International Journal of Research in Advent Technology, Vol.2, No.4, April 2014
E-ISSN: 2321-9637

Android Application for SMS Compression

Kirti Madhukar Battase, Neha Vijay Barve, Parinita Bajirao Kandekar, Lata Vitthal Sanap
IT Department, NDMVP KBTCOE, Nashik, Maharashtra, India
Email: kmbattase@gmail.com

Abstract - This paper provides the effective method for SMS compression by applying Huffman encoding technique. As we know every character in SMS is mostly encoded in 7 bit and the maximum capacity of one SMS is only 1120 bit. Previously used Arithmetic encoding technique requires additional memory space in compressed data to save arithmetic coding probability table for decompressing the data. Also it requires high precision and effective encoder-decoder to calculate and represent its respective code. Because of these reasons arithmetic coding technique is inefficient in practical use as it requires more memory space. So the new technology for SMS compression is developed by applying Huffman Encoding Technique. This mechanism is developed specially for mobile phones that run on Android Operating system. Both users require this App on their Mobile and also registered themselves for using App.

Index Terms — Compression, Huffman encoding technique, SHA-1, SMS.

1. INTRODUCTION

This paper uses Huffman coding technique to generate the frequency table. Frequency table consist of character count and its respective prefix code. The premise behind Huffman coding is that more frequently occurring symbols are coded using shorter code words. To accomplish this, variable length code words are assigned to symbols based on their frequency of occurrence.

1.1. Need

The need of this paper is to optimize the maximum character capacity of SMS body. Every character in SMS (Short Message Service) is mostly encoded in 7 bit encoding technique. The maximum capacity of one SMS is only 1120 bit (i.e. 160*7=1120 bits). This SMS capacity is very small. The previous technique used to compress the SMS called Arithmetic coding technique. It requires additional memory space in compressed data to save arithmetic coding probability table. Also it requires high precision and effective encoder-decoder to calculate and represent its code number. Because of these reasons arithmetic coding technique is inefficient in practical use. So the new technique is developed by using Huffman coding method.

2. LITERATURE REVIEW

Arithmetic coding is a compression mechanism that works by converting a data message to a real code number between 0 and 1. To compress a data, arithmetic coding requires a probability table of characters contained in the data. Probability table is a table containing probability range of existing characters in a data which is built based on the existing characters frequency in the data itself. The smaller the range to generate code number, the higher bit number is needed to represent the code number. Arithmetic coding only needs usual arithmetic operation to compress and decompress a message [1]. On arithmetic coding, compression encoding is not done to every single character but it is done straight to the message itself. Basically, arithmetic coding is able to compress a message with compressed-message result near to the message entropy value.

There are two main weaknesses of arithmetic coding. The first one is that it needs memory space in compressed-data to save its range probability table for decompressing the compressed-data. If arithmetic coding is used to compress small-size data, the existence of the range probability table in compressed-data will make the compressed-data bigger in size than the original data itself [1]. The second weakness of arithmetic coding is that it requires encoder-decoder with high-precision value. Encoder is a machine to compress a message; meanwhile decoder is a machine to decompress a compressed-message. If encoder or decoder doesn't have ability to calculate long mantissa with precision value, the decompressed messages of compressed-message can be different from the original message.

3. HUFFMAN CODING TECHNIQUE

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 is “A Method for the Construction of Minimum-Redundancy Codes" [9]. 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. Huffman was able to design the most efficient compression method of this type: no other mapping of individual source symbols to unique strings of bits will produce a smaller average output size when the actual symbol frequencies agree with those used to create the code [5]. The running time of Huffman's method is fairly efficient; it takes O (n /log n) operations to construct it. For a set of symbols with a uniform probability distribution and a number of members which is a power of two, Huffman coding is equivalent to simple binary block encoding, e.g., ASCII(American Standard Code for Information Interchange) coding. Huffman coding is such a widespread method for creating prefix codes that the term "Huffman code" is widely used as a synonym for "prefix code" even when such a code is not produced by Huffman's algorithm [7].

Although Huffman's original algorithm is optimal for a symbol-by-symbol coding (i.e. a stream of unrelated symbols) with a known input probability distribution, it is not optimal when the symbol-by-symbol restriction is dropped, or when the probability mass functions are unknown, not identically distributed, or not independent (e.g., "cat" is more common than "cta"). Other methods such as arithmetic coding and LZW (Lempel-Ziv-Welch) coding often have better compression capability: both of these methods can combine an arbitrary number of symbols for more efficient coding, and generally adapt to the actual input statistics, the latter of which is useful when input probabilities are not precisely known or vary significantly within the stream [6]. However, the limitations of Huffman coding should not be overstated; it can be used adaptively, accommodating unknown, changing, or context-dependent probabilities. In the case of known independent and identically distributed random variables, combining symbols reduces inefficiency in a way that approaches optimality as the number of symbols combined increases.

3.1 Our Approach

This paper uses Huffman coding technique to generate the frequency tables. The premise behind Huffman coding is that more frequently occurring symbols are coded using shorter code words as shown in Table 1. [2]. To accomplish this, variable length code words are assigned to symbols based on their frequency of occurrence. The coding is performed using a binary tree as shown in fig.1. First, the symbols are arranged in order of decreasing probability of occurrence. Then the two least occurring symbols are combined to form a new node. The result of this node is placed in the tree in a position that preserves the order. Then the new node is combined with the next least occurring symbol to create yet another node. This process is repeated until all nodes have been processed. Afterwards a traversal back to the tree is done to tag one branch as 0 and the other as 1. Traversal from the root node to the leaf node would give you the codeword for the symbol. The final node is designated the root [5].

Table 1. Frequency table

Symbol	**Count**	**Code**	**No. of bits**
A	15	0	15
B	7	100	21
C	6	101	18
D	6	110	18
E	5	111	15
		Total=	87

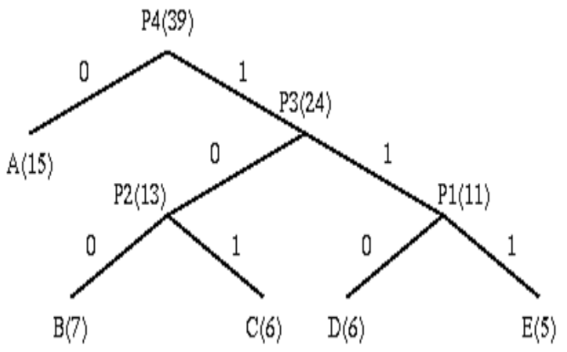


Fig. 1. Prefix code tree

In this technique, basically the frequency table will be store on server. When user wants to use this software they need to install this App and at the same time the frequency tables gets downloaded into the mobile phones [4]. The user needs to register him on this software and then he can login himself using login ID and password. After typing the message user needs to select SMS compression option, the SMS will get compress using the frequency tables. When the user enters the recipient mobile number, the App will verify that the recipient will have the same App install on his mobile or not. If the receiver has same App install on his mobile then the SMS will get send by using Internet [3]. At the receiver side the user needs to select SMS decompression option and then using the same frequency tables the SMS will get decompress. For this both the sender and receiver require Android mobiles and the App installed in their phones. This technique will provide private messaging service. Means this will not give any notification of SMS when user is offline, it will provide the message list when the user goes online.

3.2 Huffman Coding Algorithm

```
(1) Initialization: Put all nodes in an OPEN list, keep it sorted at all times (e.g., ABCDE).
(2) Repeat until the OPEN list has only one node left:
    (i) From OPEN pick two nodes having the lowest frequencies / probabilities, create a parent node of them.
    (ii) Assign the sum of the children‘s frequencies/probabilities to the parent node and insert it into OPEN.
    (iii) Assign code 0, 1 to the two branches of the tree, and delete the children from OPEN [8].
```

3.3 System Architecture

This paper represents a technique for developing an android application which will have the options of compressing and decompressing. Both the users will have to download this application on their respective android phones. As shown in fig. 2. the transmission of compressed data over cellular networks is done in a transparent way and therefore this mobile application is needed on both ends. That is because the compressed SMS will be encoded into fuzzy characters as shown in fig. 3 and cannot be understood by the receiving mobile device which does not have the same decompression application. Along with the application Huffman tree will be downloaded on their phones [2]. SMS will be sent through internet or Wi-Fi and the SMS will be compressed according to the Huffman tree which will convert the character data into compressed form and will save the bandwidth.

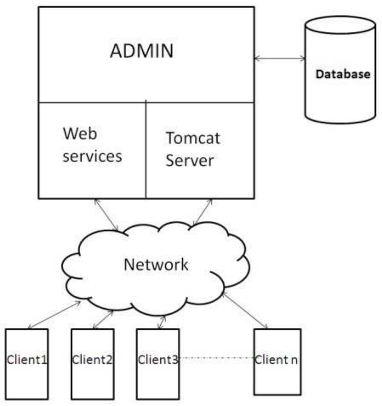
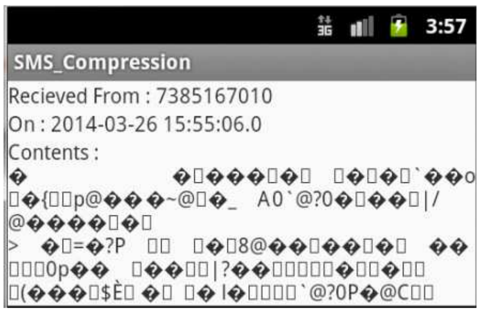


Fig.2. System Architecture



```
Recieved From : 7385167010
On : 2014-03-26 15:55:06.0
Contents :
�            �□����□�□  □�□�□`��o
□�{□□p@����~@□�_  A0`@?0�□���□|/
@������□�□
>  �□=�?P  □□  □�□8@���□���□�□  ��
□□□0p��   □���□□|?���□□□□□�□□�□□
□(����□$È□ �□ □� l�□□□□`@?0P�@C□□
```

Fig.3. Compressed SMS.

3.4 Application

This application is mainly develop for Android phones. As till now there is no any SMS compression technique is practically used in mobile phones because of the limitation they have. This technique will provide 50 to 100% compression ratio depending on the occurrences of frequently used words. This App will be supportable for all Android 2.3.3 and the later versions.

4. EVALUATION

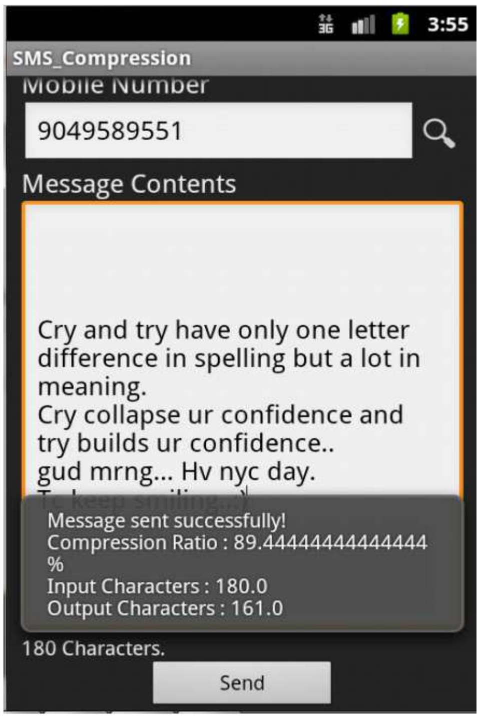
There is a formula for calculating compression efficiency of SMS compression technique using Huffman coding scheme.

$$Compression\ \% = \frac{compressed\ SMS\ size}{Original\ SMS\ size} \times 100.$$

Eq.(1).

In this case original SMS size represents the size of SMS before compression and compressed SMS size represents size of SMS after compression.

Testing process is conducted in order to find out how much compression ratio is given by this technique. This technique is implemented by using an emulator as shown in fig. 4. As we are going to implement this scheme for Android mobile phones so Android Virtual Device Manager of SDK (Software Development Kit) is being used for getting the outcome.



9049589551

Cry and try have only one letter difference in spelling but a lot in meaning.
Cry collapse ur confidence and try builds ur confidence..
gud mrng... Hv nyc day.
Tc keep smiling..:)

Message sent successfully!
Compression Ratio : 89.44444444444444 %
Input Characters : 180.0
Output Characters : 161.0

180 Characters.

Fig. 4. Original SMS.

In above SMS, total no of characters are 180 and total number of characters after compression are 161. By using above formula percentage compression can be calculated as follows:

$$Compression\,\% = \frac{161}{180} \times 100.$$

$$= 89.4444\%$$

5. FUTURE SCOPE

This SMS compression technology presented in this paper is compatible with English language only hence in future we can upgrade this application to support other languages also like French, Marathi, Malayalam, Chinese, Tamil etc [4]. This paper develops a technique mainly for mobile phones having Android OS, later we can develop it to run on other mobile phones having different OS.

6. CONCLUSION

Huffman table which is built using Huffman tree is used for compressing the SMS. Huffman Encoding proves to be a very efficient compression technique. Compression ratio depends on the SMS, it will produce a good compression if the characters used in SMS are frequently occurring and it will require small code. This paper has represented a successful technique for SMS compression which is user friendly, cost effective and time saving.

REFERENCES

[1] Affandi A.; Saparudin; Erwin (2011). “The Application of Text Compression to Short Message Service Using Huffman Table” Vol.6 No.1

[2] Aprianto, D. (2007). “Performansi Modifikasi LZW (Lempel-Ziv-Welch) Untuk Kompresi Teks.” [unpublished thesis], Department of Informatics, Bandung Institute of Technology.

[3] Hashemian, R. (2005). “Direct Huffman Code and Decoding Using The Table of Code-Lengths.” [unpublished thesis], Northen Illionis University.

[4] Huffman, D.A., (1952), A Method for the construction of Minimum Redundancy Codes, Proceedings of the IRE, pp. 1098-1101.

[5] Husodo A. Y.; Munir R. (2011), “Arithmetic Coding Modification to Compress SMS” International Conference on Electrical Engineering and Informatics, Bandung, Indonesia.

[6] Mahmoud, Tarek M et al. *Hybrid Compression Encryption Technique for Securing SMS.*

[7] Mohd A.; Min W. T. (2008), “Short Messaging System (SMS) Compression On Mobile Phone-SMS zipper” Jurnal Teknologi, 49(D) 173–187 © Universiti Teknologi Malaysia.

[8] Ong G. H.; Chong W. T., “Compressing Chinese Text files using an Adaptive Huffman Coding scheme and a Static Dictionary of Character pairs” Department of Information Systems & Computer Science National University of Singapore, Singapore.

[9] Patil M.; Prof. Sahu V. (2013), “A Survey of Compression and Encryption Techniques for SMS” International Journal of Advancements in Research & Technology, Volume 2, Issue 5.