IT FOR SOCIETY, Vol. 03, No. 02
ISSN 2503-2224

SMS Compression System using Arithmetic Coding Algorithm

Rosalina*, Wahyu Hidayat

President University
*Corresponding author: rosalina@president.ac.id

Abstract - *Short Message Service (SMS) has limitation in the length of its text message, which only provides 160 characters per SMS. It means that if we send more than 160 characters, it will be considered as sending more than one SMS, so that we have to spend more cost for sending SMS. On the other side, the Arithmetic Coding algorithm provides an effective mechanism for text compression. It has performed the great compression result and in many case it was considered as the better compression algorithm than other ones, such as Huffman and LZW (Lempel-Ziv-Welch). This research will implement the Arithmetic Coding algorithm to develop an application that will compress the SMS text message. The concept of Arithmetic Coding will be implemented to compress the SMS text message before it is sent from the sender to the receiver. The application is called CheaperZipper (CZ). This application will handle the sending and receiving SMS in the hand phone by preceding the compression and decompression process.*

Keywords: sms compression, arithmetic coding algorithm, text message, text compression

I. INTRODUCTION

Long time ago people said: ‘Say it with flowers….’, now it might be changed become ‘Say it with SMS….’ [1]. This statement starts to look along since more and more people that use SMS (Short Message Service) in their daily life. In the special days such as Idul Fitri, Christmas, and New Year the rate of SMS is increased rapidly. I these special days, peoples use a lot of SMS to express their happiness, hoping and wish for good luck by sending SMS to their family, relatives, and friends. During these special days the traffic of SMS in some GSM (Global System for Mobile Communication) operators show significant increase value compared to normal days. This phenomenon shows that the SMS become more popular in the community as the preferable choice in cellular communication.

In short, SMS is communication media that offers the service to deliver brief message through the mobile devices. It is categorized as strictly service, means that once people understand how to use it, hence they will tend to use this service [1]. Moreover, a research shows that 90% of the SMS in the world come from the SMS from person to person [2].

The SMS is very flexible since it does not know regional boundary, it means the SMS can be sent or accepted in all the world as long as it is in GSM (Global System for Mobile Communication) area [3]. However, beside its flexibility SMS also require the expense in its delivery and have the limitation in term of message length. SMS only provide 160 characters per one SMS. It means that if we send more than 160 characters, we must pay more for the SMS delivery. This matter will become the problem for the one who often use long SMS in their communication, because they must spend much money to send SMS, whereas the expense for one SMS is quite expensive.

Based on problems above, one of the solutions that can be offered is data compression. In brief, data compression can be defined as the process encoding given data that will reduce the size (length) of data. By using data compression the SMS text message will be compressed so that the length of the message becomes shorter than before. With this method the length of SMS can be extend to more than 160 characters for one SMS, so that directly it will decrease the cost as well. In this compression process, it is needed an additional software which will be installed in hand phone, both in sender and receiver. In the sender side, this software will compress the message before it is sent, while in the receiver side it will decompress the compressed message into original message.

II. LITERATURE STUDY

2.1 Data Compression

Data compression seeks to reduce the number of bits used to store or transmit information [**Error! Reference source not found.**]. It is used to reduce the number of bits produced by data. It has become important topic discussed by many researchers, because in the next future the data or information will become larger and larger so it needs a mechanism to save the space in the storage. In general, the task of data compression consists of two components, an encoding algorithm that takes a message and generates a "compressed" representation (hopefully with fewer bits), and a decoding algorithm that reconstructs the original message or some approximation of it from the compressed representation. These two components are typically intricately tied together since they both have to understand the shared compressed representation [**Error! Reference source not found.**].

There are many types of algorithm in data compression, such as: Huffman, Arithmetic Coding, LZ77 and its variant (LZ78, LZW, GZIP), Dynamic Markov Compression (DMC), Block-Sorting Lossless, Run-Length, Shannon-Fano, PPM (Prediction by Partial Matching), Burrows-Wheeler Block Sorting, Half Byte, etc.

In data compression there are 3 factors that will determine the quality of the compression result:

Compression Ratio

Compression Ratio is a value used to describe the reduction in size of the data. In other word, it is the ratio of the file sizes of a compressed file to an uncompressed file. For example, if we start with a 100 KB text file and compress it to 20 KB, the compression ratio is 20/100 = 0,2. Typically, the lower the compression ratio value will show the better the compression.

Compression Time

Compression time is the time that is needed by the computer to run the encoding and decoding process respectively. The faster compression time, the better compression is.

Memory Requirement

During the encoding and decoding process the computer will need amount of memory. The smaller the memory space used, the better the data compression is.

III. METHODOLOGY

The SMS communication occurs between the sender and the receiver through wireless medium. Generally, there are only two main processes, sending SMS and receiving SMS. The sender types the SMS message and gives the command to send the SMS through wireless medium. On the receiver side, SMS will be received from the wireless medium so that finally the receiver can read the SMS message. The Figure 3.1 shows the flow of one execution SMS process between the sender (User A) and the receiver (User B):

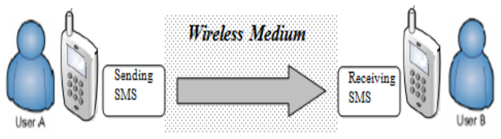


Figure 3.1 Figure 3.1 Sending and Receiving SMS

The cost for each delivery will depend on how long the SMS is. The longer the SMS, the more expensive the cost is. In fact, there is no facility in mobile device that can control the length of the message or event makes it shorter. Therefore, it is needed an application that can control the length of SMS and act as compressor to make the SMS become shorter.

New system offers the solution to insert the compression process in between the sending and receiving process. The SMS will be compressed first before it sent to the receiver. After being compressed, the compressed message will be sent to the receiver through the wireless medium. On the receiver side, the SMS will not read directly, but there will be decompression process preceded. The decompression process will translate back the compressed message into original message. The Figure 3.2 shows the detail process of new system.

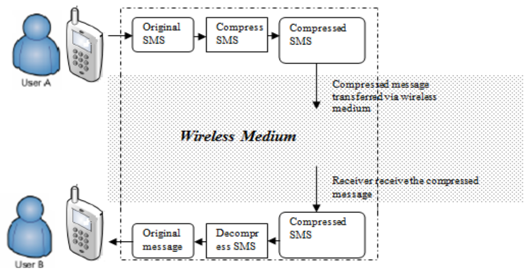


Figure 3.2 Sending and Receiving SMS in New System

3.1 Arithmetic Algorithm Analysis

The Arithmetic Coding will be combined with Prediction by Partial Match –start (PPM*). The result is expected will get maximal to handle the repetition characters or words, since the PPM* works better in compressing the repetition text.

3.1.1 Data Model

The data compression concept usually works at all ASCII characters. It is because the total ASCII character to build the data is 256 characters (0 - 255 characters). However, in the text format (English text format) not all ASCII characters are used to construct the text, except for other language such as German, and France. In English text format, it only uses the characters from character 32 (space) to character 126(~) plus new line ('\n') characters. Hence, the total of characters used for English text format only 96 characters. Therefore, it will improve the compression ratio if the number of characters is decreased. Instead of using 256 characters, it will better use only 96 characters by eliminating the rest characters. Figure 3.3 show the process of ASCII character conversion from 256 characters to 96 characters.

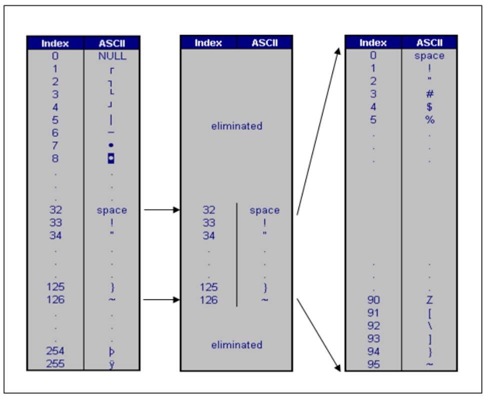


Figure 3.3 Data model for English text.

3.1.2 Flow Analysis

In this case, SMS compressor will use the adaptive one to avoid sending additional bit in compressed message. Another consideration is the SMS typed by user can be different pattern, so it is better to adapt the model with the pattern text rather than define the fix frequency symbol.

A. Encoding

The encoding process of Arithmetic Coding will be started by creating the model. Then the process will go to initiation process and during the encoding each symbol will perform scaling process followed by updating the mode. Figure 3.4 describe the flow chart in encoding process.

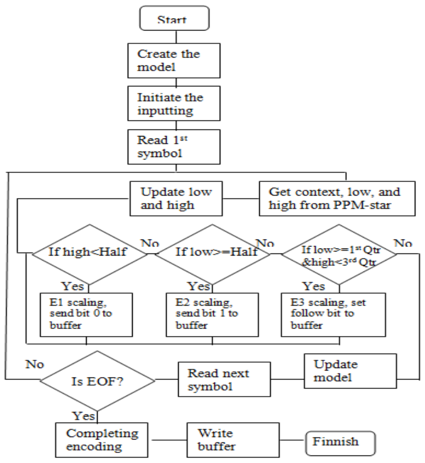


Figure 3.4 Flowchart for Encoding

The Arithmetic Coding works at 95 (from 32 to 126) characters + 1 character as new line and 1 character for indicating End Of File (EOF). Hence, the total characters are 97 characters. The Arithmetic Coding combined with PPM* to predict the context for each characters in SMS. The process will start by initiating the symbol, context, and tries and then it will read all characters of SMS to be encoded character per character until the last character. In the encoding process, the context for each character will be predicted before being encoded by Arithmetic Coding algorithm.

```
encodeSMS(SMS)
1: Initialization symbols, contexts, and tries
2: n←0
3: length←SMS.length()
4: total←97
5: while n<lenth and ai next characters of SMS
       do Encode_Symbol(ai)
6: encode_symbol(EOF)
7: done_encoding()

encodeSymbol(symbol)
1: context←predictcontext(symbol)
2: encode_context(context, symbol)
3: low←low bound of symbol
4: high←high bound of symbol
5: arithmetic_encode (low,high,total)
6: update(context,symbol)
```

Figure 3.5. Algorithm of encoding characters

B. Decoding

The decoding process is nearly the same with the encoding process. However, instead of reading each characters of message there will be process for reading bytes as described in Figure 3.6.

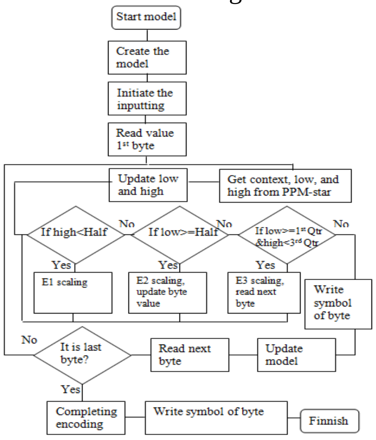


Figure 3.6 Flowchart for Decoding

The same as the encoding process, decoding process also work at 97 ASCI characters. The decoding process uses the same model with encoding one. At the first time of decompression The *S_value* will be initiated a value described by first 16 bytes of SMS. After that it will continue reading byte per byte of SMS.

```
decodeSMS(SMS)
1: Initialization symbols, contexts, and tries
2: S_value←value of first 16 bytes.
3: total←97
4: while ai is next byte and ai not last byte
     do symbol=Decode_Symbol(ai)
     write_out(symbol)

decodeSymbol(symbol)
1: context←predictcontext(symbol)
2: decode_context(context, symbol)
3: low←low bound of symbol
4: high←high bound of symbol
5: arithmetic_decode (low,high,total)
6: update(context,symbol)
```

Figure 3.7 Algorithm of decoding characters

IV. RESULT

4.1 Implementation

A. Main Form

This form will become the main form of CZ that is loaded first time when CZ runs. From this form the user can choose the menus New SMS, Inbox, Sent, Contacts, Setting, About, and Exit.

When this form is activated, it will listen automatically at the port 6677 to catch SMS comes through this port. The listening process will run every 0.01 second done by class SMSCatcher. If it is incoming SMS, CZ will translate the sender number and its message. The sender number will be checked if it is already in the contact list or not. If it is not yet in the contact list, CZ will automatically save that number to the contact list. CZ also will check the SMS whether it is compressed SMS format or not. If it is compressed SMS format, then CZ will decompress it before save it into inbox, otherwise save directly the SMS to inbox.

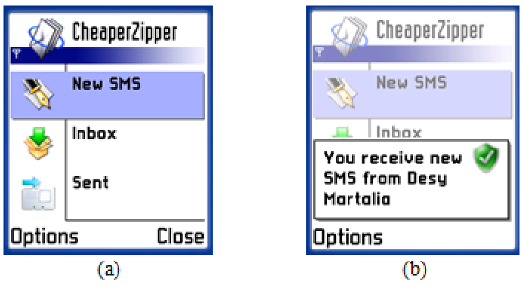


Figure 4.1 (a) The main form, (b) Alert of incoming SMS

B. New SMS

In this item the user can type the SMS to be sent to the receiver. The SMS will be compressed run timely every 0.05 second and if the SMS is changed. The result of the compression will be displayed directly on the screen of the user, so that the user will know the length of typed SMS so far. The compression process will only occur when the number of characters is greater than 152 characters.

There will be 2 validations in the process. The first validation will check whether the SMS characters are in the correct format or not. The correct format means the SMS only contain characters ASCII from “space” (32) to “~” (126) and new line “\n”. While, the second validation will check whether the SMS has already the maximum length of SMS or not.

After typing the SMS, the next part is inputting the destination address. To input the destination address the user can type manually or browse from the contact list.

Before sending the message, CZ will match the destination address with the list of contacts. If it the member of contacts, CZ will send compressed SMS, otherwise the SMS will be sent without any compression. Figure 4.2 show the screen shoot when the destination address is not member of contacts. It will check also the SMS length, if the length is greater than 152 (160-8 bytes) then the SMS will be send in binary format, otherwise it will be sent in text format.

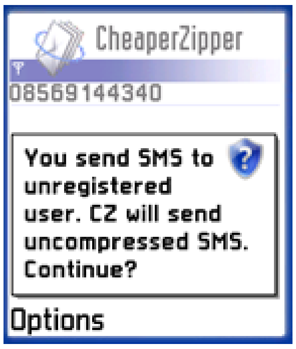

CheaperZipper
08569144340
You send SMS to unregistered user. CZ will send uncompressed SMS. Continue?
Options

Figure 4.2 Unregistered user alert in New SMS

C. Inbox

Once the incoming SMS is received by CZ, CZ will automatically record this message and save into application database called Datastore. The inbox will load all contents in the Datastore and display them as the list. From this inbox user can perform open the detail of SMS, reply, forward, delete, and delete all SMS. In order make it more informative, there will be differentiation between read SMS and unread SMS, by giving different image.

At the first time, the CZ will load all received SMS from DataStore in class RecordStore. If the SMS is already read by the user, then display it in the list with read image; otherwise display it in the list with unread image. And when a received SMS is read by the user, it will be set the flag to be considered as read SMS.

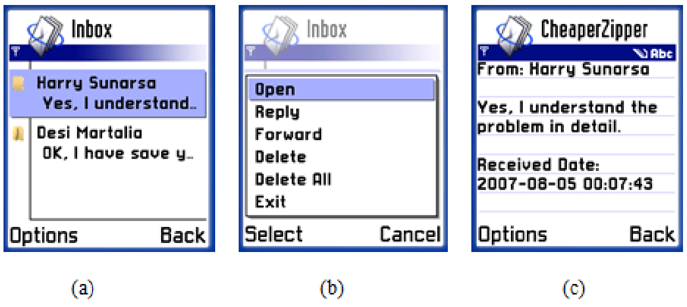


Figure 4.3 (a) Inbox list, the open folder icon indicating read SMS, otherwise unread SMS. (b) The menus that can be chosen from the Inbox. (c) The detail of SMS

D. Sent

This part becomes the history of sending SMS activity performed by the user. When the user performs sending SMS, a new record will be stored to the DataStore as sent SMS. From this part user can perform open the detail SMS, forward, delete, and delete all SMS. The data of sent SMS will be loaded from DataStore and then display all item in sent list.

E. Contacts

Contacts will store the hand phone number that has installed CZ inside. The data will be loaded from the DataStore and displayed in form of list. The user has to register the phone number and its name manually. User can create new contact, delete a contact and send SMS directly to certain chosen contact.

F. Setting

The setting will include the setting of language, maximum SMS length, and sound. For the language CZ so far only provide English, other language such as Indonesia, German, and France could be added in the future work to improve the compression result. Maximum SMS length will set the maximum length of SMS that user can type. For instance, if the user chooses 1 SMS (133 characters) means that the user can type the SMS at most 133 characters, otherwise the CZ will erase the rest of characters.

The user also can change the sound to indicate incoming SMS has been received. This feature allows the user to browse the available sound files in the hand phone. The data will be loaded from the DataStore and any changing of setting will be saved in the DataStore as well.

4.2 The testing of SMS compression process

The SMS compression process is started from reading SMS until getting the compressed SMS. The testing for this process was carried to test the compression at different length of SMS and at different type of SMS but the same SMS length.

A. Compression Ratio

In Figure 5.1 the compression ratio is given at different type of SMS. The test was run by compressing different type of SMS with the same length of characters. The length for each type of SMS was set to 310 characters (3 SMS). Figure 5.1 shows the testing for the compression ratio versus SMS type in Nokia 7610 and Sonny Ericsson W700. It shows that the value of compression ratio is the same both in Nokia 7610 and Sonny Ericsson W700. It shows also that the compression ratio get maximal value when applied to repetition SMS and get minimum value at non-repetition SMS. It is because the function of PPM-star that improve the compression for repetition text (words or characters).

The Figure 4.4 shows the compression ratio of normal SMS with different SMS length. The y axis of Figure 4.4 shows the compression ratio and x axis contain the length of SMS. The result shows that the more length of SMS, the compression ratio tends to be better. It is because the possibility of repetition of words or characters is bigger so that the compression can work maximal.

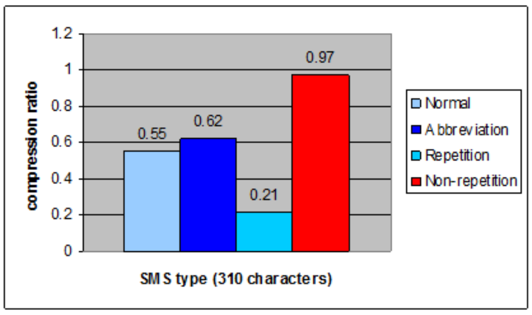


Figure 4.4 Compression ratio at different SMS type.

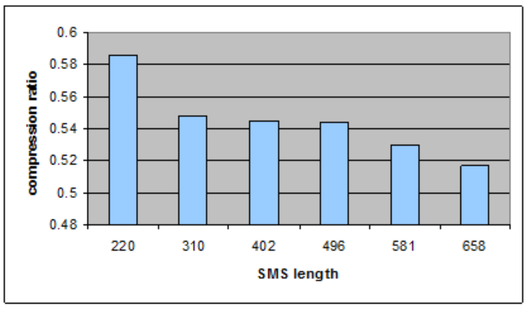


Figure 4.5 Compression ratio at different length of SMS.

From the Figure 4.4 and Figure 4.5, it shows that the SMS content is much more important in term of compression ratio rather than the length of SMS it self and not depend on the model of the hand phone.

B. Compression Time

The result of testing of compression time is shown in Figure 4.6 and Figure 4.7. Unlike the compression ratio that will get better at repetition SMS, but the compression time will take longer time if used to compress repetition SMS. It is understandable because there will be many looping process if there are many repetition characters or words. However, this condition is not occurred for the non-repetition, since the repetition SMS still need more time to compress event the compression ratio is the worst. It is because non repetition SMS needs more time to create the tries to store the context of characters. Regarding to SMS length, the compression time will increase doe to the increasing of SMS length as shown in Figure 4.7.

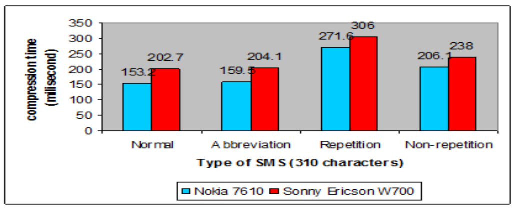


Figure 4.6 Compression time at different type of SMS.

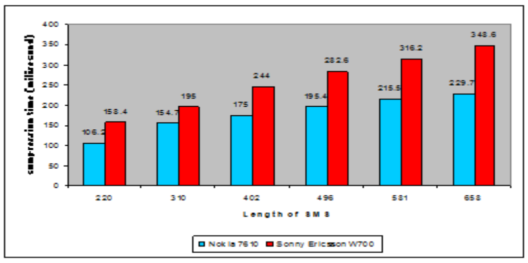
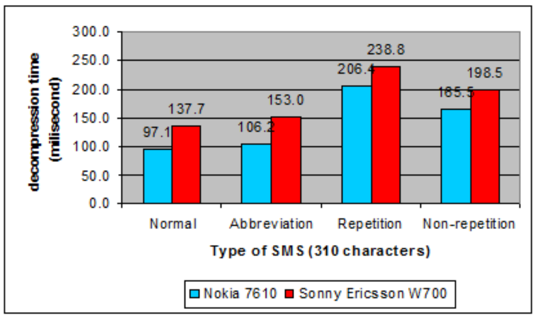


Figure 4.7 Compression time at different length of SMS

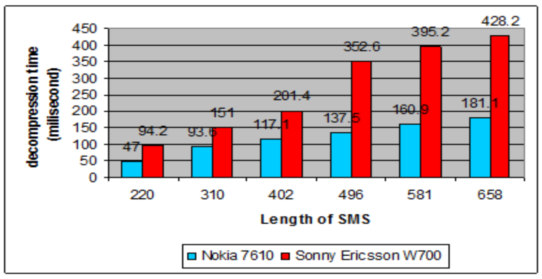
4.3 The testing of SMS decompression process

A. Compression Time

The time needed to translate back compressed SMS into original SMS of different SMS type is given in Figure 4.8. It shows that to translate compressed SMS from repetition SMS need more time than other types. For the same type of SMS but at different length of SMS, the decompression time will increase if the length of compressed SMS increases as well. This condition is shown in Figure 4.9.



(a)



(b)

Figure 4.8. (a) Decompression time at different SMS type, (b) Decompression time at different SMS length.

4.4 Length of SMS

The last testing investigated the ability of CZ to extend the length SMS. The testing was carried out by compressing some normal SMS and abbreviation SMS so that the compression result must reach 133 characters (1 SMS). The result of this testing is given in Figure 4.9. The result shows that from 20 of SMS (10 normal SMS and 10 abbreviation SMS), the average of compression ration is 0.6. Hence, CZ can extend the length of SMS from 160 characters into 133 / 0.605 = 221 characters.

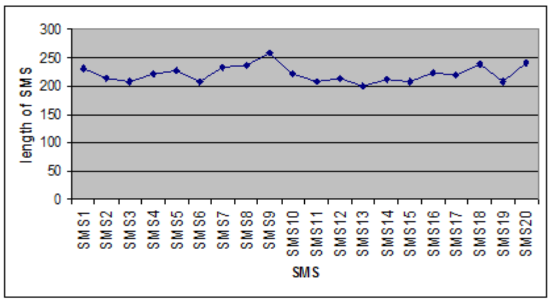


Figure 4.9 Length of SMS to reach 133 characters.

V. CONCLUSION

This research has been successfully implementing the idea of SMS compression in mobile device. However, this work still has big possibility to growth and to be enhanced in the future. In order to improve the quality of the work, the following improvement can be done in the future work:

- Improve the model of data compression algorithm.
  The growth of data compression algorithm is improved from time to time in term of compression ratio gained. The Arithmetic Coding algorithm has much challenge to be researched in order to improve its performance.
- Testing for energy consumption (battery) and sending time.
  This research only run the testing for three parameters; compression ratio, compression time, and memory usage. Actually there are still some parameters that can be tested to measure the performance of the SMS compression application, such as energy consumption (battery) and sending time.
- Energy consumption means the effect of the compression and decompression process into the power of mobile device. The sending time will compare the time needed to transfer SMS from the sender to the receiver, between SMS with compression and SMS without compression.
- Multi language compression

The SMS compression in this research only focused on English text, so that it will not work well on other language, such as German, France and Chinese. It will be better if the application can work at any languages so that other user from different country also can use this SMS compression application.

- Add model of mobile device in the testing phase.
  As mention before, the testing only run in Nokia 7610 and Sonny Ericsson W700. Although, theoretically the application can run in all model of mobile devices, but it has not proved yet in real testing. Therefore, it will make more general and objective if run the testing in other model of mobile device, such as Motorola, Seamen, LG, Samsung, and other vendors.
- Integrate SMS Compression System with SIM Card
  In order to make easier in installation process, it is possible in the future to insert SMS compressor tool (CheapeZipper) into SIM Card. Hence, when the user plug in the SIM Card into the hand phone, the SMS Compression System will be installed automatically on the hand phone. This idea can be implemented by making cooperation with some GSM providers.

REFERENCES

[1] T. Satriyantono, “Say It With SMS”, http://satriyantono.net, April 2007

[2] T. Satriyantono, “Aplikasi mobile berbasis SMS di era GPRS dan 3G“,http://satriyantono.net, April 2007.

[3] A.Sugiharto, “J2ME Wireless Messaging System“, http://kuliahwireless.blogspot.com, April 2007.

[4] I. Witten, R. Neal, and J.Cleary, “Arithmetic coding for data compression“, Comm. of the ACM, 1987.

[5] Nelson M, “The Data Compression Book“, M&T Books, 1995.

[6] Guy E. Blelloch,I “Introduction to Data Compression”, Computer Science Department Carnegie Mellon University, October 16, 2001

[7] John G. Cleary and Ian H. Witten, “Data Compression Using Adaptive Coding and Partial String Matching”, 1982

[8] G. Howard, “Analysis of Arithmetic Coding for Data Compression“, Brown University - Department of Computer Science, November 1992.

[9] G. Cleary, “Unbounded length contexts for PPM”, Department of Computer Science, University of Waikato, New Zealand.

[10] S. Rein, “Compression of Short Text on Embedded Systems“, Technical University of Berlin, September 2006.

[11] Yuku, “SMS Compression,” http://www.kejut.com/kompresisms, February 2005