NEW APPROACHES FOR XML DATA COMPRESSION

Márlon A. C. Teixeira[1], Rodrigo S. Miani[2], Gean D. Breda[2],
Bruno B. Zarpelão[2] and Leonardo S. Mendes[2]
[1]Acre Federal Institute of Education, Science and Technology (IFAC), Rio Branco, Brazil
[2]School of Electrical and Computer Engineering, University of Campinas (UNICAMP), Campinas, Brazil

Keywords: Data Compression, Data Exchange, Extensible Markup Language (XML), Interoperability.

Abstract: Integration of information systems is essential to organizations. Therefore, it is necessary to make different technologies interoperate. Extensible Markup Language (XML) is often used for data exchange because it is self-descriptive and platform-independent. However, XML is a verbose language which may bring problems related to the size of documents. This work proposes two new approaches for XML data compression and compares our solutions with three algorithms: WAP Binary Extensible Markup Language (WBXML), Xmill and Efficient XML Interchange (EXI). The comparison is based on compression rate and compression time for files with different sizes.

1 INTRODUCTION

Extensible Markup Language (XML) is a *de facto* standard to exchange data in heterogeneous environments. However, problems with the size of XML documents are common due to the redundancy contained in their repeated tags (Ng et al., 2006) (Augeri et al., 2007). A possible solution to reduce the size of XML documents is the application of data compression algorithms. Yet, XML-conscious compressors like WBXML (WAP Binary Extensible Markup Language) (XBXML, 1999), XMill (Liefke and Suciu, 2000) and EXI (Efficient XML Interchange) (EXI, 2011) bring about, as a result, compressed documents which are not in the XML format. This way, one of the great advantages of the XML language, which is the diversity of APIs (Application Programming Interfaces) that can be found for parsing and querying XML data, is annulled.

In the present work, we propose two approaches for the compression of XML documents. The first solution, entitled Schema-aware algorithm, is based on the elimination of redundant tags of the document structure. The final compression outcome is still a document in the XML format. So, it is possible to handle the compressed document by using the various XML parsing and querying solutions available. The second solution is a hybrid algorithm which applies the general purpose compressor GZIP (Gailly and Addler, 2011) on the result of the compression achieved from the Schema aware algorithm.

To assess both propositions, we have carried out a comparative with the WBXML, Xmill and EXI algorithms taking in consideration two metrics: compression rate and compression time. These two metrics were collected during the compression of files of 8 different sizes.

The remainder of the paper is structured as follows. In section 2, the Schema aware algorithm is presented. In section 3, the Hybrid algorithm is presented. Section 4 brings the comparison between the two proposed solutions and the compressors WBXML, XMill and EXI. At last, section 5 frames the final considerations on this work.

2 SCHEMA-AWARE ALGORITHM

The Schema-aware algorithm consists of the elimination of redundant tags which are in the same level of the document, without neither compressing the data, nor the tags existent in the document. To demonstrate how this redundancy occurs, we shall use the following XML document:

```
<customers>
  <customer>
    <organization>Org1</organization>
    <type>Type1</type>
    <name>Name1</name>
  </customer>
  <customer>
    <organization>Org2</organization>
    <type>Type2</type>
    <name>Name2</name>
  </customer>
</customers>
```

The XML document can be represented as a tree, as in Figure 1. By looking at the tree structure, one realizes that the nodes of the same level in the tree have the same tags. For this reason, plenty of redundant information is sent through the network and stored in the destinations, as well. So, it is exactly on the elimination of the redundant tags of the same level that the Schema-aware algorithm has its base for reducing XML document size. By applying the algorithm in order to have the tag redundancy eliminated in each level of the tree, the nodes of the same level will become a single node. Therefore, the redundant tags are eliminated and the pieces of information inside the nodes are stored in this unique resulting node.

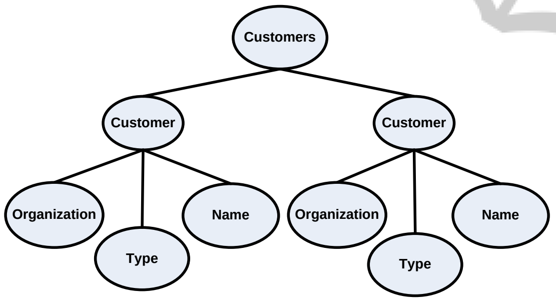

Figure 1: XML document shown as a tree.

A detail needs to be considered in the information storing process in the resulting node: the data location is of extreme relevance, since the file decompression is carried out based on the order of the elements. The Figure 2 tree stands for the result of the compression of the XML document in Figure 1.

The following XML document shows the result of the compression carried out by using the Schema Aware algorithm. Clearly, there was a remarkable reduction in the size of XML file, mainly in the use of tags.

```
<customers>
  <customer>
    <organization>[Org1, Org2]
                                </organization>
    <type>[Type1, Type2]</type>
    <name>[Name1, Name2]</name>
  </customer>
</customers>
```

Schema-aware algorithm, differently from the WBXML, XMill and EXI compressors, generates the same representation of the original XML file in another XML file of smaller size. This is an advantage, since self-descriptive characteristic of the XML language is maintained and the tools which handle XML can still be used.

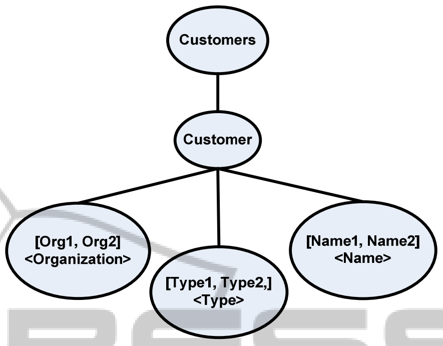

Figure 2: Resulting compressed document.

3 HYBRID ALGORITHM

The hybrid algorithm is a variation of the Schema-aware algorithm. This variation consists of the application of a generic purpose data compressor on the result of the Schema-aware algorithm compression. The data compressor chosen was the GZip, which is broadly used and spread out in the proper literature (Snyder, 2010). Figure 3 presents the Hybrid algorithm diagram.

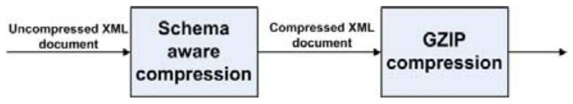

Figure 3: Hybrid algorithm diagram.

The GZip compressor has been chosen because it is one of the most popular data compressors (in the market). It is widely used even in specialized XML file compressors, for instance, the XMill (Lifke and Suciu, 2000). The objective of applying the GZip is the compression of information and tags, not done in the first approach, where only the redundancy of existent tags in the same level of the XML tree was removed.

With this change, it is expected an improvement in the compression rate, followed by a possible degradation in the performance of the method regarding to the compression time. Beyond the impact on the performance resulted from this new approach, other alterations arise due to this change. The resulting file of the compression is no longer a

XML document. In other words, for the information to be extracted, it is necessary the utilization of a GZip decompressor. Therefore, there is a loss of legibility leading to a bigger impact on the interoperability. No less important is the impossibility of using the available tools for handling the XML language.

4 RESULTS

In this section, we present the evaluation of the following XML compression algorithms: Schema-aware, Hybrid, EXI, WBXML and XMill. The scenario of tests was built in accordance with the tests proposed by Cokus and Winkowski (2002). The authors used a range of XML document size from 6,010 bytes (approximately 6 Kbytes) to 11,421,822 bytes (approximately 11 Mbytes).

In the same way, five sets of files were created in our work. Each set contains eight files with the following sizes: 1 – 268 Kbytes; 2 – 538 Kbytes; 3 – 1073 Kbytes; 4 – 2238 Kbytes; 5 – 4482 Kbytes; 6 – 8957 Kbytes; 7 – 17922 Kbytes and 8 – 35850 Kbytes.

In these file sets, it was applied the EXI, WBXML, XMill, Schema-aware and Hybrid algorithms. Also, the following criteria were assessed: compression rate and compression time.

4.1 Compression Rate

In this section, we will evaluate the compression rate achieved by the methods. The five file sets were submitted to the compression methods and, for each file, the result of the compression rate was analyzed. These results are exposed in the Figures 4, 5 and 6.

The analysis of the graphs in the Figures 4, 5 and 6 shows that the EXI method has reached the best compression rate for all the sizes of the analyzed files. By comparing the rates obtained by the XMill method and Hybrid algorithm, one can notice very similar outcomes. The WBXML method has reached the worst rates in all the tests.

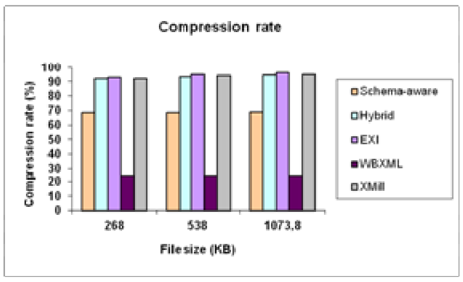

Figure 4: Compression rate for the 268, 538 and 1073.8 KB files.

The Schema-aware algorithm has reached a reasonable compression rate, despite having lower results than the other algorithms, except the WBXML's.

Analyzing the results achieved by the WBXML method and the Schema-aware algorithm, one can realize that the Schema-aware algorithm reaches roughly rates of up to 68.81% and the WBXML compressor reaches rates of up to 24.26%, which means a meaningful difference.

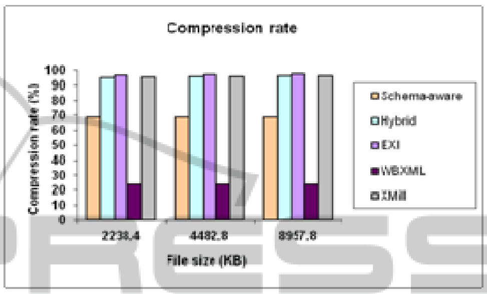

Figure 5: Compression rate for the 2238.4, 4482.8 and 8957.8 KB files.

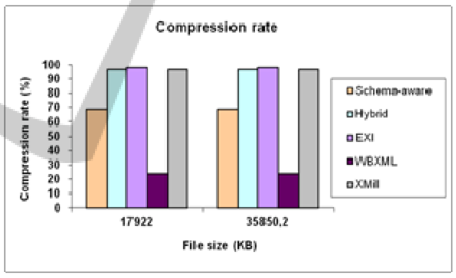

Figure 6: Compression rate for the 17922 and 35850.2 KB files.

4.2 Compression Time

Another point analyzed was the compression time. The compression algorithms were executed on a computer with a processor at 1.33 GHz and 1 GB of system RAM. The results are presented in Figures 7, 8 and 9.

According to Figures 7, 8 and 9, one can realize that the XMill compressor has presented the best performance for all the files and in all trials. One factor that may explain this result is the algorithm implementation, based on C++. The other compression algorithms were implemented using Java language. C and C++ compilers output platform-specific native code, while Java compilers output machine-neutral bytecode. Therefore, C++ programs usually have better performance than Java programs.

According to the results, we can also notice that the WBXML has obtained better performance than the Schema-aware and Hybrid algorithms for files which vary between 268 and 2238.4 Kbytes and worse performance for files which vary between 17922 and 35850.2 Kbytes. The performance difference is of 35% when compared to the Schema-aware algorithm and of 52% when compared to the Hybrid algorithm.

The EXI compressor has shown the worst performance in all file sizes and all trials carried out. The biggest difference of performance, compared to XMill, is of approximately 96% and regarding to WBXML it goes around 88%.

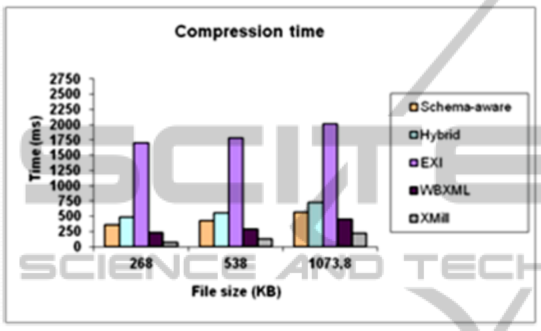

Figure 7: Compression time for the 268, 538 and 1073.8 KB files.

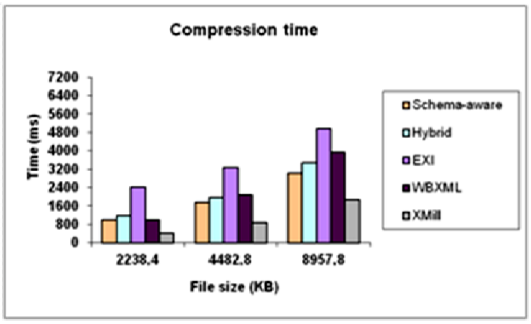

Figure 8: Compression time for the 2238.4, 4482.8 and 8957.8 KB files.

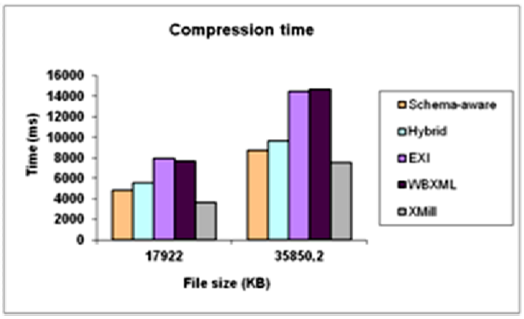

Figure 9: Compression time for the 17922 e 35850.2 KB files.

5 CONCLUSIONS

This paper proposed two algorithms for XML documents compression: Schema-aware algorithm and Hybrid algorithm. Tests were designed and applied to compare the two proposed approaches with the following XML conscious compressors: EXI, WBXML and XMill. The items compression time and compression rate were evaluated.

No method was good enough in all requirements. The method which reached the best compression rate was the EXI. On the other hand, it is the slowest one compared to the other two methods that most get close to its compression rate, namely the XMill and the Hybrid algorithm. Both the XMill and Hybrid algorithm have reached compression rates very much alike and also better performances compared to the EXI. Finally, a remarkable characteristic of Schema-aware approach is the preservation of interoperability. The Schema-aware algorithm outputs an XML document after the compression, while other investigated methods output new formats.

REFERENCES

Augeri, C. J., Mullins, B. E., Baird III, L. C., Bulutoglu, D. A., Baldwin, R. O., 2007. An Analysis of XML Compression Efficiency. *In 2007 Workshop on Experimental Computer Science (ExpCS)*. ACM New York.

Cokus, M., Winkowski, D., 2002. “XML sizing and compression study for military wireless data”. In *XML Conference and Exposition*.

EXI - Efficient XML Interchange, 2011. W3C Recomendation, http://www.w3.org/TR/exi/.

Gailly, J. and Adler, M., 2011. GZIP. http://www.gzip.org/.

Huffman, D. A., 1952. A Method for the Construction of Minimum-Redundancy Codes. *In Proceedings of the I.R.E*, p. 1098-1102.

Liefke, H. and Suciu, D., 2000. XMill: An efficient compressor for XML data. *In Proceedings of the ACM SIGMOD International Conference on Management of Data*, p. 153–164.

Ng, W., Lam, W., Cheng, J., 2006. Comparative Analysis of XML Compression Technologies, *World Wide Web: Internet and Web Information Systems*, v. 9, p. 5-33.

Open Mobile Alliance, 2011, http://www.openmobilealliance.org/.

Snyder, S. L., 2010. *Efficient Xml Interchange (EXI) Compression and Performance Benefits: Development, Implementation And Evaluation*. PhD thesis, Naval Postgraduate School Monterey, California.

Winzip, 2011. http://www.winzip.com/.
WBXML - WAP Binary XML Content Format, 1999. W3C NOTE, http://www.w3.org/TR/wbxml/.