Editorial Manager(tm) for Wireless Networks
Manuscript Draft

Manuscript Number:

Title: Energy-efficient Cluster-based Cooperative FEC in Wireless Networks

Article Type: Manuscript

Section/Category:

Keywords: Wireless networks; cluster; link layer cooperation; code combining

Corresponding Author: Su Yi, Ph.D.

Corresponding Author's Institution:

First Author: Su Yi

Order of Authors: Su Yi; Shivkumar Kalyanaraman; Babak Azimi-Sadjadi

Manuscript Region of Origin:

Abstract:

Energy-efficient Cluster-based Cooperative FEC in Wireless Networks

Su Yi, Shivkumar Kalyanaraman, and Babak Azimi-Sadjadi
Department of ECSE, Rensselaer Polytechnic Institute, TROY, NY 12180
yis@alum.rpi.edu, {shivkuma,babak}@ecse.rpi.edu

***Abstract*— In this paper, we introduce a novel link layer cooperation technique in noisy wireless networks to improve overall system throughput and reliability, and to reduce the cost of retransmission and energy consumption. Under a cluster-based network design, code combining [1] is used together with FEC to improve the link layer reliability. This approach is different from how code combining is used in the conventional hybrid ARQ, which is in a sequential way. The analytical results and the simulations show that with the cooperation of nodes in a clustering network, the link reliability will be greatly improved with the same power consumption. We also show that not only transmission power is greatly reduced, but also the aggregate power consumption for a successful transmission and reception. Moreover, a lower transmission power implies lower interference thus potentially increase the network capacity.**

***Index Terms*— Wireless networks, cluster, link layer cooperation, code combining.**

I. Introduction

Multi-hop ad hoc networks or sensor networks are increasingly essential for commercial infrastructures, military settings, crisis monitoring, and public safety. Our objective in this paper is to make these networks more powerful by increasing their overall throughput, service capability, and individual unit flexibility. This work provides further benefits by improving the power efficiency of the network nodes, which implies smaller, more affordable units, permitting the deployment of more units.

This paper aims towards adaptively using distributed cooperation techniques in wireless multi-hop ad hoc or sensor networks. These techniques are intended to improve overall system throughput, reduce the cost of node elements, and extend the units' service lives. To this end, tools from multi-disciplinary areas of error correction and detection (FEC), combining techniques, cluster-based forwarding and routing, are to be employed.

This work is supported in part by NSF under contract number NSF-ITR 0313095, and by a grant from Intel Corp.

In this paper we present a new link layer cooperation scheme for multi-hop wireless networks and sensor networks to improve the overall channel quality for each transmitter/receiver pair. For this, we propose to extract diversity gain out of the redundancy inherently present in all broadcast network transmission, such as wireless sensor networks, and direct those gains for chosen receiver nodes. The redundancy in such systems is present since the signal carried over such a channel is received (if not necessarily detected) by all nodes within transmission radius. Thus, in this distributed cooperative paradigm, packets are not relayed from one network node to the next, but from one cluster of nodes to the next cluster of nodes, until it reaches its destination. Grouping the network nodes into collaborating groups enables the system to gain from cooperative diversity, cooperative error recovery, network-layer cluster auto-configuration and cooperative cluster-based routing. This research strives to design cooperation techniques that permit the nodes for such networks to be as simple, small, flexible, long-lasting, and affordable as possible.

We consider a large collection of autonomous nodes or terminals that communicate with each other by forming a multi-hop wireless network and maintaining connectivity in a decentralized manner. Cooperation among nodes can be done in different communication layers. Fig.1 shows cooperation in the physical layer and in the link layer.

In the physical layer, cooperative nodes share their information to improve the channel quality using transmit and/or receive diversity (Fig.1a and 1b). Physical layer cooperation has been studied recently under the subject name of "cooperative diversity". In cooperative diversity the transmitting nodes use the nodes in the neighborhood of the transmitter and the receiver as relays [3], [4], [5], [6], [7], [8], [9], [10], active scatterers [11], or simply clusters of cooperating nodes [12], [2], to reduce the adverse effect of multipath fading in the wireless channel.

In this paper we take a different approach and we use cooperation in the link layer. If the Signal to Noise Ratio (SNR) of the received signal is moderately high,

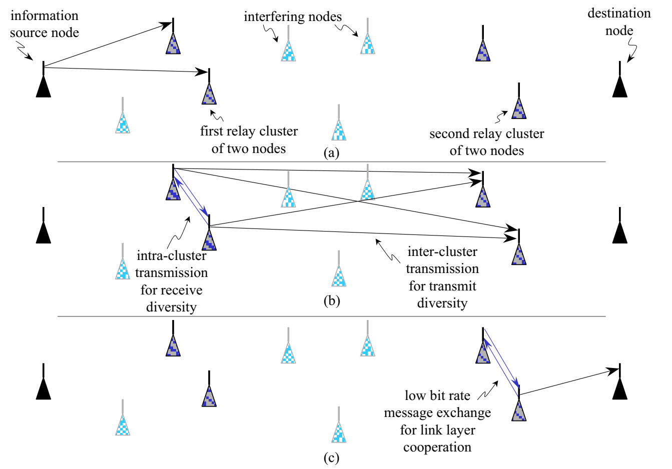


Fig. 1. Transmitting nodes group into cooperative clusters to relay the information from the source to the destination [2]. (a) The information source reaches the first relay cluster. (b) The nodes in the relay cluster share their information for diversity gain. Then they relay the information to the next cluster. (c) The next cluster has a reliable channel with the destination node, hence there is no need of physical layer cooperation. A single node can relay the information to the final destination node.

one can avoid physical layer cooperation to save on the bandwidth used for information sharing and synchronization [12], [2] and instead use the link layer cooperation to increase the overall throughput of the network. In the link layer cooperative transmission the cooperating nodes decode the received packets (instead of the individual bits/symbols done in the physical layer cooperation) and participate in the cooperative transmission of the error-free packets. The link layer cooperation can be implemented in following stages depending on the quality of the link:

Stage 1: Cluster head decides if cooperation is necessary. Unlike the node to node cooperative cluster transmission, a packet is successfully received if at least one node in the cluster receives the packet without error. The nodes with the error free packet send their status to the cluster head using a low bit rate message. The cluster head chooses one of the nodes with the error free packet to forward that packet to the next cluster.

Stage 2: FEC and Code combining among cluster nodes. If no node receives the packet successfully, the cooperating nodes can combine their erroneous packets and use code combining techniques [1], [13], [14] to reconstruct the packet. FEC can be designed over the entire frame to facilitate code combining.

Stage 3: ARQ or transmit diversity. If the reconstruction is unsuccessful the master node sends an ARQ to the previous cluster for the packet retransmission. Or, if the forward channel quality is too low for any individual transmitter, the master node can recruit several transmitting nodes to use cooperative transmit diversity. If the number of nodes with error free packets is not enough to satisfy the desired BER the cooperating nodes share their data with other nodes in the cluster to recruit them for cooperation.

The main technique in this link layer cooperation is the use of the well-known code combining. In the conventional type I hybrid ARQ scheme with code combining, the repeated packets are sent upon each request [15]. This retransmission based method can be considered a redundancy in time. In our new cooperative link layer paradigm, retransmission can be greatly reduced or avoided by making use of the wireless broadcast nature. In fact, the retransmission is replaced by information sharing among the nodes in the receiving cluster. In other words, we use the existing parallel channels between the transmitting node and the receiving nodes for code combining. This can be called redundancy in space. This method is well-suited for interactive real-time communication streams where waiting for retransmission introduces unacceptable delay and jitter. However, the cost for the node cooperation is the extra power and bandwidth used for the intra-cluster communication.

The rest of this paper is organized as follows: The performance analysis for the link layer cooperation is given in Section II. In Section III we present our simulations and results and in Section IV we touch upon some design issues for the cooperative network. Finally in Section V we give our concluding remarks and we lay out future work.

II. Performance Analysis for Link Layer Cooperation

A. Preliminaries

We now assume nodes are already clustered using some existing clustering protocol, like LEACH [16] and there are enough nodes in one cluster to cooperate. Moreover, all the cluster nodes are in one hop distance to the cluster head. The packets in each cooperative node will be sent to the cluster head for code combining. So the number of repeated packets is identical to the number of cooperative nodes. Throughout the whole chapter, L represents the number of nodes joining the cooperation. This is equivalent to the repeated packets in code combining.

In the cooperative cluster, the member nodes will transmit their received packets to the cluster head if necessary. The distance between the nodes in the cluster is much smaller than the distance between the transmitter and the receiver from different clusters. Therefore, the required intra-cluster transmission power is much smaller than the power of the inter-cluster transmission. In general the bit error rate for inter-cluster channel and intra-cluster channels are different. Let p_1 and p_0 be the bit error rate for the inter-cluster channel and intra-cluster channel, respectively. Therefore, a single bit traveling from the source to the cluster head via a member node, has the bit error probability equal to $p = p_1 + p_0 - p_1 p_0$.

How the code combining technique is used among the cooperation procedure can be illustrated in Figure 2.

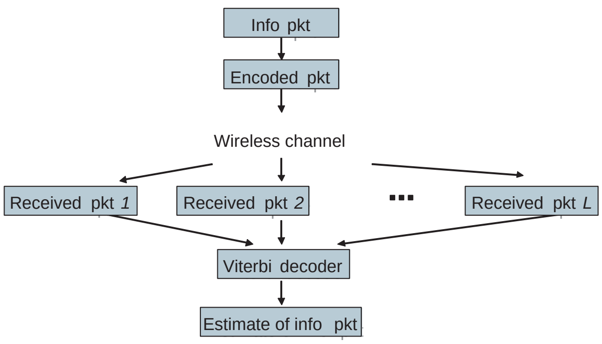


Fig. 2. Block Diagram of Link Layer Cooperation with Code Combining Technique

Code combining [1] represents a technique for combining L repeated packets encoded with a code of rate R to obtain a lower rate, R/L, and thus more powerful, error-correcting code, capable of allowing more channel errors. Code combining is designed to work in a very noisy environment, where conventional diversity combining concepts [17], [18], [19] can easily break down. One feature of code combining is that the maximum-likelihood (ML) decoder will select the codeword m which maximizes the conditional probability between the received sequence $\mathbf{r}$ and the repeated codeword denoted by $\mathbf{v}_m$. Repeated codewords are transmitted over BSC channels with bit error rate p_i for $i = 1, 2, \cdots, L$. The decoding function can be written as

$$\max_m \left\{ \mathbf{P}[\mathbf{r}|\mathbf{v}_m] = \prod_{i=1}^{L} (1 - p_i)^{N - d_{mi}} p_i^{d_{mi}} \right\} \quad (1)$$

where d_{mi} is the number of bit disagreements for the ith codeword, and N is the pre-combined codeword length. An alternate way to write (1) is

$$\min_m \sum_{i=1}^{L} w_i d_{mi} \quad (2)$$

where weight (reliability factor) $w_i = \log \frac{1-p_i}{p_i}$.

B. Code Combining with Convolutional Codes in a Uniform Channel Condition

It can be seen from (1) that if a block code is used for code combining, the complexity of the decoder depends greatly on the number of codewords, which increases exponentially with the codeword length n. Therefore, to reduce the decoding complexity, we want the codeword length to be small. This will limit the use of block codes, since block codes are efficient in large blocks. For this reason, code combining is generally used for convolutional codes or for short block codes. Due to the large amount of redundancy short block codes deliver for data networks, we propose convolutional codes for the cooperative FEC in this paper. Nevertheless it is still possible that block codes are used. For completeness, we include the performance of code combining with block codes in Appendix. For the rest of this section we analyze the performance of the code combining technique for convolutional codes. We adopt the notation used in [20].

Let's use a rate 1/3 nonsystematic feedforward convolutional encoder with memory order $m = 2$ as example. The block diagram of the encoder is shown in Fig.3. This encoder consists of $k = 1$ shift register with $m = 2$ delay elements and with $n = 3$ modulo-2 adders. The modulo-2 adders can be implemented as Exclusive-OR (XOR) gates. Since modulo-2 addition is a linear operation, the encoder is a linear system.

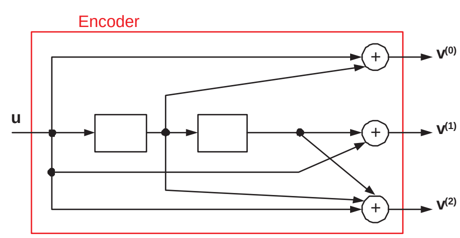


Fig. 3. A (3,1,2) Binary Nonsystematic Feedforward Convolutional Encoder

If we look the encoder in Fig.3 as a black box with $k = 1$ input bit and $n = 3$ output bits, L identical encoders then form a virtual low rate $1/3L$ nonsystematic feedforward convolutional encoder. This 1-input $3L$-output virtual encoder is shown in Fig.4. The interesting feature is that the output bits of each identical sub-encoder are identical too. This encoder does not actually exist since each packet is encoded using the (3,1,2) sub-encoder. It only represents the encoding structure when we put the L repeated packets together. When some error occurs in one or more of the repeated packets, the output bits will be no longer the same.

If the cooperating nodes are close (relative to the

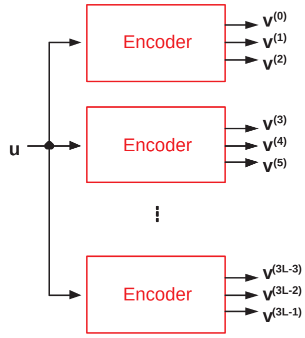


Fig. 4. The Virtual ($3L$,1,2) Convolutional Encoder with L Combined Encoder

distance between the transmitting node and the cluster head) to each other and close to the cluster head, the signal to noise ratios for all nodes are almost the same. In this case, the received packet weights w_i used in the code combining technique are the same for all the cooperative nodes, thus can be ignored. This scenario is referred as *uniform channel condition*.

For general convolutional codes with maximum likelihood decoding (Viterbi algorithm), the *bit error probability*, P_b, that is, the expected number of information bit errors per decoded information bit, is used to evaluate the performance of Viterbi algorithm. This bit error probability can be approximated by (upper bound):

$$P_b \approx B_{d_{free}} \left[2\sqrt{p(1-p)} \right]^{d_{free}} \tag{3}$$

where $B_{d_{free}}$ is the coefficient of $X^{d_{free}}$ in the bit *weight*[1] *enumerating function* (WEF) $B(X)$, and d_{free} is the *minimum free distance*.

In code combining the decoder receives L corrupted copies of the transmitted packets. A k-input n-output convolutional code with rate $R = k/n$ with L repeated packets, can be modelled by a k-input nL-output convolutional code with rate R/L. The Viterbi decoder for this rate R/L convolutional code has exactly the same trellis structure as the original rate R convolutional code. The only difference is how the metric for each branch of the trellis is calculated. Therefore, the decoder for the code combiner and the decoder for the original convolutional code have the same order of complexity. Furthermore, it is easy to see that the WEF of the R/L rate convolutional code, $B_L(X)$, has the following relation with the WEF of the original code:

$$B_L(X) = B(X^L) \tag{4}$$

[1]It is unfortunate that we use the term “weight” both for the measure of the quality of a link (w_i) and for the number of ones in a binary sequence (d or $W(\cdot)$).

Hence the lowest power of X in $B_L(X)$ is Ld_{free}, i.e., $d_{free}(L) = Ld_{free}$, and

$$P_b(L) = B_{d_{free}} \left[2\sqrt{p(1-p)} \right]^{Ld_{free}} \tag{5}$$

In this expression, p refers to the transition probability of a BSC channel.

C. Code Combining with Different Channel Conditions

The assumption made in Section II-B is mainly valid when code combining is used together with hybrid ARQ, where the same channel is used for packet retransmission. However, in a cluster-based cooperation system, the channel condition can vary significantly among nodes. This is due to the different path losses caused by the different distances between receiver nodes and the transmitter. For this reason, the packets received with higher SNR should have higher weights in the decoder at the master node. The following part in this section will discuss the performance analysis of the weighted code combining. The results depend on the well-known performance bound for convolutional codes using Viterbi decoding, which is described in the following fact:

Fact 1: Using the analysis of the maximum-likelihood path selection on a trellis diagram, the error probability of a convolutional code with optimum decoding can be upper-bounded using a union bound, by the sum of the error probabilities of each of the paths. The bit-error probability, that is, the expected number of information bit errors per decoded information bit, can be approximated by:

$$P_b < \sum_{d=d_{free}}^{\infty} B_d P_d \tag{6}$$

B_d is the total number of nonzero information bits on all weight-d paths, divided by the number of information bits k per unit time (i.e., the coefficient of the weight-d term in the bit WEF $B(X) = \sum_{d=d_{free}}^{\infty} B_d X^d$ of the decoder). P_d is the event error probability for the weight-d path. This bound is tight, because P_d is very small. Therefore the union bound is the dominant part for the whole probability of error. $\diamond$

B_d is determined by the encoder. P_d has a nice expression for ordinary Viterbi decoding over a BSC channel. In weighted code combining, the result for P_d is more complicated. We assume that the decoder is aware of the channel condition for each cooperative node (this can be achieved by piggybacking extra bits during intra-cluster transmission process). Using the channel conditions, the decoder assigns the weight $w_i = \log \frac{1-p_i}{p_i}$

to the ith repeated packet according to the channel error rate p_i, for $i = 1, ..., L$.

A path with weight d would have the weight Ld when the code combining of order L is used. Let the *pseudo codeword* made of bits in these d positions for the correct path be $\mathbf{v}$, the corresponding pseudo codeword for the incorrect path be $\mathbf{v}'$, and the received set of packets be $\mathbf{r} = \{\mathbf{r}_1, \cdots, \mathbf{r}_L\}$. $\mathbf{r}_i$ is the ith received repeated packet. The path metric for $\mathbf{r}$ and $\mathbf{v}$ is given by

$$M(\mathbf{r}|\mathbf{v}) = \sum_{i=1}^{L} w_i d(\mathbf{r}_i, \mathbf{v}) \quad (7)$$

where $d(\mathbf{x}, \mathbf{y})$ is the Hamming distance between codewords $\mathbf{x}$ and $\mathbf{y}$.

For a weight-Ld path, a first event error will be made if, in the Ld positions in which the correct and incorrect path differ, the path metric for the incorrect path is less than that of the correct path (so the decoder wrongly chooses the incorrect path). The probability of such event is given by

$$\mathbf{P}[M(\mathbf{r}|\mathbf{v}') < M(\mathbf{r}|\mathbf{v})] = \mathbf{P}\left[\sum_{i=1}^{L} w_i d(\mathbf{r}_i, \mathbf{v}') < \sum_{i=1}^{L} w_i d(\mathbf{r}_i, \mathbf{v})\right]$$

From the linear property of the convolutional codes, the all-zero path is always assumed to be the correct path and the non all-zero path is the incorrect path. Therefore, $\mathbf{v}$ consists of d zeros and $\mathbf{v}'$ consists of d ones. Thus, $d(\mathbf{r}_i, \mathbf{v}) = W(\mathbf{r}_i)$ and $d(\mathbf{r}_i, \mathbf{v}') = d - W(\mathbf{r}_i)$, where $W(\mathbf{r})$ represents the Hamming weight of the received packet $\mathbf{r}$. So we have

$$\begin{aligned}\mathbf{P}[M(\mathbf{r}|\mathbf{v}') < M(\mathbf{r}|\mathbf{v})] &= \mathbf{P}\left[\sum_{i=1}^{L} w_i (d - 2W(\mathbf{r}_i)) < 0\right] \\ &= \mathbf{P}\left[\sum_{i=1}^{L} w_i W(\mathbf{r}_i) > \frac{d}{2}\sum_{i=1}^{L} w_i\right]\end{aligned}$$

If there is a tie between the metrics of the paths, decoder will randomly choose one. Let $c_{Ld} = \frac{d}{2}\sum_{i=1}^{L} w_i$, and $S = \sum_{i=1}^{L} w_i W(\mathbf{r}_i)$. Therefore, the probability of decoding error is given by

$$P_{Ld} = \mathbf{P}[S > c_{Ld}] + \frac{1}{2}\mathbf{P}[S = c_{Ld}] \quad (8)$$

S is the weighted sum of L binomial random variables with different parameter sets (d, p_i). We make use of the generating function to calculate the Probability Mass Function (PMF) of random variable S:

$$\begin{aligned} G_s(z) &= \mathbf{E}\left[z^{\sum_{i=1}^{L} w_i W(\mathbf{r}_i)}\right] = \prod_{i=1}^{L} G_{W(\mathbf{r}_i)}(z^{w_i}) \\ &= \prod_{i=1}^{L} (1 - p_i + p_i z^{w_i})^d = \sum_k p_S(k) z^k \quad (9)\end{aligned}$$

The coefficient $p_S(k)$ is the probability of $S = k$. Therefore,

$$P_{Ld} = \sum_{k > c_{Ld}} p_S(k) + \frac{1}{2} p_S(c_{Ld}) \quad (10)$$

Thus, based on Fact 1 and (4), we have the following theorem:

Theorem 1: The upper bound for the bit-error probability of the distributed code combining method, P_b, is given by:

$$P_b < \sum_{d=d_{free}}^{\infty} B_d P_{Ld} \quad (11)$$

where B_d is the coefficient of the weight-d term in the bit WEF $B(X)$ of the original convolutional code, and B_{Ld} is given by (10). $\diamond$

Since p_i is small, P_{Ld} decreases greatly as d increases. P_b is generally dominated by the first several terms of the summation in (11), or even the first term $B_{d_{free}} P_{d_{free}}$. So this union bound is tight, and numerical results show the first several terms of the summation in (11) can be a good estimation of real P_b. There are $(d+1)^L$ terms in the right hand side of (9). For $L \leq 10$, the computation time of P_{Ld} is quite tolerable. Some results will be shown in the simulation section.

An example of the Viterbi decoding procedure in an unequal error condition environment is shown in Fig.5. This trellis diagram is for a (3,1,2) code with an information sequence of length $h = 3$. In this example, we assume $L = 3$. For each output, since the original code has $n = 3$ bits, so the combined code in one time unit has 9 output bits, in the order of channel No.1, No.2, and No.3. Assume the reliable factor (weight) $w_1 = 1$, $w_2 = 2$, and $w_3 = 3$. Suppose an all-zero sequence is sent, and the received sequence $\mathbf{r} =$ (000 001 111, 000 100 101, 001 010 001, 000 000 000, 000 000 000). Since the codeword in the trellis structure is a repeated (3 times) codeword, we use one copy - the first 3 bits on each branch - in the following notation. let $\mathbf{v}$ represent the correct path sequence (000, 000, 000, 000, 000), and $\mathbf{v}'$ represent the sequence of the highlighted path in Fig.5 (111, 101, 011, 000, 000). Split $\mathbf{r}$ into $\mathbf{r}_1$, $\mathbf{r}_2$, and $\mathbf{r}_3$ according to channels, then we will have

$$\begin{aligned} M(\mathbf{r}|\mathbf{v}) &= w_1 d(\mathbf{r}_1, \mathbf{v}) + w_2 d(\mathbf{r}_2, \mathbf{v}) + w_3 d(\mathbf{r}_3, \mathbf{v}) \\ &= 1 \cdot 3 + 2 \cdot 3 + 3 \cdot 6 = 24 \end{aligned}$$

Likewise

$$M(\mathbf{r}|\mathbf{v}') = 1 \cdot 6 + 2 \cdot 4 + 3 \cdot 1 = 17 < M(\mathbf{r}|\mathbf{v})$$

Thus the all-zero path is eliminated. Note if otherwise the channel condition were equal ($w_1 = w_2 = w_3$), then $M(\mathbf{r}|\mathbf{v}) < M(\mathbf{r}|\mathbf{v}')$ and the highlighted path would be

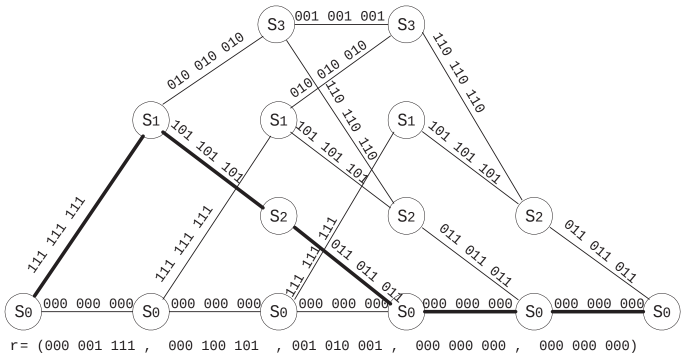


Fig. 5. The Viterbi Algorithm for Code Combining with Unequal Error Probability

eliminated. Using the same algorithm to check the other paths we can decide that the highlighted path is the final survivor, $\hat{\mathbf{v}} = (111, 101, 011, 000, 000)$. This surviving path corresponds to the decoded information sequence $\hat{\mathbf{u}} = (100)$. Note that the final $m = 2$ branches in any trellis path correspond to 0 inputs and hence are not considered part of the information sequence.

III. Simulations

In order to evaluate the performance of the cooperative networks, a set of random nodes representing the networks nodes are chosen according to the network topology as follows: the transmitter and the receiver cluster head are fixed nodes and are 250 meters apart. The cluster is formed around the cluster head in a circle with radius of 50 meters. The cooperative nodes are randomly placed as a uniform distribution inside the cluster. The topology of the simulated network is shown in Fig.6.

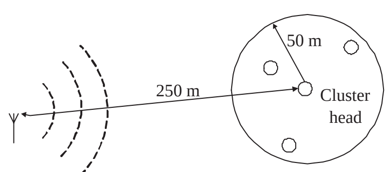


Fig. 6. Topology of the simulated network

In the following simulations the decoded bit-error rate P_b, is calculated using Theorem 1 from section II. The channel model used is Rayleigh fading channel, and Binary Phase Shift Keying (BPSK) is used for modulation [21]. It is shown in [22] that in a frequency-nonselective Rayleigh fading channel with Additive White Gaussian Noise (AWGN), the probability of error for using BPSK is

$$p = \frac{1}{2}\left(1 - \sqrt{\frac{\overline{\gamma_b}}{1 + \overline{\gamma_b}}}\right) \qquad (12)$$

where $\overline{\gamma_b}$ represents the average signal-to-noise ratio. In general $\overline{\gamma_b} = \frac{E_b}{N_0}\mathbf{E}[\alpha^2]$ where α is the attenuation factor due to fading.

A. Link Layer Decoding Performance

We use different levels of power for inter-cluster and intra-cluster transmission because the distance between cluster nodes and the cluster head are at most 1/5 of the radio distance for inter-cluster transmission. Let PD represent the difference between the power used by the cluster nodes and the power at the sender node, in dB. We consider two cases where PD=10dB and 20dB, i.e. the cluster nodes use a transmit power that is 10dB and 20dB less than the sender transmit power, respectively. This means the SNR level is at least 4.5dB ($10\log(250/50)^{3.5} - 20$ dB = 4.5 dB) higher than the signal received from the sender. For each power level, the simulation takes 100 runs and finds the average decoded bit-error rate. A (2,1,3) convolutional code is used for code combining with Viterbi decoding at the cluster head. The decoded bit-error rate P_b with weighted code combining at the cluster head is plotted as a function of L in Fig.7. The SNR is measured at the receiver, i.e., the cluster head. Therefore the SNR is proportional to the sender transmission power. Changing PD from 10dB to 20dB does not change the overall performance of the code combining technique significantly. The change is negligible when the sender transmits at a considerably high power, e.g., in this simulation when SNR=8dB.

We also tried different cluster radii for the simulations. For PD=20dB, we simulated the cluster radii of 50m and 100m. The decoded bit-error rate is plotted in Fig.8. A larger cluster radius leads to a worse decoding performance since some cluster nodes may be too far from the sender node. However, it is shown in both Fig.7 and Fig.8 that the decoded bit error rate decreases sharply when L increases. A system designer should take this fact into account when deciding about the maximum number of the cooperation nodes.

To provide a reliable link performance, a very low bit error rate is desired. In another round of simulations, a couple of fixed decoded bit-error rates, 10^{-7}, 10^{-6}, and 10^{-5}, are set to be the objectives. The choice of the desired P_b mainly depends on the frame size. For each random topology, the sender power level is adjusted to achieve the desirable P_b. Cluster nodes use 20dB less power than the sender node (PD=20dB). We plot the required SNR at the cluster head as a function of cluster size to compare the dB gain of the cooperative code combining technique, as shown is Fig.9. Note when $L = 1$ it means there is no cooperation. So the difference

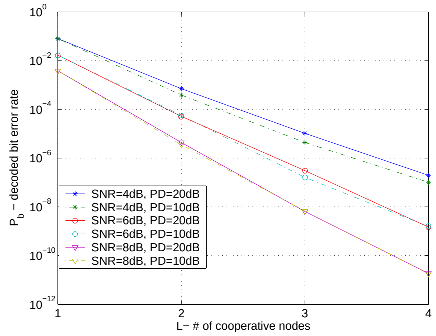


Fig. 7. Decoded bit-error rate P_b vs. number of cooperative nodes L. PD is the amount of power deduction of the intra-cluster transmission upon the inter-cluster transmission.

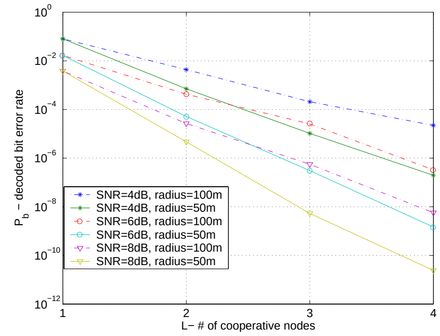


Fig. 8. Decoded bit-error rate P_b vs. number of cooperative nodes L with different cluster radius. Smaller cluster radius has a better performance.

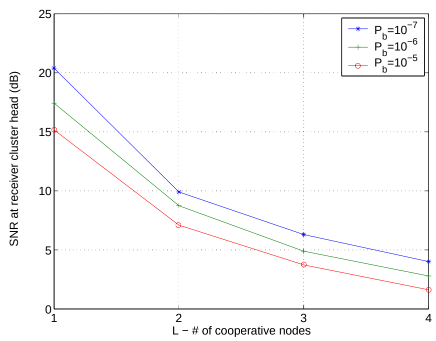


Fig. 9. SNR vs. number of cooperative nodes L. With a fixed objective P_b, the required SNR decreases with the increase of the cluster size L.

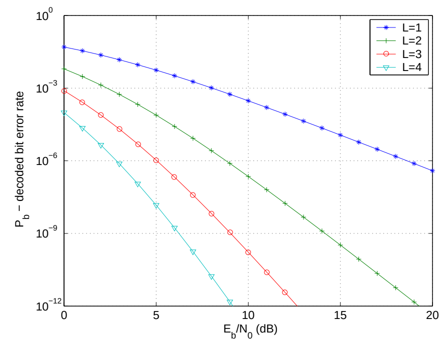


Fig. 10. Decoded bit error rate P_b vs. SNR (dB) with different cluster size L. The dB gain with the use of cooperation over ordinary FEC is substantial.

between the SNR of cooperation and non-cooperation is very similar to the concept of *coding gain*.

To view the coding gain more clearly, we plot the probability of error versus the signal to noise ratio (E_b/N_0 at the cluster head) of different cooperative levels ($L = 1 \sim 4$), shown in Fig.10. From this figure, it is shown that the decoding performance of cooperative FEC with code combining ($L > 1$) has a obvious dB gain over that of the non-cooperative FEC ($L = 1$).

B. Energy Consumption

Generally clustering is proposed to solve scalability problem, node mobility problem in wireless ad hoc networks. Clusters can maintain a relatively stable effective topology. The membership in each cluster changes over time in response to node mobility, node failure or new node arrival. Clustering techniques are expected to achieve better scalability since most of the topology changes within a cluster are hidden from the rest of the network. In addition, clustering can be extremely effective in multicast, broadcast, and pear-to-peer communication. Clustering can support data fusion and data dissemination/dissipation. In sensor networks, some clustering protocols are claimed to save energy expenditure. The clustering in our scheme is for the cooperation purpose, yet it can save energy as well.

The cost for the cooperation is the energy consumed at the cluster nodes. To take this into account, we will model the aggregate energy spent in transmitter together with all the cluster nodes for successfully transmitting one bit.

We use a simple energy consumption model to evaluate the total cost of the communication, including the transmitting and the receiving [23]. The average energy consumption of the radio in transmission process can be described by:

$$E_t = P_{tx}T_{tx} + P_{out}T_{tx} \qquad (13)$$

and the average energy consumption of the receiving system can be expressed as

$$E_r = P_{rx}T_{rx} \qquad (14)$$

where $P_{tx/rx}$ is the power consumption of the transmitter/receiver, P_{out} is the output transmit power, $T_{tx/rx}$ is the transmit/receive on-time (actual data transmission time). Note that $T_{tx/rx} = l/R$, where l is the packet size and R is the data rate in bits per second. Also note that if R_c is the code rate then the number of information bits in a packet is $l' = lR_c$. To transmit an l'-bit information message, the radio expends $(P_{tx} + P_{out})\frac{l'}{RR_c}$ and to receive l' bit message, the radio expends $P_{rx}\frac{l'}{RR_c}$. The electronics power P_{tx} and P_{rx}, are the amount of power spent in the transmitter electronics circuitry, depending on digital coding, modulation, filtering and spreading of the signal, while P_{out} is the amount of energy spent in the RF amplifiers to counter the propagation loss. Here P_{out} takes into account the constant factor in the path loss term, as well as the antenna gains of the transmitter and the receiver. When receiving a packet, only the receiver circuitry is invoked.

In general, with the assumption on fading channel and the modulation scheme, the probability of error is a function of signal to noise ratio E_b/N_0. After knowing how strong the signal is, then we can convert E_b/N_0 to carrier to noise ration using the equation:

$$\frac{C}{N} = \frac{E_b}{N_0}\frac{f_b}{B_w} \tag{15}$$

where f_b is the bit rate, and B_w is the receiver noise bandwidth.

Since we now have the carrier to noise ratio, we can determine the necessary received carrier power after we calculate the receiver noise power. The thermal noise power N_{th} is computed using Boltzmann's equation:

$$N_{th} = kTB \tag{16}$$

where k is Boltzmann's constant $= 1.380650 \times 10^{-23}$ J/K, T is the effective temperature in Kelvin, and B is the receiver bandwidth. Therefore, $N_{th} = (1.380650 \times 10^{-23}$ J/K $\times 290$K$\times 1$ MHz $= 4 \times 10^{-15}$ W $= 4 \times 10^{-12}$ mW = -114dBm.

The receiver has some inherent noise in the amplification and processing of the signal. This is referred to as the receiver noise figure N_{rx}. So the receiver noise level will be:

$$N(\text{dBm}) = N_{th} + N_{rx} \tag{17}$$

We can now find the carrier power as $C = C/N \times N$, or in dB $C = C/N + N$. This is how much power the receiver must have at its input. To determine the transmitter amplifier power P_{out}, we must account for the path loss that we are building in to the system.

The log-distance path loss model has been used extensively in the literature [21]. The average large-scale path loss for an arbitrary transmitter-receiver separation is expressed as a function of distance by using a path loss exponent γ, which indicates the rate at which the path loss increases with distance. The value of the propagation loss exponent γ is highly dependent on the surrounding environment (usually between 2 to 4). The average path loss for a T-R separation with distance d can be expressed by

$$\overline{PL}(\text{dB}) = \overline{PL}(d_{ref}) + 10\gamma \log \frac{d}{d_{ref}} \tag{18}$$

where d_{ref} is the close-in reference distance and can be based on a free space assumption from the transmitter to d_{ref}. So, assume no system loss, with unity antenna gain, the path loss for the reference distance is given by

$$\overline{PL}(d_{ref}) = 10 \log \frac{(4\pi)^2}{\lambda^2} \tag{19}$$

where λ is the wavelength in meters.

Finally, adding the path loss to the receiver carrier power will give the required transmitter amplifier power:

$$P_{out}(\text{dBm}) = C + \overline{PL} \tag{20}$$

In our link layer cooperation scheme, d_0 and d_1 are the average intra-cluster and inter-cluster distance, respectively. Since $d_0 < d_1$ or even $d_0 \ll d_1$, they correspond different level of path loss, leading to different power consumption of radio amplifier. The energy spent for successfully relaying one packet during a single hop is:

$$\begin{aligned} E_c &= [(P_{tx} + P_{out})\frac{k}{RR_c} + \bar{L}P_{rx}\frac{k}{RR_c} \\ &\quad + (\bar{L}-1)(P_{tx} + P_{out}^{(0)})\frac{k}{RR_c} + (\bar{L}-1)P_{rx}\frac{k}{RR_c} \\ &\quad + (P_{tx} + P_{out})\frac{k'}{R} + P_{rx}\frac{k'}{R}]\bar{T}_c \end{aligned} \tag{21}$$

where $\bar{L}$ is the average number of the cooperative nodes in a cluster, R_c is the FEC code rate used for code combining, k' represents the control packet length for ACK, ARQ request, or request intra-cluster cooperation, $\bar{T}_c$ is the average transmission times for a packet, including first transmission and retransmissions. Note $P_{out}^{(0)}$ represents the power requirement for radio amplifier used in intra-cluster transmission.

The expression in first line of the equation indicates the inter-cluster communication (one node sends and $\bar{L}$ cluster nodes receive); the second line indicates intra-cluster communication (all the cluster nodes except the cluster head send and the cluster head receive); and the last represents the ACK or ARQ request.

If $\bar{L} = 1$, expression for E_c describes the energy consumption for a non-cooperative network, which is exactly the case with conventional hybrid ARQ with the same FEC coding but without cooperation. The communication energy spent in a single hop is for this hybrid ARQ is:

$$E_h = [(P_{tx}+P_{out})\frac{k}{RR_c}+P_{rx}\frac{k}{RR_c} +(P_{tx}+P_{out}+P_{rx})\frac{k'}{R}]\bar{T_h} \quad (22)$$

where T_h is the counterparts of T_c, the average transmissions for a packet in a single hop in the non-cooperation scenario. Keep in mind that although expressions are in the similar forms, with or without cooperative FEC, the power levels for meeting the requirement of probability of error are very different.

Table I gives the parameter settings used in our simulations. The parameters about the energy model are from [23].

γ	3.5
f_b	2 Mbps
B_w	1 MHz
λ	0.122 m
N_{rx}	10 dB
d_{ref}	1 m
d_1	250 m
d_0	50 m
R_c	1/2
P_{tx}	81 mW
P_{rx}	180 mW
l	500 bytes

TABLE I
VARIABLES IN THE ENERGY MODEL

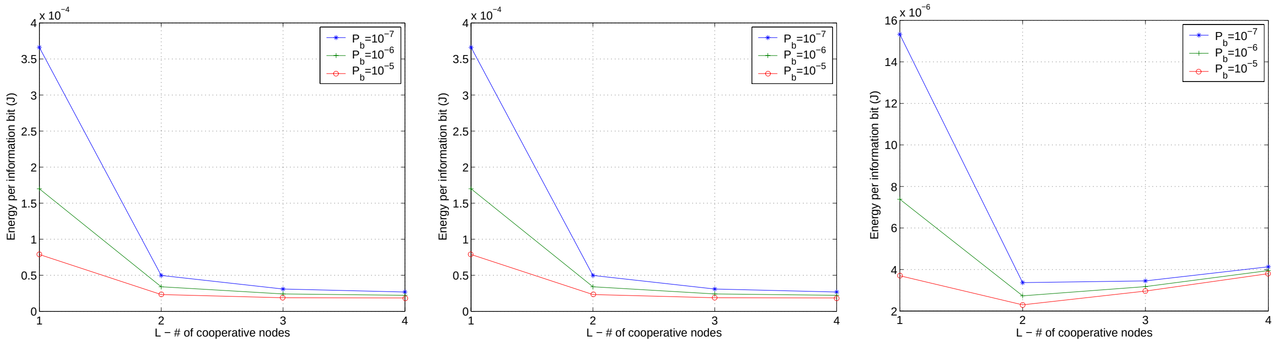
Given the above energy model, we use the topology in Fig.6 to simulate the average energy consumption per useful information bit for different number of cooperative nodes. Three plots in Fig.11 illustrate the results for different source to destination distances d_1 and cluster radii d_0. Note Fig.11(a) and Fig.11(b) are almost the same. This shows that cluster radius is not a significant factor as far as energy consumption is concerned. Also note that the scale of Fig.11(c) is different from that of the other two. This shows the distance from sender to receiver is the major factor for the power consumption. In Fig.11(c), the distance between sender to receiving cluster head is relatively small, thus power used to combat path loss is relatively low. The power consumed in transmit/receive electronics circuitry begins to take more effect. This is why the energy per information bit increases slightly after $L = 2$. Therefore, from the prospective of power efficiency, cluster size L is not necessarily very big. Nevertheless, cooperative FEC offers energy saving over traditional forwarding. In addition, the cooperative FEC is more efficient when long distance transmission is needed.

The above simulations are just some case studies to illustrate how cooperation can increase the decoding performance. If the channel quality is better than the channel used in these simulations, we may choose a code with a higher rate than 1/2 used in the above examples. In fact, such a low code rate as 1/2 will bring too much overhead in ad hoc networks. Obviously codes with lower rates have a better performance in terms of the decoded error rate. Given the desired P_b, and the channel condition, we can choose the appropriate operating point (code rate and cluster size) to meet the needs. A higher rate convolutional code can be achieved using punctured codes, which is a simple operation on a lower rate code without additional complexity. Likewise, the result can be extended to the case with longer distances between the transmitter and the receiver, different node density etc. The cluster size may be adapted to the channel condition and the code rate.

IV. NETWORK LAYER DESIGN

The higher layer design is not the main interest of this paper, so we summarize some layer-3 related work and express our thoughts on this topic. The concept of cooperative FEC can be used to consider an interesting paradigm shift in layer-3 forwarding and routing: cooperative cluster-based routing. There is an important difference between our proposed forwarding protocol and other routing protocols that introduce the notion of a "cluster". Our use of clusters is for enabling cooperative error recovery at layer-2. In the conventional routing schemes, "clusters" are used for routing scalability, i.e. to facilitate hierarchial routing. Traditionally forwarding of packets at the network layer happens from node to node. There has been a copious amount of research into various aspects of ad-hoc wireless network routing, especially routing scalability, geographic and trajectory routing [24], [25], [26], [27], [28], [29]. The key difference in our model is that packets will be forwarded ad-hoc on a cluster-to-cluster basis and not on a node-by-node basis. Moreover, we propose to give the source freedom to explicitly choose source-routes (eg: as an extension of DSR [30]) or trajectory routes (eg: as in location-driven or geographic routing) around which the cooperative cluster-based forwarding is performed. Fig.12 illustrates the cooperative cluster-based forwarding ideas. The figure shows a source that can explicitly pick a path along the trajectory and form a cluster as its first relay cluster.

An interesting aspect of our proposed method is that these forwarding cooperative clusters are formed on the fly. In particular, the source (or the previous-cluster-hop)



(a) d_1=250m, d_0=50m (b) d_1=250m, d_0=100m (c) d_1=100m, d_0=50m

Fig. 11. Energy per useful information bit plotted versus number of cluster nodes given different decoding probability of error P_b. Note that $L = 1$ corresponds to non-cooperative networks.

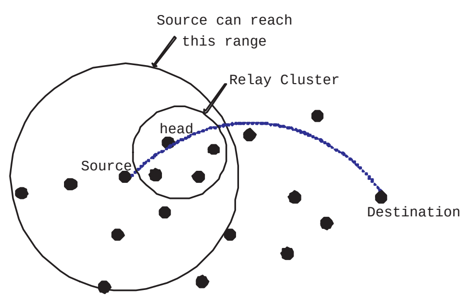


Fig. 12. Routing Example: Forming the Cluster along the Trajectory.

can actually encode a cluster boundary into the packet header.

Since the cluster-based routing and forwarding have been extensively studied by researchers, this cooperative FEC can adopt most of the traditional cluster-based routing architectures and build the cooperation based on these routing protocols. Recently Biswas and Morris [31] proposed an opportunistic routing protocol which involves one sender and choice of multiple receivers when forwarding packets. This follows our basic motivation of cooperation - nodes with unreliable links can help each other - and can be integrated to our cooperation scheme.

At this point, we intend to ignore some secondary considerations in this paper. These considerations include the contribution of idle-listening on the RF channel to the energy consumption, and contention resolvent at the cluster head. We regard these issues as secondary because it is well accepted the energy consumed by listening is much less than that by transmission or reception, and contention problem can be solved by intra-cluster transmission MAC, e.g., TDMA or CDMA. It is also worth mentioning that the decoding energy is not covered in our energy consumption consideration. There is a good reason behind this. Cooperative FEC decoding does no more decoding than ordinary FEC decoding. Given our previous argument that the complexity of each code combining decoding is as same as each basic code decoding, the energy spent for cooperative decoding keeps unchanged. As far as we consider scenarios with same FEC codes, the decoding energy expenditure is a constant factor for all the cooperation schemes.

V. Conclusions

In this paper we analyzed the decoding performance of the cluster-based cooperative networks with a code combining technique. This cooperative FEC exploits the fundamental feature of the wireless medium: its broadcast nature. Simulation results from various aspects show this cooperation architecture is effective in improving the link performance and reducing the energy consumption. This result is promising in that the reduced power requirement leads to less interference caused by a transmission, thus can improve the capacity of the wireless networks.

The results in this paper are under the consideration of a single hop network. Yet they are applicable to a multi-hop network as well. However, more problems will be involved, such as the effect of interference, MAC design, and so on. Our future work will look into the detailed cross layer design of the network, including cooperation-intended cluster-based routing, medium access issues in the intra-cluster communications, network performance from all aspects, and more information theoretic analysis of the coding technique and network capacity.

The vision of our work is to develop enabling core technology for cooperative wireless networks and fuel the interdisciplinary effort which is required to make co-operation at each level a reality. Our preliminary results indicate that this approach achieves a quantum leap in the performance/cost trade off. The future focus of our work is on designs which explicitly exploit physical layer, data link layer, and network layer cooperation among nodes.

APPENDIX
CODE COMBINING WITH BLOCK CODES

To show the performance of code combining, it is easier to start with block codes. We use Golay code with maximum-likelihood decoding as an example. Suppose the original code is (24,12), a rate 1/2 block code.

The minimum Hamming distance d_{min} of the new set of codewords will increase linearly with the number of repeats. Error correcting capability can be given as $t = \lfloor \frac{d_{min}-1}{2} \rfloor$. The minimum distance and the error correcting capability of the repeated Golay code is listed in Table II.

number of repeats L	code dimensions (Ln, k)	minimum distance d_{min}	error correcting capability t
1	(24,12)	8	3
2	(48,12)	16	7
3	(72,12)	24	11
4	(96,12)	32	15
8	(192,12)	64	31
16	(384,12)	128	63

TABLE II
MINIMUM DISTANCE OF A REPEATED GOLAY CODE AND ITS ALGEBRAIC ERROR-CORRECTION CAPABILITIES

In general, for a binary symmetric channel (BSC) with and error probability of p_1 we can define the random variable

$$U_i^1 = \begin{cases} 1, & \text{with probability } p_1 \\ 0, & \text{with probability } 1-p_1. \end{cases}$$

The number of errors in a codeword of length n is

$$\mu_1 = \sum_{i=1}^{n} U_i^1 \tag{23}$$

We define the random variable ν_1 as

$$\nu_1 = \frac{\mu_1}{n} \tag{24}$$

So random variable ν_1 has a mean of p_1 and a variance of $(1-p_1)p_1/n$. When $n \to \infty$, $\nu_1 \sim N(p_1, \frac{(1-p_1)p_1}{n})$.

In the cooperative FEC, the member nodes will transmit their received packet to the cluster head if necessary. Therefore all the repeated packets will travel from the sender to the cooperative node and then to the cluster head, except that the packets directly received by cluster head, only have to travel from sender to the cluster head. Thus among L repeated packets, one of them has error probability p_1, and the other $L-1$ have error probability $p = p_1 + p_0 - p_1 p_0$. Similarly with U_i, we define another Bernoulli trail as

$$U_i = \begin{cases} 1, & \text{with probability } p \\ 0, & \text{with probability } 1-p. \end{cases}$$

There are $(L-1)n$ bits in the packets transmitted from all the member nodes. The head node has it's own packet with n bits. Let $\mu = \sum_{i=1}^{(L-1)n} U_i$, and $\nu = \frac{\mu}{(L-1)n}$. So $\nu \sim N(p, \frac{(1-p)p}{(L-1)n})$.

Now let's calculate the decode error rate for the original codes and the combined codes. Golay code is a perfect code, so for a combined code capable of correcting t errors in the combined packet, the probability of decoded error (word error rate) is

$$p(e) = \mathbf{P}[\mu_1 + \mu > t] = \mathbf{P}[n\nu_1 + (L-1)n\nu > t] \tag{25}$$

Let $\theta = n\nu_1 + (L-1)n\nu$. The linear function of independent Gaussian random variables is still Gaussian.

$$\begin{aligned} \mathbf{E}[\theta] &= n\mathbf{E}[\nu_1] + (L-1)n\mathbf{E}[\nu] \\ &= np_1 + (L-1)n(p_1 + p_0 - p_1 p_0) \end{aligned} \tag{26}$$

$$\begin{aligned} \mathbf{Var}[\theta] &= n^2\mathbf{Var}[\nu_1] + (L-1)^2 n^2 \mathbf{Var}[\nu] \\ &= np_1(1-p_1) + (L-1)n(1-p_1)(1-p_0) \\ &\quad (p_1 + p_0 - p_1 p_0) \end{aligned} \tag{27}$$

Therefore $\theta \sim N(np_1 + (L-1)n(p_1+p_0-p_1p_0), np_1(1-p_1) + (L-1)n(1-p_1)(1-p_0)(p_1+p_0-p_1p_0))$. So

$$p(e) = \mathbf{P}[\theta > t] = Q(\frac{t - \mathbf{E}[\theta]}{\sqrt{\mathbf{Var}[\theta]}}) = Q$$
$$(\frac{t - np_1 - (L-1)n(p_1+p_0-p_1p_0)}{\sqrt{np_1(1-p_1) + (L-1)n(1-p_1)(1-p_0)(p_1+p_0-p_1p_0)}})$$

where $n = 24$ in this example and the function $Q(x)$ is defined as

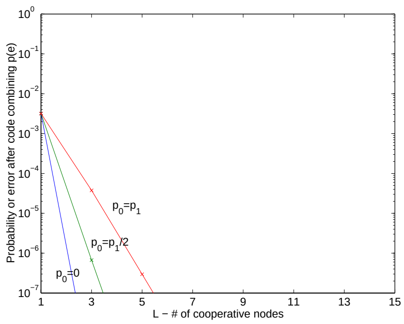
$$Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-y^2/2} dy \tag{28}$$

Fig.13 plots the decoded error rate as a function of the number of cooperative nodes, L, under different channel error rates. In this example, channel error rate around 8.5% is a threshold that code combining can work well or not. If the channel error rate is greater than 8.5%, code combining can not handle the errors any more. Let's take a look at the algebraic error-correction ratio t/Ln of the combined code.

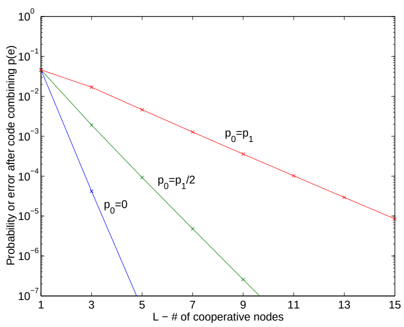
$$\frac{t}{Ln} \approx \frac{d_{min}}{2Ln} = \frac{L d_{min}^o}{2Ln} = \frac{d_{min}^o}{2n} \tag{29}$$

where d_{min}^o corresponds to the minimum distance of the original code (no combining).

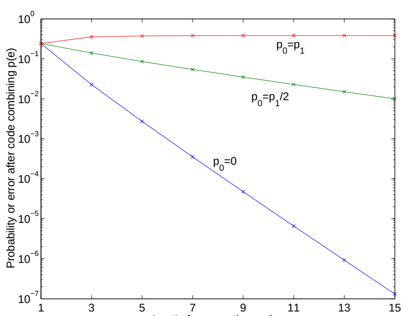
So this ratio will depend on the code chosen. In this example, $d_{min}^o = 8, n = 24$, so the algebraic error-correction ratio is roughly $1/6 (16.7\%)$. Our threshold on p_1 is lower than this due to the error accumulation at the relaying link ($p \approx 2p_1 = 17\%$). If the channel condition is better than the threshold (8.5%), cooperation among nodes can reduce the decode error rate greatly and thus avoid retransmission. To our knowledge, 8.5% bit error rate is generally far above the ordinary wireless channel. Depending on the channel error rate of noisy channel, the choice of the number of cooperative nodes can be made according to the performance prediction.



(a) $p_1 = 3\%$



(b) $p_1 = 5\%$



(c) $p_1 = 8.5\%$

Fig. 13. Decoded Word Error Rate $p(e)$ vs. Number of Cooperative Nodes L, where $L = 1$ Corresponds to the Scenario with No Code Combining

References

[1] D. Chase, "Code combining–a maximum-likelihood decoding approach for combining an arbitrary number of noisy packets," *IEEE Trans on Communications*, vol. 33, no. 5, pp. 385–393, May 1985.

[2] B. Azimi-Sadjadi and A. Mercado, "Diversity gain for cooperating nodes in multi-hop wireless networks," in *Proceedings of IEEE Vehicular Technology Conference 2004-Fall*, Los Angeles, CA, Sep 2004.

[3] A. Sendonaris, E. Erkip, and B. Aazhang, "User cooperation diversity–part I: System description," *IEEE Transactions on Communications*, vol. 51, no. 11, November 2003.

[4] ——, "User cooperation diversity–part II: Implementation aspects and performance analysis," *IEEE Transactions on Communications*, vol. 51, no. 11, November 2003.

[5] P. Anghel and G. Leus and M. Kaveh, "Distributed Space-Time Coding in Cooperative Networks," *5th Nordic Signal Processing Symposium*, October 2002.

[6] J. N. Laneman and G. W. Wornell, "Exploiting Distributed Spatial Diversity in Wireless Networks," *in Proc. Allerton Conf. Commun., Contr., and Computing*, October 2000.

[7] ——, "Distributed Space-Time Coded Protocols for Exploiting Cooperative Diversity in Wireless Networks," *IEEE Trans. Inform. Theory*, vol. 49, no. 10, October 2003.

[8] J. N. Laneman and D. N. C. Tse and G. W. Wornell, "Cooperative Diversity in Wireless Networks: Efficient Protocols and Outage Behavior," *IEEE Trans. Inform. Theory*, vol. 50, no. 12, Dec 2003.

[9] T. Hunter and A. Nosratinia, " Cooperation Diversity Through Coding," *Proc. International Symposium on Information Theory*, June 2002.

[10] A. Stefanov and E. Erkip, "Cooperative space-time coding for wireless networks," *In Proceedings of IEEE Information Theory Workshop*, April 2003.

[11] A. Scaglione and Y. Hong, "Opportunistic large arrays: Cooperative transmission in wireless multihop ad hoc networks to reach far distances," *IEEE Transactions on Signal Processing*, vol. 51, no. 8, pp. 2082–2092, August 2003.

[12] A. Mercado and B. Azimi-Sadjadi, "Power efficient link for multi-hop wireless networks," in *41st Annual Allerton Conference on Communication, Control, and Computing*, Oct 2003.

[13] J. Hagenauer, "Rate-compatible punctured convolutional codes (RCPC codes) and their applications," *IEEE Trans on Communication*, vol. 36, no. 4, pp. 389–400, April 1988.

[14] S. B. Wicker and M. D. Bartz, "Type-II hybrid-ARQ protocols using punctured MDS codes," *IEEE Trans on Communication*, vol. 42, no. 2/3/4, pp. 1431–1440, February/March/April 1994.

[15] J.M.Wozencraft and M.Horstein, "Coding for two-way channels," in *Proc. 4th London Symp. On Information Theory*, Butterworths, Washington, D.C., pp. 11–23.

[16] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Trans on Wireless Communications*, vol. 1, no. 4, pp. 660–670, Oct 2002.

[17] B. A. Harvey and S. B. Wicker, "Packet combining systems based on the Viterbi decoder," *IEEE Trans on Communication*, vol. 42, no. 2/3/4, pp. 1544–1557, February/March/April 1994.

[18] P. Sindhu, "Retransmission error control with memory," *IEEE Trans on Communication*, vol. 25, no. 5, pp. 423–429, May 1977.

[19] G. Benelli, "An ARQ scheme with memory and soft error detectors," *IEEE Trans on Communication*, vol. 33, no. 3, pp. 285–288, March 1985.

[20] S. Lin and D. J. Costello, *Error Control Coding, 2nd Ed.* Pearson Education, 2004.

[21] T. S. Rappaport, *Wireless Comunications: Principles and Practice, 2nd Ed.* Upper Saddle River, NJ: Prentice Hall PTR, 2002.

[22] J. G. Proakis, *Digital Comunications 4th Ed.* New York, NY: McGraw-Hill, 2001.

[23] E. Shih, S.-H. Cho, N. Ickes, R. Min, A. Sinha, A. Wang, and A. Chandrakasan, "Physical layer driven protocol and algorithm design for energy-efficient wireless sensor networks," in *Proceedings of the 7th Annual International Conference on*

Mobile Computing and Networking (MobiCom), Rome, Italy, Jul 2001, pp. 272–287.

[24] B. Karp and H. Kung, "GPSR: Greedy perimeter stateless routing for wireless networks," in *Proceedings of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, Boston, MA, Aug. 2000, pp. 243–254.

[25] J. Li, J. Jannotti, D. D. Couto, D. Karger, and R. Morris, "A scalable location service for geographic ad-hoc routing," in *Proceedings of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, New York, NY, Aug. 2000, pp. 120–130.

[26] L. Blazevic, S. Giordano, and J.-Y. L. Boudec, "Anchored path discovery in terminode routing," in *Proceedings of the Second IFIP-TC6 Networking Conference (Networking 2002)*, Pisa, May 2002, pp. 141–153.

[27] L. Blazevic, J.-Y. L. Boudec, and S. Giordano, "A scalable routing scheme for self-organized terminode network," in *Communication Networks and Distributed systems modelling and Simulation conference (CNDS)*, San Antonio, Texas, Jan. 2002.

[28] D. Niculescu and B. Nath, "Routing on a curve," in *Proceedings of Workshop on Hot Topics in Networks (HOTNETS-I)*, Princeton, NJ, Oct. 2002.

[29] M. Yuksel, R. Pradhan, and S. Kalyanaraman, "An implementation framework for trajectory-based routing in ad-hoc networks," 2005.

[30] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad-hoc wireless networks," in *Mobile Computing*, T. Imielinski and H. Korth, Eds. Kluwer Academic Publishers, 1996, ch. 5.

[31] S. Biswas and R. Morris, "ExOR: Opportunistic multi-hop routing for wireless networks," in *Proceedings of ACM SIGCOMM'05*. ACM, August 2005, pp. 133–143.