Neighbor discovery with reception status feedback to transmitters

Ramin Khalili
EPFL IC-LCA
Switzerland.
ramin.khalili@epfl.ch

Dennis L. Goeckel
ECE Department, UMASS
Amherst, MA
goeckel@ecs.umass.edu

Don Towsley
CS Department, UMASS
Amherst, MA
towsley@cs.umass.edu

Ananthram Swami
Army Research Lab
Adelphi, MD
a.swami@ieee.org

***Abstract*—Neighbor discovery is essential for the process of self-organization of a wireless network, where almost all routing and medium access protocols need knowledge of one-hop neighbors. In this paper we study the problem of neighbor discovery in a static and synchronous network, where time is divided into slots, each of duration equal to the time required to transmit a hello message, and potentially, some sort of feedback message. Our main contributions lie in detailing the physical layer mechanism for how nodes in receive mode detect the channel status, describing algorithms at higher layers that exploit such a knowledge, and characterizing the significant gain obtained. In particular, we describe one possible physical layer architecture that allows receivers to detect collisions, and then introduce a feedback mechanism that makes the collision information available to the transmitters. This allows nodes to stop transmitting packets as soon as they learn about the successful reception of their discovery messages by the other nodes in the network. Hence, the number of nodes that need to transmit packets decreases over time. These nodes transmit with a probability that is inversely proportional to the number of active nodes in their neighborhood, which is estimated using the collision information available at the nodes. We show through analysis and simulations that our algorithm allows nodes to discover their neighbors in a significantly smaller amount of time compared to the case where reception status feedback is not available to the transmitters.**

I. Introduction

Wireless sensor and ad-hoc networks are increasingly being developed and deployed in civilian and military scenarios. Wireless networks are very attractive as they can be deployed without spending the time, money, and energy involved in setting up a wired network [12], [13]. In such a network, nodes coordinate among themselves to establish routes and communicate with each other. However, route construction generally requires nodes to know their one-hop neighbors after they are deployed. Knowledge of one-hop neighbors is also essential to perform medium-access control and to construct minimum weight (e.g. energy) spanning trees. Hence, in a self-organized wireless network, before the execution of any other protocol (such as topology control, medium access and routing protocols) can begin, nodes must execute an initialization phase during which they discover their one-hop neighbors, henceforth referred to as **neighbor discovery**. This phase must proceed as quickly as possible to allow the network to begin operation.

The most commonly used neighbor discovery algorithm is probabilistic, with nodes randomly exchanging neighbor discovery packets with their neighbors. The neighbor discovery packet is often broadcasted using a CSMA (Carrier Sense Multiple Access) protocol to reduce the probability of multiple nodes accessing the channels at the same time [9], [10]. If simultaneous channel accesses occur, collisions occur and colliding packets are assumed to be lost. These Aloha-like discovery algorithms are slow and suffer from the hidden terminal problem, which decreases their reliability. Recently, several slotted-time based neighbor discovery algorithms have been proposed [1]–[3], [5], [6], [8], [15], [17]. Time is slotted in order to enhance reliability and increase discovery speed; at each slot, each node decides (based on a deterministic or random decision process) to transmit or to listen to other transmissions. Note that the probability that two or more nodes transmit in a slot is greater than zero and thus collisions will occur in the network. In previous protocols [1]–[3], [5], [6], [8]–[10], [15], [17], nodes do not have knowledge of whether their transmissions have been received by all neighbors, and thus nodes must continue to transmit to discover neighbors with some desired reliability.

In [16], we considered the scenario that nodes have a reliable collision detection mechanism. We proposed an algorithm in which nodes stop transmitting packets as soon as they learn about the reception of their packets by their neighbors. For a single clique of n nodes and using a highly idealized PHY layer, we showed that:

- Using such an algorithm, each node discovers all its $n-1$ neighbors in an expected time equal to ne that yields at least $\ln n$ improvement over Aloha-like algorithms and is within a factor e of the optimal.
- Lack of synchronization among nodes results in at most a factor of two slowdown in the algorithm performance. We proposed an asynchronous version of our algorithm.

This work was supported in part by the NSF under grant CNS-072186, by NCCR-MICS (a center supported by the Swiss National Science Foundation), and the International Technology Alliance sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the National Science Foundation, the US Army Research Laboratory, the U.S. Government, the UK Ministry of Defense, or the UK Government. The US and UK Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

- Despite starting execution at different time instants, each node can discover all its neighbors. We also relaxed the requirement that the number of nodes be known.

As stated above, in [16] we assumed that nodes are all in transmission range of each other and transmission errors due to fading are negligible. Here, we take a more detailed look a the PHY layer where we consider the general case of channels with fading. We consider the network setting, where nodes are not all in transmission range of each other. The main contribution of the current paper can be summarized as follows.

- We propose a reliable and practical collision detection mechanism that provides collision information to the transmitters.
- We introduce an efficient retransmission scheme to deal with losses due to fading, where the collision information is used to enhance the performance of the scheme.
- We show through simulations that our algorithms allow nodes to discover their neighbors much faster than Aloha-like algorithms.

For the sake of simplicity, we consider that nodes are synchronous on slot boundaries and start the neighbor discovery process at the same time instants. However, the algorithms in [16] can be used in conjunction with what is proposed here in an analogous fashion to relax these assumptions.

In particular, we introduce a physical layer energy detection mechanism that gives nodes the ability to estimate their reception status when they are in receive mode. Such a mechanism has been widely used by medium access and physical layer protocols. Examples include CSMA/CA in which the decision that a node is permitted to either transmit or has to defer its transmission depends on whether it senses the channel as idle or busy. In this paper we consider more complex cases in which nodes are not only able to distinguish between busy and idle channels, but are also capable of discerning between errors and collisions. We detail one possible physical layer configuration for such. Using this energy detection mechanism, we propose a simple feedback mechanism that provides collision information to the transmitters. With the feedback, we only need to carry one bit of information to the transmitters; whether collisions occurred during their transmissions or not. Hence, we do not need to schedule receivers; rather, they simultaneously transmit NACK packets during a slot that is reserved for feedback if they detect collision in the previous transmission slot. If a transmitter detects energy during the feedback slot corresponding to its transmission slot, it concludes that a collision has occurred during its transmission.

After introducing the feedback mechanism, we discuss the slotted version of the discovery algorithm proposed in [16], where we consider that nodes are all in transmission range of each other, they know the number of their neighbors, and transmissions are over additive white Gaussian noise (AWGN) channels. We then address the case of channels with fading, where a collision is not the only source of loss in the network and packets can be lost due to transmission errors. We propose an efficient retransmission scheme to enhance the reliability of our discovery algorithm, where we use the collision information to increase the probability that no collisions occur during the retransmission of a packet. Finally, we consider that nodes do not know the exact number of their neighbors. Hence, nodes need to estimate this number to adjust their transmission probabilities. We show that collision information is useful for this estimation procedure and can be used by nodes to adjust their transmission probabilities. We evaluate the performance of our algorithm in the network setting, where we relax the last assumption that nodes are all in transmission range of each other. We show that our algorithm allows nodes to discover their neighbors in a significantly smaller amount of time compared to Aloha-like discovery algorithms.

The goal of this paper is to clearly demonstrate the utility of physical layer feedback in facilitating neighbor discovery. Hence, we take a clean Aloha-like approach as a baseline, rather than one of the more complicated algorithms of [1]–[3], [5], [6], [8], [15], [17]. In addition, we note that [16] has shown that such an approach is order optimal in the single clique case without fading, and we believe that these results also hold for the network setting with fading channels as assumed in this paper. Hence, we relegate comparison of our algorithms with more advanced algorithms to the future.

The rest of this paper is structured as follows. In the following section we describe the network setting and assumptions for our study. Section III describes Aloha-like neighbor discovery algorithms and their analysis. In Section IV, we introduce our energy detection mechanism. We then propose our feedback mechanism and discuss its overhead cost in the rest of Section IV. Section V introduces our algorithms. We evaluate the performance of our algorithms in the network setting in Section VI. Finally, Section VII concludes this paper and discusses possible future research.

II. Model and assumptions

Consider a network of n nodes where nodes are not mobile. Each node is equipped with a transceiver that enables the node to transmit and receive a signal. Nodes are distinguishable by unique identifiers (ID) that can be MAC addresses or geographic positions provided by GPS. The goal is to propose an algorithm in which nodes in the network discover their neighbors, i.e. determine their identifiers, as quickly as possible. Neighbor discovery algorithms function by exchanging messages between nodes, where these messages contain the identifiers of transmitters. Hence, if a node receives a message, it discovers the transmitter of the message. In this paper we do not consider gossip-based approaches, hence the hello message contains only the identifier of the transmitter. We assume that two nodes are neighbors if they are in transmission range of each other, i.e. the signal transmitted by one of the nodes can be heard by the other node. Stated more precisely, we assume that when a node transmits, the *average* (over time) received energy at its neighbors is greater than a threshold.

We assume that time is divided into slots and nodes are perfectly synchronized. This synchronization can be obtained through a base station that broadcasts clock messages to the nodes or by using GPS clock synchronization [4], [7]. During any given time slot, a node can either be in transmit or receive mode but not both; i.e. nodes are half-duplex. A collision occurs in such a setting if two or more neighbors transmit at the same time, and the collided packets are assumed to be lost. However, a collision is not the only source of packet loss in the network where, due to fading and interference from non-neighboring nodes, the received signal-to-interference plus noise ratio (SINR) between any two nodes is time-varying; thus, there will be transmitted packets that cannot be successfully decoded by all neighbors even if no collision occurs in the network. We denote by P_e the expected packet loss rate due to fading. As we discuss in Section IV, a receiver can distinguish between successful reception, collision, transmission error, and the case that a channel is idle. However, this information is not automatically available at transmitters; hence, a transmitter can never be sure about the reception of its discovery message without feedback from receivers. In this paper, we first briefly study random discovery algorithms without feedback. We then consider in detail the case that collision information is made available to transmitting nodes.

III. Random protocol without feedback

Each slot is assumed to have duration equal to the time required to transmit a packet, where L (in bits) is the size of a discovery packet. Consider a random node i. At every slot, i selects one of these modes; with probability p_i the node transmits and with probability $(1 - p_i)$ it chooses to receive (listen). When a node is in receive mode and if it successfully receives a transmission from a neighbor, the node records the identity of the neighbor. We refer to this algorithm as Algorithm 0. Let n_i be the number of neighbors of node i; $p_i^* = \frac{1}{n_i}$ is the value of p_i that maximizes the probability of discovering a neighbor within a given time (see [15] and [6] for more details).

IV. Reception Status Feedback

An important assumption underlying the current work is the ability of receiving nodes to distinguish between successful reception, collision, transmission error, and a free channel. The idea is that the amount of energy received during a given time slot at a receiver can be used to determine the activity during that time period. Our energy detection mechanism performs as follows. First, a receiving node checks whether the received signal power during a given listening period, averaged over the period, is above a given threshold τ_1. If the answer is "yes", the receiving node attempts to decode the packet; if the decoding is successful, the node recovers the ID of its neighbor and considers the status a success, else, it decides the status was collision. If the average received power at a node is less than τ_1, the node needs to decide if something is on the channel to distinguish between the error and free states. The output of the energy detector is then compared to a second threshold τ_0, which is set above the power of the noise floor but below τ_1. If the received signal power is above τ_0, the node estimates its status as error, else it decides that its reception status is free (see Figure 1). In this section, we first describe one possible physical layer architecture for the generation of such. We then propose a feedback mechanism that provide this information to transmitters.

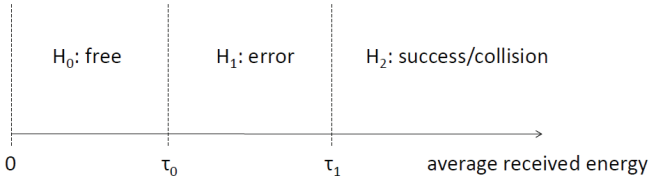

Fig. 1. Receivers need to decide between three hypothesis; free, error, and success/collision. We propose an energy detection mechanism at the PHY layer of nodes that compares the average received energy, γ, with two thresholds τ_0 and τ_1, where $\tau_0 < \tau_1$. A receiver considers its reception status as free if $\gamma < \tau_0$ and as error if $\tau_0 \leq \gamma < \tau_1$, o.w., it attempts to decode the packet; if decoding is successful its reception status is success, else, it concludes that a collision occured.

A. Energy Detection Mechanism

A potential scheme is provided based on the assumption that discovery messages are transmitted using (potentially coded) binary phase-shift keying (BPSK); however, it will be clear that similar results are easily obtained in an analogous fashion for other modulation schemes.

Consider packets of L bits transmitted by a set of K active nodes, that are in transmission range of the receiver of interest, during a given time slot. The transmitted signal from the k^{th} active node during a given time interval is given by:

$$s_k(t) = \sum_{l=0}^{L-1} \sqrt{E_s}(-1)^{b_l^{(k)}} p(t - lT_s) \qquad k = 0, 1, \ldots, K-1$$

where E_s is the transmitted energy per symbol, $b_l^{(k)}$ is the l^{th} data bit of the "hello" message for the k^{th} user, $p(\cdot)$ is the unit-energy pulse shape, and T_s is the symbol time. We will assume that the gain (inverse of the attenuation caused by path-loss and shadowing) β on each of the paths is identical, and define the average received energy from a given transmitter as $E_r = \beta E_s$. Assuming independent slow frequency-nonselective Rayleigh fading between each of the active transmitters and the receiver of interest, the multipath fading on a given packet from transmitter k is a (scalar) complex zero-mean Gaussian random variable h_k [14]. Then, working any phase offset (and time synchronization in the narrow-band case) between transmit and receive oscillators into the complex variable h_k, the (complex) received signal during a given time slot is:

$$y(t) = \sum_{k=0}^{K-1} \sqrt{\beta} h_k s_k(t) + v(t) + i(t) \tag{1}$$

where $v(t)$ is an additive white Gaussian noise process with (two-sided) power spectral density $\frac{N_0}{2}$ and $i(t)$ is a Gaussian

process with power density I that represents interference from non-neighboring nodes. After pulse-matched filtering and symbol-spaced sampling, the elements of the (complex) received vector for a given packet are given by:

$$y_l = \sum_{k=0}^{K-1} \sqrt{E_r} h_k (-1)^{b_l^{(k)}} + v_l + i_l, \quad , l = 0, 1, \ldots, L-1$$

where v_l and i_l are zero-mean complex Gaussian random variables with variance $\frac{N_0}{2}$ and I, respectively.

Modern error control codes demonstrate a threshold effect; that is, if there is no collision ($K = 1$) and the received signal-to-noise-interference ratio (SINR) $\frac{E_r|h_0|^2}{N_0+2I}$ is above a given threshold $\frac{\gamma_d}{N_0}$, the packet decodes with high probability. Hence,

$$P_e = 1 - P(|h_0|^2 \geq \frac{(N_0+2I)\gamma_d}{N_0 E_r}) = 1 - e^{-\frac{(N_0+2I)\gamma_d}{N_0 E_r}}. \quad (2)$$

Likewise, if the received SNR is below that threshold, it is quite unlikely that the packet will be decoded. In the latter case, the receiver wishes to determine the status of the channel. In the absence of a model for the number of colliding users and of their sequences, one could use an energy detector. The output of the energy detector, normalized by L, is:

$$\begin{aligned} \gamma &= \frac{1}{L} \sum_{l=0}^{L-1} |y_l|^2 \\ &= \frac{1}{L} \sum_{l=0}^{L-1} |\sum_{k=0}^{K-1} h_k \sqrt{E_r} (-1)^{b_l^{(k)}} + v_l + i_l|^2. \end{aligned}$$

In particular, as detailed below, our feedback mechanism requires the receiver to decide between the following three hypotheses, listed in increasing order of (expected) received energy (see Figure 1):

$$\begin{aligned} H_0 &: \text{free} \\ H_1 &: \text{error} \\ H_2 &: \text{success / collision} \end{aligned}$$

The statistics of γ under the three hypotheses can be easily evaluated and a minimum error classifier for the 3-class problem derived. We concentrate on a threshold detector which is close to optimal in the case of low SNR. Since "success" can be determined by the packet decoding successfully, the thresholding on γ need only try to detect a collision, and, hence, with a slight abuse of notation, γ under the three hypotheses can be written as:

$$\begin{aligned} H_0 &: \gamma = \frac{1}{L} \sum_{l=0}^{L-1} |v_l + i_l|^2 \\ H_1 &: \gamma = \frac{1}{L} \sum_{l=0}^{L-1} |\tilde{h}_0 \sqrt{E_r} (-1) * b_l^{(0)} + v_l + i_l|^2 \\ H_2 &: \gamma = \frac{1}{L} \sum_{l=0}^{L-1} |\sum_{k=0}^{K-1} h_k \sqrt{E_r} (-1)^{b_l^{(k)}} + v_l + i_l|^2 \end{aligned}$$

where $\tilde{h}_0$ has probability density function of the random variable h_0 conditioned on the event $\{|h_0|^2 < \frac{(N_0+2I)\gamma_d}{N_0 E_r}\}$. The logical hypothesis test is then given by:

$$\begin{aligned} \gamma < \tau_0 &: \text{Decide } H_0 \\ \tau_0 \leq \gamma < \tau_1 &: \text{Decide } H_1 \\ \tau_1 \leq \gamma &: \text{Decide } H_2 \end{aligned}$$

where τ_0 and τ_1 are two thresholds to be selected.

The thresholds τ_0 and τ_1 can be set to minimize the total error probability. When L is large, the decision statistic γ can be approximated as a real Gaussian variable with means μ_i and variance μ_i^2 under the i-th hypothesis, where

$$\begin{aligned} \mu_0 &= \frac{N_0}{2} + I, \\ \mu_1 &= \mu_0 + \vartheta E_r, \\ \mu_2 &= \mu_0 + K E_r, \end{aligned}$$

and $\vartheta = (1 - e^{-\frac{(N_0+2I)\gamma_d}{N_0 E_r}})(\frac{(N_0+2I)\gamma_d}{N_0 E_r} + 1) < 1$. Hence, the threshold τ_0 could be set to $\mu_o + \frac{\vartheta E_r}{2}$ and τ_1 to $\mu_0 + \frac{(1+\vartheta)E_r}{2}$. In practice, nodes can use an estimate of I to set their thresholds.

Let $P_{i\to j}$ be the probability of selecting hypothesis j when hypothesis i is correct. First, consider $P_{0\to j}$, $j = 0, 1, 2$. Under H_0, γ has a central chi-square distribution with $2L$ degrees of freedom, and thus it is straightforward to show (see, for example, [11, pg. 43]):

$$\begin{aligned} P_{0\to 2} &= P(\gamma \geq \tau_1 | H_0) \\ &= e^{-\frac{L\tau_1}{N_0+2I}} \sum_{k=0}^{L-1} \frac{1}{k!} \left(\frac{L\tau_1}{N_0+2I}\right)^k, \\ P_{0\to 1} &= P(\tau_0 \leq \gamma < \tau_1 | H_0) \\ &= e^{-\frac{L\tau_0}{N_0+2I}} \sum_{k=0}^{L-1} \frac{1}{k!} \left(\frac{L\tau_0}{N_0+2I}\right)^k \\ &- e^{-\frac{L\tau_1}{N_0+2I}} \sum_{k=0}^{L-1} \frac{1}{k!} \left(\frac{L\tau_1}{N_0+2I}\right)^k, \\ P_{0\to 0} &= 1 - P_{0\to 2} - P_{0\to 1}. \end{aligned}$$

Next, consider $P_{1\to j}, j = 0, 1, 2$. Under H_1, the distribution of γ is independent of the value of the data bits; hence, $\{b_l^{(0)}, l = 0, 1, \ldots, L-1\}$ can be assumed to be 0. It is then easy to observe that, conditioned on $\tilde{h}_0$, γ is non-central chi-square with $2L$ degrees of freedom and (see, for example, [11, pg. 45]):

$$\begin{aligned} P_{1\to 2} &= P(\gamma \geq \tau_1 | H_1) \\ &= E_{\tilde{h}_0}[P(\gamma \geq \tau_1 | \tilde{h}_0, H_1)] \\ &= E_{\tilde{h}_0}\left[Q_L\left(\sqrt{\frac{2|\tilde{h}_0|^2 L E_r}{N_0+2I}}, \sqrt{\frac{2L\tau_1}{N_0+2I}}\right)\right], \end{aligned}$$

$$
\begin{aligned}
P_{1\to 1} &= P(\tau_0 \le \gamma < \tau_1 | H_1) \\
&= E_{\tilde{h}_0}[P(\tau_0 \le \gamma < \tau_1 | \tilde{h}_0, H_1)] \\
&= E_{\tilde{h}_0}\left[Q_L\left(\sqrt{\frac{2|\tilde{h}_0|^2 L E_r}{N_0 + 2I}}, \sqrt{\frac{2L\tau_0}{N_0 + 2I}} \right) \right] \\
&- E_{\tilde{h}_0}\left[Q_L\left(\sqrt{\frac{2|\tilde{h}_0|^2 L E_r}{N_0 + 2I}}, \sqrt{\frac{2L\tau_1}{N_0 + 2I}} \right) \right], \\
P_{1\to 0} &= 1 - P_{1\to 2} - P_{1\to 1},
\end{aligned}
$$

where $Q_L(\cdot,\cdot)$ is Marcum's generalized Q function. Finally, consider $P_{2\to j}, j = 0, 1, 2$. Under H_2, the most likely chance of not detecting that collision occurs when there are only two users; hence, assume $\gamma = \sum_{l=0}^{L-1} |h_0\sqrt{E_r}(-1)^{b_l^{(0)}} + h_1\sqrt{E_r}(-1)^{b_l^{(1)}} + v_l + i_l|^2$. Here, we require only the probability of not detecting a collision (without regard to what type of error), which we denote $P_{2\to\{0,1\}}$, which allows the following bounding. The randomness of the data bits (over l) provide a form of diversity, and hence performance is lower bounded if the bits are assumed constant; that is, if the data bits in the sets $\{b_l^{(0)}, l = 0, 1, \ldots, L-1\}$ and $\{b_l^{(1)}, l = 0, 1, \ldots, L-1\}$ are assumed to be equal to zero. Then, employing this bound and the distribution of non-central chi-square random variables again, we have

$$
\begin{aligned}
P_{2\to\{0,1\}} &= P(\gamma \le \tau_1 | H_2) \\
&\le P(\sum_{l=0}^{L-1} |\sqrt{E_r}[h_0(-1)^{b_l^{(0)}} + h_1(-1)^{b_l^{(1)}}] + \\
&\qquad v_l + i_l|^2 < L\tau_1) \\
&\le P(\sum_{l=0}^{L-1} |h_0\sqrt{E_r} + h_1\sqrt{E_r} + v_l + i_l|^2 < L\tau_1) \\
&= E_{h_0,h_1}[P(\sum_{l=0}^{L-1} |h_0\sqrt{E_r} + h_1\sqrt{E_r} + \\
&\qquad v_l + i_l|^2 < L\tau_1) | h_0, h_1] \\
&= E_{h_0,h_1}\left[1 - Q_L\left(\sqrt{\frac{2|h_0 + h_1|^2 L E_r}{N_0 + 2I}}, \sqrt{\frac{2L\tau_1}{N_0 + 2I}} \right) \right]
\end{aligned}
$$

and

$$
P_{2\to 2} \ge E_{h_0,h_1}\left[Q_L\left(\sqrt{\frac{2|h_0 + h_1|^2 L E_r}{N_0 + 2I}}, \sqrt{\frac{2L\tau_1}{N_0 + 2I}} \right) \right].
$$

B. Feedback Mechanism

Now we show how receivers provide collision information to transmitters. We assume that time is divided into slots, each of duration equal to the time required to transmit a packet of size $(1+\alpha)L$ bits, where $0 < \alpha \le 1$. We divide each such slot into two sub-slots as depicted in Figure 2. The first sub-slot is assigned to nodes to broadcast their discovery packets of size L bits. The second sub-slot is used to send NACK packets of size αL bits. Receiving nodes transmit NACKs in the second sub-slot if they detect a collision during the first sub-slot. Note that these NACKs may collide when two or more nodes transmit in the second sub-slot. Nodes that transmitted during the first sub-slot, and receivers that did not detect a collision, listen to the channel during the second sub-slot and simply perform energy detection; if they detect their reception status as either success, collision, or error, they conclude that NACK packets were transmitted in the second sub-slot and that a collision occurred in the first time slot.

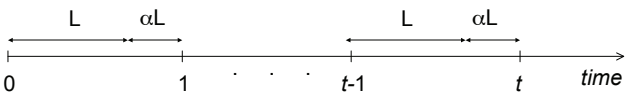

Fig. 2. Each transmission slot is divided into two sub-slots; the first sub-slot with a duration equal to the time that one requires to send a discovery packet of size L bits and the second sub-slot with duration equal to αL bits, which is the length of a NACK packet. Transmitting nodes transmit in the first sub-slot, and the second sub-slot is assigned to receiving nodes to transmit NACKs.

In theory, we only need one bit of feedback; hence, the corresponding overhead is negligible, i.e. $\alpha \ll 1$. However, in practice we need to consider constraints such as switching delay, synchronization complexity, and other physical layer constraints to make a transmission feasible. In particular, in a slotted communication protocol it is typical for the slots to be of equal length as this simplifies synchronization. In the approach described here where only one bit of information must be returned to complete the communication, then there would be an inefficiency arising from the rest of the acknowledgment slot being unused. We could increase efficiency by making the slot size smaller and using multiple short slots for the transmit period and a single short slot for the acknowledge message; however, a dramatic reduction in slot size to the "few bits" level could possibly result in synchronization issues and make it difficult to detect an acknowledgment from a distant node in such a short slot. For the sake of simplicity we consider $\alpha = 0.5$, which is a reasonable value for α when L is on the order of a few hundred bits.

C. Parametric example

Let $\gamma_d = 6\ dB$ and $E_r/N_0 = 16\ dB$; for $I = 0$ this results to $P_e = 0.1$ which is a reasonable value for packet loss rate in wireless networks [14]. Also, let $L = 200$ bits and $\alpha = 0.5$. At the first sub-slot of each slot, we have $\{P_{0\to 2} = 0,\ P_{0\to 1} = 0,\ P_{0\to 0} = 1\}$, $\{P_{1\to 2} \approx 0,\ P_{1\to 1} \approx 1,\ P_{1\to 0} \approx 0\}$, and $\{P_{2\to 0,1} \approx 0,\ P_{2\to 2} \approx 1\}$. At the second sub-slot we have $\{P_{0\to 2} \approx 0,\ P_{0\to 1} \approx 0,\ P_{0\to 0} \approx 1\}$, $\{P_{1\to 2} \approx 0,\ P_{1\to 1} = 1 - P_{1\to 0},\ P_{1\to 0} = 8 \times 10^{-7}\}$, and $\{P_{2\to 0,1} \approx 0,\ P_{2\to 2} \approx 1\}$, where transmitted NACK packets have a size of $\alpha \times L = 100$ bits. We can see from this example that our energy detection mechanism performs almost perfectly when $I = 0$ (e.g., inside a single clique where nodes are all in transmission range of each other as we assume in Section V). However, this mechanism is less reliable when the interference from non-neighboring nodes is not negligible (we will detail this in Section VI).

V. Random Discovery Algorithms with reception status Information at Transmitters

In this section we show how nodes in the network can use collision information to efficiently discover their neighbors. We explain the idea behind our algorithms for a single clique of n nodes, where neighbor discovery consists of determining the IDs of all nodes in the network. We start with the simplest case where the exact number of nodes in the network, n, is known by all nodes, and additive white Gaussian noise (AWGN) is assumed. We then extend our algorithm to channels with multipath fading. Finally, we consider the case where n is not known *a priori* and hence nodes must estimate n to determine their transmission probability.

A. AWGN Channels

We first consider the case of an AWGN channel where the average and instantaneous received SINRs between any two nodes are the same. Thus any loss between neighbors is caused by collisions. In this case, the probability that a receiver fails to detect a collision is essentially zero (see Section IV). Hence, if a node transmits a discovery message and receives no NACKs from receivers, it concludes that its transmission was successful. During the execution of neighbor discovery, nodes divide into *active nodes* and *passive nodes*. Here an active node is one that does not believe that it has been discovered and continues to actively participate in neighbor discovery. A passive node, on the other hand, is one that was once active but believes that it has been discovered by its neighbors and stops participating actively in neighbor discovery. Our algorithm functions as follows: once a node learns that its packet has been received by its neighbors, it stops transmitting discovery messages but continues to listen to the channel for transmissions from its neighbors. Therefore, the number of nodes attempting to transmit their hello messages, i.e. the number of active nodes, decreases over time. As stated before, we showed in [16] that this algorithm is in the order of optimal in the single clique case without fading.

Let $\mathcal{N}_h$ be the set of nodes that successfully transmit their messages in the first h time slots; and $\mathcal{N}_h^C$ be the complementary set of $\mathcal{N}_h$. $N_h = \mid \mathcal{N}_h \mid$ is the number of successful transmissions, success events, that occur by time h. Obviously, N_h is non-decreasing in h, $N_0 = 0$, and $0 \leq N_h \leq h$, as only a single successful transmission can occur in each time slot. $\mathcal{N}_{i,h}$ denotes the set of nodes discovered by node i in the first h time slots where $N_{i,h} = \mid \mathcal{N}_{i,h} \mid$. At time $h+1$, i executes the following algorithm; if $i \in \mathcal{N}_h^C$ it transmits with probability $p_{i,h}$ and listens with probability $1 - p_{i,h}$, otherwise, it listens to the channel. $p_{i,h}^* = \frac{1}{n - N_{i,h}}$ is the optimal value of $p_{i,h}$ that maximizes the probability of discovering a neighbor within a given time (see [16]). Note that, assuming that all nodes start with knowledge of n, the feedback protocol of the previous section guarantees that $\mathcal{N}_h$ is known by all nodes at time $h+1$, i.e., $\mathcal{N}_{i,h} = \mathcal{N}_h$ for any h and i.

We compare Algorithm 1 with Algorithm 0 through simulations where $P_e = 0$. Using Algorithm 1, each time slot is of length $(1+\alpha)L$ bits, while without feedback a slot is only of size L bits. We account for this difference in our evaluation. Figure 3 depicts the probability of non-discovery; the probability that a random node i does not discover neighbor j within time t time slots, for both of the protocols. We plot the results versus time in units of L bits slots with the overhead of αL accounted for in the proposed algorithm (i.e. k slots of the proposed algorithm requires $(1+\alpha)k$ time units).

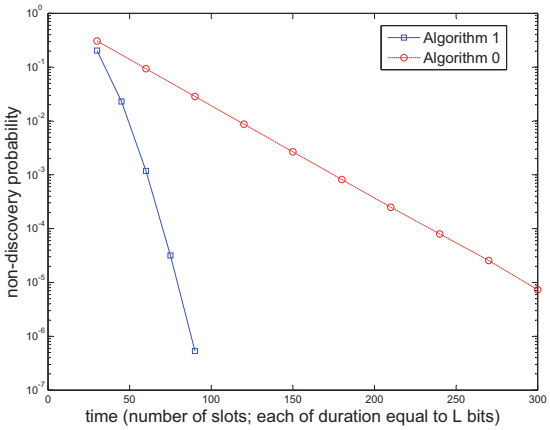

Fig. 3. The probability that a random node i does not discover its neighbor j within t slots (the non-discovery probability) is depicted using Algorithm 1 and the results are compared with the case that we apply Algorithm 0. The results are shown for networks of 10 nodes where $P_e = 0$. α represents the feedback overhead where each slot of the proposed algorithm in this section needs $1+\alpha$ time units. We set $\alpha = 0.5$.

The results clearly indicate that the probability of not discovering a neighbor decays much faster for the case that transmitters have side information about the reception of their messages. We see that the expected time to reach to a decay probability equal to 10^{-5} is up to 3 times less for the case that we use Algorithm 1, thus confirming our analytical results in [16]. We did simulations under Matlab where we applied Algorithm 0 and Algorithm 1. We consider a perfect energy detection mechanism at the physical layer and, per above, we suppose that transmission over links is errorless. To calculate the non-discovery probability at t, we count, at each node, the number of neighbors that have not been discovered after t slots, sum the obtained numbers at all nodes, and divided the sum by $10 \times 9 = 90$ where 10 is the number of nodes and 9 is the number of neighbors per each node. Every point obtained through simulation in Figure 3 corresponds to an average over 10^8 simulations. The confidence intervals are very small so we do not report them here.

B. Channels with fading

We consider the more general case of channels with fading. In this case, if a transmitter broadcasts a discovery packet and does not detect energy in the second sub-slot, it cannot conclude that its packet has been received by all neighbors, since with probability P_e, the expected packet loss probability between two neighbors, a neighbor may fail to decode the packet. In this section, we propose an extension of Algorithm 1 to address this problem.

In Algorithm 2, nodes switch to the passive mode only

after they have transmitted their packets at least b times, without detecting a collision during these transmissions. We propose an efficient retransmission scheme, where we use the collision information at transmitters and the ability of receivers to estimate their reception status to increase the probability that no collisions occur during the retransmission of a packet. Algorithm 2 operates as follows: if i successfully transmits a packet at time h (i.e. no collision occurs during the transmission of the packet), it retransmits the packet again at time $h+c$ with a probability p_r if $b_i < b$, where b_i is the number of times that node i has transmitted the packet without collision, b and c are positive integers, and $0 \leq p_r \leq 1$ (in this section, we set $p_r = 1$). Nodes that are in receiving mode at time h and detect their reception status as either success or error will assign to receiving mode at time $h+c$. Hence, the probability that a collision occurs during the retransmission of the packet is smaller than the case that all active nodes can transmit at time $h+c$. Once i transmits its packet b times without a collision, i.e. $b_i = b$, it switches to passive mode. Also, similar to Algorithm 1, if i is in receiving mode at time h and it receives a packet, it saves the identifier of the transmitter and if it is a new neighbor then $N_{i,h+1} = N_{i,h} + 1$. We assume enough temporal diversity that the multipath fading can be assumed to be independent from slot to slot, and thus set $c = 1$.

We applied Algorithm 2 in Matlab where we use the physical model proposed in Section IV with $I = 0$. The simulation results show that $p_{i,h} = \frac{1}{n - N_{i,h}}$ is the value of $p_{i,h}$ that maximizes the probability of discovering a neighbor within a given time. Hence, using Algorithm 2 we set $p_{i,h} = \frac{1}{n - N_{i,h}}$. Figure 4 depicts the non-discovery probability for the scenario described in the previous section with $n = 10$, $\alpha = 0.5$, and $P_e = 0.1$. Every point on a simulation curve corresponds to the average over 10^7 simulations, and shows the 95% confidence intervals. For each point on a curve we find the b^* that minimizes the non-discovery probability. The results indicate the performance of Algorithm 2 as it reaches a desired non-discovery probability much faster than Algorithm 0.

The parameter b^* can be estimated by $\left\lfloor \frac{\ln(P_{nd})}{\ln(P_e)} \right\rfloor + 1$, where P_{nd} is the desired non-discovery probability and $\lfloor A \rfloor$ is the nearest integer less than or equal to A. A rough explanation for this estimation is the following: if a node transmit a packet b^* times without a collision, then the probability that at least one of its neighbors fails to discover the node is equal or greater than $(P_e)^{b^*}$; hence, b^* need to be greater than $\left\lfloor \frac{\ln(P_{nd})}{\ln(P_e)} \right\rfloor$ to guarantee a non-discovery probability less than P_{nd}. The simulation results show that this is a good estimation hence we use it in our simulation experiments.

C. Estimating the number of neighbors

Until now, we have assumed that each node knows how many neighbors it has (in the case that all nodes are in transmission range of each other; this assumption is equivalent to saying that n is known to all the nodes). However, in many scenarios this number is not known *a priori*; hence, nodes need

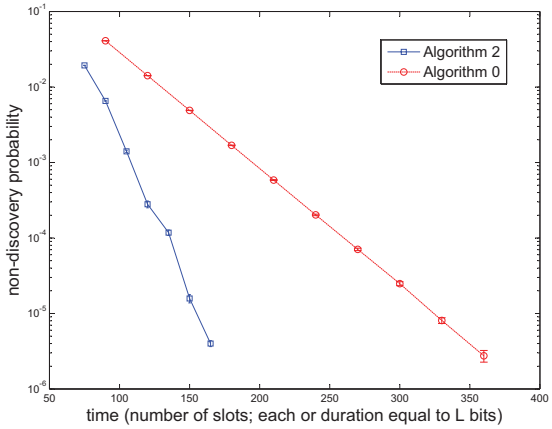

Fig. 4. The non-discovery probability of Algorithm 2 is compared with Algorithm 0 where $n = 10$ and $P_e = 0.1$.

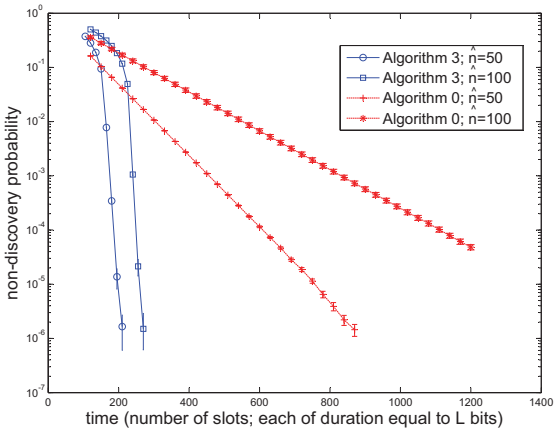

Fig. 5. The non-discovery probability of Algorithm 3 is compared with Algorithm 0, where $n = 10$, $P_e = 0.1$, and $\hat{n} = 50$ and 100. $\hat{n}$ is the initial estimate of the number of nodes in the network used by nodes.

to estimate this value to adjust their transmission probability. Previous studies show that overestimation or underestimation of this value degrades the performance of random discovery algorithms [6], [8], [15].

In this section, we extend Algorithm 2 to use collision information provided by our feedback mechanism to adjust the estimation of the number of neighbors of a node[1]. Let $\hat{n}_{i,h}$ be the node i estimate of the number of its neighbors at time h. We assume that there is a known upper bound for the number of neighbors of nodes in the network, $\hat{n}$, and this is used by the nodes at $h = 0$; i.e. $\hat{n}_{i,0} = \hat{n}$ for any i. Suppose $h \geq 1$ is a slot that is not assigned for retransmission and assume i is an active node at this slot. Using Algorithm 2, the probability that i transmits in h is $p_{i,h} = \frac{1}{(\hat{n}_{i,h} - N_{i,h})}$ where $(\hat{n}_{i,h} - N_{i,h})$ is i's estimate of the number of active nodes in its neighborhood. Recall that $N_{i,h}$ is the number of passive neighbors of i at time h. We described in Algorithm 2 how i

[1]We showed in [16] that not knowing n results in no more than a factor of two slowdown in the performance of discovery algorithms. We also proposed discovery algorithms that achieve this performance. However, algorithms in [16] rely on the assumptions that transmissions are over AWGN channels and nodes are all in transmission range of each other; hence, they cannot be directly applied here and need to be modified.

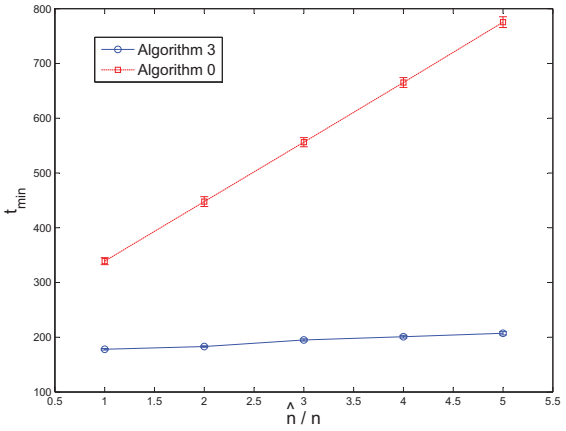

Fig. 6. t_{min} is the minimum time to reach a non-discovery probability less than 10^{-5}. The results are shown for $n = 10$ and $P_e = 0.1$.

estimates $N_{i,h}$. In Algorithm 3, we add a supplementary step to Algorithm 2 to use the collision information available at i to adjust its estimate of $\hat{n}_{i,h}$. Algorithm 3 functions as follows; at time h, if this time slot is not assigned for retransmission, then:

- $\hat{n}_{i,h+1} = \hat{n}_{i,h} + 1$ if i detects a collision at time h.
- $\hat{n}_{i,h+1} = \hat{n}_{i,h} - 1$ if i detects a free channel at time h.

The rest of Algorithm 2 remains unchanged.

Using Algorithm 3, i increases its estimate of the number of its neighbors by one if it detects a collision, as underestimation of the number of neighbor nodes increases the collision probability. It decreases $\hat{n}_{i,h}$ by one if it detects a free channel, where overestimation of $\hat{n}_{i,h}$ increases the probability that nobody transmits in a time slot. This algorithm is analogous to TCP in that TCP adjusts the transmission rate of transmitters (end users) to the level of congestion observed in the network, where here the probability that an active node transmits in a given time slot is adjusted to the level of collision seen by the node in previous slots. Algorithm 3 uses a simple additive increase additive decrease mechanism. This may not be the best algorithm, but simulation results show that it performs well compared to discovery algorithms without feedback. Note that applying Algorithm 0, nodes always transmit with the same probability, i.e. $p_i = \frac{1}{\hat{n}}$ for any i.

Figure 5 depicts the decay error probability of Algorithm 3, obtained through simulations, with the 95% confidence intervals. We also compare it to Algorithm 0. The results are shown for $n = 10$, $\alpha = 0.5$, $P_e = 10^{-5}$, and for $\hat{n} = 50,\ 100$. From Figures 5 and 4, we can observe that not knowing n results in at most a factor of two slowdown compared to the case that n is known *a priori* in which we apply Algorithm 2. The results also indicate that using Algorithm 3, nodes discover their neighbors much faster than Algorithm 0. In Figure 6, we depict t_{min}, the minimum time required to guarantee a non-discovery error probability less than 10^{-5}, versus $\frac{\hat{n}}{n}$. We see that increasing $\frac{\hat{n}}{n}$, t_{min} increases much faster using Algorithm 0.

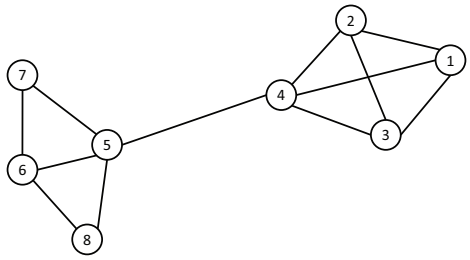

Fig. 7. A collision that occurs due to simultaneous transmissions of nodes 4 and 5 cannot be detected as none of the receiving nodes is in the transmission range of both the transmitters.

VI. Evaluation of Algorithm 3 in the network setting

In this section we consider scenarios in which nodes are not all in transmission range of each other. Figure 7 is an example of such a setting, where the lines show one-hop connectivity. If the separation between nodes 4 and 5 is large enough, then a transmission from node 4 (5) will degrade the received SNR at the local clusters $N_5 = \{6, 7, 8\}$ ($N_4 = \{1, 2, 3\}$) only slightly so that successfully decoding is possible and no collision is detected.

Recall that p_r is the probability that a transmitter retransmits its packet in its assigned retransmission slot as defined in Section 5.2. Now suppose the case that nodes 4 and 5 transmit at the same time and consider that all the other nodes in the network are in the receiving mode. Collisions occur due to simultaneous transmission of 4 and 5, but none of the nodes in the network can detect the collision. Hence 4 and 5 consider their transmissions as success, i.e. they conclude that there was no collision during their transmissions. The other nodes in the network either receive the packet transmitted by their neighbor or detect their reception status as error (the probability that some of the nodes detect their reception status as free is so small that we ignore it to make the example easier to follow.). Using Algorithm 3 with $p_r = 1$, nodes 4 and 5 both retransmit their packets at the next slot. Obviously, they will not detect a collision during their retransmissions so they continue the retransmission procedure until they decide to switch to the passive mode, i.e. nodes 4 and 5 will keep retransmitting in successive slots and will never discover each other. Simulations over different topologies confirm our observations from this example: using Algorithm 3 with $p_r = 1$, there is a small, but non-negligible, probability that a node will not discover all of its neighbors.

Here, we propose a backoff mechanism to solve this problem. Setting $p_r < 1$ achieves this. A very small value of p_r will degrade the algorithm since it increases the time to discovery. Hence, there is a p_r^* that minimizes the decay probability for a given t (see Sec. 5.2 for definition of t). However, finding p_r^* analytically is very difficult;hence, we relegate this to future work, and here show the performance of our algorithm for $p_r = 0.5$.

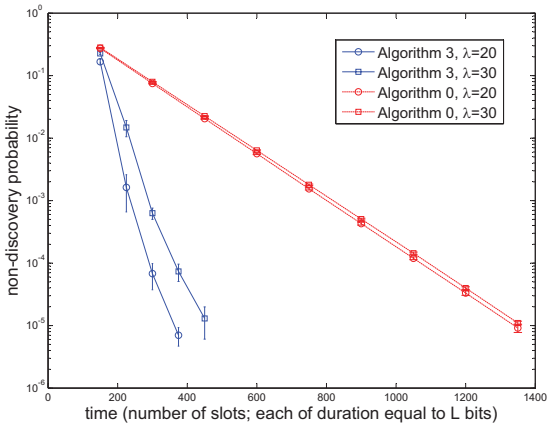

Fig. 8. The non-discovery probability using Algorithm 3 compared with the case that Algorithm 0 is used. The results are proposed for the network setting scenarios described in this section.

We consider scenarios where the position of nodes are distributed according to a Poisson point process of constant spatial intensity $\lambda \in R^2$, in a square of size 1×1. Nodes are neighbors if the distance between the nodes is less than r, the transmission range. Hence, if i is a neighbor of j, j is also a neighbor of i. However, if i discovers j, it does not mean that j has discovered i, as j needs to receive discovery packets from i to discover it. To calculate the decay probability at t, we count the number of neighbors at each node that have not been discovered in the first t slots; sum these values at all the nodes; and then divide the sum by the sum of the number of neighbors of each node in the entire network. We use the model proposed in Section IV-A for the physical layer transmissions, where we estimate I with $I = KN_0/2$, where K is the number of non-neighboring nodes of the receiver of interest that transmit in the slot and $N_0/2$ is the energy received from a non-neighboring interfering node. As we stated in Section IV-C, our collision detection mechanism is less reliable in the scenario we consider in this section compared to the single clique of Section V. Hence, we expect some degradation in the performance of our algorithm. However, as we will see, this degradation is negligible compared to the gain of using our algorithm.

In Figure 8, we depict the average of non-discovery probabilities that we obtained for 200 random scenarios, with the 95% confidence intervals, where for each scenario we run the simulation 10^6 times. We consider scenarios with $\lambda = 20,\ 30$ and $r = 0.3$. The results are shown for $\alpha = 0.5$, $\frac{E_r}{N_0} = 16$dB, and $\gamma_d = 6$dB. For each point on a curve we find the b^* that minimizes the decay probability. We set $\hat{n}_0 = 100$. The results clearly indicate the utility of feedback in the process of neighbor discovery. We observe again that the probability of non-discovery decays much faster when transmitters have access to collision information.

VII. Discussion and Conclusions

In this paper we have shown that reception status feedback can be obtained and is of great utility in reducing the time for discovering neighbors. The simulation and analytical results indicated that the expected time to discover all the neighbors using the discovery algorithm with feedback is smaller than that of algorithms that do not have access to collision information.

According to the retransmission scheme in Section V, a transmitter is discovered by its neighbors after sending multiple discovery packets. While real neighbors are identified more reliably with such a scheme, nodes far away from the transmitter in a network setting could be wrongly identified as legitimate neighbors because they would receive one of the discovery packets with a non-negligible probability. Note that the average number of discovery packets received from a node by the neighbors of the node is roughly $b(1 - P_e)$, where b is the retransmission threshold and P_e is the average transmission error probability. Hence, one can propose a threshold-based decision mechanism such that a node decides that another node in the network is its neighbor if the number of packets it has received from that node is greater than a threshold. We relegate the analysis of such a mechanism to the future.

References

[1] D. Baker and A. Ephremides. The architectural organization of a mobile radio network via a distributed algorithm. *IEEE Trans. Com.*, 29:1694–1701, 1981.

[2] S. A. Borbash, A. Ephremides, and M. J. McGlynn. An asynchronous neighbor discovery algorithm for wireless sensor networks. *Elsevier Ad Hoc Networks*, 5:998–1016, 2007.

[3] V. Dyo and C. Mascolo. A node discovery service for partially mobile sensor networks. *IEEE International Workshop on Sensor Network Middleware*, November 2007.

[4] J. Elson and D. Emin. Time synchronization for wireless sensor networks. *Parallel and distributed symposium*, April 2001.

[5] L. Jun and G. Dongning. Neighbor discovery in wireless ad hoc networks based on group testing. *Forty-Sixth Annual Allerton Conference*, September 2008.

[6] A. Keshavarzian, E. Uysal-Biyikoglu, F. Herrman, and A. Manjeshwar. Energy-efficient link assessment in wireless sensor networks. *IEEE Infocom*, 2004.

[7] J. Mannermaa, K. Kalliomaki, T. Mansten, and S. Turunen. Timing performance of various gps receivers. *In Proceedings of the 1999 Joint Meeting of the European Frequency and Time Forum and the IEEE International Frequency Control Symposium*, April 1999.

[8] M. McGlynn and S. Borbash. Birthday protocols for low energy deployment and flexible neighbor discovery in ad hoc wireless networks. *ACM MobiHoc*, 2001.

[9] T. Narten, E. Nordmark, and W. Simpson. Neighbor discovery for ip version 6. *RFC 2461, Internet Engineering Task Force*, December 1998.

[10] T. Narten, E. Nordmark, W. Simpson, and H. Soliman. Neighbor discovery for ip version 6 (ipv6). *RFC 4861, Internet Engineering Task Force*, September 2007.

[11] J. G. Proakis. *Digital communication*. McGraw-Hill, 1995.

[12] R. Ramanathan and J. Redi. Ad hoc networking with directional antennas: A complete system solution. *IEEE JSAC*, 23, March 2005.

[13] J. Redi, S. Kolek, K. Manning, C. Partridge, R. Rosales-Hain, R. Ramanathan, and I. Castineyra. Javelen an ultra-low energy ad hoc wireless network. *Elsevier Ad Hoc Networks*, pages 108–126, 2008.

[14] D. Tse and P. Viswanath. *Fundamentals of Wireless Communication*. Cambridge University Press, 2005.

[15] S. Vasudevan, J. Kurose, and D. Towsley. On neighbor discovery in wireless networks with directional antennas. *IEEE Infocom*, 2005.

[16] S. Vasudevan, D. Towsley, D. Goeckel, and R. Khalili. Neighbor discovery in wireless networks and the coupon collectors problem. *Mobicom 2009*.

[17] R. Zheng, J. Hou, and L. Sha. Asynchronous wakeup for ad hoc networks. *ACM MobiHoc*, 2003.