Energy-efficient Link Assessment in Wireless Sensor Networks

Abtin Keshavarzian, Elif Uysal-Biyikoglu
Information Systems Lab
Department of Electrical Enginerring
Stanford University,
Stanford, CA 94305
Email: abtink@stanford.edu
elif@stanford.edu

Falk Herrmann, Arati Manjeshwar
Robert Bosch Corporation
Research and Technology Center
4009 Miranda Avenue,
Palo Alto, CA 94304
Email: Falk.Herrmann@us.bosch.com
Arati.Manjeshwar@rtc.bosch.com

***Abstract*— For energy-constrained stationary wireless networks of sensors, selection of links with high quality rate helps to ensure reliable long-term operation. During the implementation of a protocol targeting industrial applications of such systems, it was found that it is advantageous to acquire accurate information about the availability and quality of the RF communication links prior to the network topology formation. "Link assessment" as part of the initialization process, accomplishes this task by assessing a sufficient number of packets exchanged between neighboring nodes.**

This paper introduces and analyzes two different approaches to link assessment: The first approach is a random non-deterministic scheme that allows for a probabilistic guarantee of collision-free packet exchange. An alternative method is described which employs 'constant-weight codes' and provides a deterministic guarantee of success. In particular, a special class of constant-weight codes, known as optical orthogonal codes, are considered. Since, these codes are cyclically permutable, they make the link assessment process simpler, and therefore they are preferred over other codes. We evaluate the performance of these methods based on their energy consumption, time duration, and implementation complexity.

I. Introduction

Sensor networks are widely used in both civil and military applications such as security management, surveillance, automation, and environmental monitoring. So far, most commercially deployed systems utilize wire based communication. However, in recent years there has been tremendous interest in both industry and academia in self-configuring wireless networks [1]–[5], [10], [11], [24]. Main motivations are to reduce installation cost, gain flexibility, allow for unobtrusive installation, and enable entirely new applications such as tracking and wireless interrogation. The recent developments have been fuelled by advances in low cost and low power RF communication, as most envisioned systems are battery operated and expected to be successful only if low cost.

A wireless sensor network may be generalized as a distributed system consisting of hundreds of sensor nodes each equipped with a wireless radio transceiver along with application-specific sensors and signal processing hardware. There may or may not be a central control unit, *i.e.*, a base station. Due to the typically short range of low-power RF transceivers, communication takes place via multi-hop through neighboring nodes. The information flow may be from the sensor nodes to a network access point, *e.g.*, the base station, or from the network access point to sensor nodes, or among sensors themselves. In order to fully utilize the benefits of a multi-hop RF network, the system should also be self-configuring and able to adapt to environmental changes by reconfiguration.

The nodes are typically small and battery operated and are expected to live for at least couple of years. Therefore, energy-efficiency is a crucial factor for all tasks performed throughout the lifetime of the system. Energy can be saved by communicating over reliable links and by avoiding collision of packets, which eliminates the necessity of re-transmission.

Wireless sensor networks are still in the development stage with some industrial interest, *e.g.*, Ember, Intel, ABB, Sensicast, Parc, Bosch. Most industrial applications for wireless sensor networks are low cost and deal with stationary scenarios. In spite of the low cost of these wireless sensor nodes (typically less than $5) it is still not low enough to allow the use of redundant sensors as proposed in current academic research. This implies that there is still a need to gather information from *every* available sensor, requiring a reliable path to every sensor in the network. During the implementation of a protocol targeting industrial applications of stationary wireless sensor networks, it was found that it is advantageous to acquire accurate information about the availability and quality of the RF communication links *prior* to the network topology formation. This task is accomplished by the "link assessment process".

The link assessment process includes discovery of all nodes and the available links between them, and grading the quality of these links. The latter can be achieved by estimating parameters such as packet success rate or signal strength, which may be determined by assessing a sufficient number of packets exchanged between neighboring nodes [24]. This information can then be used to make routing decisions and form a reliable multi-hop network topology.

The paper is organized as follows: In the following section the network scenario and link assessment requirements and assumptions are described. In Section III, a scheme where each node uses a random pattern is described and analyzed. In section IV, deterministic patterns are considered. Constant-weight codes are employed to generate these patterns for link assessment. It is shown that optical orthogonal code (which is a special class of constant-weight codes) is preferred over other codes. Section V concludes this paper.

II. Link Assessment

In this section we describe the network model and assumptions, define the link assessment process , motivate the need for it and discuss the previous related works.

A. Network model

Consider N nodes which are capable of wireless transmission and reception of data, installed in a certain area. The wireless links or connections of such network can be modelled as a *weighted directed* graph $G = (\mathcal{V}, \mathcal{E})$, called the connectivity graph. The $\mathcal{V}$ represents the set of nodes, and $\mathcal{E}$ the set of edges. An edge from node i to node j is present, if node j can hear node i's transmission with an acceptable quality. In other words, the connectivity graph shows all the available links in the network and the neighbors of every node. Although in many cases the links are bi-directional, we use the more general model of a directed graph, *i.e.*, the link from node i to node j is treated differently from the link from node j to node i. It is even possible that there is an edge from node i to node j, while there is no reverse edge from j to i. This can happen when node i and j are within hearing range of each other but there is a strong interferer/jammer near node i.

The weight of the edges/links represents the "quality" of the links, which can be defined in many different ways. Depending on the application, quality may be defined as a function of one or more of the following physical layer parameters : (1) average received signal strength (2) signal to noise/interference ratio (SNR/SIR), (3) average bit error rate, (4) average packet success rate. For example the knowledge of packet success rate, may be required by routing or topology formation and scheduling algorithms [1]–[4], [6], [7]. Based on the application and hardware capabilities of nodes, different methods can be employed to estimate these parameters. In general, to have a good estimate for any particular link in the network, one needs to transmit and assess sufficient number of packets on that link [24]. For example, a simple method to compute an estimate of packet success rate on a link from node i to node j is as follows: Node i transmits a predefined number (say 300) of packets. Depending on the number of packets received by node j, one can estimate that packet success rate as the ratio of number of packets received to the number of packets transmitted. Note that the larger the number of exchanged packets, the more accurate, the estimates will be, however, more time and energy will be consumed.

The main goal of the "link assessment" process is *to discover the weighted connectivity graph of network.* Each node should not only find all its neighbors (the nodes that it can hear from), but it also should assess all the available links and measure their qualities. As described above, definition of quality is application dependent. However, no matter what definition is used or which parameters should be estimated, we assume that the following two conditions are satisfied:

(a) some fixed number of packets should be transmitted on each link: The required number of packets basically depends on the parameters that should be estimated, and the accuracy needed for these estimates. For example, if we just want to discover the neighbors of a node (with no link grading), the node should correctly receive only one (or two) packet(s) from each of its neighbors, so transmission of only a couple of packets is sufficient. However, for a good estimate of packet success rate or bit error rate, hundreds of packets may required.

(b) intra network interference/collision should be avoided: the number mentioned in part (a) corresponds to the number of *collision-free* packets that are required for the accurate estimation and quality measurement.

Collision occurs at a particular node, when two (or more) of its neighbors send packets simultaneously. When collision occurs, we assume that all colliding packets are lost, although it is possible that the receiver correctly decodes the packet from the transmitter which has the strongest *received* signal power (and it is also typically the closest transmitter to the receiver). This is called the "capture effect" ([8], [9]), and ignoring this effect is actually a pessimistic assumption, therefore the number of packets received by a node can be larger than what will be designed for later.

Combining the above two conditions implies that to perform a successful link assessment, one needs a protocol which guarantees that between *any* two neighboring nodes in the network at least some fixed number of collision-free packets are exchanged[1], and this process should be done while the network is being discovered and the neighbors are being found. The three key constraints in designing such methods are (1) the amount of energy consumed by the link assessment protocol, (2) the total time spent, and (3) the complexity of its implementation and the amount of memory it requires. We need to design a method which is energy-efficient, takes a reasonable time and is fairly simple such that it can be implemented in each sensor node which typically has very low computational power and memory.

B. Motivation

The link assessment process combines two goals: Neighbor discovery and link grading. The knowledge of one-hop neighbors (or in some cases two-hop neighbors or the entire connectivity graph) is the main requirement in many routing or time scheduling or topology formation algorithms [3], [6], [7] in ad hoc or sensor networks.

It has been justified that information about the quality of the links leads to a more reliable and energy-efficient topology or a much better routing method [1], [2], [4]. These schemes acquire the link quality data *during* the operation of the

[1]Note that the term "exchanged" will be meaningless if we deal with a uni-directional link, but it is assumed that the same number of packets is required to be transferred over such links.

network, inherently assuming that the nodes have a relatively high duty cycle. However, many sensor networks (including the system which motivated the present work) use extremely low duty cycles, *i.e.*, below 0.1%. Therefore reliable link grading during network operation is not feasible. Moreover, in such systems, nodes can be in receive mode, only during active communication periods. So promiscuous listening cannot be used. Hence, link grading *prior* to the topology formation is preferred.

Link assessment must be done energy-efficiently. Even though link assessment might be a one time operation during the network lifetime, any energy saving during this phase, is significant since in very low duty cycle applications, 100 packets may translate into 100 days of operation.

C. Assumptions

Our methods for link assessment are based on the following assumptions:

1) *The nodes are stationary,* so that the neighbors of a specific node and the corresponding link qualities do not change significantly during the link assessment process.
2) *All nodes use the same transmission power, the same frequency channel, and the same modulation technique.* Note that any kind of power control or frequency reuse requires some sort of coordination in the network. This requires the knowledge of the connectivity graph which is not available before link assessment.
3) *Each node has a unique identification number*[2]. This can be a factory-assigned number (part of the serial number of the product), or it can be selected randomly from a large enough range such that uniqueness is ensured with high probability. However, it is preferred (but it is not essential) that the id numbers are in the range from 1 to N (number of nodes in the network). So a distributed method for unique id assignment is needed. Some studies related to this problem can be find in [10].
4) *There is a known upper bound on the number of neighbors,* denoted by n_{max}. This is a key parameter in designing the link assessment process. Clearly, a protocol designed for a large value of n_{max} will work for a smaller value as well, but it will take longer time and consume more energy than necessary.
5) *The nodes are synchronized to each other.* Based on the application, different methods can be used to achieve synchronization among the nodes (see [11] for more details). One approach is to have a general broadcast clock message, then the nodes can be synchronized by resetting their internal clock upon reception of this message. Another method is to run a distributed clock synchronization algorithm before the link assessment begins to achieve synchronization between nodes.
6) *Total time is divided into time slots.* Each slot is designed to accommodate some fixed number of packets (each of fixed size), and a guard time. The guard time should be larger than the clock skew between any two nodes. We select the time slot long enough (accommodate more packets in one time slot) such that the guard time is negligible in comparison with the duration of one time slot. Clearly, shorter time slots (with fewer packets in each of them) can be used with more accurate synchronization. For a constant total time duration, shorter time slots implies larger number of them and link assessment offers better performance when larger number of time slots are used. Hence, accurate synchronization is essential for good performance.
7) *At each time slot, each node can be in one of three states: sleep, transmit, or receive.* In sleep mode the node shuts down the receiver/transmitter circuit in order to preserve energy. If a node is in transmit mode during a time slot, it will send as many packets as it can. The content of the packets is not important for the link assessment process, however, the receiver should be able to recognize the sender of the packet. When a node is in receive mode during a time slot, it listens to the channel to receive packets from other nodes. As mentioned before, we assume that colliding packets are all lost, so if two (or more) neighbors of a node (which is in receive state) are both in transmit mode at the same time slot, we assume that no packet will be received by the node during that time slot.

[2]Local uniqueness is actually sufficient.

D. Problem Definition

A particular time slot is called a "collision-free" (or clear) time slot for the link from node i to node j, if in that time slot, node i is in transmit state, node j is in receive mode and all neighbors of node j (except node i) are either in sleep state or in receive state.

The number of time slots needed for successful completion of link assessment process, denoted by C, is defined as the ratio of the number of packets required to measure the quality of the links to the number of packets within a time slot. For example, if there are 10 packets in each time slot, and for link quality measurement we need 300 packets, then $C = 300/10 = 30$ collision-free time slots must be assigned to every link in the network. As mentioned above the number of packets in each time slot depends on the accuracy of the synchronization.

Hence, the link assessment problem reduces to finding *a protocol which ensures that every link in the network gets at least C collision-free time slots.*

E. Energy Consumption

In each mode the node will consume a definite amount of energy. Typically, the energy consumed in sleep mode is very small in comparison with the energy used in the other two states. Hence, we neglect this energy consumption in our analysis. Let E_{TX} (E_{RX}) denote the energy consumed by a node which is in transmit (or receive) state for one time slot. To simplify the energy expressions, the ratio of these two terms is defined to be

$$\beta = \frac{E_{TX}}{E_{RX}}. \quad (1)$$

Although it seems that a node should consume significantly more energy for transmission than reception, in low-power (or short range) transceivers, the value of β typically is between 1 to 3. So energy consumed for reception is not negligible as it is in other systems[3]. This is a key difference between low-power sensor networks and other ad hoc networks.

In this part, we describe an ideal situation in which the minimum amount of energy is consumed by each node in the network.

Each node is required to transmit in at least C time slots. If node i $(i = 1, \ldots, N)$ has n_i neighbors, it must also listen to each of its n_i neighbors for C time slots, *i.e.*, it should listen in at least $n_i C$ time slots. Hence, the energy consumed by this node is at least $CE_{TX} + n_i CE_{RX}$. Therefore, using (1) node i consumes at least $(\beta + n_i)CE_{RX}$. The minimum total energy consumed by all nodes in the network is then given by:

$$E_{\text{Total},min} = \sum_{i=1}^{N}(\beta + n_i)CE_{RX}. \quad (2)$$

Defining $\bar{n} = \frac{1}{N}\sum_{i=1}^{N} n_i$ to denote the average number of neighbors, the above expression can be written in the following form:

$$E_{\text{Total},min} = N(\beta + \bar{n})CE_{RX}. \quad (3)$$

Note that no feasible method can achieve this value, and it is just used as a benchmark for judging the energy usage of other methods.

F. Related Work

To the best of our knowledge, link assessment (as a combination of neighbor discovery and link grading) has not yet been considered as a separate phase of a wireless network protocol. The general method usually proposed for neighbor discovery is a time-scheduled serial search [4]–[6][4] : The nodes broadcast in a pre-determined order, some packets informing their neighbors of their existence. Although the number of packets transmitted by each node is small, the main problem with this method is the fact that all nodes need to be awake and listen to the channel for the entire time that this process is going on. This time is directly proportional to the number of nodes in the network which can become inconveniently long for a large network. As mentioned before, amount of energy consumed by the node for listening to the channel is comparable to the energy used for transmission. So by being awake the nodes will waste significant amount of energy. To get a feeling of why a simple time-scheduling is not feasible for large networks, consider the following example[5]: Assume that each time slot is 800 *m*sec long, the network has $N = 1000$ nodes, and $C = 30$ collision-free time slots per each node is required. By using a simple time-scheduled method, the link assessment will take $1000 \times 30 \times 0.8 = 24,000$ seconds which is more than 6.5 hours during which all nodes should to be awake. Using our proposed methods the link assessment can be done in almost 15 minutes.

III. Random Protocol

As mentioned in the previous section, in every time slot each node can be in one of the three states: transmission (shown by TX), reception (shown by RX) or sleep (shown by SL). In the random method we assume that at each time slot, each node randomly selects to go to one of these three states. With probability P_{TX} the node will go to transmit state, with probability P_{RX}, it will choose receive state, and with probability P_{SL} the node will sleep. Clearly $P_{TX}+P_{RX}+P_{SL} = 1$. The selection of the state at current time slots, is assumed to be independent of the past states. The designer of the system, should decide about the value of "design parameters", which include these probabilities and the total number of time slots devoted to the link assessment process, based on the value of n_{max} (maximum number of neighbors in the network) and other known network parameters (such as β defined in (1)).

This method is a very simple, which makes it perfect from the implementation point of view. In fact, the only thing the nodes are required to do, is to generate random numbers and decide about their states in each time slot. Since this method is random, we get a probabilistic guarantee on the success of the protocol, however the probability of success can be very close to one.

The designing results in optimal performance when $P_{SL} = 0$, *i.e.*, no node is in sleep state during the process. In other words, when the random protocol is used, the ability of the nodes to go to sleep mode, is completely useless. The main advantage of this method is that the design parameters do not depend on the number of nodes in the network and therefore the design is scalable with respect to the number of node, *i.e.*, any design which works for a network with 100 nodes, can be used for a network with 3000 nodes or even larger number of nodes, as long as other network parameters (most importantly, n_{max}) in the two networks are the same. However, this method is very sensitive to the value of n_{max}, and an over estimate of this parameter can considerably degrade the performance.

Consider a link from node i to node j. The probability that a particular time slot is collision-free for this link is lower bounded by[6]:

$$P_{\text{cf}} = P_{TX}P_{RX}(1 - P_{TX})^{n_{max}-1}. \quad (4)$$

[3]For example 'CC1000' chip from Chipcon, approximately consumes 2μA in sleep mode, $10m$A in receive mode, and $16.5m$A in transmit mode, so $\beta = 1.65$.

[4]Some of the methods described in [4], [5] are not exactly a time-scheduled search but they are similar. While they may work fine for small networks, the amount of time they will take is inconveniently long for networks with thousands of nodes.

[5]The numbers given in this example are close to the actual values used in the project which motivated this work.

[6]Note that in (4), n_{max} is used in place of the actual number of neighbors of the node, and since n_{max} is larger than the actual value, this expression is a lower bound. Using this lower bound obviously leads to a conservative design.

Let $C_{i,j}$ denote the total number of collision-free time slots for the link from node i to j. Since the states at different time slots are independent, $C_{i,j}$ can be modelled as a binomial random variable:

$$C_{i,j} \sim \text{Binomial}(F, P_{\text{cf}}), \tag{5}$$

where F represents the total number of time slots. The main design goal is to make sure that all links in the network get at least C collision-free time slots. We say a link is lost if it gets less than C collision-free time slots. The probability of this event is:

$$\begin{aligned} P_{\text{L}} &= \mathbb{P}(C_{i,j} < C) \\ &= \sum_{n=0}^{C-1} \binom{F}{n} (P_{\text{cf}})^n (1 - P_{\text{cf}})^{F-n}. \end{aligned} \tag{6}$$

The average energy consumed by any node can be written as:

$$\begin{aligned} \mathbb{E}(E_{\text{node}}) &= F P_{TX} E_{TX} + F P_{RX} E_{RX} \\ &= F(\beta P_{TX} + P_{RX}) E_{RX}, \end{aligned} \tag{7}$$

and the total average energy consumed in the entire network is then:

$$\mathbb{E}(E_{\text{Total}}) = NF(\beta P_{TX} + P_{RX}) E_{RX}. \tag{8}$$

The design problem reduces to the following optimization problem:

- Find values for P_{TX}, P_{RX}, and F such that the probability of losing a link (P_{L}) is kept small and bounded, while the average energy consumed in each node, $\mathbb{E}(E_{\text{node}})$, and total number of time slots, F, are minimized.

Let $P_{L,max}$ denote the maximum acceptable value for P_L. A small $P_{L,max}$ (*e.g.*, 10^{-6}) ensures that with a very high probability all links in the network are assessed[7]. Computational methods can be used to solve the above optimization problem and find the exact solution. However, by carrying out some simplifications one can obtain the following *approximate* solution:

$$F \approx \frac{2C + q^2 + q\sqrt{q^2 + 4C}}{2\frac{1}{n_{max}}\left(1 - \frac{1}{n_{max}}\right)^{n_{max}}}, \tag{9}$$

$$P_{TX} \approx \frac{1}{n_{max}}, \tag{10}$$

$$P_{RX} \approx 1 - \frac{1}{n_{max}}, \tag{11}$$

where $q = Q^{-1}(P_{L,max})$ and $Q(x)$ (the normal gaussian tail) is defined in (36). Details of the derivation of the above expressions are given in Appendix I. As mentioned before we see that the $P_{SL} \approx 0$ and the result does not depend on the number of nodes N.

[7]With $P_L = 10^{-6}$, we can roughly say that on average one out of one million links will be lost.

As an example, let $n_{max} = 25$ ($\bar{n} = 15$), $C = 30$, $\beta = 2.5$ and $P_{L,max} = 10^{-6}$. Using the approximate expressions, one obtains

$$F = 4834 \quad P_{TX} = 0.04 \quad P_{RX} = 0.96,$$

changing $C = 1$, the number of time slots reduces to $F = 1704$.

To test how energy efficient this design is, we compare the average total energy with the ideal case described in Section II-E. The ratio of (8) to (3) is,

$$\frac{\mathbb{E}(E_{\text{Total}})}{E_{\text{Total},min}} = \frac{F(\beta P_{TX} + P_{RX})}{C(\beta + \bar{n})} \tag{12}$$

For $C = 30$ this ratio is 9.76 (9.9dB), and for $C = 1$ it is equal to 103.2 (20.1dB). We can generally say that although this method may have an acceptable performance for large values of C, it performs very poorly for small C.

IV. Code-Based Protocol

In the method described in the previous section, each node randomly generates a pattern based on which it switches between different states. The main idea in this section is *to assign a deterministic pattern instead of randomly generated one to each node*. The use of deterministic patterns makes the link assessment process more efficient in time and energy. Also it gives a deterministic guarantee of the success of the process. First we define "constant-weight codes", and explain how they can be used as patterns for the link assessment process. Then, we restrict our attention to a special class of these codes which are cyclically permutable (also known as optical orthogonal codes). Although any constant-weight code can be used for link assessment, due to the additional properties of optical orthogonal codes, they make the process of pattern assignment very simple, and therefore they are preferred over other codes.

The idea of using codes as transmission patterns in wireless ad hoc network has been studies before, but it was used to solve a different problem. Initially, Chlamtac and Farago in [20] and later in a generalization, Ju and Li in [21], proposed the concept of using codes to form a topology-transparent scheduling in ad hoc networks. The code they use, which we will briefly describe in Section IV-D, is actually a special class of constant-weight codes. We believe that our method can also be applied to their problem.

A. Constant-Weight Codes

An (F, W, d) constant-weight codes is a binary code of length F, minimum Hamming distance d, whose codewords have constant weight W. Each codeword consists of F bits, with W ones and $F - W$ zeros. The Hamming distance between two codewords is defined as the number of bits in which the two codewords are different. These codes were extensively studied in [12].

These codes can be used for link assessment process. Each codeword is uniquely assigned to one of the nodes in the network. Each bit in a codeword, stands for one time slot, 'one' bits correspond to the transmit state, while 'zero' bits show the receive state. So, the length of the code F, also represents the total number of time slots. We assume that the nodes will never use sleep state. The following theorem, states the property that the code must have so that it can be used in link assessment:

Theorem 1: If the weight of the code satisfies the following inequalities then the link assessment process will be successful, *i.e.*, each link in the network will get at least C collision-free time slots:

$$C \leq W \leq \left(\frac{n_{max}\frac{d}{2} - C}{n_{max} - 1} \right). \tag{13}$$

Proof: The left inequality is trivial: Since we need C collision-free time slots, and each node is in transmit state for W time slots (according to the codeword assigned to the node), clearly, the weight of the code, W, should be larger than C.

The proof of the other inequality is based on the distance property of constant-weight codes. Consider two different codewords, and let the Hamming distance between these two codewords be d_o. We claim that d_o is an even number: Since the two codewords have the same weight thus they have the same number of ones. Consider the bit positions in which the first codeword has bit one, and the second codeword is zero. Corresponding to each such position there exists a bit position in which the first codeword has a zero and second codeword has one. Hence the total number of bit positions that they can be different, *i.e.*, the distance between the two codewords, should be even.

Let λ_o show the number of bit positions in which the two codewords both are one. Note that in $\frac{d_o}{2}$ of the W positions that the first codeword has bit one, the second codeword is zero. Hence the number of bit positions that the two codewords have bit one, is $\lambda_o = W - \frac{d_o}{2}$. Based on the definition of constant-weight code, $d_o \geq d$, thus $\lambda_o \leq \lambda$ where

$$\lambda = W - \frac{d}{2}. \tag{14}$$

λ shows the maximum number of bit positions in which any two codewords of an (F, W, d) constant-weight code, both have bit one at the same position. Since bit one stands for transmit state, λ also shows the maximum number of time slots that any two nodes in the network can be in transmit state during the same time slot.

Observe that the right inequality in (13) can be written in the following form:

$$\begin{aligned} W &\geq C + n_{max}\left(W - \frac{d}{2}\right) \\ &= C + \lambda\, n_{max}. \end{aligned} \tag{15}$$

So to complete the proof, one need to show that if the above inequality holds, all nodes get at least C collision-free time slots. Consider an arbitrary link in the network from node i (transmitter) to node j (receiver) . In the worst case, node j has $n_{max} - 1$ neighbors (excluding node i). Node i will be in transmit state in exactly W time slots. Each neighbor of node j, can collide with at most λ transmission time slots of node i, so at most $\lambda(n_{max} - 1)$ of transmission time slots of node i can be lost due to collision. If node j (receiver) is in transmit mode at the same time slot as node i, again the time slots will be lost. However, the number of such time slots is again bounded by λ. So, in the worst case, λn_{max} of the W time slots in which node i is in transmit state, can be collided and missed. Hence, if W is larger than $C + \lambda n_{max}$, at least C collision-free time slot will remain. ■

The maximum size (which is defined as the number of codewords) of a constant-weight code set is denoted by $A(F, W, d)$. The value of $A(F, W, d)$ is in general not known, but a number of lower and upper bounds have been established. See [12]–[15] for summaries of best bounds known today. Since each node in the network must have one unique codeword, the size of the code must be larger than the number of nodes in the network (N), *i.e.*, $A(F, W, d) \geq N$.

Any (F, W, d) constant-weight code, which satisfies (13) and has enough codewords for all the nodes in the network, can be used for the link assessment process. By selecting the code set properly, the link assessment process can be done very efficiently.

Note that, once the codewords are assigned to the nodes, the operations performed by each node, are very simple. The nodes should change their states according to the pattern assigned to them. However, the main challenge is the codeword assignment. As mentioned in Section II-C, we assume that each node has (or is given) a unique identification number. If this number is factory-assigned or assigned to the node while the protocol is written in the node's memory (*i.e.*, when the node is programmed), and network parameters such as n_{max}, C, and N are available before the programming is done, then an appropriate codeword can be computed in advance and assigned to the node during the programming. However, typically id numbers are assigned *after* the nodes' installation and programming, so the pre-computation of the codewords is not possible. Clearly, the whole codeword table can not be stored in all nodes in the network, as it can require huge amount of memory. Therefore, the nodes must construct the codewords by themselves after the installation based on the id number assigned to them. The nodes usually have very low computational power, so the codeword construction algorithm should be very simple. Hence, we consider special classes of constant-weight codes that are easy to generate.

B. Optical Orthogonal Codes

Optical orthogonal codes (OOC) are constant-weight codes which are also cyclically permutable [16]–[19]. In a cyclically permutable code, all codewords are cyclically distinct and have full cyclic order. Optical orthogonal codes were first introduced by J.A. Salehi in [17], [18] for an application in optical code-division multi-access spread-spectrum communication systems.

An (F, W, λ) OOC C is a family of (0,1) sequences of form $\mathbf{a} = (a_0, a_1, \ldots, a_{F-1})$ with length F and weight W, that have cross-correlation and out-of-phase autocorrelation values which do not exceed λ, *i.e.*,

The Auto-correlation property: For any $\mathbf{a} \in \mathcal{C}$:

$$\sum_{i=0}^{F-1} a_i a_{(i+j) \bmod F} = \begin{cases} \leq \lambda & \text{if } j = 1, \ldots, F-1 \\ = W & \text{if } j = 0 \end{cases} \tag{16}$$

The Cross-correlation property: For any two different codewords $\mathbf{a}, \mathbf{b} \in \mathcal{C}$ that are not cyclic shift of each other and any $0 \leq j < F$:

$$\sum_{i=0}^{F-1} a_i b_{(i+j) \bmod F} \leq \lambda \tag{17}$$

For example, the simplest set is (5,2,1) OOC code with the following ten codewords:

$$\begin{aligned} \mathcal{C} \;\; = \{ \;\; & 10100, 01010, 00101, 10010, 01001 \\ & 11000, 01100, 00110, 00011, 10001 \; \} \end{aligned}$$

Note that any (F, W, λ) OOC is an (F, W, d) constant-weight code with $d = 2W - 2\lambda$. It is easy to see that any cyclic shift of a codeword in $\mathcal{C}$ is another valid codeword, so to generate the whole codeword set, we need to store only one codeword from each group that are cyclic shift of one another, *e.g.*, in the above example, all codewords can be generated from cyclic shifts of 10100 and 11000. This property of optical orthogonal makes the codeword assignment process very simple. Only a few main codewords are stored in (or given to) all nodes, then different nodes, based on their id number, use different shifts of the stored codewords. Since F is typically in the order of hundreds, by storing just one codeword, we can generate hundreds of other codewords and support hundreds of nodes in the network.

Given the values of F, W and λ, the largest possible size of an (F, W, λ) OOC is denoted by $\Phi(F, W, \lambda)$. Except for some special cases, the exact value of this function is not known. However, upper and lower bounds are available [16], [19].

As optical orthogonal codes are special class of constant-weight codes, the result obtained in Theorem 1 in Section IV-A is valid, and the condition on W is even simpler (similar to (15)):

$$W \geq C + \lambda n_{max}. \tag{18}$$

The energy consumed by each node is the same and is given by:

$$\begin{aligned} E_{\text{node}} &= W E_{TX} + (F - W) E_{RX} \\ &= \left[F + W(\beta - 1)\right] E_{RX}, \end{aligned} \tag{19}$$

and the total energy used by the network is:

$$E_{\text{Total}} = N \left[F + W(\beta - 1)\right] E_{RX}. \tag{20}$$

The energy usage of the new method is again compared with the ideal case from Section II-E:

$$\frac{E_{\text{Total}}}{E_{\text{Total},min}} = \frac{F + (\beta - 1)W}{(\bar{n} + \beta)C}. \tag{21}$$

The designer must choose parameters W and F and λ based on network requirements using the network parameters C, n_{max}, β, and N. The design problem is again an optimization problem:

- Find F, W and λ to minimize $F + W(\beta - 1)$ (from (19)) such that (18) and the following inequality are satisfied:

$$\Phi(F, W, \lambda) \geq N. \tag{22}$$

The above condition guarantees that all N nodes in the network get a unique codeword.

Since no closed form solution for $\Phi(F, W, \lambda)$ is available, this optimization problem can not be solved analytically. However, we can solve it by numerical methods. In Appendix II, we show that for typical network parameters, the following approximate solution is very close to the optimum design:

$$\lambda = 1, \tag{23}$$

$$W = C + n_{max}, \tag{24}$$

$$F \approx W^2. \tag{25}$$

Let us present some examples in this part and demonstrate how the OOC-based method outperforms the random scheme. For $C = 30$ and $n_{max} = 25$, any (F,55,1)-OOC with $F > 2980$ can be used. To make sure that sufficient number of codewords can be found, we set $F = 4000$, then the energy consumed is 7.77 (8.8dB) times the ideal case. Note that if we can find one OOC codeword, since all shifts of it can also be used, we can support up to 4000 nodes. Recall that the random protocol requires $F = 4834$ time slots and consumes 9.76 (9.89dB) times of the energy of the ideal case. For smaller values of C, even more performance improvement can be achieved. With $C = 1$, we can use a (660,26,1) OOC set which consumes 39.9 (16dB) times the energy of the ideal case, while the random method requires $F = 1704$ with 103.2 times energy of ideal case. Both the total number of time slots and the energy usage are considerably decreased.

C. Under-weight codes

In Theorem 1, we give the necessary conditions on weight of the code in order for the link assessment process to finish successfully. In case of OOCs the condition is $W \geq \lambda n_{max} + C$. But what happens if a smaller value of W is used? The situation described and used in the proof of Theorem 1, is the worst possible case. The probability that this worst case happens in an actual network, is very small. It this section we show that we can still get a good performance by using codes with smaller weight. However, one no longer gets a 100% guarantee of success. We use the concept of probability of loss as defined in (6) and select the parameters such that the probability of missing a link in the network is very small.

Assume that an $(F, W, 1)$ OOC $\mathcal{C}$ is used and the codewords are distributed randomly among the nodes, but each node has a unique codeword. From [17], [18] it is known that for any two $\mathbf{a}, \mathbf{b} \in \mathcal{C}$, if U has uniform distribution, then:

$$\mathbb{P}\left(\sum_{i=0}^{F-1} a_i a_{(i+U) \bmod F} = 1\right) = \frac{W(W-1)}{F-1} \leq \frac{W^2}{F} \tag{26}$$

$$\mathbb{P}(\sum_{i=0}^{F-1} a_i b_{(i+U) \bmod F} = 1) = \frac{W^2}{F}. \qquad (27)$$

Hence, the probability that packets from two neighboring nodes collide at one of the F time slots is less than $\frac{W^2}{F}$. Note that according to the property of the OOCs, two neighbors can collide at most at one time slot. Since the codewords are assigned to nodes independently, if a node has n neighbors, the number of neighbors that collide with it (denoted by m) will have binomial distribution as follows:[8]

$$m \sim \text{Binomial}(n, \frac{W^2}{F}). \qquad (28)$$

It is possible that some of the collisions happen in the same time slot (*i.e.*, three of four nodes transmit together), but in the proof of Theorem 1 we assumed that all collisions occurred in separate time slots. To compute the probability of this worst case and the exact number of collided time slots, we use the following ball-bin model:

There are W bins (corresponding to the W transmission time slots of the main node) and m balls (corresponding to the collided time slots). The balls are thrown independently and uniformly into the bins. An empty bin corresponds to a collision-free time slot. We want to find the probability $p_m(e)$ that after dropping m balls into W bins, there are e empty bins. Clearly:

$$p_0(e) = \begin{cases} 1 & \text{if } e = W \\ 0 & \text{otherwise} \end{cases} \qquad (29)$$

We get e empty bins after m drops, if

1) either there are e empty bins after $(m-1)$ balls and the last ball selects one of the occupied bins which happens with probability $(W-e)/W$,
2) or there are $(e+1)$ empty bins after $(m-1)$ drops and the last ball goes to one of the empty bins which occurs with probability $(e+1)/W$.

This leads to the following recursive expression for $p_m(e)$:

$$p_m(e) = \frac{W-e}{W} p_{m-1}(e) + \frac{e+1}{W} p_{m-1}(e+1). \qquad (30)$$

Note that m (the number of balls) is also a random variable according to (28), so by averaging over m, we obtain the overall probability of getting e empty bins (or equivalently e collision-free time slots):

$$p(e) = \sum_{m=0}^{n} p_m(e) \binom{n}{m} \left(\frac{W^2}{F}\right)^m \left(1 - \frac{W^2}{F}\right)^{n-m}. \qquad (31)$$

Recall that P_L shows the probability that we get less than C collision free time slots (or empty bins), so:

$$P_L = \mathbb{P}(e < C) = \sum_{e=0}^{C} p(e). \qquad (32)$$

[8]each neighbor has a chance of $\frac{W^2}{F}$ to collide and they are independent of each other.

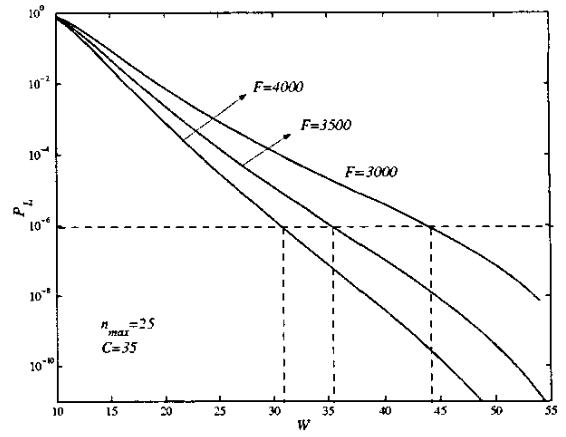

Fig. 1. Loss probability versus code weight, W for different values of code length F, with $n_{max} = 25, C = 35$.

Figure 1 shows P_L versus W for different values of F with $n_{max} = 25$, $C = 35$. Note that using the previous method, an optical orthogonal code with $W \geq 60$ must be used which required an $F \approx 3600$. However, from this figure it can be observed that with $F = 3000$ any $W > 44$ can be used, while we ensure that $P_L < 10^{-6}$. For $F = 3500$, it is necessary that $W > 35$, and for $F = 4000$, any $W > 32$ will be sufficient. As it is expected, with increase of F, the acceptable range of values for W becomes larger. If for instance, a (3000,45,1) OOC code is used, the total time reduces from 3600 time slots to 3000 time slots, and energy consumption is reduced from 3.83 (5.8dB) times the energy of the ideal case to 3.18 (5.0dB) times the energy of the ideal case.

D. Other type of codes

In this section we describe another class of constant-weight codes, which has a simple algorithmic construction method. They were introduced in [20], [21] for an application in scheduling in wireless multi-hop networks. The construction method uses Galois fields ([22]) represented by GF(q) where q shows the number of elements in the field [9]. q must be of the form $q = p^m$, where p is a prime and m a positive integer. If $m = 1$ the elements of GF(q) can be represented by numbers $0, 1, 2, \ldots, q-1$ and the operations are addition and multiplication modulo q.

To generate a codeword, consider a polynomial of order k: $f(x) = a_k x^k + \ldots + a_1 x + a_0$ where $a_i \in$ GF(q). The resulting code has length $F = q^2$ and weight $W = q$. Divide the $F = q^2$ bit positions into q groups of each of size q. Enumerate the groups from 0 to $q-1$. Each group consists of q bits, where only one of them will be set to one and the other $q-1$ bits will be zero. Find all pairs $(x, f(x))$ for all q possible values of $x \in$ GF(q). Note that $f(x)$ is also in GF(q). Each value x corresponds to one of the groups, while the value $f(x)$ shows the position of the bit within that group that should be

[9]We assume that the reader is familiar with these concepts, further details are widely available, *e.g.*, [22].

set to one. Hence a codeword can be constructed from each polynomial. For more details see [21].

To clarify the method, we give an example. Consider $GF(3) = \{0, 1, 2\}$ and polynomial $f_1(x) = x + 1$. Then we have the following pairs: $\{(0, 1), (1, 2), (2, 0)\}$ which gives us the following codeword: $(010 - 001 - 100)$. Polynomial $f_2(x) = x^2$ results: $\{(0, 0), (1, 1), (2, 1)\}$ and codeword $(100 - 010 - 010)$.

Now consider all distinct polynomials of degree less than λ. Each such polynomial generates a different codeword. The set of codewords generated in this way, has this property that the number of positions that any two codewords have bit one at the same place, is less than λ. So this set is a constant-weight code with minimum Hamming distance $d = 2q - 2\lambda$.

To prove the above property, consider two different polynomials $f_1(x)$ and $f_2(x)$. The number of positions that their corresponding codewords have colliding ones, is the same as the number of times, the two pairs $(x, f_1(x))$ and $(x, f_2(x))$ are equal. In other words, we need to find the number of times $f_1(x) = f_2(x)$, *i.e.*, the number of roots of $f_1(x) - f_2(x)$. Since $f_1(x)$ and $f_2(x)$ are polynomials of degree λ or less, so is their difference and thus it can have at most λ roots.

These codes have similar properties as optical orthogonal codes. The length of the code in both cases is square of the weight (in case of OOC, the length should be larger than W^2). They have typically the same number of codewords. The main advantage of these codes is their algorithmic construction method. However, we believe that optical orthogonal codes are more suitable for link assessment, as the code generation is much simpler in case of OOCs: We need to store a couple of codewords and then different nodes will use different shifts of the stored codewords, while for the other codes, each node must choose a polynomial and do some computations in GF(q) to construct the codeword.

E. Enhancement of the protocol

In this section, the concept of sparse OOCs is introduced then we use it to modify the above scheme to obtain a more energy-efficient protocol.

An OOC with length F and weight W is *"sparse"* if the W ones in the binary sequence representation of any codeword in the set are distributed almost evenly among all F positions, and are not concentrated in one section of the sequence. In other words, if a code is sparse there are approximately $W/2$ ones in the first half of *any* of its codewords.

For large value of C, the use of sparse OOC allows us to further reduce energy consumption by slight changes to the protocol: The same method is used for half of the time slots, *i.e.*, for $F/2$ time slots. Every node in the network then has approximately $W/2$ opportunities to receive packets from each of its neighbors. Note that, ignoring fading and other effects, just one collision-free time slot is required for each link in order to find a neighbor. Hence, after $F/2$ time slots, all nodes know all of their neighbors (and their id numbers) with very high probability. However, they may not yet have completed measuring the link qualities as it requires the total of C collision-free time slots.

As the nodes know their neighbors' id number, they can construct their codewords and therefore their pattern (the nodes just use the same method, they used to find their own pattern). In the second half of the time, all nodes suspend to sleep mode. A node wakes up only when it has to transmit, or when it deduced from its neighbors' patterns that only one of them is transmitting, *i.e.*, when no collision occurs. In this way, the nodes avoid being active and losing power when either collision happens, or no transmission takes place.

One can approximately compute how much energy will be saved by introducing this modification: Consider a node with n neighbors. The amount of energy used by each node in the original method is given by (19). In the new method, during the first $F/2$ time slots the consumed energy is similar as in the original scheme. However, during the second $F/2$ time slots, the node will transmit in approximately $W/2$ time slots only, and so will each of its neighbors. In the worst case from the energy saving point of view, there is no collision at all and therefore the node will be awake listening to the channel for $n\frac{W}{2}$ time slots. This implies

$$\begin{aligned} E'_{\text{node}} &\leq \frac{F + (\beta - 1)W}{2} E_{RX} + \frac{W}{2}(n + \beta)E_{RX} \\ &= \frac{F + W(2\beta + n - 1)}{2} E_{RX}. \end{aligned} \tag{33}$$

Hence, the energy saved is at least

$$E_{\text{node}} - E'_{\text{node}} \geq \frac{F - (n + 1)W}{2} E_{RX}. \tag{34}$$

The above result is only valid when $F > (n+1)W$. In the case that this condition does not hold, still some energy will be saved but the above expression will be meaningless.

As an example, the case for $C = 30$ is considered with patterns generated from a (4000,55,1) OOC set. Assuming $\beta = 2.5$ and $n = \bar{n} = 15$, in the original scheme, each node will consume $4082.5E_{RX}$. After introducing this new feature, the energy used in each node is reduced to approximately $2522.5E_{RX}$.

The above method may also be applied to different fractions of the overall link assessment time. For example, the nodes may all be awake for the first $F/3$ time slots, and may then reconstruct the codewords and wake up only when it is necessary for the remaining $2F/3$ time slots. By decreasing the initial part, more energy can be saved, however, the chance of losing some of the neighbors increases.

V. Conclusion

Our paper highlights the importance of performing 'link assessment' (prior to the network topology formation) which acquires information about availability and quality of RF links. This information is required to make routing decisions in order to form a reliable multi-hop network.

The main objective of this work has been to reduce time and power consumption of this process, while ensuring a sufficient accuracy of the collected data. First, a protocol has been introduced using a random pattern for shifting between transmit, listen, and sleep states for each node. The results show that it is not beneficial to utilize the sleep state of the nodes during this process.

Next, it has been described how constant-weight codes can be used to construct optimized patterns for transmission and reception. The Hamming distance property of these codes helps to achieve improved performance over the random scheme. In particular optical orthogonal codes as a special class of constant-weight codes were considered. It was shown that due to properties of OOCs, the pattern assignment can be done very easily, therefore they are preferred over other codes. Moreover, it was shown that, by introducing sparse OOCs and modifications to the protocol, the power consumption can further be reduced.

ACKNOWLEDGMENT

The authors would like to thank Neda Beheshti, Dhanajay Lal, Balaji Prabhakar, Mayank Sharma, Bhaskar Srinivasan, Lakshmi Venkatraman, and Sina Zahedi for valuable input and discussion.

REFERENCES

[1] A. Cerpa and D. Estrin, "ASCENT: Adaptive Self-Configuring Sensor Network Topologies.", *Proceeding of 21st International Annual Joint Conference of IEEE computer and communication Societies (INOFCOM 2002)*, June 2002.

[2] M. D. Yarvis, W. S. Conner, L. Krishnamurthy, J. Chabra, B. Elliot, and A. Mainwarning, "Real-World Experiences with an Interactive Ad Hoc Sensor Network," *Proceeding of International Workshop on Ad-Hoc Networking (IWAHN 2002)*, August 2002.

[3] P.R. Kumar, "New Technological Vistas for Systems and Control: The Example of Wireless Networks," *IEEE Control System Magazine*, pp.24-36, February 2001.

[4] K. Sohrabi, J. Gao, V. Ailawadhi, and G.J. Pottie, "Protocols for self-organization of a wireless sensor network,", *IEEE Personal Communications*, vol. 7, no. 5, pp. 16-27, Oct. 200.

[5] F.J. Block, and C.W. Baum, C.W., "Energy-efficient self-organizing communication protocols for wireless sensor networks," *IEEE Military Communications Conference, 2001. MILCOM 2001*, vol. 1, pp. 362-266, Oct 2001.

[6] A. Ephremides and T.V. Truong, "Scheduling broadcasts in multi-hop radio networks," *IEEE Trans. Comminication*, vol. 38, no. 4, pp. 456-460, April 1990.

[7] B. Hajek and G. Sasaki, "Link Scheduling in Polynomial Time," *IEEE Trans Info Theory*, vol. 34, no. 5, pp 910-917, Sept 1988.

[8] L.G. Robetrs, "ALOHA Pakcet system with and without slots and capture," *Computer Communication Rev.*, vol. 5, pp. 28-42, April 1975.

[9] M. Zorzi and R. Rao, "Capture and retransmission control in mobile radio," *IEEE Journal on Selected Areas in Communications*, Vol. 12, No.8, pp. 1289-1298, Oct. 1994.

[10] J.R. Smith, "Distributing Identity,", *IEEE Robotics and Automation Magazine*, Vol. 6, No. 1, pp. 49-56.

[11] J. Elson and D. Estrin, "Time synchronization for wireless sensor networks," *Parallel and Distributed Processing Symposium.*, Proceedings 15th International .pp. 1965-1970, April 2001

[12] A.E. Brouwer, J.B. Sherer, N.J.A. Sloane, and W.D. Smith, "A new table of constant weight codes," *IEEE Transaction on Information Theroy*, vol. 36, pp. 1334-1380, Nov. 1990.

[13] R.L. Graham and N.J.A. Sloane, "Lower bounds for constant weight codes," *IEEE Transaction on Information Theory*, vol. 26, pp.37-43, Jan. 1980.

[14] E. Agrell, A. Vardy, and K. Zeger, "Upper bounds for constant weight codes", *IEEE Transactions on Information Theory*, vol. 46, no. 7, pp. 2373-2395, Nov. 2000.

[15] E. Agrell, A. Vardy, and K. Zeger, "A table of upper bounds for binary codes", *IEEE Transactions on Information Theory*, vol. 47, no. 7, pp. 3004-3006, Nov. 2001.

[16] S. Bitam, T. Etzion, "Constructions for optimal constant weight cyclically permutable codes and different families," *IEEE Trans. on Information Theory*, vol. 41, No. 1, Jan 1995.

[17] J.A. Salehi, "Code-division mutiple-access techniques in optical fiber netwroks - Part I: Fundumental principles," *IEEE Transaction on Comm.*, vol. 37, No. 8, pp. 824-833, Aug. 1989.

[18] F.R.K. Chung, J.A. Salehi, and V.K. Wei, "Optical Orthogonal Codes: Design, analysis, and applications," *IEEE Trans. Inform. Theory*, vol. 35, pp. 595-604, May 1989.

[19] H. Chung and P. V. Kumar, "Optical Orthogonal Codes-New Bounds and an optimal construction," *IEEE Trans. Inform. Theory*, vol. 36, pp. 866-873, July 1990.

[20] I. Chlamtac and A. Farago, "Making Transmission Schedules Immune to Topology Changes in Multi-hop Packet Radio Networks," *IEEE/ACM Trans. Networking*, vol. 2, pp. 23-29, Fen. 1994.

[21] J. Ju and V.O.K. Li,"An Optimal Topology-Transparent Scheduling Method in Multihop Packet Radio Networks," *IEEE/ACM Trans. Netwroking*, vol. 6, no. 3, pp. 298-306, June 1998.

[22] S. Lin and D.J. Costello, *Error Control Coding: Fundamentals and Applications*, Prentice Hall: Englewood Cliffs, NJ, 1983.

[23] A. Papoulis ,Probability and Statistics, *Prentice Hall*, 1st edition. 1990.

[24] D. Lal, A. Manjeshwar, F. Hermann, E. Uysal, and A. Keshavarzian, "Measurement and Characterization of Link Quality Metrics in Energy Constrained Wireless Sensor Networks," *to appear in Globecome 2003.*

APPENDIX I

In this appendix expressions are derived for (9), (10), and (11). This is carried out in three steps. The first step is to approximate P_L in (6) and to get an expression for P_{cf} as a function of F and other parameters. Next, F is considered as a fixed known parameter, and P_{TX} and P_{RX} are determined such that $\mathbb{E}(E_{\text{node}})$ is minimized while satisfying the condition $P_L \leq P_{L,max}$. In a third step, the best value for F is determined. n is used instead of n_{max} in all the following equations.

It is known that under certain conditions a binomial distribution can be approximated as a normal (Gaussian) random variable [23]. It is also known that $C_{i,j}$ has a binomial distribution (see (5)) with mean $\mu = FP_{\text{cf}}$ and variance $\sigma^2 = FP_{\text{cf}}(1 - P_{\text{cf}})$. Hence, P_L can be approximated as follows:

$$\begin{aligned} P_L &= \mathbb{P}(C_{i,j} < C) \\ &\approx Q\left(\frac{\mu - C}{\sigma}\right) \\ &= Q\left(\frac{FP_{\text{cf}} - C}{\sqrt{FP_{\text{cf}}(1-P_{\text{cf}})}}\right), \end{aligned} \tag{35}$$

where $Q(x)$ is defined as

$$Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^{\infty} e^{-\frac{t^2}{2}} dt. \tag{36}$$

Since $Q(x)$ is a decreasing function, $P_L \leq P_{L,max}$ can be written as

$$\frac{FP_{\text{cf}} - C}{\sqrt{FP_{\text{cf}}(1-P_{\text{cf}})}} \geq Q^{-1}(P_{L,max}). \tag{37}$$

Let $q = Q^{-1}(P_{L,max})$, then solving for P_{cf} one can obtain

$$P_{\text{cf}} \geq \frac{2C + q^2 + q\sqrt{q^2 + 4C\left(1 - \frac{C}{F}\right)}}{2(F+q)}. \tag{38}$$

Assuming that F is known and using the above expression, one can derive a lower bound on P_{cf}. $\mathbb{E}(E_{\text{node}})$ is minimized (see (7)) while satisfying (38). Let the right-hand side of (38) be represented by ξ. Combining it with (4), we obtain:

$$P_{TX}(1-P_{TX})^{n-1}P_{RX} \geq \xi. \tag{39}$$

The optimization problem reduces to finding P_{TX} and P_{RX} such that (39) holds and $\beta P_{TX}+P_{RX}$ is minimized. After some simplification, we find that the optimum P_{TX} should satisfy the following equation:

$$\beta x^2(1-x)^n - \xi(1-nx) = 0, \tag{40}$$

where P_{TX} is replaced by x and $P_{RX} = \frac{\xi}{x(1-x)^{n-1}}$. This equation can be solved using numerical methods. Results of computations indicated that an approximation of $P_{TX} \approx \frac{1}{n}$ is accurate . Using this result and combining it with (39), one can obtain

$$P_{RX} = \frac{\frac{2C+q^2+q\sqrt{q^2+4C\left(1-\frac{C}{F}\right)}}{2(F+q)}}{\frac{1}{n}\left(1-\frac{1}{n}\right)^{n-1}}. \tag{41}$$

Substituting this expression in (7) one obtains

$$\mathbb{E}(E) = F\left(\frac{\beta}{n} + \frac{2C+q^2+q\sqrt{q^2+4C\left(1-\frac{C}{F}\right)}}{2\frac{1}{n}\left(1-\frac{1}{n}\right)^{n-1}(F+q)}\right)E_{RX}. \tag{42}$$

It can bee seen that $\mathbb{E}(E_{\text{node}})$ is an increasing function of F, hence the best value for F which minimizes the above expression is the smallest value possible. It is known that $P_{RX}+P_{TX} \leq 1$, hence

$$P_{RX} \leq 1 - \frac{1}{n}. \tag{43}$$

Using the above expression combined with (41) and the two simplifying assumptions that $(1-\frac{C}{F}) \approx 1$ and $F+q \approx F$, the following expression can be derived as the best selection of parameter F:

$$F \approx \frac{2C+q^2+q\sqrt{q^2+4C}}{2\frac{1}{n}\left(1-\frac{1}{n}\right)^n}. \tag{44}$$

Note that the above two assumptions are reasonable. C and q typically are in the order of some tens, while F typically is in the order of thousands. Furthermore, note that the smallest value of F is obtained when in (43) equality holds. Hence, $P_{RX} \approx 1-\frac{1}{n}$.

APPENDIX II

Assuming that λ is known and fixed, the best value for W is the smallest value, *i.e.*, $W = \lambda n_{max}+C$, since with a larger W, a larger F is needed to get the same number of codewords. The Johnson upper bound states that [17]–[19]:

$$\Phi(F,W,\lambda) \leq \left\lfloor \frac{F(F-1)(F-2)\dots(F-\lambda)}{W(W-1)(W-2)\dots(W-\lambda)} \right\rfloor. \tag{45}$$

We assume Johnson's bound is valid and then approximate and use it to find F as a function of W and λ.

$$\Phi(F,W,\lambda) \approx \frac{F^\lambda}{(W-\lambda)^{\lambda+1}}. \tag{46}$$

Combining it with (22), one obtains the inequality

$$F \geq \left\lceil (W-\lambda)(N)^{\frac{1}{\lambda+1}} \right\rceil. \tag{47}$$

In [18], [19], it is shown that one of the necessary conditions to get a non-empty code set, is:

$$F \geq \frac{W^2}{\lambda}. \tag{48}$$

Both (47) and (48) must hold in order to get the desired number of codewords. However, in most practical situations with typical network parameters, (48) is the more restricting condition. Hence it can be assumed that $F \approx \frac{W^2}{\lambda}$. Now we have both F and W as functions of λ and other known parameters. Therefore, one can write E_{node} in (19) as a function of λ,

$$\frac{E_{\text{node}}}{E_{RX}} = \frac{(n_{max}\lambda+C)^2}{\lambda} + (\beta-1)(n_{max}\lambda+C). \tag{49}$$

This expression is convex with respect to λ, and the value of λ which minimizes it, can be determined :

$$\lambda = \frac{C}{\sqrt{n_{max}^2+n_{max}(\beta-1)}} \leq \frac{C}{n_{max}}. \tag{50}$$

Although it is possible that for some network parameters the best design leads to a $\lambda > 1$, in most practical situations one will find $\frac{C}{n_{max}} < 1$. Hence, the best option is to set $\lambda = 1$.