COMPRESSION SCHEME FOR INCREASING THE LIFETIME OF WIRELESS INTELLIGENT SENSOR NETWORKS

Dragos Ioan Sacaleanu[a], Rodica Stoian[a], Dragos Mihai Ofrim[a,b], Nikos Deligiannis[b,c]

[a]Faculty of Electronics, Telecommunications and Information Technology,
Universitatea Politehnica din Bucuresti, Iuliu Maniu 1-3, 061071, Bucharest, Romania.
[b]Department of Electronics and Informatics, Vrije Universiteit Brussel, Pleinlaan 2, B-1050 Brussels, Belgium.
[c]Interdisciplinary Institute for Broadband Technology (IBBT), Gaston Crommenlaan 8, B-9050 Ghent, Belgium.

ABSTRACT

Wireless intelligent sensor networks (WISN) are characterized by the ability of self-organization and auto-coordination for long periods of time. In most cases, the limitation of time function is given by the energy supply of the sensors. Taking into account that the greatest amount of energy is spent in transmission, many studies are concentrated on minimizing the amount of transmitted bits. In order to perform this, data compression algorithms are introduced in data processing. This paper presents a scheme that combines a new extrapolation prediction algorithm and a simple Huffman compression algorithm in order to prolong the lifetime of the WISN. Compared with other data compression schemes, the new scheme has obtained better experimental results in terms of number of bits transmitted and compression ratio.

Index Terms — Wireless sensor networks, energy saving, prediction, data compression

1. INTRODUCTION

An intelligent sensor is characterized by compensation, computation, communication, integration and validation. With these features, two or more sensors can create wireless intelligent sensor networks that are able to self organize and self coordinate. Due to these characteristics, the diversity of applications implemented with such networks is growing every day. In most cases, the system has to work properly for long periods of time without external help. Since the modules are placed on location where wireless communication is imposed, often the power source is a battery with limited resources.

Energy saving in WISN is a big concern among researchers because the major problem for designing such a network is the function time. A sensor consumes energy during processing, transmission and sensing. Depending on the hardware architecture, the sensor consumes more than twice as much energy during transmission than in processing [1]. This, together with the fact that the energy spend for transmitting a bit is the same with processing 1000 instructions [2], shows that it is more convenient to process the information in order to transmit less bits. For this, a viable solution is data compression.

In many WISN applications like monitoring and data acquisition, where the sensors transmit the data to the sink periodically, a certain degree of data correlation in time is registered. Given this fact, a prediction algorithm could be introduced in order to compress and transmit only the difference between the predicted value and the measured value. This way, the data transmitted could be coded with a smaller number of bits, leading to energy saving and prolonging the lifetime of the network.

2. RELATED WORK

Data compression is a very common technique used in WISN to minimize the quantity of bits transmitted. A multitude of compression algorithms can be implemented in WISN like Sensor-LZW, arithmetic coding or Huffman algorithm.

The Huffman compression algorithm is one of the most used and can be implemented in a variety of ways. We can implement the Static Huffman algorithm [3] which is a simple compression algorithm which reduces memory and computational resources on a WSN node. The algorithm uses a predefined table of probabilities. The Modified Adaptive Huffman algorithm [4] has a tree structure with leaves that represent sets of symbols with the same frequency. The data is coded traversing the tree from the root to the leaf containing the acquired data. Also, in [5], authors present a Dynamic Huffman algorithm suited for changing statistics of highly correlated data.

In order to increase the efficiency of these algorithms, prediction methods for future data were introduced before compression. In [6] the authors present a model of prediction calculated from the latest three values acquired. From these values, the algorithm calculates the lowest, the highest, and the medium value. At the end it is transmitted the difference between one of the calculated values and the actual one, depending on its position from these.

This paper presents a new compression scheme composed of an extrapolation predictor and a Static Huffman algorithm. The paper is organized as follows: Section 3 deals with the description of prediction algorithms and Huffman algorithms. In Section 4, experimental results are shown by comparing the scheme proposed in [6] with author's scheme, and, finally, in Section 5, some observations and concluding remarks are given

3. COMPRESSION SCHEMES

3.1. Prediction algorithms used for WISN

The data acquired with a sensor can be similar to a signal sampled with a frequency proportional with the acquisition step. The samples are quantized with a rate depending on the resolution of the analog to digital converter. In many cases, the values of the monitored parameter are correlated in time with a certain degree. This way, the future acquired value, at the time t+1, can be predicted from the previous ones with an error:

$$e(t+1)=s(t+1)-\hat{s}(t+1), \quad (1)$$

where s(t+1) represents the measured value at the time t+1 and $\hat{s}(t+1)$ represents the predicted value for the time t+1.

Depending on the type of application, different prediction algorithms that have to take into account the limited processing capability of the microcontroller can be used in WISN.

3.1.1. Differential prediction

The differential prediction algorithm [2], [4] uses a linear technique. For the estimation of the sample at the time t+1, $\hat{s}(t+1)$, the value of the previous sample, s(t), is used.

$$\hat{s}(t+1)= s(t) \quad (2)$$

The transmitter forwards the value of the difference between the measured value and the predicted value. With the help of this difference, the receptor calculates the actual measured value. This way it is transmitted only a difference between two consecutive data that can be compressed with a much smaller number of bits. This method has provided very good results in highly correlated data.

3.1.2. Median prediction

In [6], the authors present a model of prediction calculated from the latest three values acquired. For these values, the algorithm calculates the lowest, l, the highest, h, and the medium value, m. Depending on the position of the acquired value, x, from the ones calculated before, the transmitted data is the difference between x and l, if $x<l$, h, if $x>h$ or m, if $l\leq m\leq h$.

For decoding, in order to know which difference to choose, two bits of control are introduced by the coder at each transmission.

3.1.3. Prediction by extrapolation

In the method described before, the algorithm calculates a difference from one of three parameters obtained from the previous three values acquired. The main disadvantage is that at each transmission, two extra bits of control are transmitted.

In general, in monitoring applications, the values acquired are highly correlated in time and the next data could be predicted from the previous ones. To do this, we propose an extrapolation prediction technique that estimates the next value from the previous three values acquired.

Due to time correlation, for a small number of acquisition steps, the shape of the diagram could be approximated on a line. The predicted value can be extracted from this line at the next acquisition step.

Due to the limited processing capabilities of the sensor, the line can be better approximated from three previous values: v_{n-2}, v_{n-1}, v_n. If we consider two values, the error could be very significant.

To obtain the predicted value, $\hat{v}_{n+1}$, the algorithm uses the following equation:

$$\hat{v}_{n+1} = -\frac{3}{4}v_{n-2} + \frac{1}{2}v_{n-1} + \frac{5}{4}v_n \quad (3)$$

This represents the equation of the line that crosses through the point representing the median value of v_{n-2} and v_{n-1}, and the point that represents the median value of v_{n-1} and v_n.

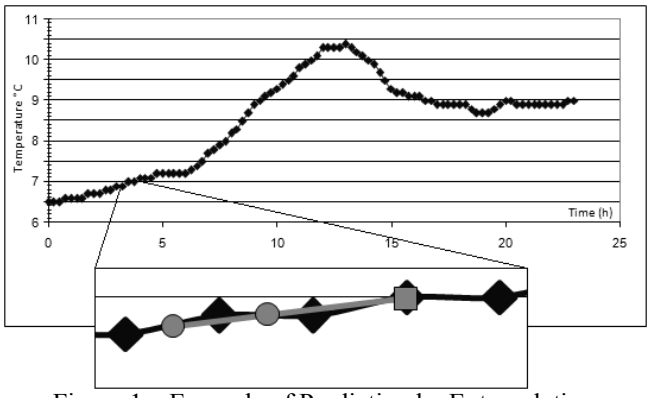

Figure 1 – Example of Prediction by Extrapolation for a temperature acquisition

Figure 1 depicts an example of prediction by extrapolation. The diamond-shaped dots represent the values of the temperature acquired in a day. The round shape dots represent the average values of two consecutive data acquired. These points give the line where the predicted value will be situated at the time t_{n+1}. Here, the predicted value is market with a square shape dot. The compression algorithm will compress the difference between the predicted value found as described, and the actual measured value.

3.2. Huffman coding algorithms suited for WISN

Huffman coding is a lossless data compression method. The basic idea of this algorithm is that the measured values that occur frequently are represented with a smaller set of bits than those which occur rarely. For that, an alphabet containing a link between bit sequences of variable sizes and measured values is mapped.

3.2.1. Simple Huffman Algorithm

Simple Huffman algorithm [3] reduces computational resources and the memory used in a WSN sensor. The algorithm performs lossless compression by encoding the data that has to be transmitted more compactly based on its statistical characteristics.

The value of a parameter measured by a sensor, m_i, is applied to an analog-to-digital converter (ADC) and transformed in a binary representation, r_i, on R bits, where R is the resolution of the ADC. In order to transmit a smaller bit stream, for each new acquisition the algorithm computes the difference between the actual value measured and the previous one:

$$d_i = r_i - r_{i-1}. \quad (4)$$

For the first acquisition, the previous value is considered to be the middle value of the 2^R possible discrete values.

The value of the difference d_i that has to be transmitted is composed from two sets of bits:

- s_i: the first set of bits codifies the number n_i of bits used to represent d_i. The value of n_i is obtained from d_i as follows:

$$n_i = \lceil log_2(|d_i|) \rceil \quad (5)$$

The exception is when d=0. In this case, n_0 is considered 0. The Huffman coding is used on n_i to generate the s_i variable-length symbols. In this case, it is considered that the occurrence probabilities of the differences decrease, as the values increase. The maximum value of n_i will be equal to R.

- a_i: the second sequence of bits is the binary representation of d_i and it can be generated in three ways:
  1. when $d_i > 0$, a_i corresponds to the n_i low-order bits of the two's complement representation of d_i;
  2. when $d_i < 0$, a_i corresponds to the n_i low-order bits of the two's complement representation of $d_i - 1$;
  3. when $d_i = 0$, a_i has no representation and s_i is coded as 00.

The bit stream formed by the concatenation of s_i and a_i is transmitted to the next hop from the network.

3.2.2. Dynamic Huffman algorithm

Like the Static Huffman algorithm, the Dynamic Huffman algorithm [5] is based on a similar dictionary but in this case, it is permanently updated with respect to the probabilities of the incoming data. At the start, the algorithm begins with a default probability-difference correspondence.

The composition of the bit stream that codifies the differences d_i is made in the same way as in simple Huffman algorithm. However, the dictionary has some changes: to code n_i=1, two bits are used instead of three. Consequently, the length of the bit sequence for n_i>4 grows with one bit, but for n_i=1, where the incoming probability of the value is much higher, decreases with one bit. The dictionary used $\mathcal{D}$ is shown in Table I. The length of the dictionary depends on the ADC resolution. In this case, a 12-bit one is considered.

TABLE I. DICTIONARY $\mathcal{D}$ USED FOR COMPRESSION

n_i	s_i	d_i
0	00	0
1	01	-1,+1
2	100	-3,-2,2,3
3	101	-7,…,-4,+4,…,7
4	110	-15,…,-8,8,…,15
5	1110	-31,…,-16,16,…,31
6	11110	-63,…,-32,32,…,63
7	111110	-127,…,-64,64,…,127
8	1111110	-255,…,-128,128,…,255
9	11111110	-511,…,-256,256,…,511
10	111111110	-1023,…,-512,512,…,1023
11	1111111110	-2047,. . .,-1024,+1024,. . .,+2047
12	11111111110	-4095,. . .,-2048,+2048,. . .,+4095

In order to introduce a permanent update of the data appearance probability, two vectors are considered:

- q[j] - includes all the possible values of the differences d_i that can be found in the dictionary;
- p[j] – includes the number of appearance of the differences from q[j], with respect to j.

When the acquisition starts, q[j] has the following representation:

$$q_{init}[j] = \{0,1,-1,2,-2,3,-3,\ldots, \frac{(j-1)+1}{2}, -\frac{j-1}{2}, \frac{j+1}{2}, -\frac{j}{2}\}, \quad (6)$$

where $j = \overline{0, 2^{R+1} - 2}$.

With the data arrival, the position of the differences from the vector chances considering the numbers of previous incomings. When a difference is calculated, it is retained its position, j, from q[j]. With the help of this, the data that was in that position at initialization is obtained. That data is compressed and transmitted to the receiver. The values of the $q_{init}[j]$ can be easily obtained from j as follows:

- if j=0, q[0]=0;
- if j is even, $q[j] = -\frac{j}{2}$; (7)
- if j is odd, $q[j] = \frac{j+1}{2}$. (8)

At the receiver, the data is decoded and the position in the vector where the value of the desired difference is held is found. This way, the length of d_i code words is permanently changed based on the incoming values, and the differences with high probability are coded on low sizes of bits.

3.3. Huffman compression schemes suited for WSN

In order to obtain better results and to transmit a smaller sequence of bits on a sensor from WSN a compression scheme derived by combining a prediction algorithm with a Huffman compression algorithm can be implemented. Further, are presented two existing compression scheme and one proposed by the authors.

3.3.1. The Median Predictor based Data Compression (MPDC)

As described in [6], this compression scheme consists of three steps:

1. Selector: Three previous values are selected together with the current value.
2. Median Predictor: It calculates the median, the lowest and highest values among three previous values selected. Then, depending on the position of the current value from these, it calculates a difference.
3. Huffman Coder: The Static Huffman algorithm is applied on the difference calculated at the previous step.

3.3.2. Differential Predictor & Dynamic Huffman algorithm (DPDH)

In DPDH the Dynamic Huffman algorithm [5] is used. The prediction algorithm considers that the next acquired value is equal to the previous one. It computes the difference between them and the result is then sent to the compression algorithm. After compression, the data is transmitted to the next hop.

3.3.3. Extrapolation Predictor & Static Huffman algorithm (EPSH)

As shown in Figure 2, this scheme combines the prediction by extrapolation algorithm and the Static Huffman algorithm.

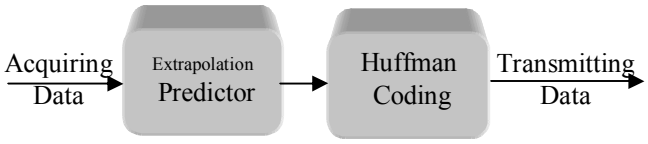

Figure 2. Extrapolation Predictor & Static Huffman algorithm scheme

In the extrapolation predictor block it is calculated the difference between the acquired data and the data obtained through extrapolation from three previous ones. The value obtained is compressed with Static Huffman algorithm, but based on the dictionary described in table I.

4. EXPERIMENTAL RESULTS

The scheme presented can be implemented on sensors in a WISN. Some experiments were made on a Visual Basic platform where the processing part of a sensor was simulated. The performances of the schemes were analyzed in number of bits required to transmit the acquired data and compression ratio. During simulation, the attention was focused only on the bits required to compress the data. The bits used for encapsulation were not considered.

Two sets of data were considered, representing the temperature values collected during two whole days (1 Jan. 2012 and 2 Jan. 2012). The data was taken from the meteorological station Băneasa-Bucharest [7]. Considering the acquisition step of 10 minutes, in a day the sensor should acquire 144 values. With a resolution of 0.1ºC and a range of possible values between -25 ºC and 35 ºC, the uncompressed data can be coded to a minimum of 10 bits. Thus, the total amount of bits for transmitting 144 uncompressed values is of 1440 bits.

In Figure 3 it can be seen the results obtained by the three compression schemes described. In light grey it can be seen the number of bits transmitted for the data acquired on 1 Jan. 2012 and in dark grey it can be seen the number of bits transmitted for the data acquired on 2 Jan. 2012. Also it is shown a comparison with the number of bits transmitted for uncompressed data.

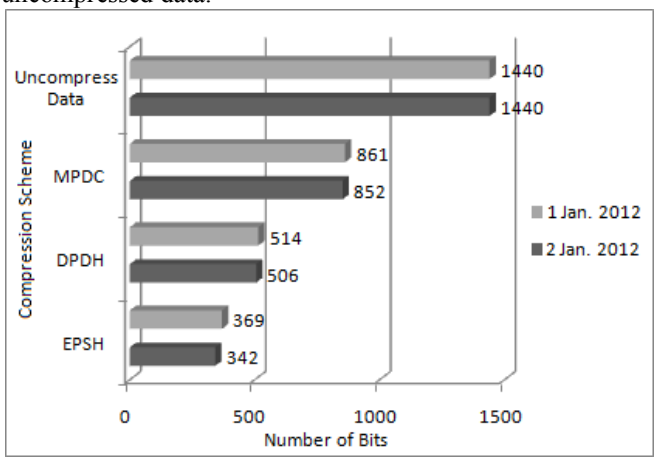

Figure 3. Number of bits transmitted for set 3 of data.

From the data obtained we can observe that EPSH scheme obtained the better results, followed by DPDH scheme and MPDC scheme. With EPSH had been send approximately a quarter from the bits transmitted without compression.

From these results it can be obtained the compression ratio of each scheme. This could be calculated with the formula presented in [3]:

$$Cr = 100\left(1 - \frac{Cs}{Os}\right) \quad (9)$$

where *Cr* is the compression ratio, *Cs* is the compressed size and *Os* represents the original size.

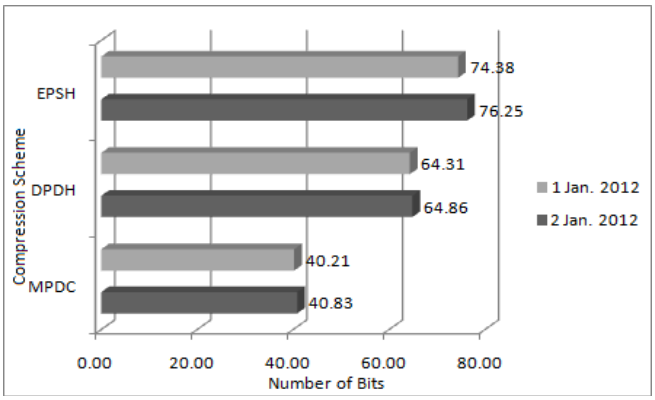

Figure 4. Compression ratio

As it can be observed in Figure 4, in both cases of data acquisition, the EPSH scheme obtained the best result with a compression ratio of approximately 75%. The DPDH scheme recorded a compression ratio of approximately 65 % while MPDC only 40%.

If we consider n acquisitions, as n grows large, we can state that the complexity for all three presented algorithms, on acquisition level, is O(n), in Big O notation. But the time used to complete a transmission differs from an algorithm to another. Strictly concerning the prediction method, at each step, EPSH makes 6 operations, DPDH makes one, while MPDC has a variable number of operations since it has to calculate the minimum, the median and the maximum value from the three previous ones, and then compare the measured value with all of these in order to compute the difference. In the best case scenario, this number exceeds 12 operations. Regarding the Huffman compression algorithm the number of operations increases significantly. While EPSH and MPDC use the same algorithm with a slight difference on the dictionary, the computational and memory needs for DPDH rises in order to adapt the statistics. In DPDH a procedure is introduced that, in the worst case, triples the used memory to introduce the vector with the updated order of data and the vector including the statistics, thus increasing the number of operations by a 2^{R+1} factor, where R is the resolution of the analog-to-digital converter. In the best case scenario, where the initial statistics do not change, the number of operations of DPSH approaches the one for EPSH. Thereby, considering the prediction method and the compression algorithm, in the best case scenario for MPDC and DPDH, the number of operations is higher than the number of operations used by EPSH. This means that less time is spent on processing, leading to less time spent by the microcontroller in awake state, in turn, leading to prolonging the lifetime of WISN.

4. CONCLUSIONS

In WISN, great attention has to be paid to energy consumption. In order to preserve power, the nodes have to transmit the data in a compact form. The solution is to process and compress the data acquired on a smaller number of bits. Consequently, prediction algorithms, compression algorithms or compression schemes that combine these two solutions were introduced.

In this article, the authors propose a compression scheme obtained by combining an Extrapolation Prediction algorithm and a Simple Huffman compression algorithm. Experimental results obtained reveal an improvement in the reduction of the number of bits transmitted comparing with other existing compression schemes.

The number of instructions executed by the microprocessor to implement EPSH is under one hundred. In terms of cost of energy, that means less than transmitting one bit and the result is a compression ratio of about 75%. These results highlight the fact that the EPSH scheme can be a suitable compression scheme for WISN.

5. ACKNOWLEDGEMENTS

The work has been funded by the Sectoral Operational Programme Human Resources Development 2007-2013 of the Romanian Ministry of Labour, Family and Social Protection through the Financial Agreement POSDRU/88/1.5/S/60203 and by the FWO Flanders projects G.0391.07 and G.0146.10.

6. REFERENCES

[1] C. M. Sadler, and M. Martonosi, "Data compression algorithms for energy-constrained devices in delay tolerant networks", *Proc. SenSys: 4th Int. Conference on Embedded networked sensor systems*, pp. 265–278, 2006

[2] K. C. Barr, and K. Asanovic, "Energy-aware lossless data compression", *ACM Transactions on Computer Systems*, Vol. 24, No. 3, Pages 250-291, August 2006

[3] F. Marcelloni, and M. Vecchio, "A Simple Algorithm for Data Compression in Wireless Sensor Networks", *IEEE Communications Letters*, Vol. 12, No. 6, pp. 411-413, June 2008

[4] C. Tharini, and P. Vanaja Ranjan, "Design of Modified Adaptive Huffman data compression algorithm for Wireless Sensor Network", *Journal of Computer Science*, Science Publications, Vol. 5, pp. 466-470, 2009

[5] D.I. Sacaleanu, R. Stoian, D.M. Ofrim, "An Adaptive Huffman Algorithm for Data Compression in Wireless Sensor Networks", *10th International Symposium on Signals*, Circuits and Systems (ISSCS), 2011 June 30 2011 - July 1 2011

[6] Ashish K. Maurya, Dinesh Singh, Anil K. Sarje, "Median Predictor based Data Compression Algorithm for Wireless Sensor Network", *International Journal of Smart Sensors and Ad Hoc Networks*, Volume-1, Issue-1, pp. 62-65, 2011

[7] http://www.meteoromania.ro/