Effective Compression Technique by Using Adaptive Huffman Coding Algorithm for Xml Database

Ms. Rashmi N. Gadbail¹Prof.V.S.Gulhane²

Sipna’s college of Engg. And Technology,Amravati.
Corresponding Adresses
rashmi_gadbail@rediff.com, v_gulhane@rediffmail.com

Abstract: The Extensible Markup Language (XML) is one of the most important formats for data interchange on the Internet. XML documents are used for data exchange and to store large amount of data over the web. These documents are extremely verbose and require specific compression for efficient transformation. In this proposed work we are enhancing the existing compressors which uses Adaptive Huffman coding. It is based on the principle of extracting data from the document, and grouping it based on semantics. The document is encoded as a sequence of integers, while the data grouping is based on XML tags/attributes/comments. The main disadvantage of using XML documents is their large sizes caused by highly repetitive (sub) structures of those documents and often long tag and attribute names. Therefore, a need to compress XML, both efficiently and conveniently to use. The re-organized data is now compressed by adaptive Huffman coding. The special feature of adaptive Huffman coding algorithm is that, it has extremely accurate compression as well as it eliminates the repetition of dictionary based words in xml database. Using Adaptive Huffman algorithm, we derived probabilities which dynamically changed with the incoming data, through Binary tree construction.

Keywords: Compression, decompression, Efficient XML compression and decompression, Adaptive Huffman coding.

1. Introduction

The Extensible Markup Language (XML) is one of the most important formats for data interchange on the Internet. XML documents are used for data exchange and to store large amount of data over the web. These documents are extremely verbose and require specific compression for efficient transformation. In this proposed work we are enhancing the existing compressors which uses Adaptive Huffman coding. It is based on the principle of extracting data from the document, and grouping it based on semantics[1]. The document is encoded as a sequence of integers, while the data grouping is based on XML tags/attributes/comments. The main disadvantage of using XML documents is their large sizes caused by highly repetitive (sub) structures of those documents and often long tag and attribute names. Therefore, a need to compress XML, both efficiently and conveniently to use. The design goal of Effective compression of XML database by using Adaptive Huffman coding is to provide extremely efficient and highly accurate compression of XML documents while supporting "online" usage. In this context, "online" usage means: (a) only one pass through the document is required to compress it, (b) compressed data is sent to the output stream incrementally as the document is read, and (c) decompression can begin as soon as compressed data is available to the decompressor. Thus transmission of a document over a heterogeneous systems can begin as soon as the compressor produces its first output, and, consequently, the decompress or can start decompression shortly thereafter, resulting in a compression scheme that is well suited for transmission of XML documents over a wide-area network.

2. Related work

Various sophisticated algorithms have been proposed for lossless text compression. A very promising development in the field of lossless data compression is the Burrows-Wheeler Compression Algorithm (BWCA), introduced in 1994 by Michael Burrows and David Wheeler. The algorithm received considerable attention since of its Lempel-Ziv like execution speed and its compression performance close to state-of-the-art PPM algorithms. A preprocessing method is performed on the source text before applying an existing compression algorithm. The transformation is designed to make it easier to compress the source file. The star encoding is generally used for this type of preprocessing transformation of the source text. Star-encoding works by creating a large dictionary of commonly used words expected in the input files. The dictionary must be prepared in advance, and must be known to the compressor and decompressor.

Several proposals and references there in make use of the observation that the pioneering work in this domain was XGRind which was based on static Huffman coding. XGRind was the first XML-conscious compression scheme to support querying without full decompression[7] Element and attribute names are encoded using a byte-based scheme, and character data is compressed using static Huffman coding. Use of the latter technique significantly slows down the compression process, since two passes over the original document are required (first to gather probability data for the
compression model, and a second time to perform the encoding according to the generated model. XPRESS also supports querying of compressed data and claims to achieve better compression than XGRID. [8] However, it uses a semi-adaptive form of arithmetic coding which also necessitates two passes over the original XML document.

3. Proposed work and objectives:

3.1 Compression Techniques for XML Database

- Lossless Compression

Lossless compression techniques provide exact recovery of the original data from their compressed version. Any information contained in an original cannot be lost during compression and reconstruction. These techniques are used widely in applications to save storage space and network bandwidth. Since an XML compressor needs to preserve all data content, only lossless compression techniques can be used.

3.1.1 Huffman coding

In computer science and information theory, Huffman coding is an entropy encoding algorithm used for lossless data compression. The term refers to the use of a variable-length code table for encoding a source symbol (such as a character in a file) where the variable-length code table has been derived in a particular way based on the estimated probability of occurrence for each possible value of the source symbol. It was developed by David A. Huffman "A Method for the Construction of Minimum-Redundancy Codes." Huffman coding uses a specific method for choosing the representation for each symbol, resulting in a prefix code (sometimes called "prefix-free codes"), that is, the bit string representing some particular symbol is never a prefix of the bit string representing any other symbol) that expresses the most common source symbols using shorter strings of bits than are used for less common source symbols.

In this proposed work we are enhancing the existing compressors which uses Adaptive Huffman coding. Adaptive Huffman coding for xml database compression

XML simplifies data exchange among heterogeneous computers, but it is notoriously verbose and has spawned the development of many XML-specific compressors and binary formats."[10]. We can present an XML test and a combined efficiency metric integrating compression ratio and execution speed. With the help of adaptive Huffman compression technique. The Adaptive Huffman compression is more efficient than static Huffman compression it is an important dimension for lossless data compression. In computer science and information theory Huffman coding is an entropy encoding algorithm used for lossless data compression. Basically, the Adaptive algorithm states that a received symbol’s node (current node) must be promoted (via the Update method) as the highest numbered node among nodes that are of equal weight. Simply swap the two nodes (i.e., the highest numbered node and the current node which must now become the new highest numbered node) and the tree maintains the Sibling property, as well as ensuring a correct tree. The parent of the current node then becomes the new current node, and the process continues until the root of the tree is reached.

3.1.2 Adaptive Huffman Compression Technique

The proposal here is that to design an efficient way of compressing XML documents by using adaptive Huffman coding. High compression ratio and speed is equally important. We also require the transformation of xml database to be fully reversible and decompress able so that the decompressed document is a mirror image of the original. Huffman coding requires prior knowledge of the probabilities of the source sequence. If this knowledge is not available, Huffman coding becomes a two pass procedure: the statistics are collected in the first pass and the source is encoded in the second pass. In the Adaptive Huffman coding procedure, neither transmitter nor receiver knows anything about the statistics of the source sequence at the start of transmission. Using Adaptive Huffman algorithm, we derived probabilities which dynamically changed with the incoming data, through Binary tree construction. Thus the Adaptive Huffman algorithm provides effective compression by just transmitting the node position in the tree without transmitting the entire code. Unlike static Huffman algorithm the statistics of the sensor data need not be known for encoding the data. That's why adaptive Huffman is extremely accurate with respect to the compression.
The design goal of Effective compression of XML database by using Adaptive Huffman coding is to provide extremely efficient and highly accurate compression of XML documents while supporting "online" usage. In this context, "online" usage means: (a) only one pass through the document is required to compress it, (b) compressed data is sent to the output stream incrementally as the document is read, and (c) decompression can begin as soon as compressed data is available to the decompressor. Thus transmission of a document over a heterogeneous systems can begin as soon as the compressor produces its first output, and, consequently, the decompress or can start decompression shortly there after, resulting in a compression scheme that is well suited for transmission of XML documents over a wide-area network. Compression Performance ,Compression Time (CT) and decompression time. The 

- **Objectives of proposed work are:**

To provide an Effective compression technique by using Adaptive Huffman coding algorithm

- **Adaptive Huffman coding for xml database compression**

XML simplifies data exchange among heterogeneous computers, but it is notoriously verbose and has spawned the development of many XML-specific compressors and binary formats. We can present an XML test and a combined efficiency metric integrating compression ratio and execution speed. With the help of adaptive Huffman compression technique. The Adaptive Huffman compression is more efficient than static Huffman compression it is an important dimension for lossless data compression. In computer science and information theory Huffman coding is an entropy encoding algorithm used for lossless data compression.

- **The Sibling Property of Adaptive Huffman Algorithm**

In dynamic coding, it is not enough to just have a symbol tree: the tree must be a “correct” Huffman tree. Thus, the tree is recomputed or updated to ensure a correct tree. Re computation of the tree is done dynamically, and the decoder also maintains the same tree that the encoder creates.

Adaptive Huffman Algorithm maintains a property to create a compact tree as much as possible. This is called the **Sibling Property**. Accordingly, the Sibling property ensures a Huffman tree in the fastest manner as possible; re computation of the tree always maintains the Sibling property.

The Sibling Property defines a binary tree to be a Huffman tree if and only if:

- all leaf nodes have non-negative weights (i.e., a leaf node can have a 0 weight), all internal nodes have exactly two children, and the weight of each parent node is the sum of its children’s weights; and
- the nodes are numbered in increasing order by non-decreasing weight so that siblings are assigned consecutive numbers or **rank**, and most importantly, their parent node must be higher in the numbering [Vitter 1987].

With the Sibling property, nodes are **promoted** up the tree when necessary to assign them a minimal number of bits; that is, if they are gaining weight, they are assigned shorter bit codes. Some nodes may go down according to the statistics of the source; if one node goes up, there is certainly one node which goes down because the two nodes are simply swapped in the tree positions. Node promotion involves constant swapping of nodes.

Basically, the Adaptive algorithm states that a received symbol’s node (**current node**) must be promoted (via the **Update** method) as the **highest** numbered node among nodes that are of **equal** weight. Simply swap the two nodes (i.e., the highest numbered node and the **current node** which must now become the **new** highest numbered node) and the tree maintains the Sibling property, as well as ensuring a correct tree. The parent of the current node then becomes the **new** current node, and the process continues until the root of the tree is reached.

- **Why Compress XML**
  - XML is verbose
  - XML documents has repetitive structures of data
  - Each non-empty element tag must end with a matching closing tag -- \(<\text{tag}>\text{data}</\text{tag}>\)
  - Ordering of tags is often repeated in a document (e.g. multiple records)
  - Tag names are often long

4. **Conclusion**

Adaptive Huffman coding encodes an alphabet with fixed codes. That allows us to directly search keywords in the compressed data[6]. Since the data volumes reduced, such compressing of xml data may be even faster than the original data. The resulting output of proposed work will be that xml document compresses with the help of adaptive Huffman algorithm and that compressed data will be decompressed as well and provide original xml document over a heterogeneous systems.
References


Author Biographies

First Author: Ms. Rashmi N. Gadbail received her Bachelor of computer technology in 2009 from Sipna’s COET Amravati (MS), Currently pursuing her M.E in Information Technology and working as a lecturer in IBSS C.O.E.T Ghatkhed, Amravati.

Second Author: Prof. V.S. Gulhane received his Bachelor of Computer Science in 1994 from Pusad Engg College (MS), Masters degree in computer science from RMIT Badnera, Amravati in 2005, Currently working as a Asst. Prof. in sipna’s COET Amravati.