

Low Complex and Power Efficient Text Compressor for Cellular and Sensor Networks

Frank H.P. Fitzek* Stephan Rein** Morten V. Pedersen*
Gian Paolo Perrucci* Thorsten Schneider* Clemens Gühmann**

*Aalborg University, Denmark

**Technical University of Berlin, Germany

Abstract—This paper investigates a novel text compressor for short text messages. Short text messages are used in cellular and sensor networks. The terminals or nodes are battery driven and compression is needed to save energy or to use bandwidth in an efficient manner. The presented work focuses on the trade-off between compression and energy consumption. Examples are shown where the energy to compress and decompress is lower than the energy saved during the transmission on the wireless medium. Thus the compression pays off in terms of overall energy saving. Especially for the cellular scenario we show the application of the text compressor for short message services offering more than twice the capacity compared to a standard message. A JAVA application for SMS compression on mobile phones can be downloaded free of charge for personal usage or for research purposes. The main conclusion of this paper is that the compressor has such a low complexity that in addition to the possible lower number of SMS used, significant power saving can be achieved.

I. MOTIVATION

Compression of short messages is a vital operation for low complexity entities such as mobile phones or wireless sensors. As we will show throughout this paper the compression will give benefits in terms of costs and energy consumption. The application field of a text compressor are mobile phones and wireless sensors.

In the mobile phone's world the compression would allow users to increase the number of characters of their short message service (SMS). As it seems obvious to compress these messages, state of the art text compressor based on LZ77 and others would fail to compress such small messages ending up with more data than the original [1]. Therefore a totally new concept is needed to compress short messages. The transmission of compressed data over cellular networks is done in a transparent way (so still not more than 160 characters or 140 bytes are transmitted in one SMS) and therefore a compressor/decompressor is needed on both ends.

But mobile phones are not the only field of application. Sensors are battery driven entities too, with low memory and some sort of wireless communication front end. The compression will pay off if the energy spent by the compression and decompression is lower than the energy saved for the transmission. As given in Figure 2, in case of multi path communication between source and sink, the compression and decompression is done only once, but the gain for shorter transmission time and in turn less power consumption is achieved by each hop. Another advantage of the compression

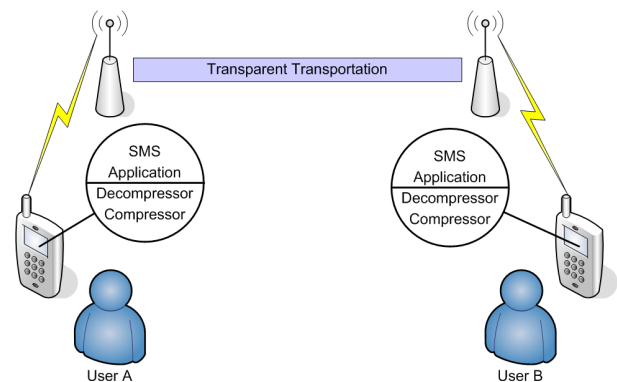


Fig. 1. Text Compressor for the Mobile Phone World.

is that the time spent on the wireless medium is shorter and therefore more capacity is available. This is especially important for information aggregation, where a central entity wakes up thousands of nodes asking them to provide some information.

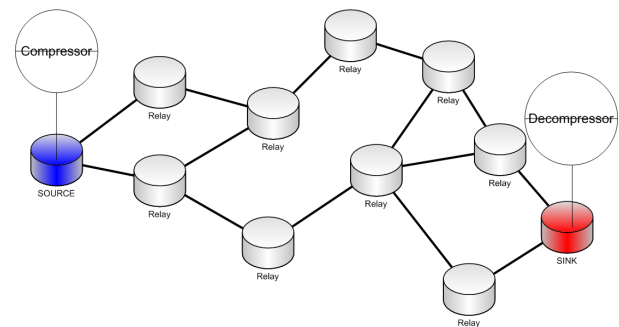


Fig. 2. Text Compressor for the Sensor World.

Therefore, compression of short messages to be conveyed over the wireless medium seems to be promising in terms of power, costs, and bandwidth savings. The complexity introduced in terms of computational power and memory usage will be investigated and reported to be low.

II. TEXT COMPRESSOR

A. Functionality

The here applied text compressor uses a trained statistical context model to arithmetically code a text message. The context model detects known sequences of characters in the

short message and estimates the probability of each symbol with the given context, as it is done in *prediction by partial matching* (PPM). The probability is then given to an arithmetic coder [2] that computes a sequence of bits. The compressor tries to code a symbol in the highest order, that is the number of characters that precede the symbol to be coded. If the context model does not contain the string (i.e., the symbol to be coded and its context), it switches to the next lower order. In the lowest order ($o = -1$) each symbol has equal probability, that is $p = 1/256$ for an alphabet of 256 different symbols, thus not resulting in any compression. The more symbols are coded in higher orders the better is the compression, as these symbols are less probable (because they are more specific) and thus are coded by less bits. To signal the decompressor the current order the encoder uses the so-called *Escape* symbols. The decompressor constructs a symbol's context from the previous decoded symbols.

The employed text compressor is novel in that it does not use a dictionary / data tree to store the single character arrays and their probability, which is generally stored as an integer *symbol count*. Instead, it uses a single data array each element containing two bytes: The first byte contains a symbol count and the second a parity check byte. The single elements are accessed by a specific hash function that assigns each character array an element of the data array. Collisions are detected by a parity check of the hash function input and the parity byte of the mapped element in the data array. We call this functionality a *data model*. To build a context model the data model is supplemented by functions to compute the complete statistics of the requested symbol, i.e., a *left*, *right*, and a *total count*. The text compressor is fully explained in [3].

B. Memory Requirement

The code book used for compression is designed to be rather small not requiring too much memory space. In Figure 3 the potential compression is given versus the message length for different sizes of the code book. We see that for very short messages the compression is not high, but with more than 10 characters the proposed compression is outperforming the existing solution represented by the PPMc curve. Furthermore we note that a code book of size 32kB is achieving already near to the optimal performance. Larger code books improve the compression, but not in a significant way.

III. MEASUREMENT

For the measurement testbed we used two types of mobile phones, namely Nokia 6600 and Nokia 6630. These phones have been chosen to investigate the performance of the text compressor for two different platforms. The text compressor is implemented in JAVA and C++ for Symbian OS. The JAVA application was done mainly to support as many phones as possible and the C++ version was done for the measuring as the object oriented approach allows a more detailed view on the complexity and energy consumption of the different mechanisms. The first goal was to verify the compression ratios and compressed output with those done on the PC beforehand.

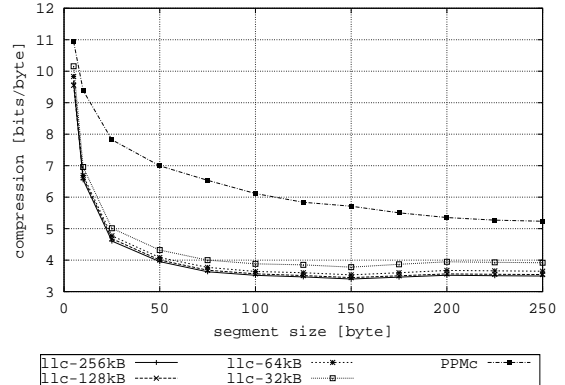


Fig. 3. Compression with different memory requirements for the compression model 2.

Furthermore the measurements are done to investigate the time needed to compress and the energy consumption for both, the compression and the sending process.

A. Measurement Testbed

The measurements were carried out with one phone connected to the Agilent 34401A multi-meter measuring the current from the Nokia battery with 3.7 V. To measure the energy consumption for compression, sending, receiving and decompression only, we wrote our own script performing each action separately. Even though the SMS compressor supports online compression, the compression for the measurements were done for predefined messages. The display lightning was suspended while we did the measurements and the messages given in Table I were used.

B. Compression Ratio

In Figure 4 the compression ratio versus the text length for different values for the compression mode is given. The compression ratio is a metric to describe how many compressed bits are needed to describe one byte. The lower the presented value is the better is the compression. A general observation is that higher modes lead to better compression ratios even if the difference with higher orders becomes smaller. Note that in some cases mode 3 may lead to slightly worse ratios than mode 2, as we have only four predefined messages. The compression ratio remains stable independently from the message length and the content of the message.

We note that in most of the cases the compression pays off with messages which are longer than eight characters. From our experience we know that the message content is much more important in terms of compression ratio than the message length. The compression ratio does obviously (and should) not depend on the mobile phone model.

C. Energy Consumption

In the following we give a detailed view on the energy consumption for the compression, sending, receiving and

TABLE I
MESSAGES USED FOR THE TESTING.

number	length	content
1	203	<i>Hi Stephan, thanks for the nice plot. I will meet Morten soon and report back how large the compression gain of this email will be. Would be nice if you could report the gain in dependency on the order.</i>
2	291	<i>The international system of units consists of a set of units together with a set of prefixes. The units of SI can be divided into two subsets. There are the seven base units. Each of these base units are dimensionally independent. From these seven base units several other units are derived.</i>
3	302	<i>However, few believed that SMS would be used as a means for sending text messages from a mobile user to another. One factor in the takeup of SMS was that operators were slow to eliminate billing fraud which was possible by changing SMSC settings on individual handsets to the SMSC's of other operators.</i>
4	454	<i>Alice is a fictional character in the books Alice's Adventures in Wonderland and its sequel Through the Looking-Glass, which were written by Charles Dodgson under the pen name Lewis Carroll. The character is based on Alice Liddell, a child friend of Dodgson's. The pictures, however, do not depict Alice Liddell, as the illustrator never met her. She is seen as a logical girl, sometimes being pedantic, especially with Humpty Dumpty in the second book.</i>

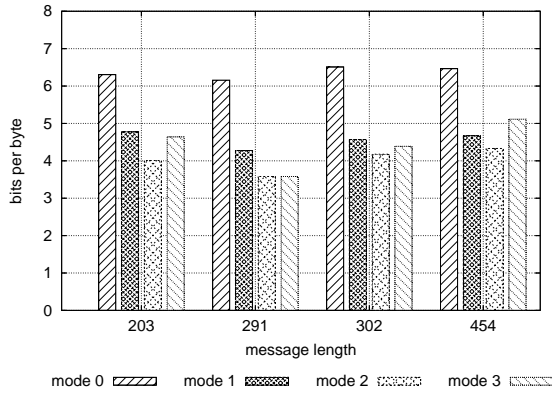


Fig. 4. Compression ratio versus compression with mode 0, 1, 2, and 3 for different message lengths.

decompression part. While the compression and decompression is only important for the measurement with compressed messages, the send and receiving part is the same for all measurements. The only difference between the compressed and uncompressed transmission is the length of the message, which has an impact on the energy used. Finally we give the results for the overall energy consumption with the split in sender and receiver side energy consumption.

1) *Compression:* In Figure 5 the energy needed to compress different messages is given at four compressions modes. On the x-axis the two phones with its four modes are given for each message length. The energy varies between 0.2 and 1.0 Joule and with an increasing message length the energy consumption increases. Generally the energy consumption

increases with a higher mode. In case the loading of the model is included the energy consumption is dramatically higher. For a message length of 203, the Nokia 6630 needs 0.1 Joule and 3.15 Joule to load model 0 and model 3, respectively.

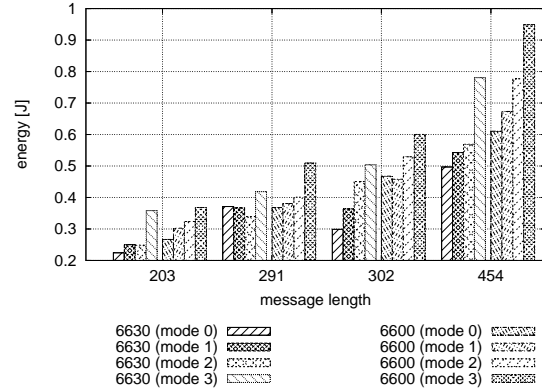


Fig. 5. Energy consumption to compress messages with different lengths at four compressions modes 0, 1, 2, and 3.

2) *Sending Process:* In Figure 6 the energy consumption of the sending process in dependency of the SMS length is given. The measurements for the two phones did not have any significant difference. For each message length multiple measurements have been performed and the mean value with a 99% confidence interval is given. If a message will not fit into one SMS, the message has to be send by multiple SMS. The energy for sending increases slightly with the message length. When a new SMS has to be used due to the message length, there is a jump in the energy consumption. The message length of one SMS is 133 byte as we need one port identification to distinguish between the uncompressed and compressed measurements.

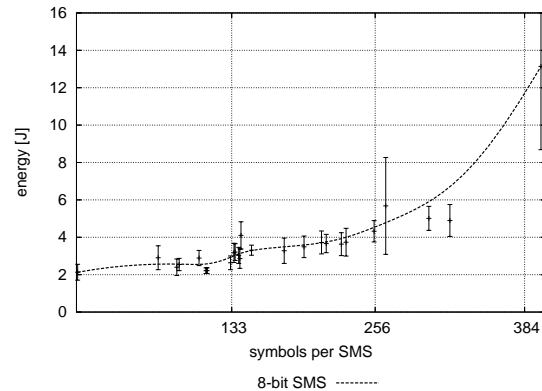


Fig. 6. Energy consumption of the sending process for binary SMS with different lengths.

3) *Receiving Process:* Figure 7 gives the energy consumption of the receiving process for binary SMS versus

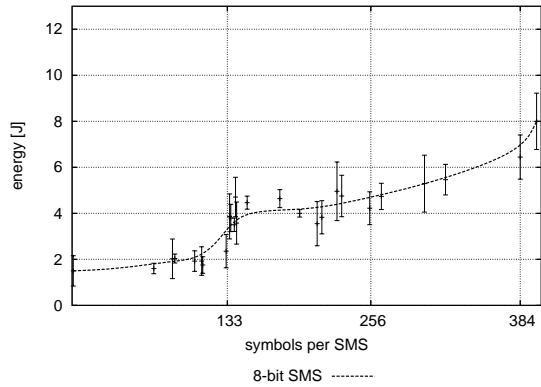


Fig. 7. Energy consumption of the receiving process for binary SMS with different lengths.

different message length. The characteristic is similar to that of Figure 6. We note that the step in energy consumption from one to two SMS is larger than the step measured for the sending process.

4) *Decompression*: Figure 8 is the counterpart measurement to Figure 5 giving the energy to decompress the message. The energy consumption for compression and decompression is similar.

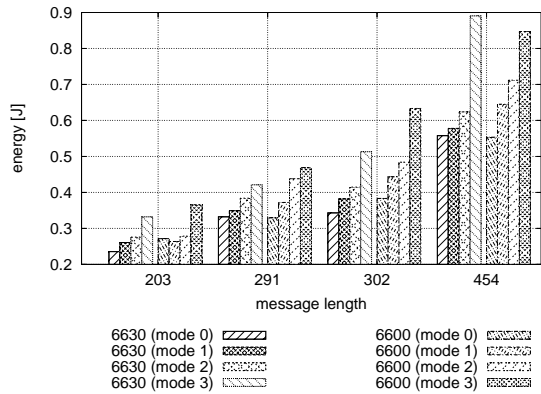


Fig. 8. Energy consumption to decompress messages with different lengths at different compression modes 0, 1, 2, and 3.

5) *Complete Transmission Chain*: Here we discuss the complete energy consumption on sender and receiver side. Figure 9 gives the energy consumption for the sender side, thus the compression with model loading and the sending process. On the x-axis the ticks are set at 133, 256, and 384 to indicate how many SMS are needed to convey the message. As we have four messages, the plot contains four curves. Each curve is composed out of five points. The most right point represents the uncompressed sending process. The second, third, fourth, and fifth point (moving to the left on the curve) represents the compressed sending with model 0, model 1, model 2, and

model 3, respectively. Each point indicates the energy needed to support this sending process and the number of SMS needed to convey the full message.

For the message with 203 characters we see that we would reduce the needed SMS messages from two down to one SMS using the compression mode 1 or higher. We further observe that the uncompressed message would need the minimum energy. By compressing this message we need more energy. This observation is not a general trend as we can see for other measurements. If we have a look at the message with 302 characters, we would reduce the needed SMS message from three to two with compression mode 0 and compression mode 1 reducing also the needed energy by nearly 2 Joule. Further compression with compression mode 3 is increasing the energy by 2 Joule compared to the uncompressed one, but on the other side we need only one SMS reducing the cost by a third. By these example we motivate different compression modes.

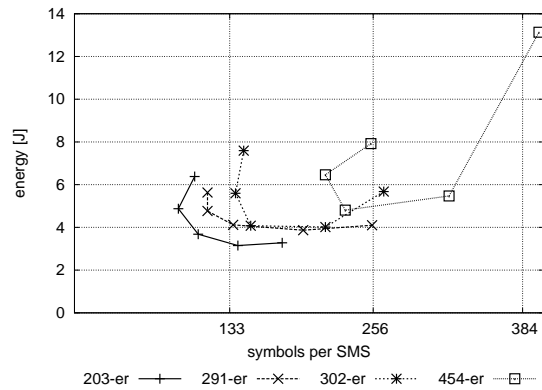


Fig. 9. Energy trade-off plot for the sender side with model loading.

In Figure 10 we see the energy consumption of the receiver side including the receiving and the decompressing process. The plot has to be read as the plot of the sender-side.

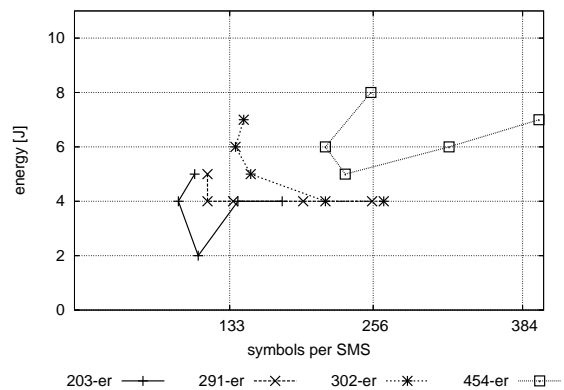


Fig. 10. Energy trade-off plot for the receiver side with model loading.

For the message length 454 the energy, including the sender and receiver, is around 20 Joule for the uncompressed transmission. For the compressed transmission (mode 2) 12 Joule are consumed if the model loading is taking into account and 8 Joule if we assume that the model is already loaded. In such a situation the energy consumption is reduced to 60% or 40%, respectively.

D. Compression Time

Even though the compression can be done in the background while the SMS is written, here we present the time needed for a stand alone compression on the Nokia 6600 and the Nokia 6630. Furthermore, we envision also the compression in the sensor world and there the compression will be started when the text message is fully available.

In Figure 11 and Figure 12 the time needed to compress or decompress is given respectively for two mobile phones under investigation. In general higher mode orders need more time and obviously the 100MHz processor of the Nokia 6600 needs more time than the 200MHz processor of the Nokia 6630. As for the energy measurements, the time for compression or decompression is dramatically smaller than the loading of the model. For a message length of 203, the Nokia 6630 needs 0.5 seconds and 11.5 seconds to load model 0 and model 3, respectively. As the model loading is taking a significant portion of energy and time, in the publicly available application the model is loaded only once while booting the phone.

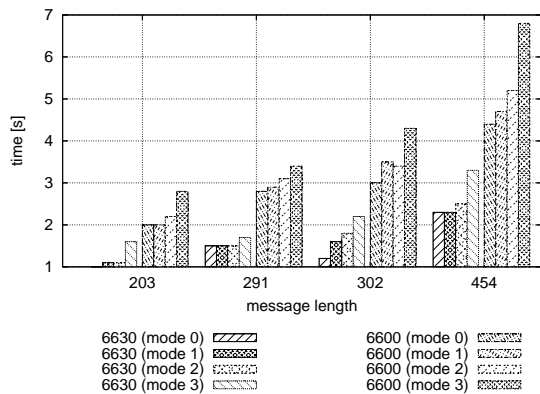


Fig. 11. Time needed to compress messages at different modes for the Nokia 6600 and Nokia 6630.

IV. CONCLUSION

In this paper we have investigated a novel low complex and power efficient text compressor that has been introduced beforehand. The compressor can be applied in cellular mobile communication systems as well as in sensor networks. The text compressor has been introduced shortly and a solid performance evaluation for the usage on mobile phones has been carried out. The main conclusion is that compression pays always off in terms of a lower number of SMS used and

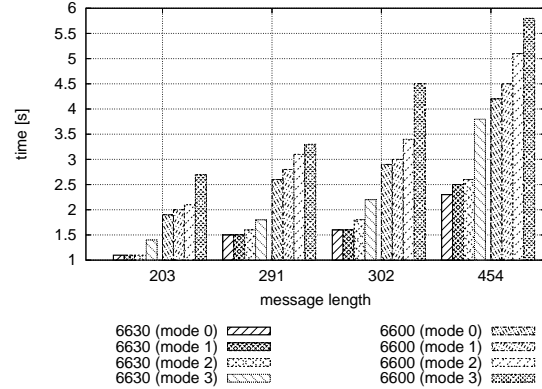


Fig. 12. Time needed to decompress messages at different modes for the Nokia 6600 and Nokia 6630.

energy savings. An application for mobile phones has been implemented for the measurements, but can also be used by other researchers for their work or users for their daily life. The application is written in JAVA and can be downloaded free of charge for personal usage at [4]. The text compressor compresses short text messages offering the customer the possibility to use more than the double capacity without being charged at additional costs. The compressor tool is needed on both ends, the sender and the receiver. The compressor is able to work at different modes. The higher the mode, the more energy is used, but also a higher compression ratio is achieved. The investigations were focused on the compression gain, the energy consumption and the time needed to perform the compression. As the compression gain depends on the message, the compressor needs a smart strategy to decide which mode to use, offering the best financial solution in the first step and secondly looking into minimizing the energy used for compression and transmission. This strategy can be derived from our measurements. The compressor should be used in all places where the costs for transmission are larger than in the compression itself and we have identified the cellular and the sensor environment as suited candidates.

V. ACKNOWLEDGEMENTS

Parts of this project have been financed by the Danish Government within the X3MP project. We are also grateful Nokia's support providing the phones to carry out the measurements .

REFERENCES

- [1] G. Korodi, J. Rissanen, and I. Tabus, "Lossless data compression using optimal tree machines," in *Proceedings of the IEEE Data Compression Conference (DCC'05)*, March 2005, pp. 348–357.
- [2] S. Rein and C. Gühmann, "Arithmetic coding—a short tutorial and free source code," Tech. Rep., Wavelet Application Group, April 2005.
- [3] S. Rein, C. Guehmann, and F.H.P. Fitzek, "Low Complexity Compression of Short Messages," in *IEEE Data Compression Conference (DCC), IEEE Computer Society Press, Snowbird, UT, Mar. 2006*.
- [4] AAU web pages, "smszipper," smszipper.kom.aau.dk, March 2006.