A Survey on Data Compression in Wireless Sensor Networks

Naoto Kimura and Shahram Latifi
Electrical and Computer Engineering, University of Nevada, Las Vegas
{naoto, latifi}@egr.unlv.edu

Abstract

Wireless sensor networks (WSNs) are resource constraint: limited power supply, bandwidth for communication, processing speed, and memory space. One possible way of achieve maximum utilization of those resource is applying data compression on sensor data. Usually, processing data consumes much less power than transmitting data in wireless medium, so it is effective to apply data compression before transmitting data for reducing total power consumption by a sensor node. However, existing compression algorithms are not applicable for sensor nodes because of their limited resource. Therefore, some of compression algorithms, which have been specifically designed for WSNs, are presented in this paper: Coding by Ordering, Pipelined In-Network Compression, low-complexity video compression, and Distributed Compression.

1. Introduction

Because of rapid advancement of semiconductor fabrication technology, electronic devices have been becoming smaller, cheaper, and requiring less power for its operation every year. One of the fields, which have benefited by the technological advancement, is the field of WSNs. Recently, a lot of researched related WSNs have been conducted and people have been realizing its unlimited applicability. For example, WSNs can be used for data collection purposes in situations such that environmental monitoring, habitat monitoring, surveillance, structural monitoring, equipment diagnostics, disaster management, and emergency response [2]. However, the technology of WSNs is still relatively premature, and it will require more extensive research and development before WSNs become practical.

The basic wireless sensor network configuration [1] is that a large number of sensor nodes are densely deployed over a sensor field. All nodes are connected by radio frequency, infrared, or other medium without any wire connection. The data collected by nodes traverses among the nodes in wireless medium. In order to realize WSNs, peer-to-peer network techniques are widely used so that it allows direct communication between any two nodes. If two devices cannot communicate directly, other nodes, located between those two nodes, will transmit a data packet from the source node to the destination node. This is called multi-hop routing. Because of their peer-to-peer communication style, no centralized point, which will control a network formation like a base station for a cellular system, is required for the network. Since no fixed infrastructure is necessary for WSNs, a network will be constructed inexpensively. Also, nodes can be added to and removed form the network easily. On the other hand, the network topology in a WSN may change drastically since nodes can be added and removed easily. The data from sensor nodes is gathered by the sink. The sink may connected to the outside world thorough Internet or satellite. Sensor nodes will be scatted over a sensor field, so the locations of sensor nodes in the field cannot be predetermined. A sensor, discussed in this paper, will be equipped with at least a power supply, sensing unit, processing unit, in which the data from the sensor unit is processed, and transmitter-receiver unit.

As it was mentioned, the size of each sensor node is expected to be small. To achieve its requirement, the size of every sensor components, such as the power source and processing and data storing memory, also has to be small. Also, a large number of sensor nodes will be often deployed to the location, where it is hard to access. It is not practical to perform maintenance operations, such as changing batteries, on deployed sensor nodes. Because of above reasons, WSNs, or more specifically each sensor node, are resource constrained. They have limited power supply,

Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'05)
0-7695-2315-3/05 $ 20.00 **IEEE**

bandwidth for communication, processing speed, and memory space. Therefore, many of studies have been conducted so far focuses on how to achieve the maximum utilization of limited sensor resource.

One field of resource utilization studies for sensor networks is data compression. Researchers seek the optimal way to compress the sensing data. By doing so, it will reduce the power consumption due to processing and transmitting data in each nodes, and thus extend the life time of sensor network. Also, by reducing data size less bandwidth is required for sending and receiving data. The data compression is one effective method to utilize limited resources of WSNs.

The remainder of this paper is organized as the following way. In the following section, I will discuss about the energy consumption of data compression and data transmission in wireless medium. Some of data compression schemes specifically designed for WSNs is presented in Section 3. Finally, Section 4 presents conclusion.

2. Energy Consumption Analysis in Wireless Medium

In terms of power consumption, operation of a wireless sensor node can be divided into three parts: sensing, processing, and transmission. Among those three operations, it is known that the most power consuming task is data transmission. Approximately, 80% of power consumed in each sensor node is used for data transmission. Thus, if we can minimize the size of data by compression, it will reduce transmission power. However, on the other hand by applying data compression more power for processing will be required to perform a compression algorithm. In order to reduce total power consumption, the sum of power consumed by transmission and processing has to be reduced. The power consumed by compressing "a" bits data string into "b" bits data string, where a > b, has to be smaller than the power consumed by transmitting "a – b" bits of data string.

In [5], an extensive research has been conducted on power consumption by data compression and data transmission in wireless communication. In their experiments, the Compaq Personal Server, which is research version of the Compaq iPAQ, was used for data collection instead of wireless sensor nodes, so their experimental results may not be totally applicable for WSNs. However, it still gives significant insights of power consumption in data processing and transmission.

The first experiment was collecting the data of power consumption by performing a simple 32-bit addition instruction and transmitting one bit. The result shows that approximately 0.4 μJ of energy was consumed by sending a single bit of data. On the centrally, 0.86 nJ of energy was consumed by executing an addition instruction. Transmitting one bit via radio medium is at least as 480 times power consuming as performing one addition instruction. The result indicates that if more than one bit is removed from an original data bit string by applying a compressing operation, which takes instructions equivalent to 480 addition instructions, than it will reduce total power consumption by a sensor node.

The actual data compression algorithms are far more complicated than series of add instructions, so the next experiment show total energy consumption by applying various existing lossless data compression on text and web page data. The data compression algorithms tested in this experiment were bzip2 (BWT algorithm), compress (LZE algorithm), LZO (LZ77), PPMd (PPM), and zlib (LZ77). Table 1 shows the number of instruction required by removing a single bit. The number of instructions for each data compression algorithms is much smaller than 480 instructions, so when those algorithms are applied, it is expected to consume less total energy than simply transmitting a data string. According to the experimental results, for most compression algorithms, compressing data before transmission reduces total power consumed. However, in some cases applying data compression increases total power consumption. This is due to accessing memory during compression execution time. Accessing memory is expensive in terms of energy consumption. Therefore, this experimental result indicates that applying the data compression before transmitting data in wireless medium is effective to reduce the amount of energy consumption. However, it is crucial to select a data compression algorithm, which requires less memory access during execution time.

	bzip2	compress	Lzo	PPMd	zlib
Compression Text Data (instructions/bit)	116	10	7	76	74
Compression Web Data (instructions/bit)	284	9	2	60	23

Table 1. Required Instruction per Removed Bit

Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'05)
0-7695-2315-3/05 $ 20.00 **IEEE**

3. Data Compression Techniques

The studies in the previous section illustrate that sending data is more power consuming than computation, and thus minimize data size before transmitting in wireless medium is effective to reduce total power consumption. Therefore, it is beneficial for WSNs to employ a data compression algorithm.

One obstruction is that most of existing data compression algorithms is not feasible for WSNs. One reason is the size of algorithms. For example, the size of bzip2 is 219 KB and the size of LZO is 220 KB. The instruction memory size of sensor node currently available is only 128 MB [10]. Another reason is the processor speed. The processing speed of sensor node is only 4 MHz [10]. On the other hand, a processing speed of PC, nowadays, is around 3 GHz. Thus, it is necessary to design a low-complexity and small size data compression algorithm for sensor networks. In this section, some of data compression schemes for WSNs are introduced.

3.1 Coding by Ordering

The Coding by Ordering data compression scheme is introduced in [7] as part of Data Funneling Routing. The compression scheme is works as follow. First, a data pass from sensor nodes in the interested region to a collector node is set up as shown in Figure 1. In Data Funneling Routing, some of sensor nodes work as a data aggregation node. For example, node A, B, and D are a data aggregation node. At an aggregation node, sensing data collected by other nodes is combined, and the aggregated data is sent to its parent node. At node D in Figure 3, data collected by node E is combined with data collected by node D itself. Then, the aggregated data is transmitted to node B.

In the algorithm, when data is combined at an aggregation node, some data is dropped. To include the information of dropped data in the aggregated data, the order of data packet is utilized. For example, four nodes (N1, N2, N3, and N4) send the data to an aggregation node (Na). The data value of each node can be any integer ranging from 0 to 5. If we decide to drop the data from N4 and express the data from N4 by ordering packets from other 3 nodes (N1, N2, and N3), there are 3! = 6 possible ordering. Therefore, by using permutation of three packets, the data value of N4 can be included in an aggregated packed without actually including the packet of N4. The possible combination of permutation and data value is presented in Table 2.

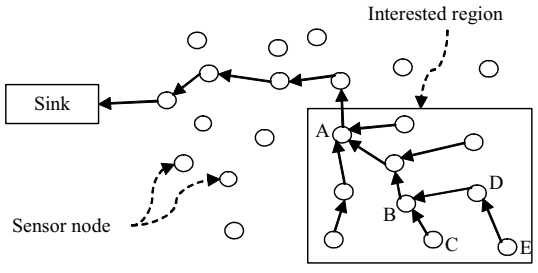

Figure 1. Data Path

Packet Permutaion	Integer Value
N1,N2,N3	0
N1,N3,N2	1
N2,N1,N3	2
N2,N3,N1	3
N3,N1,N2	4
N3,N2,N1	5

Table 2. Permutation and its Represented Integer Value

For a general case, let's assume that n is the total number of sensor nodes – each node has different a node ID, m is the number of nodes sending a packet to an aggregation node, k is the possible range of data value, and l is the number of sensor node dropped at the aggregation node. Then, the number of possible combination of IDs, which dropped nodes have, can be expressed as $\binom{n-m+l}{l}$. Since each of l nodes can take any value among possible k data values, there are k^l possible data value combinations. When combining possible IDs and data values, there is total of $\binom{n-m+l}{l}k^l$ possible values. This combination of values needs to be expressed by $(m-l)!$ permutations. Therefore, the following inequality has to be satisfied.

$$(m-l) \geq \binom{n-m+l}{l}k^l \qquad (1)$$

Theoretically, when $n = 2^7$, $k = 2^4$, and $m = 100$, approximately 44% of packets can be dropped at the aggregation node by applying Coding by Ordering. Since this method has good compression ratio and simple algorithm, it may be possible to use for WSNs. One difficulty of utilizing this scheme is that since there is no efficient algorithm mapping permutation to data value, it requires a mapping table. As the number of sensor nodes aggregated increases, the size of table increases exponentially.

Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'05)
0-7695-2315-3/05 $ 20.00 **IEEE**

3.2 Pipelined In-Network Compression

The pipelined in-network compression scheme is discussed in [6]. The basic idea is trading high data transmission latency for low transmission energy consumption. Collected sensor data is stored in an aggregation node's buffer for some duration of time. During that time, data packets are combined into one packet, and redundancies in data packets, will be removed to minimize data transmission.

For example, each data packet has the following form: <measured value, node ID, timestamp>. Then, the compressed data packet has the following form: <shared prefix, suffix list, node ID list, timestamp list>. The "shared prefix" is the most significant bits, which all measured values in combined data packets have in common. The length of shared prefix can be changed by a user based on the knowledge of data similarity. If the measured values are expected be close to each other, the length of prefix value can be set to relatively long. The "suffix list" is the list of measured values excluding the shared prefix part. The "node ID list" is the list of node identifiers and the "timestamp list" is the list of timestamp. The compression scheme is illustrated in Figure 4. In the figure, three nodes send the data packets to the compression node. At the compression node, three data packets are compressed into one packet. In this example, the length of shared prefix is set to 3. In this example, total number of bits is reduced from 33 to 27.

One advantage of this simple compression scheme is that the shared prefix system can be used for node IDs and timestamps. By doing so, more data compression can be achieved. The efficiency of data compression depends on the length of shared prefix. If we could set a long shared prefix and measured values have commonality, the compression ratio increases. However, there is no similarity in measured sensor values. Even if we could set a long shared prefix, the efficiency of Pipelined In-Network Compression will decrease. In addition, if we are combining a large amount of data packets, than a large data buffer is required to temporary store those packets. Since a sensor node has only a limited size of memory space, enough buffer space will not be available.

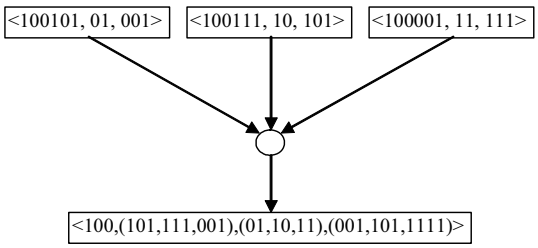

Figure 2. In-network Compression

3.3 Low-Complexity Video Compression

The low complexity video compression scheme is introduce in [8]. Since the current video encoding technologies are mostly designed on utilizing the motion estimation and compensation, it will require a high computation power, which sensor nodes are not usually equipped with. Thus, this proposed method is based on block changing detection algorithm and JPEG data compression. Figure 3 presents a block diagram of image data procession flow. This algorithm is specifically designed for the wireless video surveillance system. In this method, each video frame is divided into small blocks so that each block contains 8×8 (64) pixels. To reduce the computational complexity, only the subset of blocks (all white blocks in this case) in each frame is considered. Furthermore, in each block the subset of pixels (number assigned pixels) is examined their changes as seen in Figure 7. The number assigned to pixels indicates importance of pixels (1 = most important and 3 = least important).

The algorithm works as follow. In the first step, for one block "1" assigned pixels are compared with the referenced frame's pixels. If the difference of two pixels (D_i) is greater than a threshold value (M), a counter (P), which is set to zero initially, is incremented by one. The maximum difference between two pixels is stored as U. Then, if P becomes greater than a sensitivity parameter (S), which will be set by a user and can take an integer from 1 to 3, the block marked as an active block, and the scanning procedure moves to a next block. If $P \leq S$ after checking all "1" pixels, the

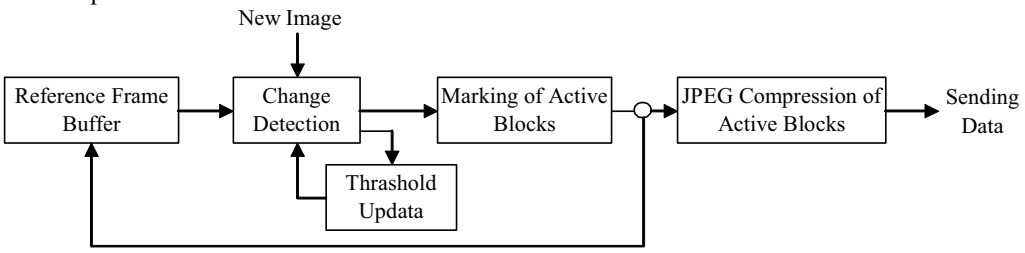

Figure 3. Image Data Processing Flow

Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'05)
0-7695-2315-3/05 $ 20.00 **IEEE**

average difference (A), where $A = \sum_{i=1}^{n} D_i \Big/ n$, is calculated. Then, if A<N and U<M, where N is another threshold value and N<M, a next block is processed. Otherwise, the algorithm starts checking "2" assigned pixels in the same block.

For "2" and "3" assigned pixels in the same block, the same procedure as for "1" assigned pixels is performed. The difference is that After checking all pixels, if A>N and U>M, the block is marked as an active block and the procedure moves to a next block for "2" assigned pixels. For "3" assigned pixels, after checking all pixels, if (A>N and U>M) or (A<N, U>M, and P>S), then the block is marked as active.

After checking a set of white blocks in one frame, non-scanned blocks (black blocks in Figure 7) are checked. If a non-scanned block is adjacent to at least two active blocks, the non-scanned block is marked as active.

Finally, all marked blocks are encoded by JPEG and transmitted in wireless sensor network. Also, the active blocks are fed to the Reference Frame Buffer to up-data the reference frame.

The experimental results indicate that the quality of image processed by this algorithm is quite similar to the image processed by MPEG-2 while the proposed algorithm achieves the certain amount of energy saving.

3.4 Distributed Compression

The basic idea behind the Distributed Compression scheme, introduce by [9] [4] is using a side information to encode a source information. For instance, there exist two sources (X and Y) as shown in Figure 4. They are correlated and discrete-alphabet independent identically distributed. Since in a sensor network, sensor nodes will be densely populated in a sensor field, this correlation condition can be satisfied easily. Then, X can be compressed at the theoretical rate of its conditional entropy, H(X|Y), without the encoder 1 accessing Y [3]. The conditional entropy can be expressed as

$$H(X \mid Y) = -\sum_{y} P_Y(y) \sum_{x} P_X(x \mid y) \log_2 P_X(x \mid y) \quad (8)$$

The general scheme of Distributed Compression is first to compose cosets, whose codevectors of source X. The distance of any two codevectors in the same coset has to be large enough. An index value is assigned to each coset. When transmitting data to a decoder, the source X only sends an index value of coset, to which the codevector belongs. The source Y sends a codevector as a side-information. The decoder looks up the coset, which has the same index received from X.

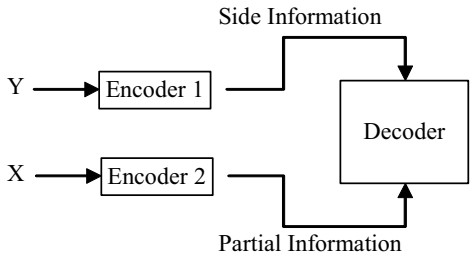

Figure 4. Distributed Compression Example

Then, the decoder selects one codevector, which has a closest value to the codevector sent by Y, in the coset.

A simple example of Distributed Compression is as follow. There are two (X and Y) 3-bit data sets. The Hamming distance between X and Y is no more than one bit. If both Encoder 2 and Decoder know Y, X can be compressed to 2 bits. Then if Y is only known by the Decoder, what will happen to the compression rate of X? According to the Distributed Compression scheme, since the Decoder know Y and X is only one Hamming distant apart from Y, it is not efficient to distinguish X=111 from X=000. As a same reason, X=001 and 110, X=010 and 101, and X=011 and 110 do not need to be distinguished from each other. These sets of two X values are grouped as 4 costes and assigned 4 different binary index numbers:

coset 1 = (000, 111) : 00
coset 2 = (001, 110) : 01
coset 3 = (010, 101) : 10
coset 4 = (011, 100) : 11

If X=010 and Y=110, the Decoder received Y=110 as a side information from Y and X=10 as a partial information form X. Then, at the Decoder, X=010 is selected from coset 2 since 110 has a Hamming distance of 2 from 110. Whether X know Y or not, X can still compress 3 bits information into 2 bits.

Distributed Compression scheme can be applied to not only discrete sources as illustrated by the above example, but also continuous sources. Also, it can be used for both lossless and lossy compression scheme. For example, 4 cosets can be formed from an 8-level quantizer. It is noticed that two codevectors in each coset are grouped so that they can have the maximum possible distance from each other.

4. Conclusion

In these days, people are discussing wide range of application areas for WSNs. In the future as technology progresses the application areas of WSNs will become much broader than now. They will become more available for people than right now. However, before

Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'05)
0-7695-2315-3/05 $ 20.00 **IEEE**

those days come there still exist many obstacles to overcome for practical use of sensor networks. One of the obstacles is limited resource of wireless nodes. In this paper, five different types of data compression schemes – Coding by Ordering, Pipelined In-Network Compression, JPEG200, low-complexity video compression, and Distributed Compression – were presented. Even though those compression schemes are still under development, experimental results indicate that their compression rate and power reduction manners are quite impressive. They are one possible method to diminish resource constrain of wireless sensor nodes.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," IEEE Communications Magazine, Volume: 40 Issue: 8, pp. 102-114, August 2002

[2] D. Culler, D. Estrin, and M. Srivastava, "Guest Editors' Introduction: Overview of Sensor Network," Computer, Volume: 37 Issue: 8, pp. 41-49, August 2004

[3] D. Slepian and J. K. Wolf, "Noiseless Coding of Correlated Information Sources," IEEE Trans. on Information Theory, Volume: IT-19, pp. 471-480, July 1973.

[4] J. Kusuma, L. Doherty, and K. Ramchandran, "Distributed Compression for Sensor Networks," In Proceedings of 2001 International Conference on Image Processing, October 2001.

[5] Kenneth Barr and Krste Asanovic, "Energy Aware Lossless Data Compression," In First International Conference on Mobile Systems, Applications, and Services, May 2003.

[6] T. Arici, B. Gedik, Y. Altunbasak, and L. Liu, "PINCO: a Pipelined In-Network Compression Scheme for Data Collection in Wireless Sensor Networks," In Proceedings of 12th International Conference on Computer Communications and Networks, October 2003.

[7] D. Petrovic, R. C. Shah, K. Ramchandran, and J. Rabaey, "Data Funneling: Routing with Aggregation and Compression for Wireless Sensor Networks," In Proceedings of First IEEE International Workshop on Sensor Network Protocols and Applications, May 2003.

[8] E. Magli, M. Mancin, and L Merello, "Low-Complexity Video Compression for Wireless Sensor Networks," In Proceedings of 2003 International Conference on Multimedia and Expo, July 2003.

[9] S. S. Pradhan, J. Kusuma, and K Ramchandran, "Distributed Compression in a Dense Microsensor Network," IEEE Signal Processing Magazine, Volume: 19, Issue: 2, pp. 51-60, March 2002.

[10] MICA Wireless Measurement System http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICA.pdf