

Multiple-Input Turbo Code for Joint Data Aggregation, Source and Channel Coding in Wireless Sensor Networks

Hasan Çam

Computer Science and Engineering Department
Arizona State University
Tempe, AZ 85287
hasan.cam@asu.edu

Abstract—In wireless sensor networks, the data collected by neighboring sensor nodes show high correlation. To exploit the data correlation for reducing the amount of data while transmitting data from sensor nodes to their base station, this paper introduces multiple-input turbo (MIT) code to implement a joint source coding and data aggregation. If there exists an explicit communication between two sensor nodes, their correlated data bits are first interleaved with each other in accordance with their correlation relation, and then are encoded and punctured. On the other hand, if there is no direct communication between sensor nodes, the data sequences are encoded with respect to side information based on the distributed source coding principles. Whenever the bit error rate needs to be improved, MIT code is used for channel coding as well. MIT code employs partial interleaving to reduce energy consumption and memory size requirements for even small-size information blocks. The simulation results show that the bit error rate performance of MIT code outperforms turbo codes to some extent, even if MIT code implements partial interleaving on interleavers.

Key words. Sensor networks, source and channel coding, data aggregation, turbo code, correlated data, partial interleaver.

I. INTRODUCTION

Recent advances in wireless networking and microelectronics have made it possible to deploy large number of sensor nodes in wireless sensor networks. Sensor nodes collect information from an area of interest, and have the capability of sensing, processing, and communication. The fact that sensor nodes are usually equipped with small batteries makes energy efficiency a major concern. Because most of the energy consumption occurs due to data transmission, it is very critical to reduce the amount of transmitted data using data aggregation and source coding.

Sensor nodes often have overlapping sensing ranges and detect common phenomena and, therefore, many of them report correlated data [1], [2], [3]. Indeed, it is likely that many intermediate sensor nodes or cluster-heads receive the same or similar data from different sensor nodes. Hence, sensor readings are often highly correlated. Data aggregation, also known as data fusion, is implemented to reduce data redundancy and/or to combine several unreliable data measurements to produce more accurate signal by enhancing the common signal and reducing the uncorrelated noise [4], [5]. For instance, data

may be aggregated using functions to compute averages, sums, median values, minimum and maximum values [6].

Source coding (i.e., compression or efficient representation of data) can be implemented to improve bandwidth and energy efficiency. For instance, when sensor nodes sense a common phenomena independently or have overlapping sensing ranges, the measurements of especially neighboring sensor nodes are often correlated. The objective of source coding is to avoid the transmission of any redundant information with minimal or no communication among sensor nodes by exploiting their data correlation.

In addition to data aggregation and source coding, channel coding should be used whenever a channel needs to be made more reliable. For a given spectral efficiency, the largest coding gain is achieved by turbo code [7], [8] in wireless systems, and turbo code achieves an exceptionally low bit error rate (BER) with a signal-to-noise ratio (SNR) per information bit (E_b/N_o) close to Shannon's theoretical limit on a Gaussian channel. As illustrated in Fig. 1, turbo code consists of two recursive systematic convolutional encoders in parallel, separated by an interleaver. The interleaver of turbo code aims

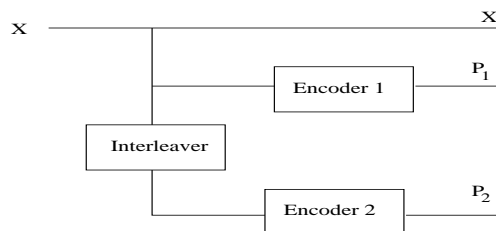


Fig. 1. Turbo code.

at: (i) improving the code strength by increasing the weight of codewords, and (ii) spreading the outputs from one decoder to provide the other decoders with loosely correlated inputs [9]. Since the coding performance of turbo code depends directly on the generation of coded sequences by the constituent encoders, the design of turbo interleaver is critical to help encoder generate a codeword with more weight [10]. But, the design of interleaver has a tradeoff between large size for high performance and small size for low latency and

small memory space because a large interleaver size results in better interleaver gain, but also leads to higher latency, larger memory size, and more processing and energy requirements. Turbo code is also suggested for distributed source coding [11] and joint source-channel decoding [12], [13]. The memory space requirements, decoding complexity [14] and energy consumption of turbo code should be reduced to make it suitable for wireless devices with limited battery power and memory space.

This paper introduces MIT code to take advantage of the correlation among the data of neighboring sensor nodes for implementing an integrated source coding, data aggregation, and channel coding over wireless sensor networks. The fact that two input sequences are interleaved in MIT code makes it possible to use small-size partial interleavers, leading to energy efficiency. To the best of our knowledge, this paper is the first to implement a joint data aggregation and source coding at the same time. Because interleaving two input data sequences enhances the randomness of data sequences entering convolution encoders, MIT code is able to employ partial interleavers to reduce energy consumption and memory size constraints for even small-size blocks that are typical in wireless sensor networks. The paper also shows how to employ MIT code for channel coding along with a joint source coding and data aggregation.

The remainder of the paper is organized as follows. Section 2 introduces MIT code. Section 3 describes how MIT code can be used for joint source coding and data aggregation. Section 4 presents the implementation of joint source-channel coding. The decoder implementation of MIT code is addressed in Section 5. Section 6 presents the simulation results, and the concluding remarks are made in Section 7.

II. MIT CODE

The proposed MIT code employs partial interleavers to reduce the complexity and to deal with multiple correlated-data inputs. Fig. 2 illustrates MIT code, where inputs X and Y represent either the correlated data of sensor nodes X and Y , respectively, or different blocks X and Y of the same input sequence. Because each of the four switches S_1 , S_2 , S_3 , and S_4 can be open (i.e., OFF) or closed (i.e., ON) independently of the other switches, the encoder structure of MIT code can have sixteen cases. Indeed, each case can be even divided into subcases since encoders can have variable-length encodings and partial interleavers can have variable sizes depending on the availability of the amount of correlated data or the number of input sources. Switches S_1 , S_2 , S_3 , and S_4 may be chosen to be closed in at least two cases: (i) X and Y represent different blocks of the same input sequence, and (ii) a sensor node has some information about the correlated data of another sensor node in case of wireless sensor networks. There is a close relationship between the type of partial permutations implemented at partial interleavers and the degree of correlation between X and Y . Note that MIT code can also implement turbo code, if needed, by setting switches properly to ON and OFF modes. Since the coding rate of MIT code

is $1/2$ compared to the coding rate $1/3$ of turbo code (without puncturing), MIT code has better throughput than turbo code. Partial interleavers have less memory storage requirements and possibly less latency. Since an encoder input is determined by interleaving two different two input sequences, the information block size does not have to be large in order to achieve low BER. In general, the packet size in wireless sensor networks is small such as 32 bytes, which requires small interleaver sizes causing the data randomness to become poor. But, a small-size interleaver leads to less decoding latency and power consumption because the complexity of turbo code design increases as the interleaver size increases. Because MIT code has two inputs, its hardware complexity should be compared with that of two parallel turbo codes each having one input. The parallel-to-serial and serial-to-parallel converters of MIT code are very simple combinational circuits.

III. JOINT SOURCE CODING AND DATA AGGREGATION

The key challenge in sensor networks is to maximize the lifetime of sensor nodes due to the fact that it is not feasible to replace the batteries of thousands of sensor nodes. Therefore, the major constraint to individual sensor node performance is energy, which is consumed primarily by sensing and communication operations. When sensor nodes sense a common phenomena independently or have overlapping sensing ranges, the measurements of especially neighboring sensor nodes are often correlated. The objective of source coding and data aggregation is to improve bandwidth and energy efficiency by avoiding the transmission of any redundant information among sensor nodes. In this section, we assume that neighboring sensor nodes A , B , and C want to send their data to a data aggregator D (or clusterhead). When sensor nodes A and B have explicit communication (i.e., direct communication), it is assumed that node A first gets the data of node B and then implements joint source-coding and data aggregation on their data before transmitting to the data aggregator.

A. Explicit Communication Between Sensor Nodes

The data of inputs X and Y of MIT code are correlated to each other because they belong to two neighboring sensor nodes A and B that directly communicate with each other. It is assumed that node B first sends its data Y to node A and, then, node A uses MIT code to implement source coding and data aggregation. Although interleavers are used to enhance the randomness of input data in turbo code, the partial interleavers of MIT code in this case are used to arrange the positions of the correlated bits of X and Y such that the bits $X_j^i, X_{j+1}^i, \dots, X_{j+k}^i$ of the interleaved input X are correlated with the bits $Y_j, Y_{j+1}, \dots, Y_{j+k}$ of the input Y , where the superscript “ i ” refers to an interleaved input and the subscripts denote the indices of the bits. Similarly, the bits $Y_j^i, Y_{j+1}^i, \dots, Y_{j+k}^i$ of the interleaved input Y are correlated with the bits $X_j, X_{j+1}, \dots, X_{j+k}$ of input X . Hence, the correlated data bits of inputs sequences X and Y enter together the convolution encoders. The encoding structure of the proposed joint source coding and data aggregation is

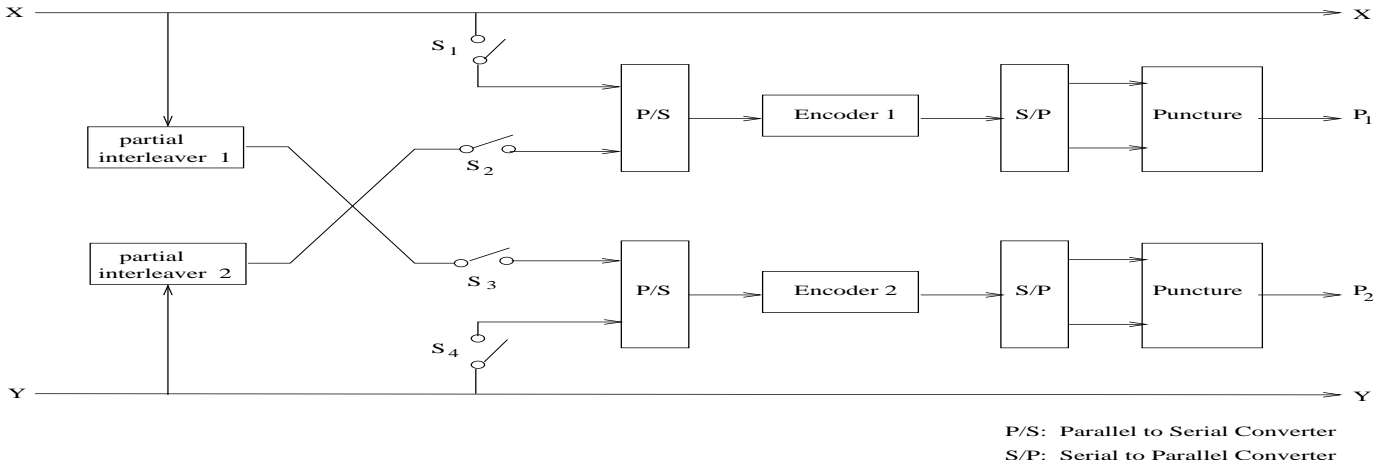


Fig. 2. The encoder structure of MIT code with multiplexers for two inputs X and Y . MIT code can be extended easily for more than two inputs by placing its multiple copies in parallel.

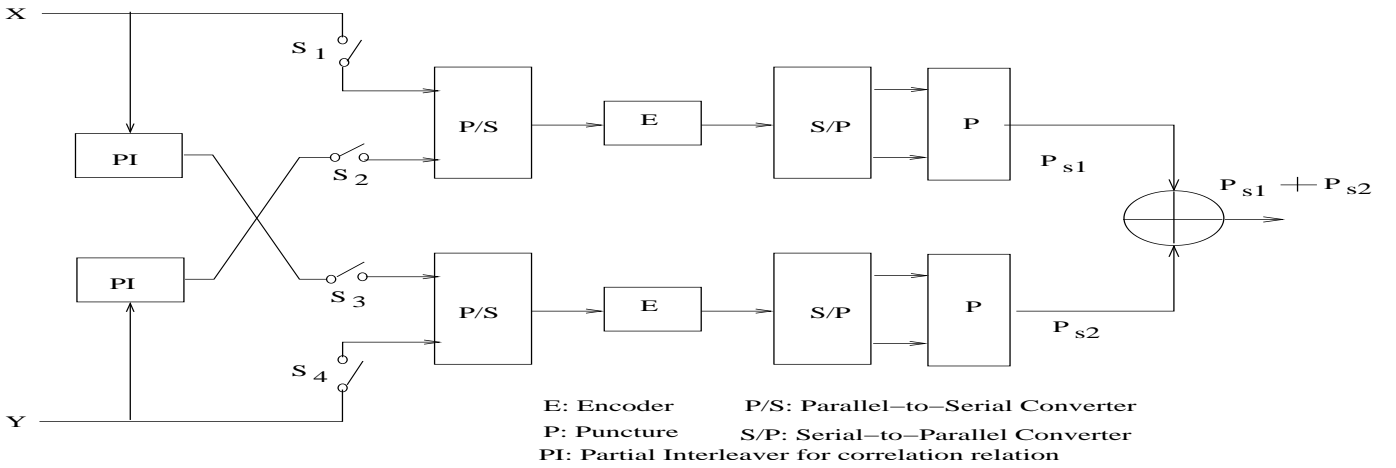


Fig. 3. The encoding structure of the joint source coding and data aggregation using MIT code. Puncturing is used to adjust the output of encoders to the desired compression rate.

illustrated in Figure 3. The systematic inputs X and Y are not included at the outputs. Puncturing is applied to the parity bits of the convolutional encoders in order to achieve the desired compression rate in source coding.

B. Distributed Source Coding

Distributed source coding [15] refers to the compression of correlated sources that do not communicate with each other but send their encoded data with respect to side information to a common decoder. The Slepian-Wolf theorem [16] shows that it is possible to compress statistically dependent signals in a distributed manner over noiseless channels to the same rate as with a system where the signals are compressed jointly. That is, separate encoding without any communication among sensor nodes (with the increased complexity at the joint decoder) is as efficient as joint encoding for lossless compression.

If each of X and Y is encoded/decoded without knowledge of any side information, the minimum rate needed to encode these two sources is $H(X) + H(Y)$, where $H(X)$ and $H(Y)$ denote the entropies (i.e., average amount of information per symbol) of X and Y , respectively. However, MIT code assumes that each of inputs X and Y knows its joint distribution

with respect to side information, say Z , sent by node C to the decoder at the data aggregator. In this case, the partial interleavers of MIT code are set up in accordance with the relation between side information Z and input sequences X and Y . The data of X and Y are encoded with respect to the side information Z . When every k consecutive bits of X (or Y) are encoded with respect to Z using a data correlation between them such as a Hamming distance relation described in [15], $k - j$ bits are transmitted from the outputs of MIT code to the data aggregator, for $1 \leq j \leq k$. Thus, instead of transmitting $H(X) + H(Y)$ to the data aggregator, node A transmits $H(X|Z) + H(Y|Z)$, given that node C transmits $H(Z)$.

IV. JOINT SOURCE-CHANNEL CODING AND DATA AGGREGATION

The separation of source and channel coding results in performance losses. Therefore, joint source-channel coding [19] is introduced to design and operate source and channel coding in a dependent fashion for optimal allocation of resources between them. The duality between source and channel coding with side information has been recently shown in [17]. That is, the encoder (decoder, respectively) of source coding with side information at decoder is functionally identical with the

decoder (encoder, respectively) of channel coding with side information at encoder.

One important advantage of using MIT code for joint source-channel coding is that it enables the decoders of source coding and channel coding to take advantage of turbo principle because the decoders interact iteratively with each other in decoding. The decoder of source coding implements turbo principle using the correlation between two sources as prior information. As illustrated in Figure 4, channel encoding can be implemented by MIT code just after a source coding and data aggregation. As in turbo code, the partial interleavers in channel coding are used to improve the code strength by increasing the weight of codewords.

A. Partial Interleaver for Channel Coding

The size of interleaver in turbo code has great impact on its performance as well as the energy consumption because an interleaver with big size requires more memory and processing operations. Hence, two important parameters of an interleaver are its size and actual interleaving pattern. Let N denote the interleaver size that also equals the size of an information word. A partial interleaver of MIT code interleaves only K out of N bits, for $K < N$. So, the remaining $N - K$ bits of the information word are not involved with the interleaving. In other words, the contents of those $N - K$ locations do not change at all during interleaving. As far as the performance of interleaver and MIT code is considered, it is critical how those K bits of a block are chosen. The degree of correlation (or dependence) between the inputs X and Y of MIT code depends on the number of those r -bit words in inputs X and Y over their all r -bit words. The impact of correlation is controlled by a weight parameter w whose values ranges from 0 to 0.99. After the value of K is determined, the K bits of information word are interleaved in a similar way to S-random interleaver [10]. The partial interleaving algorithm is described next.

Algorithm PI

Input: An interleaver with size N , and two input sequences of X and Y with the same number of bits.

Output: A partial interleaving with size K .

begin

1. Let a denote the number of those r -bit words that are located at the corresponding r -bit positions of inputs X and Y . Also, let b denote the total number of r -bit words in X (or Y). Then, let $c = (a/b)$.
2. Let K be equal to $N - \lfloor (c \times w) \rfloor$ for a given weight parameter w ranging from 0 to 0.99.
3. Choose K integers from the set $\{1, 2, \dots, N\}$. Form the interleaver output set L consisting of these selected K integers only. Map each of these bits that are located at the positions of unselected integers at the interleaved inputs to the same locations at the interleaver outputs.
4. Let the interleaver distance S equal to $\lfloor \sqrt{K/2} \rfloor$. Set a temporary variable “count” to 0.
5. Select an integer i randomly from the set L .
6. Compare the selected integer i with the previously selected S integers. If the absolute value of the difference between

i and any of the previously selected S integers is smaller than S , then increment *count* by 1. If *count* is less than S , go to Step 5 to select another integer. Otherwise, map the i th bit of the interleaver input to the leftmost empty bit position of the interleaver output.

7. Remove the current selected integer i from the set L . If L is empty, then stop. Otherwise, set *count* to 0 and then go to Step 5.

end

B. Decoder Implementation

As seen from Fig. 5, the MIT decoder is the same as the turbo decoder except that the MIT decoder has either parallel-to-serial or serial-to-parallel converters between decoder and interleavers or deinterleavers in order to properly multiplex or demultiplex the decoders’ output bits. Similar to turbo decoding, MIT decoding is an iterative process and the BER improves as the number of decoder iterations increases at the expense of having more calculations. Indeed, the decoding complexity is mainly affected by three main factors: (i) interleaver size, (ii) the number of decoder iterations, and (iii) the number of decoder states determined by encoder constraint length. MIT code uses small interleaver size and small constraint length for convolutional encoders. To consume less power in decoding, the number of decoder iterations may be dynamically changed based on the channel conditions. The power savings of MIT decoder can also be increased significantly by reconfiguring the decoder at run time based on channel conditions, similar to as much as a 52% power saving in turbo decoder reconfiguration [18].

V. PERFORMANCE EVALUATION

This section presents the simulation results for comparing the performance of MIT and turbo codes. For MIT code, we develop a simulator by modifying the turbo code simulator given in [20]. Turbo code has one input of k bits, one interleaver, and two encoders, whereas MIT code has two inputs of k bits each, two partial interleavers, and two encoders. A block log-MAP decoder is used in the simulations. Code rate of 1/2 is simulated over an AWGN channel for both turbo and MIT codes. All the encoders with $g_0 = 07$ and $g_1 = 05$ in octal are assumed to employ a tail which causes the actual code to be reduced slightly. The size of interleaver equals the information block length ranging from 256 to 1024. Each partial interleaver of MIT code interleaves half of its input bits. A maximum of up to 8 iterations are simulated for 10,000 information blocks. The bits 0 and 1 of data are generated uniformly. Inputs X and Y are assumed not to have any data correlation initially. In this case, Figures 6 and 7 show the bit error rate (BER) performance of the turbo and MIT codes for iterations of 1, 3, and 8 for block lengths of 256 and 1024, respectively.

As seen in Fig. 6, MIT code is a bit better than turbo code in the case of one iteration in the decoder, while MIT code significantly outperforms turbo code in the case of three and eight iterations in the decoder. Similar better results are also shown in Fig. 7. Depending on the BER requirements

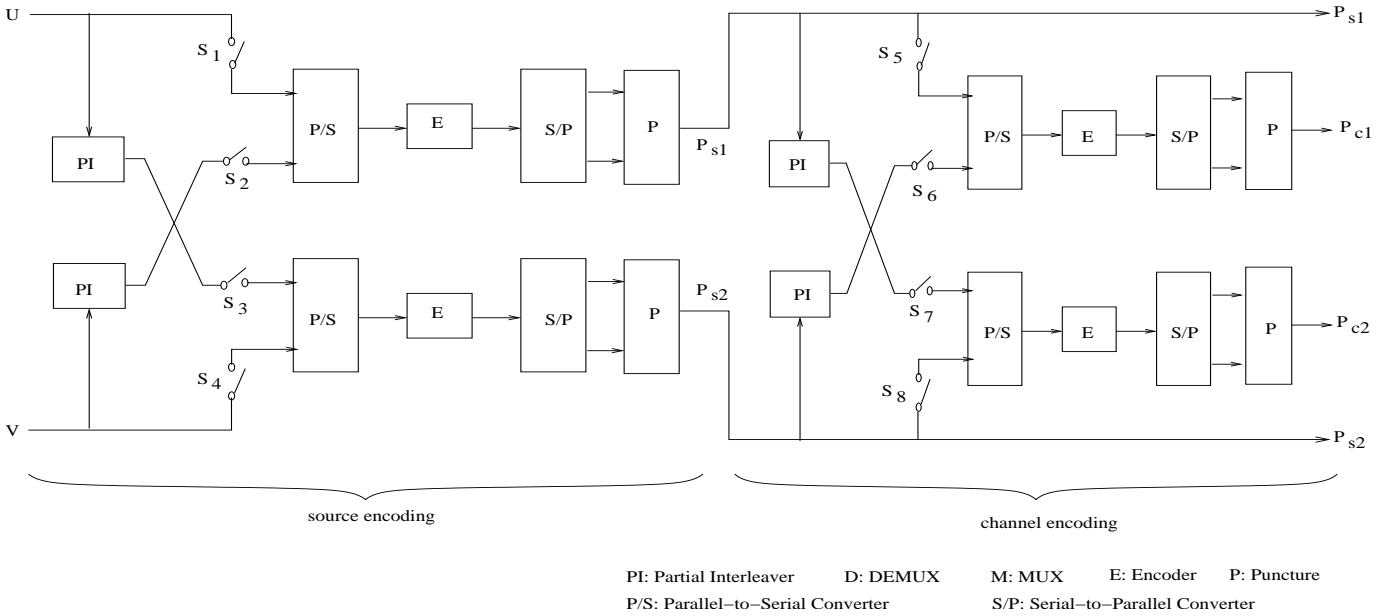
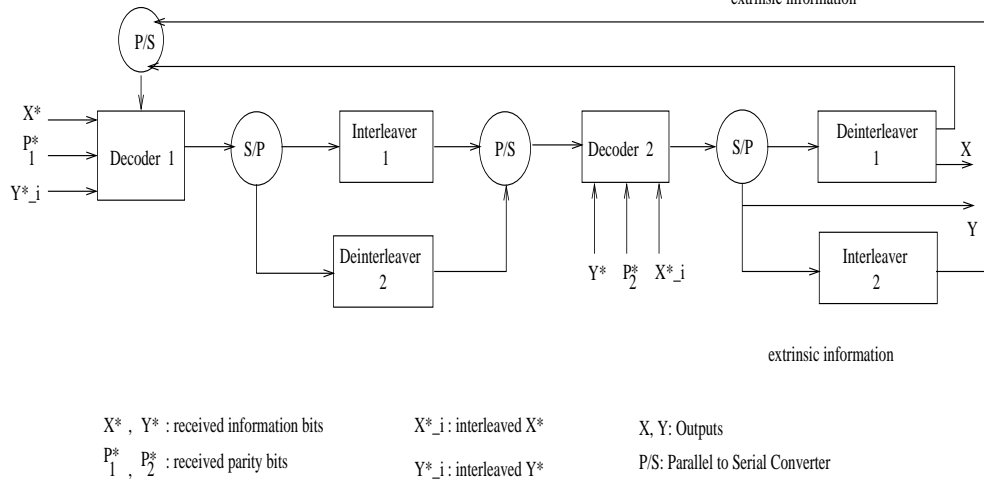


Fig. 4. The encoding structure of the joint source-channel coding and data aggregation using MIT code. The inputs U and V represent either two correlated inputs (e.g., the correlated data of sensor nodes U and V) or two different information words of the same input.



X^* , Y^* : received information bits
 P_1^* , P_2^* : received parity bits
 $X*_j$: interleaved X^*
 $Y*_j$: interleaved Y^*
 X, Y : Outputs
P/S: Parallel to Serial Converter

Fig. 5. MIT code decoder.

and that the BER difference between three iterations and eight iterations is not significant, three decoding iterations may be sufficient for some applications in order to reduce energy consumption for decoding operations. To illustrate the impact of data correlation on the BER performance of MIT code, we considered two-bit data correlation for the case where (i) each sensor measurement is expressed by two bits and (ii) the probability that nodes X and Y can produce the same sensor measurement is 0.6. In this case, as seen from Fig. 8, MIT code leads to better BER for correlated data than non-correlated data.

VI. CONCLUSIONS

This paper has presented an energy-efficient MIT code for wireless systems, in particular, wireless sensor networks. MIT code can be used for joint source coding, data aggregation, and channel coding in wireless sensor networks. MIT code allows source coding and data aggregation to benefit from

the correlated data of sensor nodes for reducing BER further. MIT code can be used to reduce the amount of data to be transmitted, leading to energy saving and better bandwidth utilization. MIT code employs partial interleaving to reduce memory space and the number of memory accesses in interleaving and deinterleaving operations at transmitter and decoder, respectively, thereby resulting in energy saving. The simulation results show that MIT code requires less E_b/N_o than turbo code for meeting a given BER requirement in case of the iterations three and eight.

REFERENCES

- [1] P. von Rickenbach and R. Wattenhofer, "Gathering correlated data in sensor networks," *Proc. of the ACM DIALM-POMC'04*, Oct. 2004.
- [2] R. Cristescu, B. Beferull-Lonzano, and M. Vetterli, "On network correlated data gathering," *Proc. of the 23rd Conference of the IEEE Communications Society (INFOCOM)*, 2004.
- [3] M. Lotfinezdah and B. Liang, "Effect of partially correlated data on clustering in wireless sensor networks," *Proc. of the IEEE International*

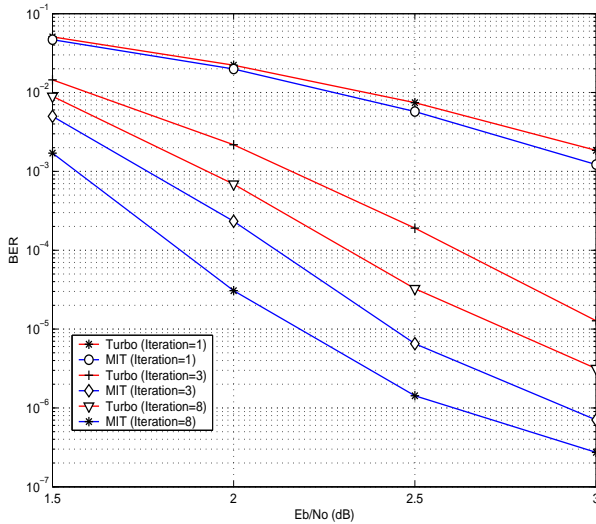


Fig. 6. BER performance of MIT and turbo codes for block length of 256 and code rate of 1/2.

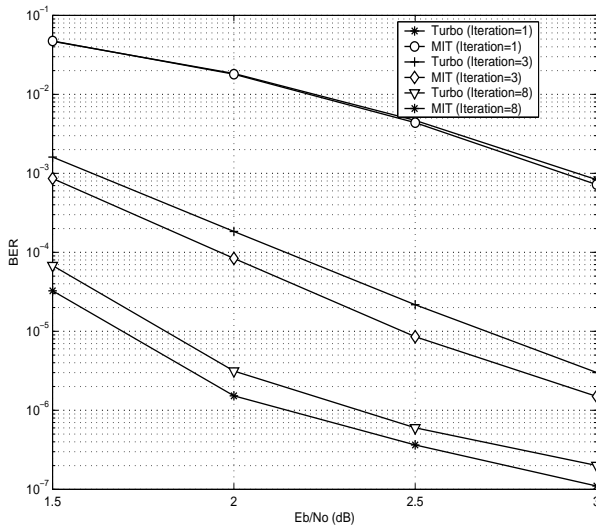


Fig. 7. BER performance of MIT and turbo codes for block length of 1024 and code rate of 1/2.

Conference on Sensor and Ad hoc Communications and Networks (SECON), Santa Clara, California, October 2004.

[4] W.R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," *Proc. of the 33rd Hawaii International Conference on System Sciences*, 2000, pp. 1-10.

[5] S. Lindsey, C.S. Raghavendra, and K. Sivalingam, "Data gathering algorithms in sensor networks using energy metrics," *IEEE Trans. on Parallel and Distributed Systems*, vol. 13, no. 9, Sep. 2002, pp. 924-935.

[6] B. Przydatek, D. Song, and A. Perrig, "SIA: Secure Information Aggregation in Sensor Networks," *Proc. of ACM SenSys 2003*, Nov. 5-7, 2003, LA, USA.

[7] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon Limit Error-Correcting Coding and Decoding: Turbo-Codes," *Proc. of ICC'93*, May 1993, pp. 1064-1070.

[8] C. Berrou and A. Glavieux, "Near Optimum Error Correcting Coding and Decoding: Turbo Codes," *IEEE Transaction on Communications*, vol. 44, no. 10, Oct. 1996, pp. 1261-1271.

[9] S. Benedetto, G. Montorsi, and D. Divsalar, "Concatenated convolutional codes with interleavers," *IEEE Communications Magazine*, Aug. 2003, pp. 102-109.

[10] D. Divsalar and F. Pollara, "Multiple turbo codes for deep-space

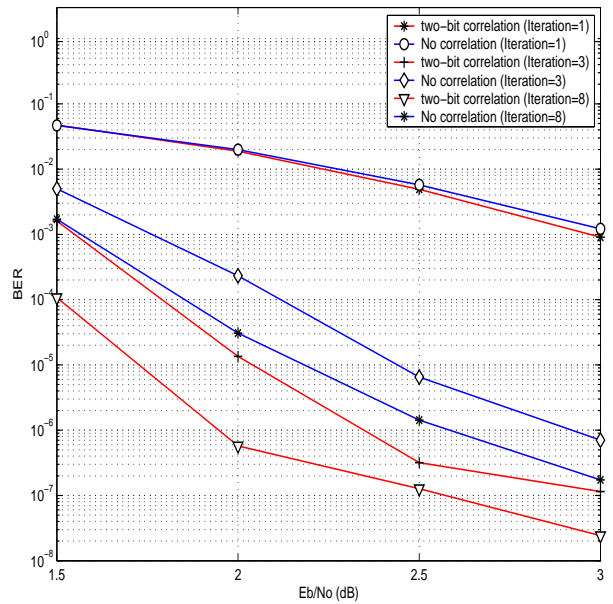


Fig. 8. Impact of two-bit data correlation on the BER performance of MIT code for block length of 512 and code rate of 1/2.

communications," JPL TDA Progress Report 42-121, Pasadena, CA, May 1995, pp. 66-76.

[11] J. Garcia-Frias and Y. Zhao, "Compression of binary memoryless sources using punctured turbo codes," *IEEE Communications Letters*, vol. 6, no. 9, Sep. 2002, pp. 394-396.

[12] Z. Peng, Y.-F. Huang, and Daniel J. Costello, "Turbo Codes for Image Transmission: A Joint Channel and Source Decoding Approach," *IEEE Journal on Selected Areas in Commun.*, JSAC-18, pp. 868-879, June 2000.

[13] W. Zhong and J. Garcia-Frias, "LDGM codes for channel coding and joint source-channel coding of correlated sources," *IEEE Trans. on Communications*, vol. 52, no. 8, Aug. 2004, pp. 1238-1241.

[14] K. Wu and L. Ping, "An improved two-state turbo-SPC code for wireless communication systems," *EURASIP Journal on Applied Signal Processing*, 2005.

[15] S.S. Pradhan, J. Kusuma, and K. Ramchandran, "Distributed compression in a dense microsensor network," *IEEE Signal Processing Magazine*, vol. 19, no. 2, March 2002, pp. 51-60.

[16] D. Slepian and J.K. Wolf, "Noiseless coding of correlated information sources," *IEEE Trans. Inform. Theory*, vol. IT-19, July 1973, pp. 471-480.

[17] S.S. Pradhan, J. Chou, and K. Ramchandran, "Duality between source coding and channel coding and its extension to the side information case," *IEEE Transactions on Information Theory*, vol. 49, no. 5, May 2003, pp. 1181-1203.

[18] J. Liang, R. Tessier, and D. Goeckel, "A dynamically-reconfigurable, power-efficient turbo decoder," *Proc. of the IEEE Symposium on Field-Programmable Custom Computing Machines*, Napa, California, April 2004.

[19] J. Garcia-Frias, W. Zhong, and Y. Zhao, "Iterative decoding schemes for source and joint source-channel coding of correlated sources," *Proc. of the Thirty-Sixth Asilomar Conference on Signals, Systems and Computers*, Nov. 2002, pp. 250-256.

[20] S.A. Barbulescu, *What a Wonderful Turbo World*, 2003, <http://people.myoffice.net.au/abarbulescu>.