

OPERA: An Optimal Progressive Error Recovery Algorithm for Wireless Sensor Networks

Saad Bin Qaisar and Hayder Radha
 Department of Electrical and Computer Engineering
 Michigan State University
 East Lansing, MI-48824, United States
 Email: {qaisarsa,radha}@egr.msu.edu

Abstract—Wireless Sensor Networks (WSNs) require robustness against channel induced errors while retransmission based schemes prove too costly for energy constrained sensor nodes. Channel coding with low rates can ensure increased reliability, and it can eliminate the need for costly retransmissions. However, low channel coding rates on an end-to-end basis (between a source sensor and a destination base-station) requires a large number of transmissions of redundant data. In this paper, we propose an Optimal Progressive Error Recovery Algorithm (OPERA) over WSNs. Under OPERA, individual intermediate sensors, which are relaying data toward the base station, partially and optimally channel-decode the incoming packets while employing a progressive decrease in parity bits as data reaches the final destination. OPERA requires significantly lesser processing than would be required for complete decoding or full decoding/encoding at the sensor nodes; and OPERA significantly reduces the total number of transmissions when compared to optimal end-to-end channel coding schemes. We use iteratively decodeable LDPC codes for this purpose. OPERA not only provides a partial processing framework, but also an algorithm to optimally map the decoding iterations over the multi hop network. We provide a comparison between our iteration assignment scheme and random iteration assignment, and show that our scheme performs considerably better. Finally, while being naturally motivated by the sensor reachback problem, we further develop a fairness-based OPERA scheme for the allocation of channel-decoding LDPC iterations to sensor nodes taking into consideration the life-expectancy of each sensor.

I. INTRODUCTION

In a multi-hop Wireless Sensor Network (WSN), reliable delivery of information to the base station is of prime importance for many applications. This motivated many researchers to explore a variety of channel coding schemes over sensor networks (see for example [17]

[18] [19]). Information quality deteriorates due to errors introduced by the channel as it reaches the final destination. Addition of parity bits to the packet can provide robustness against end-to-end channel induced errors. The cost associated though is the excess transmit (and receive) power for transmitting (and receiving) redundant bits at each intermediate node over the path. An alternative can be recovering the lost bits at the intermediate nodes through processing. ‘Full processing’ and ‘forwarding’ are the two extremes of processing within the network (in-network processing). Full processing implies complete decoding and re-encoding of information bits. In forwarding, each node just forwards the information bits it received to next node.

‘Partial processing’ provides an intermediate ground between forwarding and full processing. Allowing intermediate link nodes to perform finite complexity processing achieves a significant portion of the ultimate capacity in fairly noisy networks [2]. Meanwhile, partial recovery from errors at intermediate sensor nodes has not been studied thoroughly, and probably no work has been published in this area (to the best of our knowledge). A related context was mentioned by Fragouli et al. who discussed the benefits that can be achieved due to finite length processing at intermediate nodes [2]. We consider the generic scenario where sensors transmit data to a central collector by routing it over a multi-hop sensor network. Our objective is to optimally increase the overall throughput that finite complexity processing at intermediate nodes may provide. Specifically, we propose an algorithm that partially decodes the incoming packets at intermediate nodes along with progressive decrease in parity bits as data reaches the final destination. This work is motivated by (1) End-to-end channel coding techniques put a substantial burden on resource constrained wireless sensor networks; and (2) the reachback problem [3] [15] causes significant variation in the life expectancy of nodes; the closer the node to the base station, the lower its life is expected to be. We show that

This work was supported in part by National Science Foundation Award CCF-0515253, NSF Award CNS-0430436, MEDC Grant GR-296, and unrestricted gift from Microsoft Research.

the proposed technique provides considerably better performance than end-to-end techniques with much reduced energy cost, while addressing the reachback problem. In this paper, we use Low Density Parity Check (LDPC) codes to demonstrate our algorithm.

We refer to our scheme as Optimal Progressive Error Recovery Algorithm (OPERA). Under OPERA, an intermediate relay node partially decodes the incoming channel-coded data using lesser number of iterations than required for complete decoding. Hence, at each sensor node, a received packet is partially decoded and sent over the next hop without addition of further parity. In addition, the intermediate nodes closer to the destination progressively reduce parity bits to cut down transmission costs along with partial processing. At the final destination, a complete decoding takes place by the relatively powerful central processor. We develop a framework that optimally assigns decoding iterations to the sensor network. In addition, we propose a scheme which is proportionally fair [21] such that sensor nodes with lesser energy resources spend lesser computational and transmit energy than those that are resource rich.

Systems with Forward Error Correction (FEC) can provide objective reliability with lesser transmission power than those without FEC [18]. Due to resource limitation of the sensor nodes, only few works have been done on channel coding in wireless sensor networks. Shih et al propose a scheme based on convolutional codes for FEC in sensor network [18] whereas Sankarabramaniam et al [19] propose use of BCH codes. Mina et. al [18] presented a framework in which they consider combined source and channel coding with LDPC codes for sensor networks. All these works consider end-to-end error correction and thus cater to the worse case scenarios at intermediate forwarding links, adding significant burden over channels with higher signal to noise ratio. In terms of power consumption, transmitting an additional single bit of data is much costlier than the instructions used for computations in a sensor node [15] [20]. For example, in a recent study, a single bit of data transmission has been shown to be equivalent to 2000 computational instructions executed in a sensor node [20]. Thus, energy tradeoff between communication and computation makes a case for processing the data inside the network rather than simply transmitting the sensor readings. Therefore, for improved reliability of data from a source node to the collector and conservation of energy, in-network processing can be highly beneficial. The proposed OPERA scheme ensures this by partial decoding of packets and hence enhancing the reliability at the destination with introduction of minimal complexity as compared to end-to-end channel coding.

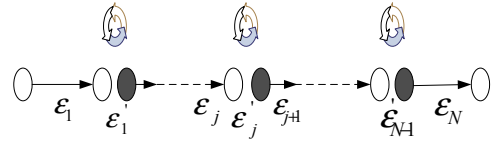


Fig. 1. A Multi hop network with in-node processing

The remainder of the paper is organized as follows. We formulate the problem in Section II. Section III presents LDPC codes and associated concentration theorem that are relevant to the OPERA approach. Section IV describes the proposed OPERA algorithm for mapping in-network iterative decoding and addresses the fairness issue through proportional fairness for energy constrained network nodes. Section V provides simulation results and discussion, with conclusion and further directions in Section VI.

II. PROBLEM SETTING

A. MultiHop Wireless Channel

A multi-hop wireless channel can be represented as a cascade of channels. Consider a line network with N nodes in cascade. All nodes are considered equally important with every node capable of decoding the received packets. The links between the nodes are assumed to be BSC with each node transmitting at power level P_T per bit. Relay nodes are allowed not only to forward the incoming information, but also to process it. In order to obtain channel error probability for a binary symmetric channel (BSC), the general expression for Signal to Interference Noise Ratio (SINR) can be given as in (1):

$$SINR = \frac{P_T}{P_A + \sum P_{int}} \quad (1)$$

where P_A is the ambient noise power, and P_{int} is the interference power of any concurrent transmissions elsewhere in the network. Sources of ambient noise may include other devices operating in the same frequency band or other networks co-located with the WSN. Let $\{n_t; t \in T\}$ be the subset of nodes simultaneously transmitting over a certain subchannel, with transmit power P_t for each node. Then the SINR at node n_j for a transmission from node n_i , $i \in T$ can be calculated as in [9]:

$$SINR = \frac{\frac{P_i}{d(n_i, n_j)^\alpha}}{P_A + \sum_{t \in T, t \neq i} \frac{P_t}{d(n_t, n_j)^\alpha}} \quad (2)$$

where $d(n_i, n_j)$ is the separation between n_i and n_j , and $\alpha > 2$. If $Q(a) = \frac{1}{\sqrt{2\pi}} \int_a^\infty e^{(-u^2/2)} du$, then the channel error probability as described in [8] is:

$$\epsilon(n_i) = Q(\sqrt{2(SINR(n_i))}) \quad (3)$$

B. Iteration Assignment

The source node n_0 generates k message bits which are encoded using a rate R code. The resulting codeword is transmitted over the first link with error probability ϵ_1 . Node n_1 performs l_1 LDPC decoding iterations and the bit error rate in the resulting packet is a function of ϵ_1 and l_1 :

$$\epsilon_1' = f(\epsilon_1, l_1) \quad (4)$$

After partial processing at the second node, the error rate in the resultant packet is:

$$\epsilon_2' = f(\epsilon_2, l_2) * \epsilon_1' \quad (5)$$

where $\epsilon_1 * \epsilon_2 = \epsilon_2(1 - \epsilon_1) + \epsilon_1(1 - \epsilon_2)$. For the cascade line network, we define the total number of decoding iterations $\Gamma(\bar{l})$ in the entire network as:

$$\Gamma(\bar{l}) = \sum_{j=1}^{N-1} l_j \quad (6)$$

where l_j is the number of decoding iterations at node n_j .

For N hops in cascade, the overall distortion measure D can be expressed as:

$$D = f(f(f(f(\epsilon_1, l_1) * \epsilon_2, l_2) * \dots * \epsilon_j, l_j) * \dots * \epsilon_{N-1}, l_{N-1}) * \epsilon_N \quad (7)$$

For $\Gamma(\bar{l}) \leq \Gamma(\bar{l})_{budget}$, a simplistic approach can be to randomly assign iterations to the network remaining in budget constraints. That may not necessarily be an optimal assignment in terms of throughput at the final destination. Thus, within the budget constraints, we intend to find an iteration assignment vector \bar{l} in such a way that the net throughput is maximized at the destination node. Conversely, $D(\Gamma)$ is minimized. We refer the tuple $[D, \Gamma(\bar{l}), \bar{c}]$ as Network Throughput Measure (NTM). The problem can be seen as a budget constrained allocation problem, such that D is minimized subject to constraint $\Gamma \bar{l} \leq \Gamma(\bar{l})_{budget}$. From the constraint highlighted above, our problem becomes similar to minimizing a distortion measure given a budget constraint under a rate distortion framework. Therefore, an algorithm providing NTM operating point with minimum distortion D while remaining within budget constraint is much sought after.

C. Sensor Reachback

Though the iteration assignment maybe optimal in throughput sense, it may introduce higher complexity at few nodes than other when we consider a network with multiple flows across sets of nodes that may not necessarily be disjoint. Especially, we do not intend to aggravate the reachback problem [4]. Therefore, its

highly desirable that any proposed throughput enhancement and reliability framework is fair to the individual nodes.

III. LDPC CODES FOR OPERA

Low Density Parity Check Codes are systematic block codes which have gained considerable attention due to their near capacity performance. Gallager provided an algorithm for decoding of LDPC codes that is near optimal [10]. The algorithm iteratively computes the distribution of variables in graph-based models, and comes under different names/variations including Sum Product Algorithm (SPA), Belief Propagation Algorithm (BPA), or more generally, Message Passing Algorithm (MPA).

For the simulations in this section, we employ rate $\frac{1}{2}$, (3,6)-regular and a degree seven density evolution optimized irregular, Progressive Edge Growth [11] based LDPC code with variable node degree polynomial $\lambda(\theta) = 0.207\theta^6 + 0.271\theta^2 + 0.522\theta^1$, message length $k = 1024$ bits. We use a log-domain Sum Product Algorithm (LSPA) [10] for iterative decoding of the code which has advantages in terms of implementation, computational complexity and numerical stability [12] which is critical in the context of sensor networks.

Both figure 2, based on our prior work in [1] and figure 3 show the expected bit error rate with variation in number of decoding iterations for different channel error probabilities averaged over 1000 runs for regular and irregular code, respectively. We see a sharp decrease in expected bit error rate within few decoding rounds. In fact, the decrease is exponential in nature.

Based on the above observations, we develop an exponentially decaying statistical model for the variation of bit error rate with decoding iterations. Equation (8) expresses the relationship between the estimated bit error rate \hat{f} for a given channel error probability ϵ and the number of decoding iterations l for our code as

$$\hat{f}(\epsilon_{ind}, l) = \alpha(\epsilon_{ind})e^{\beta(\epsilon_{ind})l} + \gamma(\epsilon_{ind})e^{\psi(\epsilon_{ind})l} \quad (8)$$

where $\epsilon_{min} \leq \epsilon \leq \epsilon_{max}$, $0 < l \leq l_{max}$, ϵ_{ind} is the corresponding index value with respect to ϵ_{min} and the coefficients $\alpha(\epsilon_{ind})$, $\beta(\epsilon_{ind})$, $\gamma(\epsilon_{ind})$ and $\psi(\epsilon_{ind})$ attain the values listed in Table I and II(see Appendix).

Therefore, significant enhancements in performance can be achieved as the number of LDPC decoding iterations are increased at the receiver provided the channel error probability is below the achievable performance bound [13] for the ensemble of codes. This result can be used to formulate an in-network processing framework to maximize the achievable reliability at the destination.

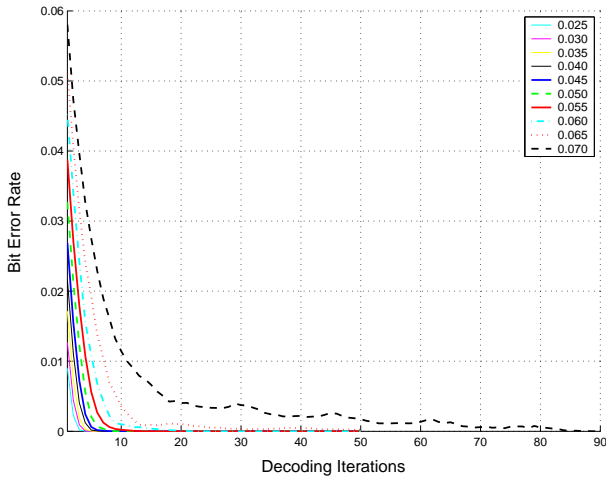


Fig. 2. Bit Error Rate as a function of Decoding iterations and Channel Error probability for a rate $\frac{1}{2}$ PEG (3,6) regular LDPC code with $\epsilon_{min} = 0.025$ and $\epsilon_{max} = 0.07$

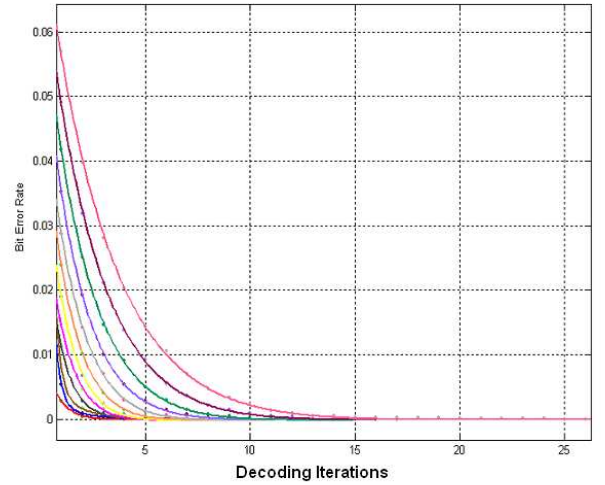


Fig. 3. Bit Error Rate as a function of Decoding iterations and Channel Error probability for a rate $\frac{1}{2}$ PEG irregular LDPC code with $\epsilon_{min} = 0.025$ and $\epsilon_{max} = 0.07$

Urbanke et.al, in [3] give a concentration theorem for LDPC codes for a wide variety of channels stated below.

LDPC Concentration Theorem

Let $P_e^n(l)$ be the expected fraction of incorrect messages which are passed in the l th iteration of LDPC decoding, where expectation is over all instances of the code, the choice of the message and realization of the noise. For any $\delta > 0$, the probability that the actual fraction of incorrect messages which are passed in the l th iteration for any particular such instance lies outside the range $(P_e^n(l) - \delta, P_e^n(l) + \delta)$ converges to zero exponentially fast in n .

The theorem asserts that (almost) all LDPC codes behave alike and so the determination of the average behavior of the ensemble suffices to characterize the individual behavior of (almost) all codes [3]. Any randomly picked code from the LDPC ensemble would have performance approaching exponentially fast in n to the mean of the ensemble. Thus, for a sufficiently large n , we can pick up any LDPC code and the results achieved would have small deviation from the ensemble mean. In subsequent sections, we use (3, 6) regular LDPC codes and the results obtained are equally valid for any randomly picked LDPC code.

IV. OPERA: AN OPTIMAL PROGRESSIVE ERROR RECOVERY ALGORITHM

For the proposed scheme, we are looking for an optimal tuple $[D, \Gamma(\bar{l}), \bar{\epsilon}]$ such that $\Gamma(\bar{l}) \leq \Gamma(\bar{l})_{budget}$ i.e. a minimal overall bit error rate that meets the budget constraint. The problem can be viewed as one that allocates the overall iteration budget $\Gamma(\bar{l})_{budget}$ among all nodes in the network such that distortion is minimum.

The computational complexity of optimally mapping the iterations to the network is high, as NTM region consists of all possible operating points obtained by choosing all possible combinations of iteration assignments within the budget constraint. The hull of the NTM region would provide the desired optimal solution. We first solve the problem for a single path line network, which can then be mapped to the entire network.

A. Dynamic Programming Approach

To find out the optimal hull of NTM region, we employ a dynamic programming approach similar to the method used in determination of the RD region for optimal quantizer design [7]. The algorithm stated is greedy in nature and it is possible that it may not find the optimum tuple $[D, \Gamma(\bar{l}), \bar{\epsilon}]$, though it does provide optimal solution under various practical scenarios ([5], [6], [7]).

The starting point of our algorithm is the case when no iterations are assigned to any intermediate nodes. Hence, D is maximum and $\Gamma(\bar{l})$ minimum. Thus the tuple $[D^0, \Gamma(\bar{l}^0), \bar{\epsilon}]$ has D^0 closest to the D axis. The algorithm then adds a single iteration to intermediate nodes, one by one in a greedy fashion, and selects the node where the minimum D is achieved. This provides the next operating point $[D^1, \Gamma(\bar{l}^1), \bar{\epsilon}]$. The procedure is repeated till $\Gamma(\bar{l}) \leq \Gamma(\bar{l})_{budget}$ is satisfied and all the iterations are now assigned. Thus, the algorithm maximizes the separation between optimal point achieved in the prior run D^- and the new points D^+ obtained in the current run for all intermediate nodes on the path. This corresponds to maximizing the gradient between the two points.

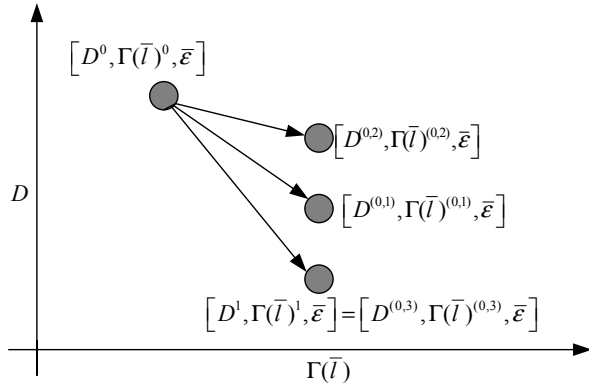


Fig. 4. Selection of Next Optimal Point $[D^1, \Gamma(\bar{l}^1), \bar{\epsilon}]$ for Dynamic Programming algorithm

Mathematically, we have:

$$\bar{l} = \underset{j=1:N-1}{\operatorname{argmax}} \left[\frac{D^- - D^+}{\Gamma(\bar{l}^-) - \Gamma(\bar{l}_j^+)} \right] \quad (9)$$

where, we take $\Gamma(\bar{l}^-) - \Gamma(\bar{l}^+) = 1$.

For extension of the algorithm to a multi hop multi path network with all nodes transmitting to a central base station, we conduct optimal iteration assignments on individual paths to obtain assignment vector and distortion for each path. The distortion is then averaged over all paths, from sources to base station, to obtain cumulative bit error rate.

B. Issue of Fairness

In our framework for a WSN, we assume the total energy spent by sensor nodes per packet delivery to the destination node, E_{total} , a function of computational energy and the transmission energy. Thus, we have

$$E_{total} = E_{Trans.} + E_{Comp.} \quad (10)$$

where, $E_{Comp.}$ is the computational energy spent per packet for partial processing and $E_{Trans.}$ is the per packet transmission energy for end to end delivery. An iteration assignment algorithm would be proportionally fair computationally if it assigns more iterations to resource rich nodes than the constrained ones. On the other hand, it would be proportionally fair in transmit energy if intermediate resource constrained nodes make lesser transmissions than resource rich nodes per packet delivery to the destination node. With the above reasoning in mind, we can exploit the architecture of LDPC codes. Since LDPC codes are systematic block codes, we can remove parity progressively from end of the block at the intermediate nodes and the resulting packets would still remain decodeable by appending randomly

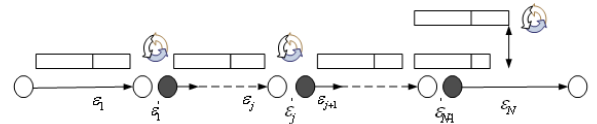


Fig. 5. Progressive decrease in parity for a multihop WSN

generated bits at the destination node. The appended parity can be thought of as corrupted version of the parity removed from the basic code at the data originating node. Since the LDPC parity check matrix remains the same across the network, for partial processing at subsequent intermediate nodes, random parity is appended and removed prior to transmission of the code block to next node. For a given WSN, we form a partition $\{N_x\}_{x \in I}$ for the N nodes classified into quantized sets based upon their energy resources, where I is the set of integers. The partitions thus formed satisfy the following two conditions:

$$N = \bigcup_{x \in I} N_x \quad (11)$$

$$N_x \cap N_y = \Phi, \{x, y\} \in I, x \neq y \quad (12)$$

Various factors can be considered in forming the quantization levels including the degree of resource limitation based upon the number of flows passing through the node and proximity to the base station. We assume from here on that closer the node to a base station, more the number of flows passing through it and costlier its energy resources.

The transmitting nodes progressively drop the parity bits as the data moves closer to the destination node. Thus, for the case of OPERA with proportional fairness, distortion D is given as:

$$D = f_{N_x(N-1)}(f_{N_x(N-2)}(f_{N_x(j-1)}(f_{N_x_1}(\epsilon_1, l_1) * \epsilon_2, l_2) * \dots * \epsilon_j, l_j) * \dots * \epsilon_{N-1}, l_{N-1}) * \epsilon_N \quad (13)$$

where $f_{N_x_j}(\epsilon_j, l_j)$ is the bit error rate we achieve at node n_j after l_j iterations for a packet transmission at modified rate $R_{N_x_j}$ corresponding to partition N_x_j the transmitting node belongs to, for a link error probability ϵ_j .

For a network with multiple flows to the base station, we assume base station with much more computational resources than individual sensor nodes. The rates assigned to each partition are a function of its energy resources and network link quality, such that the received packet still remains decodeable with high probability at the destination node with l iterations, given $l \gg l_j$.

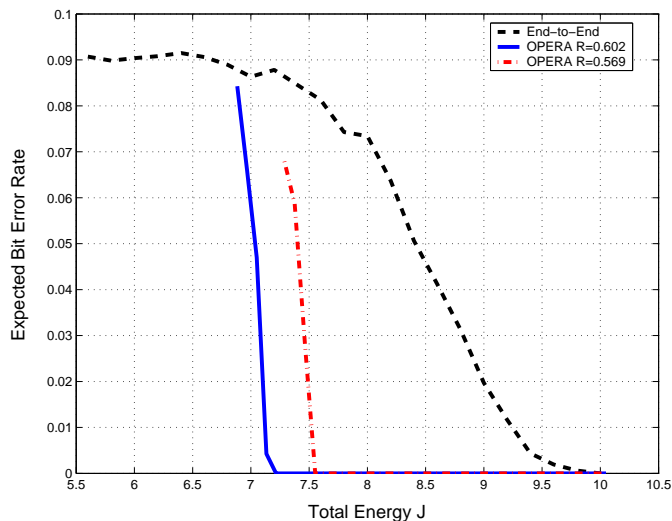


Fig. 6. End to End vs OPERA with (3,6) regular LDPC Code for a Line Network

V. SIMULATIONS AND ANALYSIS

For the proposed OPERA scheme, we assume that if $\epsilon < \epsilon_{min}$, on average, only one iteration is sufficient to completely decode the LDPC code, a behavior confirmed from the trend in figures 2, 3 and [3]. A node behaves as a forwarding node if either $\epsilon > \epsilon_{max}$ or no iteration is assigned to it. We set $\epsilon_{min} = 0.01$ and $\epsilon_{max} = 0.08$ (See [13] and [3] for tighter bound on ϵ_{max}). For a single path, we fix $\Gamma(\bar{l})_{budget} = 60$ iterations. We take per bit transmit power $P_T = 1mW$, per second, for each node. Fossorier et al. in their work on low complexity LDPC iterative decoding [14], tabulate a comparison of mathematical operations required for one iteration of various LDPC decoding algorithms including MPA. Assuming each node a sensor mote equipped with Atmel Atmega128L processor, and 2000:1 ratio between per bit transmit energy and computation energy spent per instruction [20], we provide performance curves for OPERA in the subsequent subsections.

A. End to End vs OPERA for a Line Network

We consider 100 instances of a line network for $N = 5$ with variation in separation between the nodes. The results achieved are averaged over all occurrences. We take end to end maximum and minimum rates $R_{end_{max}} = 0.731$ and $R_{end_{min}} = 0.410$. For OPERA, we consider multiple rates $R_{OPERA} = 0.602, 0.569$. All the results are averaged over 1000 runs. Figure 6 shows the performance comparison between OPERA and end to end schemes with average end to end equivalent error probability $\epsilon_{end_{eq}} = 0.0919$. It is essential to note here that the both OPERA and end-to-end results here

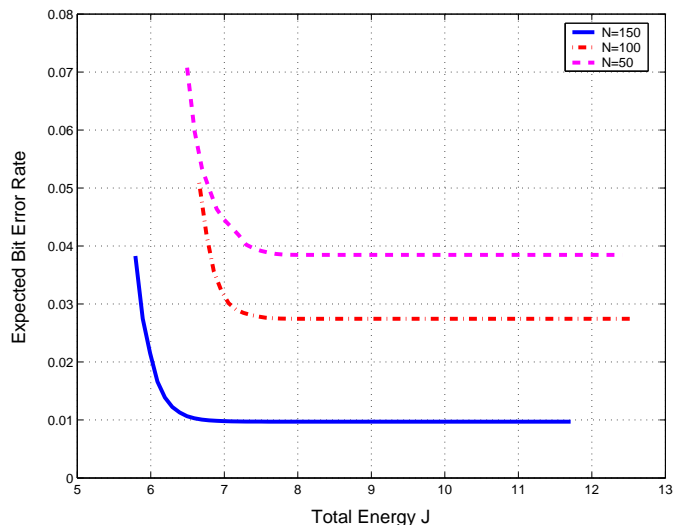


Fig. 7. Optimum curves for Expected Bit Error Rate vs. Energy spent for OPERA

are after decoding at the destination node. We assume infinite processing power at the destination node for both the schemes. Thus, for end to end case, $E_{total} = E_{Trans.}$, whereas, (10) gives the total energy for OPERA.

The results show that OPERA outperforms end to end channel decoding with much lesser energy required for a given performance.

B. OPERA with Multiple Flows

We consider a wireless sensor network consisting of N nodes spread over a $10m \times 10m$ square grid according to a random distribution. We place the base station at coordinates (5, 5) and limit the transmission range of individual sensor nodes to a maximum transmission range $r = 2m$. Since the most widely used wireless sensor network routing algorithms (DSR, AODV, Directed Diffusion) are different forms of shortest path routing algorithms, we assume that at any given time the routes from individual nodes to the base station form a tree rooted at the base station [16]. We are thus discounting the possibility of using bifurcated routing i.e. multiple paths from source to destination. Hence, we assume that all the network nodes are sending traffic back to the base station on shortest paths already established through some routing algorithm. It is important to mention here that the OPERA results, here, and in following subsections are prior to decoding at the destination node.

The plots show that it is feasible to optimally trade-off complexity/energy usage with distortion/reliability by varying the assignment of iterations. The two extremes of the Complexity-Distortion curve in Figure 7 represent the performance that is offered when two

extremes of in-network processing is employed. When we do not conduct any decoding at intermediate nodes, the accumulation of errors increases exponentially. In such a scenario a large number of packets eventually received at the collector may have zero or very little information utility. In other extreme when decoding is employed at all or almost all the intermediate nodes, the data reliability can be increased significantly, but the energy consumption is also significant. Thus, in such an operation mode, even though the throughput of a sensor network is improved, the network life-time is decreased significantly. Our proposed approach allows us to fine-granularly operate at a large number of intermediate points. Thus in an actual deployment, the operating point can be chosen in accordance with the current demands of the network. If an important event is being sensed, we may choose to operate at high in-network processing point, as against that if the network is in more passive state, we may prefer saving energy despite getting noisy readings.

The results in Figure 7 show that as the number of nodes increase the amount of in-network processing required to achieve improvement in reliability reduces. This can be explained by highlighting that, as the node density increases, the number of error prone links decrease. Since error recovery is primarily required only due to the presence of noisy links, decrease in the number of noisy links naturally leads to reduced requirement of network processing. The above observation has important implications about adapting the functional usage of a sensor network through its life-time. We illustrate our point by way of an example: Let us say that a network initially composed of 150 sensors demands

a functional usage represented by a error probability of 0.04. To support such a demand with in-network processing, we shall have to spend 5.75 Joules. With time as sensors die the density of sensors reduces, lets say our density drops to 100 sensors. At such point if the functional demand is not reduced the amount of energy that will have to be spent is infact increased to 6.8 Joules. This increase in energy spending may set-off a chain reaction eventually leading to the death of a network. Thus, as the density reduces, it may be essential to adapt the functional demands from a network. The Complexity-Distortion curves obtained by our analysis can provide important guidelines on how these demands should be reduced as sensors start dieing.

C. Random Assignment Vs OPERA

We compare the efficiency of our iteration assignment algorithm with random assignment when the decision to add iteration at a given node over the path is random such as by tossing a coin. The procedure is repeated until all the iterations are assigned to an end to end path. The plots in figure 8 show the efficiency of OPERA as compared to random assignment. The gap between the two curves indicates the enhancement in throughput we get by progressive iteration assignment through OPERA than randomly.

D. OPERA with fairness (Multiple flows)

For OPERA with proportional fairness, we form three quantization levels N_1, N_2 and N_3 . Nodes transmit at primary rate $R_{N_1} = 0.5$ when they are either a data originating node or belong to N_1 . When the code block

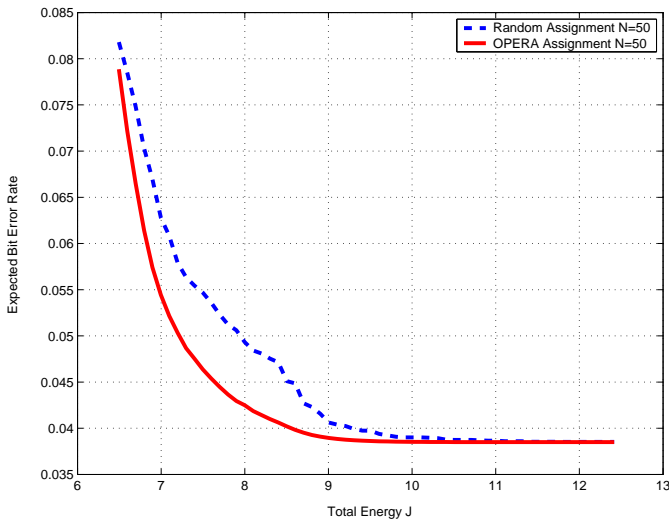


Fig. 8. OPERA vs Random Iteration assignment $R=0.5$, $N=50$, (3,6) regular PEG LDPC code

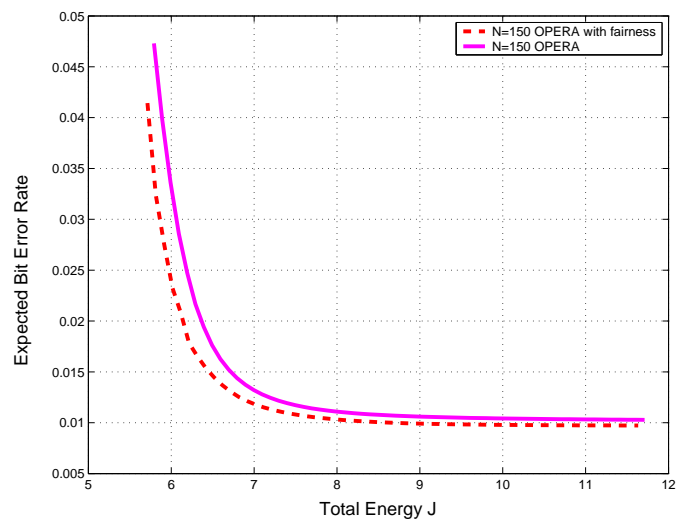


Fig. 9. OPERA with and without proportional fairness for $N=150$, (3,6) regular PEG LDPC

enters a node belonging to N_2 , parity is dropped such that $R_{N_2} = 0.504$. Similarly, for third quantization level, $R_{N_3} = 0.508$. We use uniformly generated random bits to fill in the dropped parity at intermediate processing nodes. Figure 9 shows the performance of proportionally fair OPERA with OPERA. We see that former provides very comparable performance to the latter. It is important to mention here the tradeoff in design of quantization levels. If the rate is increased significantly through parity drop with respect to primary rate, then the message may not remain decodable at the destination node. On the contrary, If very few bits are dropped, then we are not fair with resource constrained nodes. Thus, a balance should be maintained between performance and energy tradeoff when forming the quantization levels.

VI. CONCLUSION

We have presented an Optimal Progressive Error Recovery Algorithm that significantly enhances the decoding reliability at the receiver at the cost of processing within the network through partial decoding. The algorithm optimally maps the decoding iterations to the intermediate nodes and performs significantly better than random assignment. We give bounds on the performance of the algorithm for computational energy spent within the network for partial decoding of ensembles of LDPC codes. We also propose a framework that provides proportional fairness to individual nodes along with OPERA. Similar conclusions can be reached when OPERA is used with irregular LDPC codes though we have restricted our discussion in this work to regular LDPC codes due to brevity concerns.

ACKNOWLEDGMENT

The authors would like to thank Shirish Karande and Kiran Misra at Michigan State University for their valuable insights in the initial phase of the problem.

REFERENCES

- [1] Saad B. Qaisar, Shirish Karande, Kiran Misra, Hayder Radha, "Optimally Mapping an Iterative Channel Decoding Algorithm to a Wireless Sensor Network", *In Proceedings of IEEE International Conference on Communications (ICC)*, Glasgow, UK, June 2007.
- [2] Daniela Tuninetti and Christina Fragouli, "Processing Along the Way: Forwarding vs. Coding", *Proceedings of International Symposium on Information Theory and its Applications*, ISITA 2005, October 2004
- [3] T. Richardson and R. Urbanke, "The capacity of low-density parity check codes under message-passing decoding," *IEEE Transactions on Information Theory*, vol. 47, pp. 599-618, Feb. 2001.
- [4] Joao Barros and Sergio D. Servetto, "Sensor Reachback Problem", *IEEE Transactions on Information Theory*, November, 2003.
- [5] H. Radha, M. Vetterli, and R. Leonardi, "Image Compression Using Binary Space Partitioning Trees", *IEEE Trans. on Image Processing*, December 1996.
- [6] Gary J. Sullivan, "Rate Distortion Optimization for Video Compression", *IEEE Signal Processing Magazine*, November 1998.
- [7] G. M. Schuster, and A. K. Katsaggelos, "Rate- Distortion Based Video Compression, Optimal Video Frame Compression and Object Boundary Encoding", *Kluwer Academic Publishers*, 1997.
- [8] T. S. Rappaport. "Wireless Communications: Principles and Practice", *Prentice-Hall*, 1996
- [9] P. Gupta and PR. Kumar, "The Capacity of Wireless Networks", *IEEE Transactions on Information Theory*, 2000.
- [10] Robert G. Gallager , "Low Density Parity Check Codes", *PhD Thesis*, MIT, 1963.
- [11] Xiao-Yu Hu, Dieter M. Arnold, "Regular and Irregular Progressive Edge-Growth Tanner Graphs", *IEEE Trans. on Information Theory*, Vol. 51, January 2005.
- [12] H. Wymeersch, H. Steendam and M. Moeneclaey, "Log-domain decoding of LDPC codes over GF(q)", *In Proceedings of IEEE International Conference on Communications (ICC'04)*, Paris, France, June 2004.
- [13] L. Bazzi, T. Richardson, and R. Urbanke, "Exact thresholds for the binary symmetric channel and Gallagers decoding algorithm A", *IEEE Transactions on Information Theory*, Vol. 50, September 2004.
- [14] Marc P. C. Fossorier, Miodrag Mihajljevic and Hideki Imai, "Reduced Complexity Iterative Decoding of Low-Density Parity Check Codes Based on Belief Propagation", *IEEE Transactions on Communications*, Vol. 47, No. 5, May 1999
- [15] Samuel Madden, Michael J. Franklin, Joseph M. Hellerstein and Wei Hong, "TAG: a Tiny Aggregation Service for Ad-Hoc Sensor Networks", *In Proceedings of 5th Annual Symposium on Operating Systems Design and Implementation (OSDI)*, December 2002.
- [16] M. U. Ilyas and Hayder Radha, "Increasing Network Lifetime Of An IEEE 802.15.4 Wireless Sensor Network By Energy Efficient Routing", *Proceedings of IEEE International Conference on Communications (ICC)*, 2006.
- [17] M. Sartipi and F. Fekri, "Source and Channel Coding in Wireless Sensor Networks using LDPC Codes", *In Proceedings of ICSACN*, pp. 309-316, October 2004.
- [18] E. shih, S. Cho, N. Ickes, R. Min, A. Sinha, A. Wang, and A. Chandrakasan, "Physical layer driven protocol and algorithm design for energyefficient wireless sensor networks", *In Proceedings of ACM SIGMOBILE 2001*, July 2001.
- [19] Y. Sankarasubramaniam, I. F. Akyildiz, and S. W. McLaughlin, "Energy efficiency based packet size optimization in wireless sensor networks", *In Proceedings of IWSN*, 2003.
- [20] Ramesh Govindan, "IEEE COMSOC Wireless Sensor Networks Tutorial", <http://www.comsoc.org/freetutorials/nsc/>, 2006.
- [21] T. Bonald, L. Massoulie, A. Proutiere and J. Virtamo, "A Queuing Analysis of Max-Min Fairness, Proportional Fairness and Balanced Fairness", *Queuing Systems*, 2006.

APPENDIX

COEFFICIENTS $\alpha(\epsilon_{ind}), \beta(\epsilon_{ind}), \gamma(\epsilon_{ind})$ AND $\psi(\epsilon_{ind})$

TABLE I

 $\epsilon_{min} = 0.01, \epsilon_{max} = 0.08, l_{max} = 150, R = 0.5$
PEG (3,6) REGULAR LDPC

ϵ_{ind}	$\alpha(\epsilon_{ind})$	$\beta(\epsilon_{ind})$	$\gamma(\epsilon_{ind})$	$\psi(\epsilon_{ind})$
1	0.000000	0.000000	0.000000	0.000000
2	0.000000	0.000000	0.000000	0.000000
3	0.000000	0.000000	0.000000	0.000000
4	-0.694600	-0.376200	0.712600	-0.384100
5	-0.971500	-0.471100	0.998500	-0.478300
6	-1.170000	-0.481300	1.206000	-0.488300
7	-1.126000	-0.415400	1.167000	-0.422300
8	0.311300	-1.090000	-0.307900	-1.377000
9	0.175600	-0.921000	-0.266100	-1.963000
10	0.435400	-0.668200	-0.389400	-0.748000
11	0.050130	-0.365200	0.017250	-0.368100
12	0.068400	-0.271100	0.000198	0.000539
13	0.009578	-0.040290	0.063590	-0.245600
14	0.065780	-0.144100	0.005974	-0.001183
15	0.054300	-0.145700	0.022660	-0.000154

TABLE II

 $\epsilon_{min} = 0.01, \epsilon_{max} = 0.07, l_{max} = 150, R = 0.5$
IRREGULAR PEG LDPC, $\lambda(\theta) = 0.207\theta^6 + 0.271\theta^2 + 0.522\theta^1$

ϵ_{ind}	$\alpha(\epsilon_{ind})$	$\beta(\epsilon_{ind})$	$\gamma(\epsilon_{ind})$	$\psi(\epsilon_{ind})$
1	0.000000	0.000000	0.000000	0.000000
2	-0.000258	-0.610100	0.000454	-1.723000
3	0.000002	-5.950000	0.000363	-1.256000
4	0.000003	-5.950000	0.000677	-1.256000
5	-0.000077	-0.210800	0.04203	-1.317000
6	-0.015280	-1.379000	0.015970	-1.450000
7	-0.015710	-1.871000	0.015740	-1.921000
8	-0.035600	-1.675000	0.036030	-1.705000
9	-2.600000	-1.424000	2.601000	-1.425000
10	-3.202000	-1.709000	3.203000	-1.709000
11	-3.537000	-1.733000	3.538000	-1.734000
12	-0.084020	-1.636000	0.085300	-1.653000
13	0.067580	-2.064000	-0.067170	-2.050000

TABLE III

 $\epsilon_{min} = 0.01, \epsilon_{max} = 0.08, l_{max} = 150, R_{N_2} = 0.504$
PEG (3,6) REGULAR LDPC

ϵ_{ind}	$\alpha(\epsilon_{ind})$	$\beta(\epsilon_{ind})$	$\gamma(\epsilon_{ind})$	$\psi(\epsilon_{ind})$
1	0.000000	0.000000	0.000000	0.000000
2	-0.664300	-0.477300	0.680300	-0.485300
3	-0.654500	-0.419800	0.670900	-0.427800
4	-0.523300	-0.352400	0.539600	-0.360400
5	-1.462000	-0.588100	1.498000	-0.595300
6	-1.270000	-0.432800	1.311000	-0.440000
7	0.061230	-0.898800	0.000000	0.000000
8	-1.137000	-0.324100	1.183000	-0.331400
9	-1.288000	-0.313100	1.345000	-0.319900
10	-1.205000	-0.252000	1.268000	-0.259100
11	-0.966700	-0.169100	1.029000	-0.176300
12	-0.774700	-0.118300	0.836300	-0.125400
13	0.067570	-0.228000	0.000000	0.000000
14	-0.556300	-0.068210	0.625500	-0.075870
15	0.064710	-0.072830	0.003835	0.000961

TABLE IV

 $\epsilon_{min} = 0.01, \epsilon_{max} = 0.08, l_{max} = 150, R_{N_3} = 0.508$
PEG (3,6) REGULAR LDPC

ϵ_{ind}	$\alpha(\epsilon_{ind})$	$\beta(\epsilon_{ind})$	$\gamma(\epsilon_{ind})$	$\psi(\epsilon_{ind})$
1	-0.083570	0.045650	0.086160	0.0366700
2	-0.500600	-0.418700	0.513200	-0.427000
3	-0.479600	-0.333800	0.492300	-0.342000
4	-0.484000	-0.392900	0.498500	-0.401000
5	0.377700	-2.243000	-1.802000	-4.129000
6	-0.778100	-0.440300	0.807500	-0.447900
7	0.042770	-0.917000	0.000000	0.000000
8	-0.793700	-0.339400	0.825300	-0.346800
9	-0.720000	-0.306800	0.755100	-0.313700
10	-0.592300	-0.202700	0.628600	-0.210000
11	-0.375000	-0.170400	0.409400	-0.177600
12	0.101700	-0.298300	-0.070390	-0.414400
13	0.026740	-0.142400	0.009281	-0.159600
14	0.033800	-0.112100	0.004583	-0.060040
15	0.022370	-0.058430	0.014070	-0.0001169