December 2018

DOI:

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 6.887
Volume 6 Issue XII, Dec 2018- Available at www.ijraset.com

Efficient XML Retrieval Using Compression Techniques

Miss Deepali D. Rane[1], Mr. Gajanan P. Babhulkar[2]
[1,2]*Assistant Professor, Department of Information Technology, D. Y. Patil College of Engineering, Akurdi*

Abstract: ***Since XML is expressed as a text file, an obvious approach to compression is to use gzip style techniques. These methods, which leverage classic encoding algorithms, have a number of disadvantages.***
The new XML compression algorithm that is designed and will be implemented allows supporting ADXPI(Absolute Document XPath Indexing) indexing and score sharing function by a Top-Down Scheme approach which will work more efficiently than existing system.
The ADXPI Indexer will get all of structure index to construct the leaf-node index which is stored in the MySql database which can be retrieved by using Vector Space Model of MySQL Full Text. This system will help to improve efficiency and effectiveness of XML Retrieval.
This system reduces the size than current most popular compression technique GPX.
Keywords: XML, Compression, Indexing, database, vector space model

I. INTRODUCTION

Researches on XML data compression stress the reduction of XML data size. Each method has its own techniques. XML data compression can be divided into three types: 1) data compression 2) tag compression and 3) data and tag compression. XMill[1] is a technique which compresses both data and tag in order to reduce the size by starting with separating the tag, which is composed of elements and attributes, from the data, which is a character.
XGrind[3]is a technique which compressed data and tag but the user can still search for data after the compression
XPRESS uses the technique in compressing both the data and the tag. Its advantages are the same as XGrind: it can search for the data after the compression. Nevertheless, XPRESS does not use DTD.

A. Drawbacks

1) Other current systems will compress only XML data that has DTD structure so some data set that does not have DTD will result in having the user waste time in creating DTD for XML data set that they wanted to compress.
2) Can't Searched through compressed data.
3) Compression ratio is comparatively less.

II. PROPOSED SYSTEM

Our approach for retrieval of large-scale XML collection is to improve both efficiency and effectiveness of XML Retrieval.
We propose new XML compression algorithm that allows supporting Absolute Document XPath Indexing by a Top-Down Scheme approach.
It has been discovered that these steps reduce the size of the data compare to GPX, and reduce the length of Score Sharing processing time when compared to before the compression. In terms of processing time, our system required an average of one second per topic on INEX-IEEE and an average of ten seconds per topic on INEX-Wikipedia better than GPX system.

A. Problem Formulation

The proposed system has been designed and implemented to improve the efficiency and effectiveness of XML retrieval in large scale collection using compression techniques.

B. Design Goals

1) To improve the efficiency and effectiveness of XML retrieval.

©IJRASET: All Rights are Reserved

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 6.887
Volume 6 Issue XII, Dec 2018- Available at www.ijraset.com

C. System Architecture

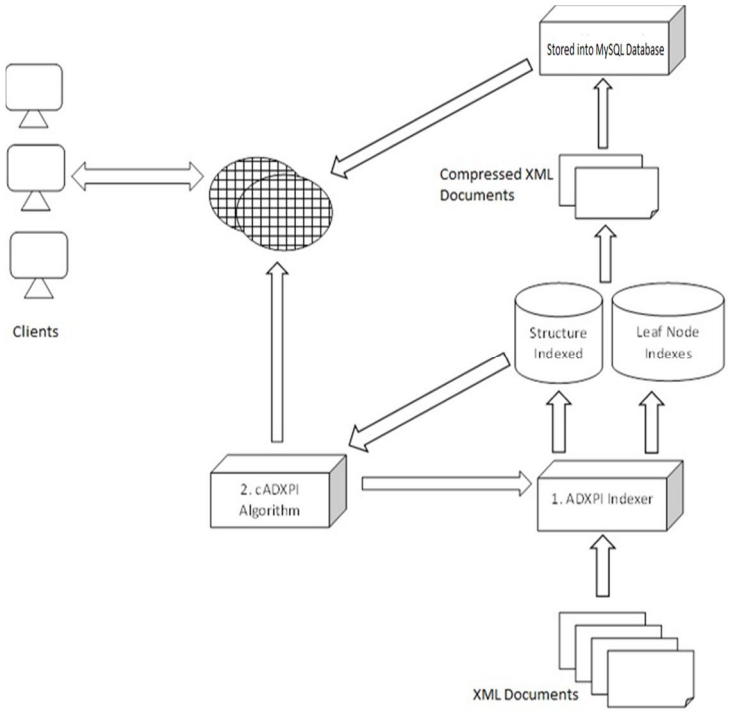


Figure.1 System Architecture

In ADXPI(Absolute Document XPath Indexing) Indexing, when new documents are entered, the Indexer parses and analyzes the tag and content data to build the list of leaf-nodes. To control overlap and reduce the cost of Joined on DBMS, we used the Absolute Document XPath Indexing (ADXPI) scheme to transform each leaf element level into a document level.

The representation of the ADXPI is more problematic, because each unique XPath is repeated in the inverted list for each term in the same node, and the XPath repeated in many files. We find out the way to encoded tags and the compression algorithm like XMill might be effective, but we considered this again to be unnecessary, particularly given the processing overheads. cADXPI Compressor analyzes the tag and counter to build the structure index store in MySQL database. We have adopted the simple compression scheme using Dictionary Mapping and easy to reconstruct the original XPath.

The cADXPI Indexer will get all of structure index to construct the leaf-node index store in MySQL database. The Leaf-Only indexing is closest to traditional information retrieval and can be scored as ordinary plain text document then we calculate the leaf element score of its context using Vector Space Model of MySQL Full Text Search.

D. Framework & Methodology

Absolute Document XPath Indexing to control overlap and reduce the cost of Joined on DBMS, we used the Absolute Document XPath Indexing (ADXPI) scheme to transform each leaf element level into a document level.DXPI is more problematic, because each unique XPath is repeated in the inverted list for each term in the same node, and the XPath repeated in many files. We find out the way to encoded tags and the compression algorithm like XMill might be effective, but we considered this again to be unnecessary, particularly given the processing overheads. We have adopted the following simple compression scheme using Dictionary Mapping and easy to reconstruct the original XPath. Finally, the database schema consists of the following tables and adding FTS index to LeafNode.

Compression Algorithm

1) Fetch all leaf node entries from the collection list.
2) For each list, create data structure to store tag name and frequency, we call Dictionary<tag,freq> data type.
3) Split all tag and counter from the leaf and add to Dictionary<tag,freq>, for instance, the leaf node is: /article[1]/body[1]/section[1]/p[1].
a) We can split them as follows;

©IJRASET: All Rights are Reserved

b) 1st tag is "article[1]", frequency is 1.
c) 2nd tag is "body[1]", frequency is 1.
d) 3rd tag is "section[1]", frequency is 1.
e) and "p[1]", frequency is 1.
4) For each tag has to check in Dictionary<tag,freq> list as follows;
a) If Dictionary<tag,freq> has contain tag then freq is accumulate by freq = freq + 1
b) Otherwise add new tag and 1 to Dictionary<tag,freq> list.
5) When already processed all of a list from 2 then create the Final Dictionary<tag,map> list by sorting freq from Dictionary<tag,freq>list. The map is a sequence of tag in Final list.
6) Return Final Dictionary<tag,map> list to store in DB.

V. RESULT ANALYSIS

In Previous system GPX system is used but the GPX search engine is using a relational database implement an inverted list data structure. It is a compromise solution provides the convenience of a DBMS at the cost of somewhat reduced performance, which may otherwise be possible.

For example, the XPath as following:`/article[1]/bdy[1]/sec[5]/p[3]

This could be represented by two expressions, a Tag-set and an Index-set as below;

Tag-set: /article/bdy/sec/p

Index-Set: 1/1/5/3

The original XPath can be reconstructed from the tag-set and the index-set. The GPX assigns to each tag set and each index-set a hash code and create auxiliary database tables mapping the hash codes to the corresponding tag-set and index-set entries. These hash tables are small enough to be held in memory and so decoding is efficient. The GPX takes 15 seconds to load all table data and takes an average of 7.2 seconds per topic. Sometimes, it takes longer than 30 seconds, depending on the type of query on a 3GHz PC with 2 GB RAM. Unfortunately, This method has not been focused on the efficiency.

It has been discovered that these steps reduces the size of the data down by 78.77 % compare to GPX, and reduce the length of score sharing processing time down by 44.18% when compared to before the compression.

VI. CONCLUSION

Proposed system, we have implemented new xml compression algorithm that allows supporting ADXPI indexing with large scale xml documents. It has been discovered that these steps reduces the size of data compared to current xml compression techniques as well retrieving and searching can be done efficiently.

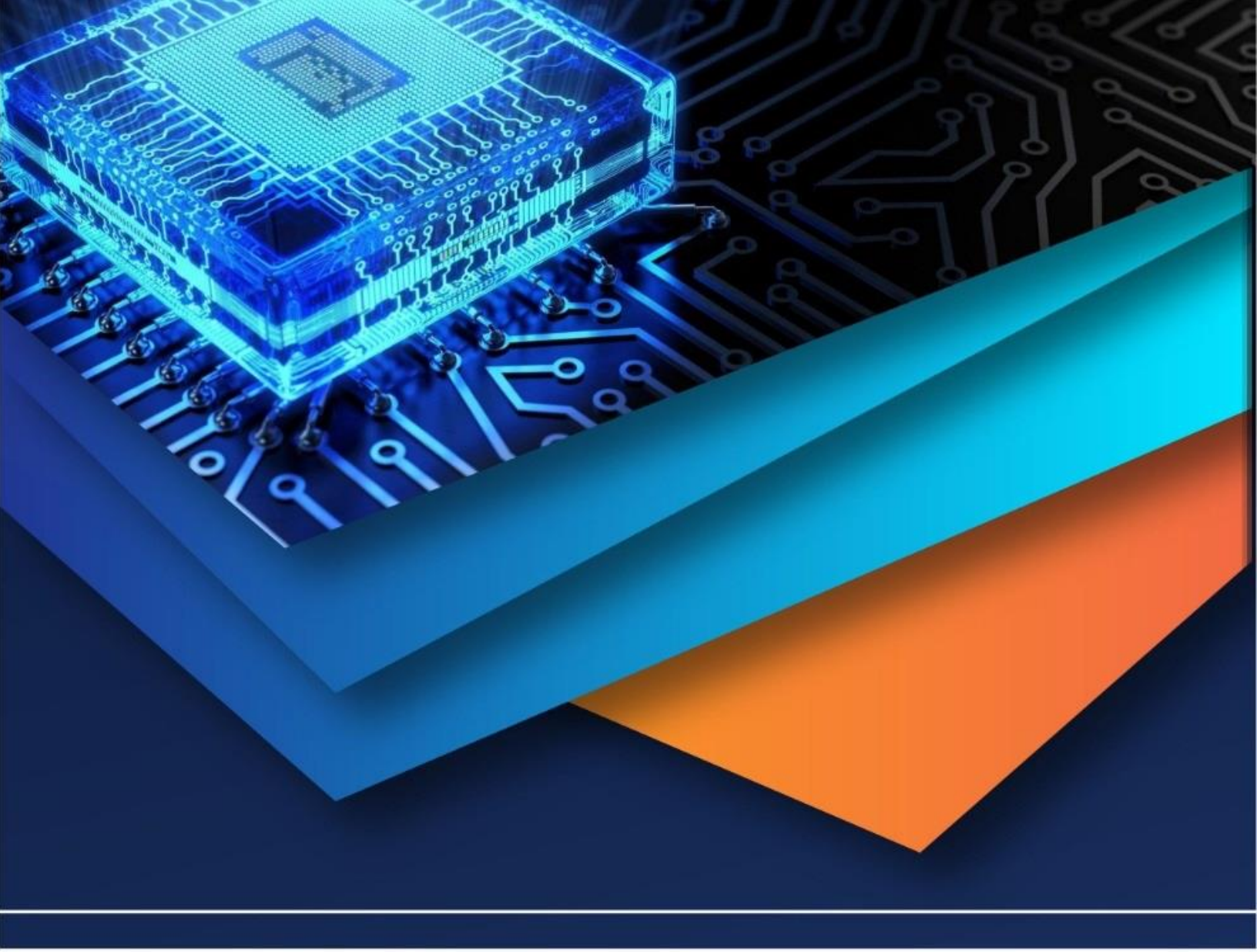
We propose new XML compression algorithm that allows supporting ADXPI indexing and score sharing function by a Top-Down Scheme approach, and a comprehensive description of our system, with performance experiments on large-scale corpora on INEX collections.

It has been discovered that these steps reduces the size of the data down by 78.77 % compare to GPX, and reduce the length of score sharing processing time down by 44.18% when compared to before the compression.

REFERENCES

[1] Liefke H. and Suciu D., "XMill: an Efficient Compressor for XML Data.," In Proceeding of the 2000 ACM SIGMOD International Conference on Management of Data, pages 153-164, May 2000.
[2] Gailly J. L. and Adler M., "gzip: The compressor data," Available at http://www.gzip.org/
[3] Tolani P. M. and Haritsa J. R., "XGRIND: A Query-friendly XML Compressor.," In Proceedings of 18th International Conference on Databases Engineering, February 2002.
[4] Min J.-K., Park M.-J., and C Chung.-W., "XPRESS: A Queriable Compression for XML Data.," In Proceeding of the 2003 ACM SIGMOD International Conference on Management of Data, pages 122-133, June 9-12, 2003.
[5] Maireang K. and Pleurmpitiwiriyavach C., "XPACK: A Grammar-based XML Document Compression," In Proceeding of NCSEC2003 the7th National Computer Science and Engineering Conference, October 28-30, 2003.
[6] Wichaiwong T. and Jaruskulchai C., "Improve XML Web Services' Performance By Compressing XML Schema tag," The 4th International Technical Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, Chiang Rai, Thailand, May 9-12, 2007.
[7] Geva, S. 2005. GPX - Gardens Point XML Information Retrieval INEX 2004. In: Fuhr, N., Lalmas, M., Malik, S., Szlavik Z. (eds.): Advances in XML Information Retrieval: Third International Workshop of the Initiative for the Evaluation of XML, Springer, Lecture Notes in Computer Science LNCS, pp. 211-223.

©IJRASET: All Rights are Reserved



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)