Extending Lifetime of Wireless Sensor Networks using Forward Error Correction

Donapudi, S U; Obel, C O; Madsen, Jan

Published in:
24th Norchip Conference, 2006.

Link to article, DOI:
10.1109/NORCHP.2006.329226

Publication date:
2006

Document Version
Publisher's PDF, also known as Version of record

Extending Lifetime of Wireless Sensor Networks using Forward Error Correction

Suresh Upandra Donapudi, Christian Olesen Obel and Jan Madsen

email:sureshupandra@ieee.org, christian@coo.dk and jan@imm.dtu.dk
Informatics and Mathematical Modelling
Technical University of Denmark
Richard Petersens Plads, building 322, DK-2800 Kgs. Lyngby, DENMARK

Abstract— **Communication between nodes in Wireless Sensor Networks (WSN) is susceptible to transmission errors caused by low signal strength or interference. These errors manifest themselves as lost or corrupt packets. This often leads to retransmission, which in turn results in increased power consumption reducing node and network lifetime.**

In this paper, we implement and evaluate a convolution code FEC with Viterbi decoding on Mica2 nodes to explore the possibility of extending the lifetime of a degrading WSN. We present results which suggest that our approach could be used in a WSN when increasing distance and channel noise degrade the network.

I. Introduction

Wireless Sensor Networks are destined to play an important role in several key areas like environmental monitoring, military and civilian surveillance, health care, etc., delivering pervasive computing and communication. Most of these environments are noisy and hence an effective technique to combat the noise could prove to be vital. Needless to say, these applications require long lived mobile nodes to set up reliable data communication in ad-hoc networks in order to communicate between base stations and distributed applications. These battery powered sensor nodes are extremely power constrained devices and hence only low-power microcontrollers and radios can be afforded. A network of these devices has limited energy ressources and to get the most out of it one often has to incorporate power management and a good communication strategy.

In a WSN, over time some nodes die due to depletion of energy, pushing the surviving nodes closer to the border of their radio's operational range. As a consequence the receiving signal strength is lower and the communication is more susceptible to channel noise which causes packet corruption. This will lead to increased retransmissons and further depletion of the remaining energy resources. Introducing Forward Error Correction on to the data transmission helps lower the number of retransmissions and thus help conserve energy in such a degrading network. This of course consumes additional resources, but when employed selectively it can prove beneficial.

This paper aims at finding a balance between the communication and computational overheads in order to prolong the life of a degrading WSN. Our scheme is evaluated in software on the Mica2 [1] platform running TinyOS [2].

II. Related Work

Most research aimed at extending the lifetime of wireless sensor networks has focused on energy-aware routing techniques. The basic power-aware routing scheme of [3] selects routes by preferring routes through nodes with long remaining battery lifetime. Although selecting the minimum energy path is of great importance, avoiding nodes with low residual energy is another important challenge, particularly towards the end of the network's lifetime. [4], [5] propose techniques to load-balance the routing in order to account for this challenge.

In this paper, we try to approach the problem of extending the lifetime by enhancing the communication at the physical layer. This could potentially allow for simple and less costly routing schemes. Jaein Jeong and Cheng Tien Ee in their work [6] propose some FEC techniques based on Linear Block Codes to correct single and double bit errors. The modified SECDED $(16, 8)$ code described in their work can correct upto two errors. However, as we gather, little work has been done to employ FEC as a solution to extend the lifetime of a network. The motivation for our work is to investigate this aspect using a different class of FEC technique based on convolution codes.

III. Coding principles

A convolution code can be described as an (n, k, K) code which encodes k information bits into n bits. The encoding can be defined as a function $f(u) = v$ of k input bits resulting in n output bits. The function $f()$ defines the convolution of the input u with the binary generating polynomial g. Hence $f()$ can be written as (1), where g_0 and g_m are =1.

$$V_j = \sum_{j=0}^{m} g_i U_{j-i} \tag{1}$$

The rate of the code is k/n and K is the constraint length of the code which represents the k-tuple stages in the encoding shift register. We use a rate $1/2$ code with constraint length 5. The generating polynomial is $g = [11.01.00.11.11]$ which can also be written as $[G_0, G_1] = [10011, 11011]$. Figure 1 illustrates the circuit for encoding. G_0 and G_1 are the two connection vectors for the shift register and the addition

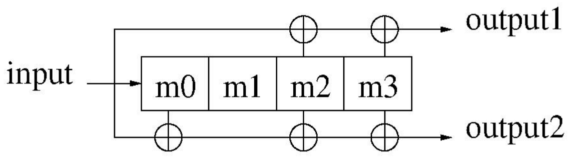

Fig. 1. Schematic of a rate 1/2 K=5 Convolution encoder.

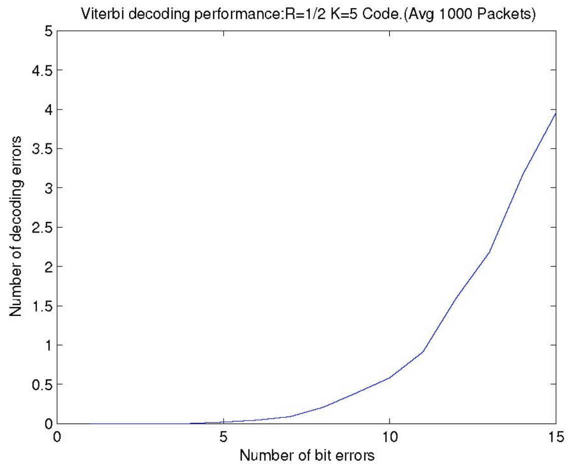

Fig. 2. Performance plot for Viterbi decoder

operation illustrated is $modulo2$ sum. The encoding process is a simple shift operation.

Convolution codes can be decoded by performing a Maximum Likelihood (ML) estimation of the received sequence. This process can be described as a search for a sequence that disagrees with the received sequence in the least number of positions [7]. This search is efficiently done using the Viterbi decoder.

A. Discussions

We have chosen the convolution code which has good distance properties. It has a free distance $d_f = 7$ and hence can correct any error pattern of t errors where, $t <= (d_f - 1)/2$ if N (length of encoded frame) is sufficiently large. This means that this code can handle even a burst of 3 bits. An $R = 1/2$ code can cope with up to 5% of Bit Error Rate (BER) in the channel. In this project a 64 bit message is encoded into 136 bits, hence we could correct up to 6 bit errors. The chances of having consecutive multiple bit errors in this short packet is very low. It can be seen from the results section of this paper that very few packets encounter a burst error longer than 3 bits. However, if the nodes were to operate in a environment which experienced bursty noise we could interleave the encoded data before transmission and de-interleave it at the receiver before decoding. This way even a burst error will be spread out over the packet as multiple single/double bit errors. Performance simulation results of chosen code can be seen in figure 2.

IV. Design and Implementation

This section presents assumptions that influenced the design and implementation of our scheme.

A. Data corruption

The radio transmission is subject to interference which causes corruption or total loss of data. Based on the nature of the interference the following scenarios arise:

- **A packet is never received.** This is most likely due to the receiver not being able to synchronize its radio to the transmission or corruption of the packet header. The synchronization could fail due to low receiving signal strength or noise interfering with the sensing of the preamble. Channel noise or packet collision can cause header corruption. If either of this happens the packet will either not be received or discarded by the application.
- **A packet's payload is corrupted.** Payload forms the bulk of the packet and hence is the most susceptible to corruption. We restrict ourselves to an FEC scheme which protects only the payload part of the packet. Protecting the header could potentially reduce the number of unreceivable packets, but it is outside the scope of this work.

In all our further calculations and experiments we restrict our analysis only to the packets that we actually receive.

B. Transmission cost model

In order to evaluate our FEC scheme we use the following simple model. Let C_t be the cost of transmitting one packet in an ideal network where the packet will not suffer any corruption and hence will not have to be retransmitted. For a network with a certain packet loss ratio p_t (corrupted-recieved ratio), the number of retransmissions would be: $p_t + p_t^2 + p_t^3 + \cdots$ Hence the effective cost of transmitting a packet, C_t', can be compued as follows.

$$\begin{aligned} C_t' &= C_t(1 + p_t + p_t^2 + p_t^3 + \cdots) \\ &= C_t \sum_{n=0}^{\infty} p_t^n = C_t \left(1 + \frac{p_t}{1 - p_t}\right) \end{aligned} \qquad (2)$$

We use the above equation (2) to calculate the net cost of transmission for each packet where the subscript t indicates the type of transmission (FEC: t = F, noFEC: t = N).

C. TinyOS Applications

The following applications are the main components of our experimental setup.

- **FecEnc** is the encoder application in this project and runs on the mobile node. When the node is powered on, the application initializes the radio and a timer. The timer is configured to fire every 256 binary miliseconds (0.25 seconds) and posts a TinyOS task that will create, encode and send a new message. Encoding is implemented as a function in NesC which encodes 8 bytes of data into 17 bytes (encoded data is 16 bytes $+$ 1 byte for trellis termination). The encoder is implemented as a shift-register-like function in software.
- **FecDec** is the decoder application that runs on the base station node. When the base station is powered on,

the application awaits for the `ReceiveMsg.receive` event. The event will post a TinyOS task which will read the encoded data from the buffer, decode it and forward it to the serial port. In order to reduce the computational complexity of the Viterbi decoder, the program uses lookup tables to perform branch and error metric calculation with minimal loops.

- **TOSbase** is a simple application that comes with the TinyOS distribution. It serves as a base station and forwards packets between it's radio and serial interface. We can view this as FecDec without the Viterbi decoding. Hence, we use this as a benchmark to estimate the additional costs due to Viterbi decoding.

V. Experiments

The experiments were carried out in two phases. Phase 1 of the experiment was to characterize the Mica2 radio (without FEC) and in phase 2, we evaluate our FEC scheme by repeating similar tests. However, results from both these phases are presented together for the sake of brevity and clarity. The test setup is an outdoor parking space in front of building 341 at the Technical University of Denmark. This location was chosen due to the high number of datalabs and wireless networks in the vicinity which we suspect would constitute a reasonably noisy environment. The setup is a Mica2 node running our **FecEnc** application and a base station running our **FecDec**. The Mica2 node sends 250 packets with FEC encoded payloads and the base station decodes and records what it receives. Measurements are taken at distances of 0m, 10m, 20, 30m, 35m and 40m and the measurements are repeated four times at every distance. The received packets are stored in a file which is compared to the known messages in order to compute the following statistics:

- Total number of packets received.
- Number of corrupt packets and number of errors per packet.
- Number of successfully decoded packets.

A. Cost calculations

Performing FEC adds additional computational burden on the nodes processor. To determine this additional cost we conducted a simple experiment in which we measured the current drawn by the node while running TOSbase, which turned out to be 16 mA averaged over 10 seconds. In a similar measurement FecDec draws 20mA. At a packet transmission rate of 4 packets per sec we can roughly assume that the cost for each packet of TOSbase is 4mAs and for FecDec 5mAs. We use these results together with the results from the following section to demonstrate the scenario where our scheme can improve energy utilization.

B. Results and Analysis

The results from phase 1 are as in the table below. They are labelled 'Without FEC' in figure 3 which shows the number of packets lost as a function of the transmission distance. From these results we see, as expected, that the number of packets successfully received decrease when the transmission distance is increased. For short distances almost all packets are received. For longer distances the border is around 30 - 40m where the communication deteriorates fast, understandably due to decreasing signal strength combined with the effect of channel noise.

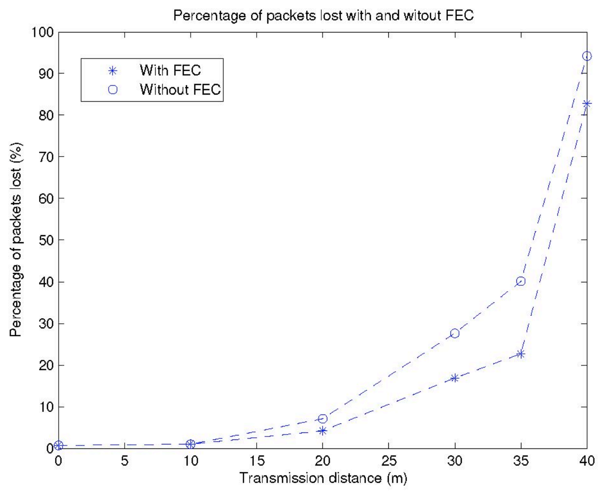

Fig. 3. Number of packets lost during transmission

The recorded loss in packets is due to two factors. One, the packets were never received and two, received packets had bit errors and failed the CRC check. Now with this point in view, if we look at the results obtained from phase 2 labelled 'With FEC' in figure 3 we can see that most of these packets with bit errors were actually successfully received. The plot in figure 4 shows that in the region of 30-40m a large percentage of the packets suffered bit errors and hence will be discarded by the CRC check without the FEC. In effect we could gain a 10 to 40 % increase in the number of successfully received packets in this range by using FEC.

Distance (m)	0	10	20	30	35	40
Recieved	993	990	958	831	773	172
Lost	7	10	42	169	227	828
With 0 errors	993	990	929	724	599	60
With 1 error	0	0	27	90	130	32
With 2 errors	0	0	2	12	25	30
With 3 errors	0	0	0	4	12	11
With 4 errors	0	0	0	0	4	10
With 5 errors	0	0	0	1	2	11
With 6 errors	0	0	0	0	0	5
more than 6	0	0	0	0	1	13
Total with errors	0	0	29	107	174	112

Of all the received packets one packet at 35m and 8 packets at 40m were decoded incorrectly owing to long error bursts and > 10 bit errors. By using the CRC flag set by the radio these erroneously decoded packets can be discarded.

The cost of transmitting a packet as a function of increasing distance is plotted in figure 5. The packet loss ratios P_N and P_F vary as a function of this distance. One should note that these could just as easily also vary with change in channel characteristics. From this figure we see that the

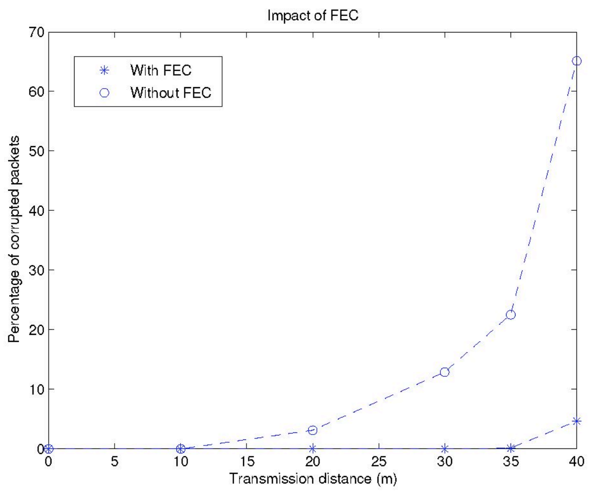

Fig. 4. Mitigation in packet corruption by FEC

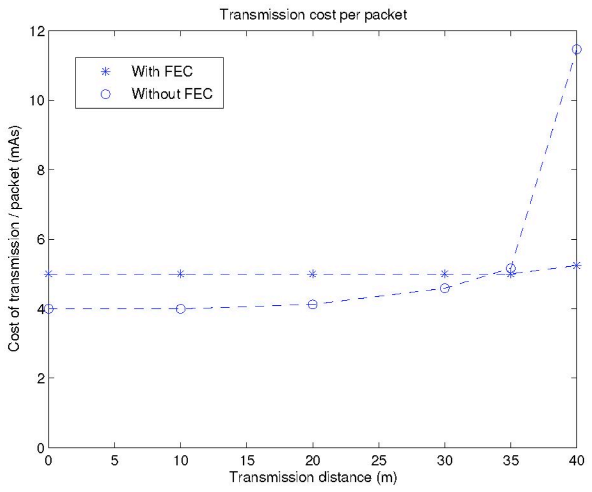

Fig. 5. Cost of packet transmission, assuming 4 packets/s

cost of transmitting a FEC packet is fairly constant and at shorter distances (lower P_F and P_N) is higher than that of a non-FEC packet. However the interesting part of the results is that at larger distances (higher P_F and P_N) the cost of transmitting a packet without FEC is far greater than that of our FEC protected packet. The border of 30-40m is where we stand to gain by using our proposed scheme.

VI. Discussions

The lifetime of any WSN will be determined by a large number of factors like the duty cycle of the nodes, the application the network is meant to serve, amount of communication, etc. For any such network we would like to squeeze the most out of the limited energy resources available and extend the lifetime as much as possible. Nodes in a network often drain their energy resources unevenly. This is because different nodes could possibly route diffrent amount of data on the network, perform different activities in their part of the network, etc. In such a case the surviving nodes could still be at a high energy level, hence there is an interest to keep the network running.

As discussed in section II, advanced routing protocols aim to ensure even depletion of energy amongst nodes in the WSN. These protocols often have large implementation and computation overheads. On the contrary, our proposed scheme suggests that we could possibly tackle this problem with a simpler strategy. The degradation of the network (increasing channel noise / increasing distance between nodes) can for instance be determined using RSSI (Received Signal Strength Indicator) and comparing it to a pre-determined threshold. This can then trigger selected nodes, regions or even the entire WSN to switch to our FEC scheme and thereby keeping the network alive for longer.

VII. Conclusion

In this paper we have demonstrated that by using our proposed FEC scheme we can increase the number of successfully received packets around the radio's operational border. We demostrate a simple method to estimate the cost of our scheme. The actual lifetime increase is best estimated through simulation or observation of a real application.

However, we note that the following issues need further investigation before we can make more conclusive deductions. A more detailed breakdown of power consumption on the node is essential to accurately measure the overhead of FEC. The percentage of packets lost completely as opposed to those that were corrupted is fairly high. An investigation into better preamble sensing and possibly a header protection scheme could give us directions to solve this problem. In such a case FEC would prove to be more beneficial since the difference in packet loss ratios for FEC and non-FEC scheme could widen.

The software approach for our FEC scheme is fast to implement, however an efficient dedicated FEC hardware co-processor could prove to be more cost efficient. This would push the transmission cost lower, enabling us to switch to this scheme much earlier and thereby saving more energy.

References

[1] Mica2 documentation, *http://www.xbow.com/Support/Support_pdf_files/MPR-MIB_Series_Users_Manual.pdf*
[2] The TinyOS website, *http://www.tinyos.net*
[3] S. Singh, M. Woo, and C. Raghavendra, *Power-aware routing in mobile ad hoc networks*, Proceedings of ACM MobiCom, 1998
[4] J.-H. Chang and L. Tassiulas, *Energy Conserving Routing in Wireless Ad-Hoc Networks*, Proceedings of IEEE INFOCOM, 2000
[5] R. C. Shah and J. Rabaey, *Energy Aware Routing for Low Energy Ad Hoc Sensor Networks*, Proceedings of IEEE WCNC, 2002
[6] J. Jeong and C. T. Ee, *Forward Error Correction in Sensor Networks*, University of California, Berkley
[7] J. Justesen and T. Hoeholdt, *A Course In Error-Correcting Codes*, European Mathematical Society, 2004