Efficient Algorithms for Neighbor Discovery in Wireless Networks

UMass Computer Science Technical Report UM-CS-2010-066

Sudarshan Vasudevan, Micah Adler, Dennis Goeckel, *Fellow, IEEE,* and Don Towsley, *Fellow, IEEE*

***Abstract*—Neighbor discovery is an important first step in the initialization of a wireless ad hoc network. In this paper, we design and analyze several algorithms for neighbor discovery in wireless networks. Starting with a single-hop wireless network of n nodes, we propose a $\Theta(n \ln n)$ ALOHA-like neighbor discovery algorithm when nodes cannot detect collisions, and an order-optimal $\Theta(n)$ receiver feedback-based algorithm when nodes can detect collisions. Our algorithms neither require nodes to have *a priori* estimates of the number of neighbors nor synchronization between nodes. Our algorithms allow nodes to begin execution at different time instants and, to terminate neighbor discovery upon discovering all their neighbors. We finally show that receiver feedback can be used to achieve a $\Theta(n)$ running time, even when nodes cannot detect collisions.**

We then analyze neighbor discovery in a general multi-hop setting. We establish an upper bound of $O(\Delta \ln n)$ on the running time of the ALOHA-like algorithm, where Δ denotes the maximum node degree in the network and n the total number of nodes. We also establish a lower bound of $\Omega(\Delta + \ln n)$ on the running time of any randomized neighbor discovery algorithm. Our result thus implies that the ALOHA-like algorithm is at most a factor $\min(\Delta, \ln n)$ worse than optimal.

I. INTRODUCTION

Wireless ad hoc networks and sensor networks are typically deployed without any communication infrastructure and are required to "configure" themselves upon deployment. For instance, immediately upon deployment, a node has no knowledge of other nodes in its transmission range and needs to discover its neighbors in order to communicate with other network nodes. Neighbor discovery is an indispensable first step in the initialization of a wireless network, since knowledge of one-hop neighbors is essential for medium access control protocols [3], routing protocols [26], [13], and topology control algorithms [20] to work efficiently and correctly.

Neighbor discovery algorithms can be classified into two categories, viz. *randomized* or *deterministic*. In randomized neighbor discovery, each node transmits at randomly chosen times and discovers all its neighbors by a given time with high probability (w.h.p). In deterministic neighbor discovery, on the other hand, each node transmits according to a pre-determined transmission schedule that allows it to discover all its neighbors by a given time with probability one. In distributed settings, determinism often comes at the expense of increased running time (see for example [4], [9]) and, in the particular case of neighbor discovery, typically requires unrealistic assumptions such as node synchronization and *a priori* knowledge of the number of neighbors [16]. We, therefore, investigate randomized neighbor discovery algorithms in this paper.

Neighbor discovery is non-trivial for several reasons:

1) Neighbor discovery needs to cope with collisions. Ideally, a neighbor discovery algorithm needs to minimize the probability of collisions and therefore, the time to discover neighbors.
2) In many practical settings, nodes have no knowledge of the number of neighbors, which makes coping with collisions even harder.
3) When nodes do not have access to a global clock, they need to operate asynchronously and still be able to discover their neighbors efficiently.
4) In asynchronous systems, nodes can potentially start neighbor discovery at different times and consequently, may miss each other's transmissions.
5) Furthermore, when the number of neighbors is unknown, nodes do not know when or how to terminate the neighbor discovery process.

In this paper, we present neighbor discovery algorithms that comprehensively address each of these practical challenges under the standard collision channel model. Unlike existing approaches that assume *a priori* knowledge of the number of neighbors or clock synchronization among nodes, we propose neighbor discovery algorithms that:

- **P1** do not require nodes to have *a priori* knowledge of the number of neighbors,
- **P2** do not require synchronization among nodes,
- **P3** allow nodes to begin execution at different time instants, and
- **P4** enable each node to detect when to terminate the neighbor discovery process.

To the best of our knowledge, our work provides the first solution to the neighbor discovery problem that satisfies all of the properties **P1-P4**. Our approach is to start with a single-hop wireless network in which nodes are synchronized and know exactly how many neighbors they have. As we will see, the analysis in such a simplistic setting yields several valuable

S. Vasudevan is with Bell Labs Research, Alcatel-Lucent, Murray Hill, NJ, 07974 USA (Email: sudarshan.vasudevan@alcatel-lucent.com).

M. Adler is with FluentMobile Inc., Boston, MA 02119, USA (Email: micah@fluentmobile.com).

D. Goeckel and D. Towsley are with University of Massachusetts Amherst, MA 02139, USA (Email: goeckel@ecs.umass.edu; towsley@cs.umass.edu).

This research was sponsored by the National Science Foundation under CNS-0721861, and the U.S. Army Research Laboratory and the U.K. Ministry of Defence under Agreement Number W911NF-06-3-0001.

An initial version of the manuscript appeared in *ACM Mobicom*, Beijing, China, September 20-24, 2009.

insights about the neighbor discovery problem. These insights allow us to progressively relax each of the assumptions leading to a complete and practical solution to the neighbor discovery problem in a multi-hop network setting.

A. Main Results

Assuming a collision channel model of communication, we obtain the following important results in this paper:

1) We first study the ALOHA-like neighbor discovery algorithm proposed in [23] in a single-hop wireless network of n nodes. We show that its analysis reduces to that of the *Coupon Collector's Problem* and that each node discovers all its neighbors in $\Theta(n \ln n)$[*] time w.h.p [†].
2) When nodes can detect collisions, we propose an order-optimal neighbor discovery algorithm that employs feedback from receiving nodes and allows each node to discover all its neighbors in $\Theta(n)$ time w.h.p. Interestingly, we find that receiver feedback can be used even when nodes cannot detect collisions and propose a novel algorithm that achieves a $\Theta(n)$ running time.
3) We next show that absence of an estimate of the number of neighbors, n, results in a slowdown of no more than a factor of two, compared to when nodes know n.
4) We further show that lack of synchronization among nodes results in at most a factor of two slowdown in the algorithm performance from the case when nodes are synchronized.
5) We then describe how neighbor discovery can be accomplished even when nodes begin execution at different time instants. Furthermore, when nodes do not know n, we propose a provably correct termination condition that allows each node to terminate neighbor discovery after discovering all its neighbors w.h.p.
6) Finally, we extend our analysis to a general multi-hop wireless network setting. Here, we establish an upper bound of $O(\Delta \ln n)$ for the running time of the ALOHA-like algorithm, where Δ is the maximum node degree in the network and n denotes the total number of nodes. We also establish a lower bound of $\Omega(\Delta + \ln n)$ on the running time for any randomized neighbor discovery algorithm. Our result thus implies that the ALOHA-like algorithm is at most a factor $\min(\Delta, \ln n)$ worse than the optimal.

B. Organization of the paper

The rest of the paper is structured as follows. Section II, describes our model and its assumptions. Section III describes the ALOHA-like neighbor discovery algorithm in the case of a single-hop network. We next present feedback-based algorithms in Section IV. In Section V, we present lower and upper bounds for the neighbor discovery problem in a multi-hop network setting. Section VI discusses a number of pertinent issues relevant to neighbor discovery. Finally, we conclude in Section VIII.

[*]Throughout this paper, $\ln$ denotes natural logarithm and $\log$ denotes logarithm to base 2.

[†]We say than an event $\mathcal{E}$ occurs w.h.p if $\lim_{n \to \infty} P(\mathcal{E}) = 1$.

II. Network Model

Let $G = (V, E)$ represent a static multi-hop wireless network, where V denotes a set of n nodes and $E \subset V^2$ the set of undirected edges in G. We do not constrain how edges are determined between node pairs. However, a common example for the definition of an edge is that an edge exists between nodes i and j if they are within transmission range of each other. The transmission range might be defined as that distance below which the signal-to- noise ratio (SNR) exceeds a fixed threshold γ, allowing node i to transmit at a fixed rate to node j.

In addition, we make the following assumptions about the multi-hop wireless network:

- **Node IDs:** We assume that the nodes have *locally unique* identifiers i.e., no two neighbors of a given node have the same identifier. For example, the identifier could be the MAC address of a node or, its location.
- **Radio Model:** Each node is equipped with a radio transceiver that allows a node to either transmit or receive messages, but not both simultaneously.
- **Collision Model:** Throughout this paper, we assume that when two or more nodes, each of which has a common receiver, transmit concurrently, a collision occurs at the receiver. We further assume that a collision is the only source of packet loss i.e., we ignore packet losses due to effects such as shadowing and fading observed in wireless channels. The collision model, although idealized, will allow us to obtain a deep understanding of the neighbor discovery problem yielding valuable insights for designing practical neighbor discovery algorithms.
- **Symmetric Edges:** Edges between nodes are assumed to be symmetric i.e., if $(i, j) \in E$, then $(j, i) \in E$.

Problem Definition: n nodes are deployed over an area without prior knowledge about the graph G. We say that a node i *discovers* node j by time t if i receives at least one message from node j by time t. Our goal is to propose efficient algorithms that allow each node $i \in V$ to *discover* all nodes j such that $(i, j) \in E$.

III. ALOHA-like Neighbor Discovery Algorithm

In this section, we consider the ALOHA-like neighbor discovery algorithm first proposed in [23]. We first study this algorithm when all n nodes in the network are arranged in a clique, and n is known to each node in the clique. Finally, we consider a slotted, synchronous system where time is divided into slots and nodes are synchronized on slot boundaries. In other words, each transmission starts at the beginning of a slot and lasts the entire duration of the slot.

Each of these assumptions will be relaxed as we proceed. Importantly, these assumptions allow us to view the ALOHA-like neighbor discovery as an instance of the *Coupon Collector's Problem*. Consequently, the time to discover the n neighbors is the same as the minimum time to collect at least one of each of n coupon types.

A. Algorithm Description

The ALOHA-like algorithm is a randomized algorithm that operates as follows. In each slot, a node independently transmits a *DISCOVERY* message announcing its ID, with probability p_{x}, and listens with probability $1-p_{\text{x}}$. A discovery is made in a given slot only if exactly one node transmits in that slot.

It has been shown in [23], [31] that the optimal value of p_{x} that maximizes the rate of discovery of neighbors is $1/n$, where n denotes the clique size. However, the question of how long it takes to discover all the neighbors when nodes transmit with $p_{\text{x}} = 1/n$ was not addressed in [23], [31], which we proceed to analyze next.

B. Neighbor Discovery As Coupon Collector's Problem

We first map the neighbor discovery problem into the classical *Coupon Collector's Problem* [25]. First, the probability of a successful transmission by node i in a given slot is

$$p = p_{\text{x}}(1-p_{\text{x}})^{n-1} = \frac{1}{n}\left(1-\frac{1}{n}\right)^{n-1} \approx \frac{1}{ne} \qquad (1)$$

Note that p is the same for each node i, $1 \leq i \leq n$.

The process of neighbor discovery maps into a coupon collector's problem as follows. Consider a coupon collector C drawing coupons with replacement from an urn containing n distinct coupons, each coupon corresponding to a distinct node in the clique. In each slot, C draws one of the n coupons (i.e. discovers a given node) with probability p, and draws no coupon (i.e., detects an idle slot or a collision) with probability $1-np$. Thus, the event that C collects n distinct coupons corresponds to the event that each node in the clique has discovered all of its $n-1$ neighbors.

We are now ready to derive $E[W]$, where W is a random variable that denotes the time required for each node to discover its neighbors. The neighbor discovery process can be thought of as consisting of a sequence of *epochs*, each epoch consisting of one or more slots. Let W_m denote the length of epoch m, $0 \leq m \leq n-1$, that starts when the m-th node is discovered and ends when the $m+1$-st node is discovered. Thus, in the m-th epoch there are $n-m$ nodes yet to be discovered, each of which has a probability p of being discovered in a given slot. It is easy to see that the epoch length, W_m, is geometrically distributed with parameter $(n-m)p$. Thus, noting that $W = W_0 + \ldots + W_{n-1}$, we get

$$E[W] = \sum_{m=0}^{n-1} E[W_m] = \sum_{m=0}^{n-1} \frac{1}{(n-m)p} = \frac{1}{p}\sum_{m=1}^{n}\frac{1}{m} \approx neH_n$$

where H_n denotes the n-th Harmonic number, i.e., $H_n = \ln n + \Theta(1)$. Therefore,

$$E[W] = ne(\ln n + \Theta(1)) = ne\ln n + O(n) = \Theta(n \ln n) \quad (2)$$

In Appendix A, we show that the error introduced in the approximate calculation of $E[W]$ above vanishes as n grows large.

C. Sharp Concentration Around Mean

We next show that W is sharply concentrated around its mean. As described in [25], we make use of the Poisson approximation to the binomial distribution. In Appendix B, we derive the sharp concentration result without relying on Poisson approximation.

Let $N_i(t)$ be a random variable that denotes the number of successful transmissions by node i in the first t slots. It is easy to see that $N_i(t) \sim \text{Binomial}(t,p)$. Using the Poisson approximation (assuming large t and small p),

$$P(N_i(t) = k) = \frac{e^{-\lambda}\lambda^k}{k!}$$

where $\lambda = tp$. Let $\mathcal{E}_i(t)$ denote the event that node i is not discovered in t slots. Therefore,

$$P(\mathcal{E}_i(t)) = P(N_i(t) = 0) = e^{-tp}$$

Substituting $p = 1/ne$ into the above equation yields

$$P(\mathcal{E}_i(t)) = e^{-\frac{t}{ne}}$$

Therefore,

$$P(\neg\mathcal{E}_i(t)) = 1 - e^{-\frac{t}{ne}}$$

We are interested in the probability that all n nodes are discovered by time t, i.e. $P\left[\neg(\cup_{i=1}^{n}\mathcal{E}_i(t))\right]$,

$$P\left[\neg(\cup_{i=1}^{n}\mathcal{E}_i(t))\right] = P\left[\cap_{i=1}^{n}(\neg\mathcal{E}_i(t))\right] \qquad (3)$$

We next show that $\{\mathcal{E}_i(t)\}_{i=1}^{n}$ can be treated as an independent sequence of events.

Lemma 1: For $1 \leq i \leq n$, and any set of indices $\{j_1, \ldots j_k\}$ not containing i, $P\left[\mathcal{E}_i(t) | \cap_{\ell=1}^{k} \mathcal{E}_{j_\ell}(t)\right] \approx P\left(\mathcal{E}_i(t)\right)$.

Proof:

$$\begin{aligned} P\left[\mathcal{E}_i(t) | \cap_{\ell=1}^{k} \mathcal{E}_{j_\ell}(t)\right] &= \frac{P\left[\mathcal{E}_i(t) \cap (\cap_{\ell=1}^{k}\mathcal{E}_{j_\ell}(t))\right]}{P\left[(\cap_{\ell=1}^{k}\mathcal{E}_{j_\ell}(t)\right]} \\ &= \frac{(1-(k+1)p)^t}{(1-kp)^t} \end{aligned}$$

Using the approximation $1 + x \approx e^x$ in the above equation yields

$$P[\mathcal{E}_i(t) | \cap_{\ell=1}^{k} \mathcal{E}_{j_\ell}(t)] \approx \frac{e^{-t(k+1)p}}{e^{-tkp}} = e^{-\frac{t}{ne}} = P(\mathcal{E}_i(t))$$

■

From Lemma 1 and (3), it follows that

$$P[\neg(\cup_{i=1}^{n}\mathcal{E}_i(t))] = (1 - e^{-\frac{t}{ne}})^n \approx e^{-ne^{\frac{-t}{ne}}}$$

Therefore,

$$P(W > t) = 1 - P[\neg(\cup_{i=1}^{n}\mathcal{E}_i(t))]$$

Letting $t = ne(\ln n + c)$, for some $c \in \Re$, we conclude

$$P(W > t) = 1 - e^{-ne^{-(\ln n + c)}} = 1 - e^{-e^{-c}} \qquad (4)$$

Observe that $e^{-e^{-c}}$ is close to 1 for large positive c and is negligibly small for small negative c, thus implying a sharp concentration around the mean. Substituting $c = \ln n, \frac{-\ln n}{2}$ into (4), and a simple application of the union bound yields

$$\frac{ne\ln n}{2} \leq W \leq 2ne\ln n \quad \text{w.h.p} \qquad (5)$$

In other words, $W = \Theta(n \ln n)$ w.h.p.

D. Unknown Number of Neighbors

We first relax the assumption that requires each node in the clique to know n. The modified ALOHA-like algorithm proceeds in *phases*, each phase consisting of one or more slots. In the r-th phase, which lasts for $2^{r+1}e\ln 2^r$ slots, each node transmits with probability $1/2^r$.

The key idea here is that nodes geometrically reduce their transmission probabilities until they enter the phase of execution appropriate for the population size n. This occurs when nodes enter the $\lceil \log n \rceil$-th phase. During this phase, each node transmits with probability $1/n$ for a duration of $2ne\ln n$ slots. From (5), we know that all n nodes will be discovered by the $\lceil \log n \rceil$-th phase w.h.p.

The total number of slots, W, until all n nodes are discovered w.h.p is therefore

$$W = \sum_{r=1}^{\lceil \log n \rceil} 2^{r+1}e\ln 2^r \leq 4ne\ln n$$

Comparing the above result with (5), we see that the lack of knowledge of n results in no more than a factor of two slowdown.

E. Asynchronous Operation

We next relax the assumption that requires a time-slotted system in which nodes are synchronized on slot boundaries. The main result we show here is that the asynchronous ALOHA-like algorithm is no more than a factor of two slower than its synchronous counterpart.

The asynchronous ALOHA-like algorithm operates as follows. Between successive transmissions, each of which is of a fixed duration τ, each node remains in receive mode for an exponentially distributed time interval with mean $1/\lambda$.

The analysis in [31] is easily extended to the case of omni-directional antennas yielding the value of λ that maximizes the rate of discovery of neighbors, which is given thus

$$\lambda = \frac{1}{2\tau n}$$

1) Algorithm Analysis: For simplicity, we start with the case when each node knows the value of n. As before, we are interested in the time until each node discovers all its neighbors, denoted by W.

We assume that the transmit duration τ is small relative to $1/\lambda$. Thus, the inter-transmission times of a node are exponentially distributed and the total traffic from the n nodes constitutes a Poisson process with rate $n\lambda$. Now, a transmission from a node at time instant t is successful only if no other transmission starts during $[t-\tau, t+\tau]$. The probability of a successful transmission, p, is therefore

$$p = e^{-2n\tau\lambda} = 1/e$$

By dividing the neighbor discovery into epochs, where epoch m of duration W_m, starts with the discovery of m-th node and ends with the discovery of the $m+1$-st node, we obtain $W = \sum_m W_m$. In epoch m, there are $n-m$ nodes are yet to be discovered. The transmissions from these $n-m$ nodes constitute a Poisson process with rate $(n-m)\lambda$, each having probability p of being successful. In other words, W_m is exponential with mean $1/((n-m)\lambda p)$. Therefore, $E[W_m] = 2\tau ne/(n-m)$, and

$$E[W] = \sum_{m=0}^{n-1} E[W_m] = 2\tau neH_n = 2\tau ne(\ln n + \Theta(1))$$

Comparing the above result with (2), we see that the asynchronous algorithm is only a factor of two slower than its synchronous counterpart.

2) Sharp Concentration Around the Mean: As described in Section III-C, we next show that W is sharply concentrated around its mean. Let $N_i(t)$ denote the number of successful transmissions from node i by time t. Let $Q_i(t)$ denote the total number of transmissions from node i by time t. The conditional pmf $P(N_i(t) = k | Q_i(t) = t)$ is then given as

$$P(N_i(t) = k | Q_i(t) = \ell) = \binom{\ell}{k} p^k (1-p)^{\ell-k}$$

Now, $Q_i(t)$ is a Poisson random variable with rate λ. Removing the conditioning, we get

$$P(N_i(t) = k) = \sum_{\ell=0}^{\infty} \binom{\ell}{k} p^k (1-p)^{\ell-k} e^{-\lambda t}\frac{(\lambda t)^\ell}{\ell!}$$

Let $\mathcal{E}_i(t)$ denote the event $\{N_i(t) = 0\}$. Therefore,

$$P(\mathcal{E}_i(t)) = \sum_{\ell=0}^{\infty} (1-p)^\ell e^{-\lambda t}\frac{(\lambda t)^\ell}{\ell!} = e^{-\lambda tp} = e^{-\frac{t}{2\tau ne}}$$

Proceeding exactly as described in Section III-C, we get,

$$P(W > t) = 1 - P[\neg(\cup_{i=1}^{n} \mathcal{E}_i(t))]$$

Therefore,

$$P(W > 2\tau ne(\ln n + c)) = 1 - e^{-ne^{-(\ln n + c)}} = 1 - e^{-e^{-c}}$$

Substituting $c = \ln n, \frac{-\ln n}{2}$ into the equation above and a simple application of union bound yields

$$\tau ne\ln n \leq W \leq 4\tau ne\ln n \quad \text{w.h.p} \tag{6}$$

In other words, $W = \Theta(n\ln n)$ w.h.p.

3) Unknown Number of Neighbors: The asynchronous algorithm can also be extended to handle the case that nodes do not know n. Again, we divide the algorithm into phases, as before. During the r-th phase, which is of duration $2^{r+2}\tau e\ln 2^r$, each node remains in the receive mode for an exponential time interval with mean $1/\lambda_r = 2^{r+1}\tau$ between successive transmissions.

It is easy to see that the $\lceil \log n \rceil$-th phase is of duration $4\tau ne\ln n$ time units. In this phase, each node transmits with rate $\lambda_{\lceil \log n \rceil} = 1/2\tau n$. From (6), we know that by the end of this phase, each node discovers all its neighbors w.h.p. Assuming that the nodes are synchronized on phase boundaries (an assumption we relax in Section III-F), the total time W required to discover all neighbors w.h.p is given by

$$W = \sum_{r=1}^{\lceil \log n \rceil} 2^{r+2}\tau e\ln 2^r \leq 8\tau ne\ln n$$

Comparing the above result with (6), we again observe at most a factor of two slowdown from the case when n is known.

F. Initiating Neighbor Discovery

So far, we have assumed that all nodes start neighbor discovery at the same time. We also assumed that the nodes are synchronized on the phase boundaries, even in the case of the asynchronous algorithm. We relax both assumptions next.

Suppose the wireless network is deployed during time interval $[t, t+\eta]$, where η is an upper bound on the deployment period and assumed to be known in advance. When nodes have access to a global clock, initiating neighbor discovery is trivial, as each node can begin execution at a globally agreed upon time instant $\tilde{t} \geq t+\eta$.

When nodes do not have access to a global clock, initiating neighbor discovery is non-trivial, since clocks at different nodes may proceed at different rates resulting in clock offsets between nodes. We assume the maximum clock offset between any two nodes in the network is bounded by δ. In reality, clock offsets between nodes can potentially grow unboundedly as clocks tick at different rates. However, neighbor discovery occurs over short time scales and therefore, it is reasonable to assume a fixed δ for the duration of neighbor discovery. Each node starts neighbor discovery when its local clock reaches $\tilde{t}$, which is determined prior to deployment. To account for clock offsets, we extend each phase by δ time units to each phase i.e., the r-th phase lasts a duration of $2^{r+2}e\ln 2^r + \delta$ time units. Thus, all nodes are simultaneously in phase r for at least $2^{r+2}e\ln 2^r$ time units, guaranteeing that each node discovers all its neighbors w.h.p when $r = \lceil \log n \rceil$, as shown in Section III-E.

To get a sense of how large δ might be, we consider Mica2 motes equipped with a 32.768 kHz quartz crystal oscillator, which has a real-time clock accuracy of ±10 ppm [27]. This corresponds to an accuracy of ±864 milliseconds per day or a maximum clock offset of 1.7 seconds per day between any two nodes. Thus, if the deployment spans a period of 3 days, $\delta =$ 5.1s. With actively compensated oscillators [22] that provide an accuracy of ± 160 milliseconds/day, $\delta =$ 1s for the same deployment period.

G. Terminating Neighbor Discovery

When the value of n is not known to a node, it cannot know when it has discovered all its neighbors and terminate neighbor discovery. We conclude the discussion of the ALOHA-like algorithm in the clique setting by describing a *provably correct* termination condition that allows each node to determine if it has discovered all its neighbors.

Let $D_{i,r}$ be the number of nodes discovered by node i in the r-th phase. Then the termination condition used by node i is as follows:

TC Halt at the end of r-th phase if $D_{i,r-1} \geq 2^{r-2}$ and $D_{i,r} < 2^{r-1}$, where $r \geq 2$.

1) Proof of Correctness of Termination Condition: We next show that the termination condition **TC** is *correct* w.h.p. i.e. each node terminates the algorithm only after discovering all its neighbors w.h.p. Let ℓ be the largest integer such that $n = 2^\ell + k, 0 \leq k < 2^\ell$. More formally, we show that when $k = 0$, each node terminates at the end of $\ell+1$-st phase and further, $D_{i,\ell+1} = n-1,\ \forall i$ w.h.p. When $k > 0$, we show that each node terminates at the end of $\ell+2$-nd phase and $D_{i,\ell+2} = n-1,\ \forall i$ w.h.p.

For simplicity of exposition, we only consider the synchronous ALOHA-like algorithm. We note, however, that the termination condition **TC** is applicable for the asynchronous algorithm as well.

Let S_r denote the number of distinct nodes that transmit successfully at least once during the r-th phase. Note that

$$D_{i,r} \in [S_r - 1, S_r],\ \forall i$$

We divide the different phases of the algorithm execution into three distinct *stages*:

1) **Stage 1**: $1 \leq r \leq \log\left(\frac{n-1}{2\ln n}\right)$
2) **Stage 2**: $\log\left(\frac{n-1}{2\ln n}\right) < r \leq \log\left(\frac{n-1}{3\ln\ln n}\right)$
3) **Stage 3**: $\log\left(\frac{n-1}{3\ln\ln n}\right) + 1 < r \leq \log n$

Our proof of correctness is accordingly divided into the following three lemmas, each of which is proved in Appendix E.

Lemma 2: For r such that $1 \leq r \leq \log\left(\frac{n-1}{2\ln n}\right)$, $S_r = 0$ w.h.p.

Lemma 3: For r such that $\log\left(\frac{n-1}{2\ln n}\right) < r \leq \log\left(\frac{n-1}{3\ln\ln n}\right)$, $S_r < 2^{r-1}$ w.h.p.

Lemma 4: For r such that $\log\left(\frac{n-1}{3\ln\ln n}\right) < r \leq \log n$, $S_r \geq 2^{r-1}$ w.h.p.

Thus, Lemmas 2 and 3 together ensure that $D_{i,r} < 2^{r-1},\ \forall i$ throughout the first two stages. Hence, each node enters the $\tilde{r} = \log\left(\frac{n-1}{3\ln\ln n}\right) + 1$-st phase w.h.p. Further, since $D_{i,\tilde{r}-1} < 2^{r-2},\ \forall i$ w.h.p, it follows from the termination condition **TC** that no node terminates the algorithm in the $\tilde{r}$-th phase w.h.p. Hence, we can ignore the $\tilde{r}$-th phase and focus only on the remaining phases instead. Now, Lemma 4 ensures that $D_{i,r} \geq 2^{r-1},\ \forall i$ w.h.p throughout *Stage 3*. Thus, no node terminates the algorithm in phase $r \leq \log n$ w.h.p. Recalling that $n = 2^\ell + k$ and noting that $\ell \leq \log n$, we conclude that each node enters the $\ell+1$-st phase w.h.p.

The probability of a successful transmission by a given node during the $\ell+1$-st phase is

$$p_{\ell+1} = \frac{1}{2^{\ell+1}}\left(1 - \frac{1}{2^{\ell+1}}\right)^{2^\ell+k-1} \geq \frac{1}{2^{\ell+1}e}$$

Since the $\ell+1$-st phase lasts $2^{\ell+2}e\ln 2^{\ell+1}$ slots and since $2^{\ell+1} \geq n$, it follows from (4) that

$$P\left(S_{\ell+1} = n\right) \geq e^{-e^{-\ln 2^{\ell+1}}} \geq e^{-e^{-\ln n}} = e^{-1/n} \qquad (7)$$

Therefore, $D_{i,\ell+1} = n-1,\ \forall i$ w.h.p i.e., each node discovers all its neighbors in the $\ell+1$-st phase w.h.p.

Consider first the case that $n = 2^\ell$ i.e. $k = 0$. From Lemma (4), and (7), it follows that

$$\{D_{i,\ell} \geq 2^{\ell-1} \wedge D_{i,\ell+1} < 2^\ell \quad \forall i\} \text{ w.h.p}$$

From the termination condition **TC**, we conclude that each node terminates the algorithm at the end of $(\ell+1)$-st phase w.h.p. Further, we have already shown that $D_{i,\ell+1} = n-1,\ \forall i$ w.h.p, as desired.

Now, let us consider the case that $n = 2^\ell + k,$ where $k > 0$. Again, it follows from (4) that

$$P\left(S_{\ell+2} = n\right) \geq e^{-1/n} \qquad (8)$$

which implies that $D_{i,\ell+2} = n-1,\ \forall i$ w.h.p. Further, from (7) and (8), we conclude that

$$\{D_{i,\ell+1} \geq 2^{\ell} \wedge D_{i,\ell+2} < 2^{\ell+1}, \forall i\} \text{ w.h.p}$$

Again, the termination condition **TC** ensures that each node terminates the algorithm at the end of the $\ell+2$-nd phase w.h.p, as desired.

We evaluated the termination condition by simulating the ALOHA-like discovery algorithm for clique sizes, n, ranging from $2 \ldots 100$. For each n, we run the simulation 100 times. The simulation results show that for every n and every simulation run, each node terminates in the correct phase as predicted by our analysis.

IV. Order-Optimal Neighbor Discovery Using Feedback

We now describe a $\Theta(n)$ neighbor discovery algorithm that exploits feedback from receiving nodes. As we will see, feedback to a transmitting node allows it to determine if it has been discovered by other nodes, which then allows it to stop transmitting.

We first describe a $\Theta(n)$ neighbor discovery algorithm employing feedback when nodes can detect collisions, i.e., each node can distinguish between a collision and an idle slot. Subsequently, we describe how feedback can be exploited to achieve a $\Theta(n)$ algorithm even when nodes cannot detect collisions. Throughout this section, we assume that all the n nodes are arranged in a clique.

A. Collision Detection-based Neighbor Discovery

The collision detection-based algorithm is presented in Algorithm 1. The key idea behind the algorithm is as follows. We divide each slot into two sub-slots. Upon successful reception of a *DISCOVERY* message in the first sub-slot, each receiving node transmits bit "1" to the source of the message (say node i) in the second sub-slot, potentially causing a collision at node i. Collision detection allows node i to detect that the second sub-slot is not idle and that its transmission in the first sub-slot was received by all other nodes. Upon doing so, it "drops out" of neighbor discovery, i.e., stops transmitting. The remaining nodes (henceforth, called *surviving nodes*) then increase their transmission probabilities in the next slot. As we will see, allowing nodes which have been discovered to drop out and requiring the surviving nodes to increase their transmission probabilities, yields a significant improvement over the ALOHA-like algorithm.

Note that since a single bit suffices for the feedback, the second sub-slot is much shorter than the first.

1) Algorithm Analysis: Let W denote the time until each node discovers all of its $n-1$ neighbors. Again, we divide neighbor discovery into epochs, where the m-th epoch, $0 \leq m \leq n-1$, is of duration W_m. It starts when the m-th node is discovered and ends upon discovery of the $m+1$-st node. Thus,

$$W = \sum_{m=0}^{n-1} W_m = \sum_{m=0}^{n-2} W_m + 1$$

Algorithm 1 Collision Detection-Based ND(i,n)

```
$b \leftarrow 0$ //Number of neighbors discovered by node $i$
$flag \leftarrow 0$ //Has node $i$ been discovered by other nodes?
$NbrList \leftarrow [\ ]$ //List of neighbors of node $i$
loop
    $p_{\text{xmit}} \leftarrow 1/(n-b)$
    if (($flag = 0$) and (Bernoulli($p_{\text{xmit}}$) = 1)) then
        Transmit $DISCOVERY(i)$ in first sub-slot
        if second sub-slot not idle then
            $flag \leftarrow 1$ //"Drop out"
        end if
    else
        if successful reception in first sub-slot then
            Transmit bit "1" in second sub-slot
            $NbrList[b{+}{+}] \leftarrow DISCOVERY.source$
        end if
    end if
end loop
```

since there is only one surviving node in the $(n-1)$-st epoch, which therefore, consists of a single slot. In general, there are $n-m$ surviving nodes in epoch m, each transmitting with probability $1/(n-m)$. The probability of a successful transmission during epoch m is

$$p_m = \frac{1}{n-m}\left(1 - \frac{1}{n-m}\right)^{n-m-1}, \quad \forall m \leq n-2$$

It can be verified that

$$\frac{1}{(n-m)e} \leq p_m \leq \frac{2}{(n-m)e}, \quad \forall m \leq n-2 \tag{9}$$

Noting that W_m is geometrically distributed with mean $1/((n-m)p_m)$,

$$E[W] = \sum_{m=0}^{n-2} E[W_m] + 1 = \sum_{m=0}^{n-2} \frac{1}{(n-m)p_m} + 1 \tag{10}$$

Substitution from (9) into (10) and further simplification yields

$$\frac{ne}{2} \leq E[W] \leq ne$$

Thus, $E[W] = \Theta(n)$. In other words, the collision detection-based algorithm outperforms the ALOHA-like algorithm by a factor of at least $\ln n$.

2) Concentration Results: We now establish an even stronger result, viz. $W = \Theta(n)$ w.h.p. We first introduce the notion of *stochastic dominance*.

Definition 1: We say that a random variable X *stochastically dominates* another random variable Y if

$$P(X \geq x) \geq P(Y \geq x),\ \forall x$$

We introduce a random variable $\tilde{W}$ that stochastically dominates W and defined as follows:

$$\tilde{W} = \sum_{m=0}^{n-1} \tilde{W}_m$$

where $\tilde{W}_m$ is a geometrically distributed random variable with mean e. The stochastic dominance of $\tilde{W}$ over W follows by noting that each $\tilde{W}_m$, which is geometrically distributed with mean e, stochastically dominates W_m, which has mean $1/((n-m)p_m) \leq e$. Thus,

$$P(W > t) \leq P(\tilde{W} > t)$$

Further, note that $\tilde{W}$ is a Pascal random variable with parameters n and $1/e$. We now introduce a random variable $X_t \sim \text{Binomial}(t, 1/e)$, which is related to $\tilde{W}$ as follows:

$$\{\tilde{W} > t\} \iff \{X_t < n\}$$

The relationship between the random variables $\tilde{W}$ and X_t follows since $\tilde{W}$ represents the waiting time until n successful events, while X_t denotes the number of successful events in t slots. Therefore,

$$P(\tilde{W} > t) = P(X_t < n)$$

We now employ the following Chernoff bound [25, pp.70] for the binomial random variable X_t:

$$P\left(X_t < (1-\varepsilon)\frac{t}{e}\right) < e^{-\frac{t\varepsilon^2}{2e}}, \quad 0 < \varepsilon \leq 1$$

Letting $t = 2ne$ and $\varepsilon = 1/2$, we get

$$P(\tilde{W} > 2ne) = P(X_t < n) < e^{-n/4} \to 0 \text{ as } n \to \infty$$

Since $P(W > t) \leq P(\tilde{W} > t)\ \forall t$, it follows that $W \leq 2ne$ w.h.p.

We next show that $W \geq ne/2$ w.h.p. We use the following Chernoff bound for X_t [25]:

$$P\left(X_t \geq \frac{(1+\varepsilon)t}{e}\right) \leq \left(\frac{e^{\varepsilon}}{(1+\varepsilon)^{1+\varepsilon}}\right)^{\frac{2t}{e}}, \quad \forall \varepsilon > 0$$

Letting $t = ne/2$ and $\varepsilon = 1$, we conclude that

$$P(\tilde{W} < ne/2) = P(X_t \geq n) \leq (e/4)^n \to 0 \text{ as } n \to \infty$$

Thus, $W \geq ne/2$ w.h.p. Using the union bound, it follows that

$$\frac{ne}{2} \leq W \leq 2ne, \quad \text{w.h.p} \tag{11}$$

In other words, $W = \Theta(n)$ w.h.p.

3) Unknown Number of Neighbors: When nodes do not know n, we again divide the algorithm into phases. The r-th phase lasts a duration of $2^{r+1}e$ slots. Each surviving node transmits with probability $1/(2^r - b)$ during this phase, where b denotes the number of nodes which have been discovered by their neighbors thus far.

In the $\lceil \log n \rceil$-th phase, there are $n-b$ surviving nodes, each transmitting with probability $1/(n-b)$. An analysis identical to the one in Section IV-A2 shows that each node discovers all of its neighbors by the $\lceil \log n \rceil$-th phase w.h.p. Hence, the total time W until each node discovers all its neighbors w.h.p is

$$W = \sum_{r=1}^{\lceil \log n \rceil} 2^{r+1}e \leq 4ne \tag{12}$$

Comparing the above result with (11), we again observe no more than a factor of two slowdown compared to the case where nodes know n.

4) Asynchronous Operation: We next describe the asynchronous collision detection-based algorithm, which is presented in Algorithm 2. Here, each transmission is of fixed duration τ and is followed by a *feedback period* of duration σ. Let $\kappa = \tau + \sigma$. We further assume that a node in the receive mode can always detect if the wireless channel is *busy* or *idle*.

As shown in Figure 1, the timeline for an asynchronous collision detection-based algorithm consists of (i) *Unsuccessful Busy Periods*, during which two or more nodes transmit concurrently; (ii) *Feedback Periods* immediately following message transmissions; (iii) *Idle Periods* during which no transmissions occur; and (iv) *Successful Busy Periods* during which exactly one transmission occurs. As in the synchronous

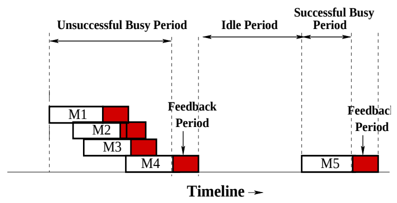

Fig. 1. Timeline of asynchronous collision detection-based algorithm

algorithm, the key idea is to allow each node that has been discovered by its neighbors to drop out and let the surviving nodes increase their transmission rate.

Algorithm 2 Asynch. Collision Detection-based ND(i,n)

```
$b \leftarrow 0$ //Number of neighbors discovered by node $i$
$flag \leftarrow 0$ //Has node $i$ been discovered by other nodes?
$NbrList \leftarrow [\ ]$ //List of neighbors of node $i$
loop
    $\lambda \leftarrow 1/(2\kappa(n-b))$
    Alarm($TIMEOUT$,Exp($1/\lambda$)) //Set Listen duration
    try:
        loop
            Listen for DISCOVERY messages
            if collision detected then
                Transmit bit "1" at the end of busy period
            else
                $NbrList[b{+}{+}] \leftarrow DISCOVERY.source$
            end if
        end loop
    end try
    catch TIMEOUT:
        if ($flag = 0$) then
            Transmit $DISCOVERY(i)$
            if feedback period idle then
                $flag \leftarrow 1$ //"Drop out"
            end if
        end if
    end catch
end loop
```

From Algorithm 2, we make two observations about the collision detection-based algorithm:

1) Unlike the synchronous version, receiving nodes transmit feedback in response to an unsuccessful transmis-

sion. The reason for this choice is as follows. As shown in Figure 1, during an unsuccessful busy period, a message transmission may overlap with the feedback period of another message. Hence, sensing energy during the feedback period has to be taken as an indication of unsuccessful transmission.

2) Our algorithm allows a node to begin transmission, despite detecting a busy period. The algorithm performance can be improved by suppressing such transmissions. However, despite allowing such an event to occur, our analysis shows that the algorithm takes only twice as long to complete neighbor discovery than its synchronous counterpart. Thus, transmission suppression only improves this constant and not the asymptotic order.

We are now ready to analyze the performance of the algorithm. Let W denote the time to discover all n nodes. As before, divide the discovery process into epochs, where the m-th epoch is of duration W_m. In epoch m, there are $n-m$ nodes yet to be discovered, each transmitting with rate $\lambda_m = 1/(2\kappa(n-m))$. Since the transmission events of an individual node constitute a Poisson process with rate λ_m, the transmission events from the $n-m$ surviving nodes also follow a Poisson process with rate $(n-m)\lambda_m$.

Now, the probability that a transmission by one of the $n-m$ surviving nodes is successful is given by

$$p_m = e^{-2(n-m)\kappa\lambda_m} = 1/e$$

Now, the successful transmission events from the $n-m$ yet to be discovered nodes follow a Poisson process with a rate given by $(n-m)\lambda_m p_m = 1/(2\kappa e)$. In other words, the random variables W_m are iid and exponentially distributed with mean $2\kappa e$.

Noting that $W = \sum_{m=0}^{n-1} W_m$, it follows that the random variable W is the sum of n iid exponential random variables. Therefore, W is an n-stage Erlang random variable with mean:

$$E[W] = \sum_{m=0}^{n-1} E[W_m] = 2\kappa n e$$

We immediately conclude that the the asynchronous version of the collision detection-based neighbor discovery is only twice as slow as its synchronous version.

We also show that the random variable W is sharply concentrated around its mean. In particular, employing Chernoff bounds for Erlang random variables (see Appendix C), it immediately follows that

$$\kappa n e \leq W \leq 4\kappa n e \text{ w.h.p} \tag{13}$$

Therefore, $W = \Theta(n)$ w.h.p.

Also, the case where nodes do not know the value of n can be handled exactly as described in Section IV-A3 and it can be shown that, that the time to discover all neighbors w.h.p is given by

$$W \leq 8\kappa n e$$

Again, comapring the above result with (13), we observe no more than a factor of two slowdown from the case when nodes know the value of n.

5) Initiating and Terminating Neighbor Discovery: Initiation of the collision detection-based algorithm can be handled identically as described in Section III-F. Unlike the ALOHA-like algorithm, termination of neighbor discovery is trivial in the case of collision detection-based algorithm. If there are any surviving nodes at the end of a phase, they proceed to the next phase. On the other hand, a node which has been discovered by its neighbors, waits an additional phase and terminates the algorithm, if it senses no energy during the entire duration of the phase.

B. Order-Optimal Neighbor Discovery Without Collision Detection

We now show that we can achieve order-optimal neighbor discovery, even when nodes cannot detect collisions. Algorithm 3 presents a neighbor discovery algorithm that achieves this goal.

The algorithm is divided into *rounds* (as indexed by variable r). Round r lasts a duration of T_r slots where

$$T_r = \frac{8n}{2^r} + 8\log 2n$$

Instead of providing a single bit of feedback, each node now includes in its *DISCOVERY* messages the ID of the most recently discovered neighbor in addition to its own ID. If a receiving node hears its ID during a round, it “drops out” of neighbor discovery at the end of the round.

The key idea behind Algorithm 3 is as follows. At the end of r rounds, where $1 \leq r < \log\log n$, there are at most $n/2^r$ surviving nodes w.h.p and the remaining nodes, which hear their IDs back, drop out. Therefore, in the $r+1$-st round, each surviving node increases its transmission probability to $2^r/n$. After $\log\log n - 1$ rounds, there are at most $n/\log n$ surviving nodes w.h.p. At this point, each surviving node simply runs the ALOHA-like neighbor discovery, which from the analysis in Section III, requires no more than

$$\frac{2ne}{\log n} \ln\left(\frac{n}{\log n}\right) = \Theta(n) \text{ slots w.h.p.}$$

Thus, assuming (for the time being) that there are at most $n/\log n$ surviving nodes after $\log\log n - 1$ rounds w.h.p, the total time W until each node discovers all its $n-1$ neighbors is given by

$$W = \sum_{r=1}^{\log\log n - 1} \left(\frac{8n}{2^r} + 8\log 2n\right) + \Theta(n) = \Theta(n) \text{ w.h.p}$$

Our challenge next is to show that, indeed, there are at most $n/\log n$ surviving nodes after $\log\log n - 1$ rounds w.h.p.

1) Algorithm Analysis: In this section, we show that Algorithm 3 runs in $\Theta(n)$ time w.h.p. To this end, we establish the following lemma.

Lemma 5: After $\log\log n - 1$ rounds, there are at most $n/\log n$ surviving nodes w.h.p.

Proof: Let $\mathcal{E}_r$ denote the event that there are at most $n/2^{r-1}$ surviving nodes at the start of the r-th round. It is easy to see that $P(\mathcal{E}_1) = 1$. We first show that

$$P(\neg\mathcal{E}_{r+1}|\mathcal{E}_r) \leq \frac{2}{n},\ \forall r : 1 \leq r < \log\log n$$

Algorithm 3 ND Without Collision Detection(i,n)

```
$b \leftarrow 0$ //Number of neighbors discovered by $i$
$flag \leftarrow 0$ //Has $i$ been discovered by other nodes?
$id \leftarrow -1$ //id of most recently discovered node
$NbrList \leftarrow [\,]$ //List of neighbors of node $i$
for $r = 1$ to $\lceil \log\log n \rceil - 1$ do
  $p_{\text{xmit}} \leftarrow 2^{r-1}/n$
  for $t = 1$ to $\lceil \frac{8n}{2^r} + 8\log 2n \rceil$ do
    if ($flag = 0$) and (Bernoulli($p_{\text{xmit}}$) = 1) then
      Transmit DISCOVERY(i,id)
    else
      if successful reception then
        if ($DISCOVERY.id = i$) then
          $flag \leftarrow 1$ //"Drop out"
        else
          $id \leftarrow DISCOVERY.source$
          $NbrList[b{+}{+}] \leftarrow id$
        end if
      end if
    end if
  end for
end for
// Switch to ALOHA-like neighbor discovery
loop
  $p_{\text{xmit}} \leftarrow \log n/n$
  if ($flag = 0$) and (Bernoulli($p_{\text{xmit}}$) = 1) then
    Transmit DISCOVERY(i)
  else
    if successful reception then
      $NbrList[b{+}{+}] \leftarrow DISCOVERY.source$
    end if
  end if
end loop
```

Let the number of surviving nodes at the start of r-th round be $n/2^{r-1}-k$, for some $k \geq 0$, each transmitting with probability $2^{r-1}/n$. Conditioned on the event $\mathcal{E}_r$ being true, let W_r denote the waiting time until an additional $n/2^r - k$ nodes hear their IDs back when each node transmits with probability $2^{r-1}/n$.

Again, we divide neighbor discovery in the r-th round into epochs, where the m-th epoch, of duration $W_{m,r}$, starts when the m-th node hears its ID back and ends when the $m+1$-st node hears its ID back. Thus,

$$W_r = \sum_{m=0}^{n/2^r-k-1} W_{m,r} = \sum_{m=k}^{n/2^r-1} W_{m,r}$$

Here, $W_{m,r}$ is geometrically distributed with mean $(n/2^{r-1} - m)p_{m,r}$, with

$$\begin{aligned} p_{m,r} &= \frac{2^{r-1}(n/2^{r-1}-m)}{n}\left(1-\frac{2^{r-1}}{n}\right)^{n/2^{r-1}-k-1} \\ &\geq \frac{2^{r-1}(n/2^{r-1}-m)}{n}\left(1-\frac{2^{r-1}}{n}\right)^{n/2^{r-1}-1} \triangleq \tilde{p}_{m,r} \end{aligned}$$

Let $\tilde{W}_{m,r}$ be geometrically distributed with mean $(n/2^{r-1} - m)\tilde{p}_{m,r}$. Clearly, $\tilde{W}_{m,r}$ stochastically dominates $W_{m,r}$ and hence,

$$W_r \leq \sum_{m=k}^{n/2^r} \tilde{W}_{m,r} \leq \sum_{m=0}^{n/2^r} \tilde{W}_{m,r}$$

Now, it can be verified that

$$\frac{2^{r-1}(n/2^{r-1}-m)}{ne} \leq \tilde{p}_{m,r} \leq \frac{2^{r}(n/2^{r-1}-m)}{ne} \tag{14}$$

Using Chernoff bounds, we obtain

$$P\left(W_r > x\right) \leq e^{-sx} \prod_{m=0}^{n/2^r} \frac{\tilde{p}_{m,r}e^s}{1-(1-\tilde{p}_{m,r})e^s}, \quad \forall s > 0$$

Substituting the upper and lower bounds for $\tilde{p}_{m,r}$ from (14) into the numerator and denominator (resp.) above yields

$$\begin{aligned} P\left(W_r > x\right) &\leq 2e^{-sx} \prod_{m=0}^{n/2^r} \frac{(\frac{n}{2^{r-1}}-m)e^s}{(\frac{n}{2^{r-1}}-m)e^s + (1-e^s)\frac{ne}{2^{r-1}}} \\ &\leq 2e^{-sx} \prod_{m=0}^{n/2^r} \frac{1}{1-2e\left(1-e^{-s}\right)} \end{aligned}$$

Letting $s = -\ln\left(1-\frac{e-1}{2e^2}\right)$ and noting that $s > 1/8$,

$$P\left(W_r > x\right) \leq 2e^{\frac{n}{2^r}+1-\frac{x}{8}}$$

Noting that the r-th round is of duration $T_r = \frac{8n}{2^r} + 8\log 2n$ slots, we conclude that

$$P\left(\neg\mathcal{E}_{r+1}|\mathcal{E}_r\right) \leq P\left(W_r > T_r\right) \leq \frac{2}{n} \tag{15}$$

Since $P\left(\mathcal{E}_1\right) = 1$, it follows from (15) that

$$P\left(\neg\mathcal{E}_{r+1}\right) \leq P\left(\neg\mathcal{E}_{r+1}|\mathcal{E}_r\right) + P\left(\neg\mathcal{E}_r\right) \leq \frac{2r}{n}$$

Since $r < \log\log n$, it follows that

$$P\left(\neg\mathcal{E}_{\log\log n}\right) \leq \frac{2\log\log n}{n} \rightarrow 0 \text{ as } n \rightarrow \infty$$

This completes the proof of Lemma 5. ■

We already know from the analysis in Section III-C that the surviving nodes (at most $n/\log n$ in total) can be discovered by the ALOHA-like algorithm in $\Theta(n)$ slots w.h.p. Putting everything together, we conclude that all n nodes are discovered in $\Theta(n)$ running time w.h.p.

2) Algorithm Extensions: Analogous to Section III, Algorithm 3 can be extended to handle asynchronous operation and the case where n is unknown. We also employ the termination condition **TC** proposed in Section III-G and evaluate it by simulating Algorithm 3. Again, we find that for each $n \in [2, 100]$, each node terminates in the correct phase, as desired.

V. The Multi-Hop Network Case

Thus far, our treatment of the neighbor discovery problem assumes single-hop networks. We now consider the more general multi-hop network setting and derive lower and upper bounds for the neighbor discovery problem.

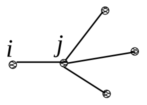

Fig. 2. Node i and node j with different number of neighbors.

A. ALOHA-like Neighbor Discovery: Upper Bound Analysis

We begin by analyzing the ALOHA-like algorithm. Recall that the wireless network is represented by a graph $G = (V, E)$, where $|V| = n$. We initially assume that all nodes know the maximum node degree Δ. Further, we assume a time-slotted system in which nodes are synchronized on slot boundaries. In each slot, each node in the network transmits with probability $1/(\Delta + 1)$.

The probability p that a node i discovers a neighboring node j in a given slot is

$$p \geq \frac{1}{\Delta+1}\left(1-\frac{1}{\Delta+1}\right)^{\Delta} \geq \frac{1}{(\Delta+1)e}$$

Let $\mathcal{E}_{ij}(t)$ denote the event that node i discovers a given neighbor j in t slots. Therefore,

$$P\left(\neg\mathcal{E}_{ij}(t)\right) \leq \left(1-\frac{1}{(\Delta+1)e}\right)^{t} \leq \quad e^{-\frac{t}{(\Delta+1)e}}$$

where the second inequality follows from the fact that $1+x \leq e^x,\ \forall x \in \Re$. Substituting $t = 3(\Delta+1)e\ln n$ into the right hand side of the above inequality yields

$$P\left(\neg\mathcal{E}_{ij}(t)\right) \leq e^{-3\ln n} = \frac{1}{n^3}$$

Applying the union bound, we conclude that

$$P\left(\bigcup_{(i,j)\in E}\neg\mathcal{E}_{ij}(t)\right) \leq \sum_{(i,j)\in E} P\left(\neg\mathcal{E}_{ij}(t)\right) \leq \frac{2n^2}{n^3} = \frac{2}{n}$$

In other words, all the edges in the network are discovered in at most $O(\Delta \ln n)$ time slots w.h.p.

Analogous to the single-hop version, it can be easily verified that the network version of the ALOHA-like algorithm can be extended to: (i) handle the case when nodes do not know Δ, (ii) allow asynchronous operation, and (iii) allow nodes to start neighbor discovery at different times. We therefore focus only on handling termination.

1) Handling Termination: The termination condition **TC** proposed in Section III-G may not work correctly in a multi-hop setting. To see why, consider the network shown in Figure 2. Here, it is possible that node i discovers node j before j discovers i. Since i has only one neighbor, it may terminate neighbor discovery before being discovered by j. We therefore propose the following change to the ALOHA-like algorithm, which doubles its running time.

We double the duration of each phase i.e. the r-th phase now lasts a duration of $2^{r+2}e\ln 2^r$ slots. During the first half of the r-th phase, each node transmits with probability $1/2^r$, as before. In the second half, each node that transmitted in the first half announces to its neighbors if it is ready to terminate the algorithm, as determined by the termination condition **TC**. At the end of the r-th phase, a node terminates neighbor discovery only if each of its neighbors is ready to terminate the algorithm.

Returning to the example in Figure 2, let us assume that node i has discovered node j in the first half of the r-th phase and is therefore, ready to terminate neighbor discovery at the end of the r-th phase, but node j is not. Node i learns of this fact from j's transmission in the second half of the r-th phase, and thus, both nodes proceed to the next phase.

Our simulation of the ALOHA-like discovery algorithm in a network setting over a wide range of node densities and over a range of node placements shows that each node terminates only after discovering all its neighbors, as desired.

B. Lower Bound Analysis

In this section, we establish a lower bound for the running time of any randomized neighbor discovery algorithm. More formally, we show that, given n and Δ, we can construct a graph $G = (V, E)$ with $|V| = n$ and maximum node degree Δ such that any randomized algorithm requires $\Omega(\Delta + \ln n)$ slots w.h.p to discover all edges in G.

Given n and Δ, we construct an input graph $G = (V, E)$ with $|V| = n$ and maximum node degree Δ as follows. Δ out of the n nodes are chosen arbitrarily and arranged in a clique, while the remaining $n - \Delta$ nodes are paired arbitrarily and arranged in a matching of size $(n - \Delta)/2$. No edges exist between the nodes in the clique and those in the matching. Each node in G is then assigned a unique ID drawn uniformly at random from the set of available IDs. For convenience, we let $M_G = (V', E')$ denote the matching in G. Note that $V', E' \subseteq V, E$, where $|V'| = n - \Delta$ and $|E'| = (n - \Delta)/2$.

Since each node needs to receive at least one transmission from a neighboring node in order to discover the neighbor, any distributed algorithm, randomized or not, has a running time of at least $\Omega(\Delta)$ on the input graph G. Thus, it suffices to show that when $\Delta = o(\ln n)$[‡], any randomized neighbor discovery algorithm has a running time of $\Omega(\ln n)$ w.h.p on the graph G. This implies a lower bound of $\Omega(\Delta + \ln n)$.

In establishing the lower bound, we assume that nodes in G can detect collisions. Since collision detection can only help reduce the discovery time, the lower bound also applies to algorithms that assume nodes cannot detect collisions.

1) Uniform Randomized Algorithms: We initially establish the lower bound for the class of *uniform randomized algorithms* i.e. each node uses the same algorithm. Consistent with the problem definition introduced in Section II, we assume that nodes do not know the IDs of their neighbors.

A more precise definition of a randomized and uniform randomized algorithm follows. Consider node $i \in V$. We introduce $h_i(t) \in (\{0, 1, c_1, c_2, w\})^t$ as the *history* observed by node i up to time t, where $t \geq 1$. Here, $w \in \{0, 1\}^b$ denotes an arbitrary bit-string of length $b > 1$ and corresponds to the contents of the message received by node i upon successful transmission by a neighbor. In the case of the ALOHA-like algorithm, w denotes the ID of the transmitting node, while in the case of the Feedback-based algorithm described in

[‡] $f(n) = o(g(n))$, if $\lim_{n\to\infty}\frac{f(n)}{g(n)} = 0$

Section IV-B, w denotes the ID of the transmitting node and that of its most recently discovered neighbor. The ℓ-th entry of $h_i(t)$, denoted by $h_i(t,\ell)$, corresponds to an event in the ℓ-th slot and is defined as follows:

$$h_i(t,\ell) \triangleq \begin{cases} 0 & \text{if } i \text{ does not transmit and does not observe} \\ & \text{any transmission,} \\ 1 & \text{if } i \text{ transmits successfully,} \\ c_1 & \text{if } i \text{ transmits and is involved in a collision,} \\ c_2 & \text{if } i \text{ does not transmit but observes} \\ & \text{a collision,} \\ w & \text{if } i \text{ observes a successful transmission} \end{cases}$$

A randomized algorithm is characterized by a per node function $p_i(t|h_i(t-1))$, that specifies the probability that node i transmits in slot t given a history $h_i(t-1)$ as defined above. A uniform randomized algorithm is one where $p_i(t|h_i(t-1)) = p_j(t|h_j(t-1))$ for all $i,j \in V$ and all t, provided that $h_i(t-1) = h_j(t-1)$.

Consider an arbitrary uniform randomized neighbor discovery algorithm $\mathcal{A}$ and consider the input graph $G=(V,E)$ constructed earlier. Let $\mathcal{T}_{\mathcal{A}}^{G}$ be the time until all edges in the G using $\mathcal{A}$. Recall that the input graph G includes a matching $M_G = (V', E')$, where $V', E' \subseteq V, E$ and $|E'| = (n-\Delta)/2$. Let $\mathcal{T}_{\mathcal{A}}^{M_G}$ denote the time required by $\mathcal{A}$ to discover the edges in M_G. Clearly, $\mathcal{T}_{\mathcal{A}}^{G} \geq \mathcal{T}_{\mathcal{A}}^{M_G}$ and hence, a lower bound for $\mathcal{T}_{\mathcal{A}}^{M_G}$ yields a lower bound for $\mathcal{T}_{\mathcal{A}}^{G}$.

We assume that each node in M_G knows n and Δ. Clearly, having more information can only reduce the running time of $\mathcal{A}$ and therefore, does not affect the lower bound. We also assume that edges $(i,j),(j,i) \in E'$ are both discovered simultaneously regardless of which node i or j successfully transmits first. Again, this assumption does not affect the lower bound.

Let the random variable $\mathcal{T}_{ij}$ denote the time until edges (i,j) and (j,i) are discovered. Then, $\mathcal{T}_{\mathcal{A}}^{M_G} \geq \max\limits_{i,j} \mathcal{T}_{ij}$.

Consider an arbitrary pair of directed edges (i,j) and (j,i) in M_G. It is important to note the following about the histories observed by nodes i and j prior to $\mathcal{T}_{ij}$, namely that they are identical. This is because the only events that can occur prior to $\mathcal{T}_{ij}$ are either idle slots or collisions involving both i and j. Consequently, $h_i(t) = h_j(t) = h(t)$, $\forall t < \mathcal{T}_{ij}$, and therefore,

$$p_i(t|h(t-1)) = p_j(t|h(t-1)) \triangleq p(t|h(t-1)), \ \forall h, \forall t < \mathcal{T}_{ij}$$

Let $\mathcal{B}_{ij}(t)$ denote the event that a successful transmission occurs, either from i to j or vice-versa, in the t-th time slot. Then, for all $h(t-1)$ and for all $t < \mathcal{T}_{ij}$, it follows that

$$P\left(\mathcal{B}_{ij}(t)|h(t-1)\right) = 2p(t|h(t-1))(1-p(t|h(t-1))) \leq \frac{1}{2}$$

Since $h(t-1)$ contains information about the outcome of slots $1,\ldots,t-1$, it follows that

$$P\left(\mathcal{B}_{ij}(t)|\neg\mathcal{B}_{ij}(1),\ldots,\neg\mathcal{B}_{ij}(t-1)\right) \leq \frac{1}{2}, \quad \forall t < \mathcal{T}_{ij}$$

Therefore,

$$P(\mathcal{T}_{ij} > t) = \prod_{\ell=1}^{t} P(\neg\mathcal{B}_{ij}(\ell)|\neg\mathcal{B}_{ij}(1)\ldots\neg\mathcal{B}_{ij}(\ell-1)) \geq 2^{-t}$$

We know that M_G has $(n-\Delta)/2$ edges. Further, the random variable $\mathcal{T}_{ij}$ depends only on the probabilistic choices of nodes i and j in each slot, which in turn depend only on the histories observed by i and j, but not on the probabilistic choices or histories of any other node. Thus, the $\mathcal{T}_{ij}$s are iid yielding

$$P\left(\max_{i,j} \mathcal{T}_{ij} > t\right) \geq 1 - \left(1-2^{-t}\right)^{\frac{n-\Delta}{2}} \geq 1 - e^{-\frac{2^{-t}(n-\Delta)}{2}}$$

Setting $t = \frac{1}{2}\log\left(\frac{n-\Delta}{2}\right)$ in the above inequality yields

$$P\left(\max_{i,j} \mathcal{T}_{ij} > \frac{1}{2}\log\left(\frac{n-\Delta}{2}\right)\right) \geq 1 - e^{-\sqrt{\frac{n-\Delta}{2}}} \to 1$$

as $n \to \infty$. Thus, $\mathcal{T}_{\mathcal{A}}^{M_G} \geq \max\limits_{i,j} \mathcal{T}_{ij} = \Omega\left(\ln\left(\frac{n-\Delta}{2}\right)\right)$ w.h.p, which yields a lower bound of $\Omega(\ln n)$ when $\Delta = o(\ln n)$. Thus, $\mathcal{T}_{\mathcal{A}}^{G} = \Omega(\Delta + \ln n)$ w.h.p.

2) Non-uniform Randomized Algorithms: We next show that the lower bound established in the previous section also applies for the class of *non-uniform randomized algorithms*, where nodes may use different algorithms. More formally, a non-uniform randomized algorithm is one in which $p_i(t|h_i(t-1))$ may not equal $p_j(t|h_i(t-1))$, even though $h_i(t-1) = h_j(t-1)$, for an arbitrary pair of nodes $i,j \in V$.

Let $\overline{\mathcal{A}} = (\mathcal{A}_1,\ldots,\mathcal{A}_n)$ be an arbitrary non-uniform randomized algorithm, where node i uses algorithm $\mathcal{A}_i$. We also assume that the node placement is independent of their IDs, i.e., given locations $\mathcal{L}_1,\ldots,\mathcal{L}_n$ of the nodes, we assume that the node at location $\mathcal{L}_k$ is equally likely to be any one of the n nodes and is, therefore, equally likely to use any one of $\mathcal{A}_1,\ldots,\mathcal{A}_n$.

Similar to [18], we reduce a non-uniform randomized algorithm $\overline{\mathcal{A}}$ to a randomized algorithm $\mathcal{Q}$, which operates as follows. Each node i chooses an algorithm $\mathcal{A}_k$ from $\overline{\mathcal{A}}$ uniformly at random and independently of any other node, and *simulates* $\mathcal{A}_k$, i.e., node i runs $\mathcal{A}_k$ as if it were node k. We next show that $\mathcal{Q}$ is uniform.

Lemma 6: The randomized algorithm $\mathcal{Q}$ is uniform.

Proof: Let i and j be an arbitrary pair of nodes such that $h_i(t-1) = h_j(t-1) = h(t-1)$. To establish that $\mathcal{Q}$ is uniform, we need to show that $p_i(t|h(t-1)) = p_j(t|h(t-1))$.

Let $p_i(h(t-1)|\mathcal{A}_k)$ denote the probability that node i observes history $h(t-1)$ given that it simulates $\mathcal{A}_k$ and $p_i(\mathcal{A}_k)$ be the probability that node i simulates $\mathcal{A}_k$. Then,

$$p_i(t|h(t-1)) = \frac{\sum\limits_{\mathcal{A}_k \in \overline{\mathcal{A}}} p_i(t|h(t-1),\mathcal{A}_k)p_i(h(t-1)|\mathcal{A}_k)p_i(\mathcal{A}_k)}{\sum\limits_{\mathcal{A}_k \in \overline{\mathcal{A}}} p_i(h(t-1)|\mathcal{A}_k))p_i(\mathcal{A}_k)}$$

Since nodes i and j both run $\mathcal{A}_k$ as node k, $p_i(t|h(t-1),\mathcal{A}_k) = p_j(t|h(t-1),\mathcal{A}_k)$. Further, since nodes i and j are both equally likely to be in any of the n locations, it follows that $p_i(h(t-1)|\mathcal{A}_k) = p_j(h(t-1)|\mathcal{A}_k)$. Finally, since $p_i(\mathcal{A}_k) = p_j(\mathcal{A}_k)$, we conclude that $p_i(t|h(t-1)) = p_j(t|h(t-1)$. The uniformness of $\mathcal{Q}$ follows immediately. ■

Let $C_{\mathcal{Q}}(\mathcal{L}_k)$ denote the algorithm chosen by the node at location $\mathcal{L}_k$ under $\mathcal{Q}$. Also, let $\overline{s} = (s_1,\ldots,s_n)$ denote an arbitrary vector of *schedules*, where each schedule s_k denotes the slots in which the node at location $\mathcal{L}_k$ transmits.

We now state a result established in [18], which also follows immediately from the preceding discussion.

Lemma 7: If $\forall k \neq \ell : C_{\mathcal{Q}}(\mathcal{L}_k) \neq C_{\mathcal{Q}}(\mathcal{L}_\ell)$, then for every $\overline{s}$:

$$P(\overline{s}|\mathcal{Q} \text{ runs}) = P(\overline{s}|\overline{\mathcal{A}} \text{ runs})$$

Consider the matching M_G defined in Section V-B1, which is of size $(n-\Delta)/2$. From $M_G = (V', E')$, we can easily derive another matching $\tilde{M}_G = (V'', E'')$ of size $\left(\frac{n-\Delta}{2}\right)^{1/4}$ such that $V'' \subset V'$ and $E'' \subset E'$, and each edge in E'' satisfies the condition: $(i,j) \in E'' \Rightarrow (j,i) \in E''$. Clearly,

$$\mathcal{T}_{\mathcal{Q}}^{G} \geq \mathcal{T}_{\mathcal{Q}}^{M_G} \geq \mathcal{T}_{\mathcal{Q}}^{\tilde{M}_G} \geq \frac{1}{8}\log\left(\frac{n-\Delta}{2}\right) \text{ w.h.p} \quad (16)$$

where the last inequality follows by noting that $\mathcal{Q}$ is a uniform randomized algorithm and by proceeding identically to the analysis in Section V-B1.

Let $\mathcal{L}_1, \ldots, \mathcal{L}_{|V''|}$ be the locations of the nodes in $\tilde{M}_G$. Then,

$$P\left(\forall k \neq \ell : C_{\mathcal{Q}}(\mathcal{L}_k) \neq C_{\mathcal{Q}}(\mathcal{L}_\ell)\right) \geq 1 - \frac{\binom{2\left(\frac{n-\Delta}{2}\right)^{\frac{1}{4}}}{2}}{n} \to 1$$

as $n \to \infty$. Therefore, it follows from (16) and Lemma 7 that $\mathcal{T}_{\overline{\mathcal{A}}}^{\tilde{M}_G} = \Omega(\ln\left(\frac{n-\Delta}{2}\right))$ w.h.p, which yields a lower bound of $\Omega(\ln n)$ when $\Delta = o(\ln n)$. This result combined with the trivial lower bound of $\Omega(\Delta)$ yields $\mathcal{T}_{\overline{\mathcal{A}}}^{G} = \Omega(\Delta + \ln n)$ w.h.p.

Our result thus implies that the ALOHA-like algorithm is at most a factor $\min(\Delta, \ln n)$ worse than the optimal.

VI. Discussion

In this section, we discuss a number of pertinent issues relevant to the proposed neighbor discovery algorithms and their analysis.

A. Neighbor Discovery Using Directional Antennas

The analysis of the ALOHA-like algorithm can also be extended to the case where nodes have directional antennas. For instance, in [31], the authors propose a variant of the ALOHA-like algorithm which operates as follows. At each slot, a node transmits with probability p_{x} by pointing its antenna, which has a fixed beamwidth θ, in a direction chosen uniformly at random from the interval $[0, 2\pi]$.

[31] derives the optimal value of p_{x} as

$$p_{\text{x}} = \frac{2\pi}{\theta(\Delta+1)}, \quad \Delta + 1 > \frac{2\pi}{\theta}$$

where Δ is the maximum node degree in the network.

To avoid repetition, we only consider the case where nodes have a directional transmitter and an omni-directional receiver. In this case, the probability that a given node i discovers a neighbor j in a given slot is given by

$$p = \underbrace{\frac{2\pi}{\theta(\Delta+1)}}_{i \text{ transmits}} \underbrace{\left(1 - \frac{2\pi}{\theta(\Delta+1)}\right)}_{j \text{ receives}} \underbrace{\left(1 - \frac{\theta}{2\pi}\frac{2\pi}{\theta(\Delta+1)}\right)^{\Delta-1}}_{\text{no other node transmits to } i}$$
$$\geq \frac{2\pi}{\theta(\Delta+1)e}\left(1 - \frac{2\pi}{\theta(\Delta+1)}\right)$$

Similar to Section V-A, we can show that all edges in a network with n nodes and maximum node degree Δ are discovered in at most $\frac{3(\Delta+1)e\ln n}{\frac{2\pi}{\theta}\left(1-\frac{2\pi}{\theta(\Delta+1)}\right)}$ slots w.h.p. Thus, when $\Delta >> \frac{2\pi}{\theta}$, we obtain a factor of $2\pi/\theta$ speed-up over the case where nodes have omni-directional antennas.

B. RFID Tag Identification

The neighbor discovery algorithms proposed in this paper can easily be adapted to solve the RFID tag identification problem, where a tag reader needs to identify the IDs of the tags in its range. In particular, the feedback-based algorithms proposed in Section IV are well-suited to address this problem and operate as follows. Each time the tag reader discovers a new tag, it announces the ID of the tag allowing it to drop out. Unlike prior work addressing the RFID tag identification problem (see Section VII for a list of references), our algorithms do not require collision detection and do not require *a priori* estimate of the number of tags. Further, the termination condition **TC** can be used to detect end of the tag discovery process, when the number of tags is not known.

C. Discovery of Asymmetric Edges

Throughout this paper, we have assumed that edges between node pairs are symmetric. We next describe a simple heuristic for discovering asymmetric edges using the ALOHA-like algorithm. In particular, if we have a pair of nodes i and j such that $(i,j) \in E$, but $(j,i) \notin E$, our heuristic allows node j to remove i from its list of neighbors. Upon termination of neighbor discovery in the r-th phase, each node runs the ALOHA-like algorithm for an identical duration as that of the r-th phase. During this phase, each node announces the IDs of all the neighbors it has discovered. Since j will not be in the list of IDs included in i's transmission, node j removes i from its neighbor list.

D. Feedback-based Algorithms for Multi-Hop Networks

It is interesting to ask whether or not the feedback-based algorithms studied in Section IV can be extended to the multi-hop network setting. There are two important obstacles that need to be overcome in this regard.

1) In a clique setting, when a node i, hears its ID back, it knows that all other nodes in the clique have discovered i, thus allowing it to drop out. In the multi-hop case, however, the presence of hidden terminals may cause a subset of i's neighbors to not receive i's transmission. Thus, i cannot drop out despite hearing its ID back.
2) In the multi-hop setting, i's dropping out needs to be signaled to its neighbors allowing them to increase their transmission probabilities, which appears non-trivial.

Exploiting receiver feedback in the multi-hop setting in a manner that yields improvement over the ALOHA-like discovery algorithm is therefore an interesting open problem.

E. Beacon-based Neighbor Discovery

In beacon-based neighbor discovery, each node transmits *BEACON* messages at fixed intervals i.e., the interval size is independent of n. To avoid synchronization between beacon transmissions, a random delay is added to the intervals. This scheme has been proposed in numerous contexts, e.g., (i) to maintain neighbor list in routing protocols such as AODV [26], DSR [13], and GPSR [15], and (ii) for topology formation in the context of Bluetooth [5] and IEEE 802.15.4 [1].

Beacon-based neighbor discovery can be thought of as a randomized algorithm in which each node transmits with probability $1/k$ at each slot, where k is fixed. For simplicity, consider a clique of n nodes. The probability of a successful transmission is then given by

$$p = \frac{1}{k}\left(1 - \frac{1}{k}\right)^{n-1} \approx \frac{e^{-n/k}}{k}$$

Similar to Section III, it can be shown that the expected time to discover all n nodes equals $ke^{n/k}(\ln n + \Theta(1))$.

To compare the performance of beacon-based neighbor discovery with the algorithms studied in this paper, we consider a Bluetooth network, where each slot is of duration 0.625 ms and beacons are transmitted once every $k = 14$ slots [5]. In a dense setting, where $n \sim 100$, beacon-based neighbor discovery is 65 times slower than the ALOHA-like algorithm and 300 times slower than the collision detection-based algorithm.

The above example illustrates the poor performance of beacon-based neighbor discovery when nodes transmit too frequently. At the other extreme, nodes may transmit very infrequently. For example, the recommended beacon interval in GPSR [15] is 1s. Assuming slots of size 0.625 ms, this corresponds to each node transmitting with probability $1/1600$. When $n = 10$, beacon-based neighbor discovery is 59 times slower than the ALOHA-like algorithm and 135 times slower than the collision detection-based algorithm.

It must also be noted that there is no obvious method for terminating the beacon-based neighbor discovery algorithm without an *a priori* estimate of network density.

F. Other Wireless Channel Models

All the results we have derived thus far are based on the assumption of a collision channel model. Extension to more general wireless channel models that incorporate fading effects and errors in transmission/reception is an interesting open direction. While the correctness of the ALOHA-like algorithm is independent of the wireless channel model used, the Feedback-based algorithms however require modification for correct operation under other wireless channel models. As an example, [17] assumes Rayleigh fading channels and proposes a simple extension to the collision detection-based neighbor discovery algorithm in which each node is required to successfully transmit *DISCOVERY* messages k times before it drops out, where k is a function of the channel parameters.

VII. Related Work

An early work on neighbor discovery is [23], which proposes a synchronous ALOHA-like neighbor discovery algorithm, identical to the one studied in this paper. More recently, an asynchronous, randomized neighbor discovery algorithm has been proposed in [6]. A feedback based neighbor discovery algorithm designed to operate in fading environments has been studied in [17]. However, the performance of these algorithms is not well-understood, even in the case of single-hop networks. Further, each of these algorithms require *a priori* estimates of node density and do not address the issue of termination of neighbor discovery.

Keshavarzian et al. [16] propose a novel, deterministic neighbor discovery algorithm. However, nodes need to be synchronized with each other and need to know the maximum number of neighbors, n, *a priori*. Furthermore, the neighbor discovery needs n^2 time slots to discover all the neighbors.

Neighbor discovery algorithms using a multi-user detection approach have been proposed in [2]. However, these algorithms require synchronization between nodes and also require each node to know the signatures of every other node in the network. An interesting approach to neighbor discovery based on group testing has been proposed in [21]. But it too suffers from the same practical limitations as [2].

There have been numerous proposals for neighbor discovery when nodes have directional antennas [31], [12], [33], [28]. In general, these solutions propose antenna scanning strategies for efficient neighbor discovery. However, none of these proposals address the practical challenges considered in this paper. Further, the analysis in this paper can be used to provide a more rigorous understanding of the directional neighbor discovery problem, as shown in Section VI.

There exists a large body of literature addressing the RFID tag identification problem (see for example [29], [14], [19], [11], [32]), where a designated tag reader needs to determine the IDs of tags in its range. *Prima facie*, the tag identification problem bears resemblance to the neighbor discovery problem. However, neighbor discovery is more challenging. First, existing tag identification algorithms typically assume that the number of tags is known *a priori*. Second, the transmissions of the tags can be more easily controlled due to the presence of the tag reader, which functions as a *master node* during the discovery process. Finally, tag identification algorithms do not address the hidden terminal problem present in multi-hop wireless networks.

Some of the literature on conflict resolution in multiple access channels is related to the neighbor discovery problem and its analysis. In particular, the deterministic *tree algorithms* in [7], [10], [30] can be adapted to perform neighbor discovery. However, these algorithms are not suitable for several reasons. The algorithms typically assume that node IDs lie in the range $[1 \ldots N]$, where N is assumed to be known *a priori*. Secondly, the performance of the algorithms scales as $\Theta(n + n\log(N/n))$, where n denotes the number of neighbors. Thus, the algorithms perform poorly when $N >> n$ (e.g. $N = e^n$). Finally, these algorithms work only in single-hop networks and, require synchronization and collision detection.

VIII. Conclusions

In this paper, we have presented efficient neighbor discovery algorithms for wireless networks that comprehensively address various practical limitations of the earlier approaches. Our neighbor discovery algorithms do not require estimates of node density and allow asynchronous operation. Furthermore, our algorithms allow nodes to begin execution at different times and also allow nodes to detect termination.

Our analysis shows a gap between the lower and upper bounds on the running time for neighbor discovery in the network case. Clearly, the quest for an order-optimal neighbor discovery algorithm remains an intriguing prospect. Of particular interest is the question of whether the feedback-based algorithms, which are order-optimal in the single-hop case, can be extended to the multi-hop network setting while outperforming the ALOHA-like algorithm. Another direction of interest is the extension of the various algorithms and the analysis presented in this paper to wireless channel models that incorporate phenomena such as fading and shadowing.

References

[1] IEEE 802.15.4 Standard Specification. http://standards.ieee.org/getieee802/802.15.html.

[2] D. Angelosante, E. Biglieri, and M. Lops. Neighbor discovery in wireless networks:a multiuser-detection approach. In *Information Theory and Applications Workshop*, pages 46–53, 2007.

[3] L. Bao and J. J. Garcia-Luna-Aceves. A new approach to channel access scheduling for ad hoc networks. In *ACM MOBICOM*, pages 210–221, 2001.

[4] R. Bar-Yehuda, O. Goldreich, and A. Itai. On the time complexity of broadcast in multi-hop radio networks: An exponential gap between determinism and randomization. *Journal of Computer and System Sciences*, 45(1):104–126, 1992.

[5] Bluetooth Specification Version 3.0 + HS.

[6] S. A. Borbash, A. Ephremides, and M. J. McGlynn. An asynchronous neighbor discovery algorithm for wireless sensor networks. *Ad Hoc Networks*, 5(7):998–1016, 2007.

[7] J. Capetanakis. Generalized TDMA: The multi-accessing tree protocol. *IEEE Transactions On Communications*, 27(10):1479–1484, 1979.

[8] H. David and H. Nagaraja. *Order Statistics*. John Wiley and Sons, 2003.

[9] A. G. Greenberg and S. Winograd. A lower bound on the time needed in worst case to resolve conflicts deterministically in multi-access channels. *Journal of the ACM*, 32(3):589–596, 1985.

[10] J. Hayes. An adaptive technique for local distribution. *IEEE Transactions On Communications*, 26:1178–1186, 1978.

[11] D. Hush and C. Wood. Analysis of tree algorithms for RFID arbitration. In *IEEE International Symposium on Information Theory*, 1998.

[12] G. Jakllari, W. Luo, and S. V. Krishnamurthy. An integrated neighbor discovery and mac protocol for ad hoc networks using directional antennas. *IEEE Transactions on Wireless Communications*, 6(3):1114–1024, 2007.

[13] D. B. Johnson and D. A. Maltz. Dynamic source routing in ad hoc wireless networks. In *Mobile Computing*, pages 153–181. Kluwer Academic Publishers, 1996.

[14] R. Kalinowski, M. Latteux, and D. Simplot. An adaptive anti-collision protocol for smart labels. Technical report, LIFL University Lille, 2001.

[15] B. Karp and H. T. Kung. Greedy perimeter stateless routing for wireless networks. In *ACM MOBICOM*, pages 243–254, 2000.

[16] A. Keshavarzian and E. Uysal-Biyikoglu. Energy-efficient link assessment in wireless sensor networks. In *IEEE INFOCOM*, 2004.

[17] R. Khalili, D. Goeckel, D. Towsley, and A. Swami. Neighbor discovery with reception status feedback to transmitters. In *IEEE INFOCOM*, pages 2375–2383, 2010.

[18] E. Kushilevitz and Y. Mansour. An $\Omega\left(D\log(N/D)\right)$ lower bound for broadcast in radio networks. *SIAM Journal on Computing*, 27(3):702–712, 1998.

[19] C. Law, K. Lee, and K. Y. Siu. Efficient memoryless protocol for tag identification. In *Proceedings of the 4th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, pages 75–84, 2000.

[20] L. Li, J. Y. Halpern, P. Bahl, Y.-M. Wang, and R. Wattenhofer. A cone-based distributed topology-control algorithm for wireless multi-hop networks. *IEEE/ACM Transactions on Networking*, 13(1):147–159, 2005.

[21] J. Luo and D. Guo. Neighbor discovery in wireless ad hoc networks based on group testing. In *Annual Allerton Conference*, 2008.

[22] Maxim DS32khz Data Sheet.

[23] M. J. McGlynn and S. A. Borbash. Birthday protocols for low energy deployment and flexible neighbor discovery in ad hoc wireless networks. In *ACM MOBIHOC*, pages 137–145, 2001.

[24] D. Mitrinovic. *Elementary Inequalities*. P. Noordhoff Groningen, 1964.

[25] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.

[26] C. E. Perkins and E. M. Belding-Royer. Ad-hoc on-demand distance vector routing. In *IEEE WMCSA*, pages 90–100, 1999.

[27] K. Pister and L. Doherty. TSMP: Time synchronized mesh protocol. In *IASTED Distributed Sensor Networks*, pages 391–398, 2008.

[28] R. Ramanathan, J. Redi, C. Santivanez, D. Wiggins, and S. Polit. Ad hoc networking with directional antennas: a complete system solution. *IEEE Journal on Selected Areas in Communications*, 23:496–506, 2005.

[29] D. Simplot-Ryl, I. Stojmenovic, A. Micic, and A. Nayak. A hybrid randomized protocol for RFID tag identification. *Sensor Review*, 26(2):147–154, 2006.

[30] B. Tsybakov and V. Mikhailov. Random multiple packet access: Part and try algorithm. *Prob. Inf Transmission (translated from Russian original in Prob. Peredach. Inform., Oct.-Dec. 1980)*, 16(4):305–317, 1981.

[31] S. Vasudevan, J. F. Kurose, and D. F. Towsley. On neighbor discovery in wireless networks with directional antennas. In *IEEE INFOCOM*, pages 2502–2512, 2005.

[32] H. Vogt. Multiple object identification with passive RFID tags. In *IEEE International Conference on Systems, Man and Cybernetics*, 2002.

[33] Z. Zhang and B. Li. Neighbor discovery in mobile ad hoc self-configuring networks with directional antennas: algorithms and comparisons. *IEEE Transactions on Wireless Communications*, 7(5-1):1540–1549, 2008.

Appendix A
Approximation Error in $E[W]$ for ALOHA-like Neighbor Discovery

In this section, we calculate bounds on the approximation error for the results derived in Section III-B.

Recall from (1), we have the following approximation

$$p = p_{\text{xmit}}(1-p_{\text{xmit}})^{n-1} = \frac{1}{n}\left(1-\frac{1}{n}\right)^{n-1} \approx \frac{1}{ne} \qquad (17)$$

Since $(1-1/n)^{n-1} \geq 1/e$, it follows that

$$p = \frac{1}{ne} + \delta_e(n)$$

where $\delta_e(n) > 0$ denotes the approximation error in (17). Using Taylor's Theorem, we can write

$$(n-1)\ln\left(1-\frac{1}{n}\right) = -1 + R(n)$$

where $R(n)$ represents the remainder term and is given by

$$R(n) = \frac{1}{n} - \frac{n-1}{2n^2(1-c)^2}, 0 \leq c \leq 1/n$$

$R(n)$ is maximized when $c = 0$ and therefore,

$$R(n) \leq \frac{n+1}{2n^2} < \frac{1}{n}$$

Therefore,

$$0 < \delta_e(n) < \frac{1}{ne}(e^{\frac{1}{n}} - 1) \qquad (18)$$

It is easy to see that $\delta_e(n) \to 0$, as $n \to \infty$.

Recalling from Section III-B that $E[W] = \frac{1}{p}H_n$, we obtain

$$E[W] = \left(\frac{1}{\frac{1}{ne} + \delta_e(n)}\right)H_n = ne$$

From (18), it follows that

$$ne^{1-\frac{1}{n}}H_n \leq E[W] \leq neH_n$$

It is easy to see that $E[W] \to neH_n$ as $n \to \infty$.

Appendix B
Sharp Concentration of ALOHA-like Neighbor Discovery

In this section, we provide a more rigorous proof of the sharp concentration result for the ALOHA-like neighbor discovery algorithm based on Boole-Bonferroni inequalities. The proof is very similar to the proofs available in [25] for the sharp concentration for the Coupon Collector's Problem.

Lemma 8: Let c be a real constant and $t = ne(\ln n + c)$ for positive integer n. Then, for any fixed positive integer k,

$$\lim_{n\to\infty} \binom{n}{k}\left(1-\frac{k}{ne}\right)^t = \frac{e^{-ck}}{k!}$$

Proof: We make use of the following inequality in our proof: For all $x, a \in \Re$, such that $a \geq 1$ and $|x| \leq a$,

$$e^x\left(1-\frac{x^2}{a}\right) \leq \left(1+\frac{x}{a}\right)^a \leq e^x$$

Let $x = -\frac{kt}{ne}$ and $a = t$. Substituting in the above inequality, we get

$$e^{-\frac{kt}{ne}}\left(1-\frac{k^2t}{n^2e^2}\right) \leq \left(1-\frac{k}{ne}\right)^t \leq e^{-\frac{kt}{ne}}$$

$$e^{-\frac{tk}{ne}}\left(1-\frac{k(\ln n + c)}{ne}\right)^{\frac{tk}{ne}} \leq \left(1-\frac{k}{ne}\right) \leq e^{-\frac{kt}{ne}}$$

Note $e^{-\frac{kt}{ne}} = n^{-k}e^{-ck}$. Also,

$$\lim_{n\to\infty}\left(1-\frac{k(\ln n + c)}{ne}\right) = 1$$

For large n, we know that

$$\binom{n}{k} \sim \frac{n^k}{k!}$$

Putting all this together yields the desired result. ∎

Theorem 1: Let W denote the time required to discover all the n neighbors. Then, for any constant $c \in \Re$ and $t = ne(\ln n + c)$,

$$\lim_{n\to\infty} P(W > t) = 1 - e^{-e^{-c}}$$

Proof: The proof is exactly the same as described in [25] and has been produced here for completeness. The event $\{W > t\} = \bigcup_{i=1}^n \mathcal{E}_i(t)$, where $\mathcal{E}_i(t)$ denotes the event that a node i is not discovered within t time slots. By the Principle of Inclusion-Exclusion, we have

$$P\left(\bigcup_{i=1}^n \mathcal{E}_i(t)\right) = \sum_{k=1}^n (-1)^{k+1}\mathcal{P}_{k,n}$$

where

$$\mathcal{P}_{k,n} = \sum_{1\leq i_1 \ldots \leq i_k \leq n} P\left(\bigcap_{\ell=1}^k \mathcal{E}_{i_\ell}(t)\right)$$

Let $\mathcal{S}_{k,n} = \mathcal{P}_{1,n} - \mathcal{P}_{2,n} + \ldots + (-1)^{k+1}\mathcal{P}_{k,n}$ denote the partial sum formed by the first k terms in this series. By the Boole-Bonferroni inequalities, for odd $k \geq 1$,

$$P\left(\bigcup_i \mathcal{E}_i(t)\right) \leq \sum_{\ell=1}^k (-1)^{\ell+1}\mathcal{P}_{\ell,n}$$

and for even $k \geq 2$,

$$P\left(\bigcup_i \mathcal{E}_i(t)\right) \geq \sum_{\ell=1}^k (-1)^{\ell+1}\mathcal{P}_{\ell,n}$$

Putting the two Boole-Bonferroni inequalities together, we can write

$$\mathcal{S}_{2k,n} \leq P\left(\bigcup_i \mathcal{E}_i(t)\right) \leq \mathcal{S}_{2k+1,n}$$

Since all k-wise intersections of $\mathcal{E}_i(t)$ are equally likely,

$$\mathcal{P}_{k,n} = \binom{n}{k} P\left(\bigcap_{i=1}^k \mathcal{E}_i(t)\right)$$

Now, the probability of intersection of the k events $\mathcal{E}_1(t), \ldots, \mathcal{E}_k(t)$ is the probability of not discovering any of the first k nodes in t time slots and is equal to $\left(1-\frac{k}{ne}\right)^t$. Therefore,

$$\mathcal{P}_{k,n} = \binom{n}{k}\left(1-\frac{k}{ne}\right)^t$$

Therefore,

$$\lim_{n\to\infty} \mathcal{P}_{k,n} = \mathcal{P}_k = \frac{e^{-ck}}{k!}$$

Let $\mathcal{S}_k = \sum_{\ell=1}^k (-1)^{\ell+1}\mathcal{P}_j = \sum_{\ell=1}^k (-1)^{\ell+1}\frac{e^{-c\ell}}{\ell!}$. Note that the right hand side of the expression for $\mathcal{S}_k$ consists of the first k terms of the power series expansion of $f(c) = 1 - e^{-e^{-c}}$. We conclude that

$$\lim_{k\to\infty} \mathcal{S}_k = f(c)$$

That is for all $\varepsilon > 0$, there exists a $k^* > 0$, such that for any $k > k^*$,

$$|\mathcal{S}_k - f(c)| < \varepsilon$$

Since $\lim_{n\to\infty} \mathcal{P}_k^n = \mathcal{P}_k$, it follows that $\lim_{n\to\infty} \mathcal{S}_{k,n} = \mathcal{S}_k$. Equivalently, for all $\varepsilon > 0$ and k, when n is sufficiently large, $|\mathcal{S}_{k,n} - \mathcal{S}_k| < \varepsilon$. Thus, for all $\varepsilon > 0$, any fixed $k > k^*$, and n sufficiently large, $|\mathcal{S}_{k,n} - \mathcal{S}_k| < \varepsilon$ and $|\mathcal{S}_k - f(c)| < \varepsilon$. Therefore,

$$|\mathcal{S}_{k,n} - f(c)| < 2\varepsilon$$

and

$$|\mathcal{S}_{2k,n} - \mathcal{S}_{2k+1,n}| < 4\varepsilon$$

$$|P\left(\bigcup_i \mathcal{E}_i(t)\right) - f(c)| < 4\varepsilon$$

This implies the desired result that

$$\lim_{n\to\infty} P\left(\bigcup_i \mathcal{E}_i(t)\right) = f(c) = 1 - e^{-e^{-c}}$$

∎

APPENDIX C
CHERNOFF BOUNDS FOR ERLANG RANDOM VARIABLE

Theorem 2: Let $X \sim$ Erlang(n, β), with mean $E[X] = n/\beta$. For all $\varepsilon > 0$,

1) $P(X \geq (1+\varepsilon)E[X]) \leq \left(\frac{1+\varepsilon}{e^{\varepsilon}}\right)^n$
2) $P(X \leq (1-\varepsilon)E[X]) \leq \left(\frac{1-\varepsilon}{e^{-\varepsilon}}\right)^n$

Proof:

$$P(X \leq a) \leq \inf_{s<0} e^{-sa} M_X(s) \tag{19}$$

where $M_X(s) = E[e^{sX}]$ denotes the moment generating function of the random variable X. For an Erlang (n, β) random variable

$$M_X(s) = \left(\frac{\beta}{\beta - s}\right)^n$$

A standard calculation shows that the value of s that minimizes the right hand side of (19) is given by

$$s^* = \beta - \frac{n}{a}$$

Therefore,

$$P(X \leq a) \leq e^{-(a\beta - n)} \left(\frac{a\beta}{n}\right)^n$$

Setting $a = (1-\varepsilon)E[X] = \frac{(1-\varepsilon)n}{\beta}$, $\forall \varepsilon > 0$ yields

$$P(X \leq (1-\varepsilon)E[X]) \leq \left(\frac{1-\varepsilon}{e^{-\varepsilon}}\right)^n, \quad \forall \varepsilon > 0$$

Since $1 - \varepsilon < e^{-\varepsilon}$, the right hand side in the above inequality goes to 0 as $n \to \infty$.

The probability bound for the upper tail can be derived similarly. In particular, for a non-negative random variable X and $a > 0$, we have

$$P(X \geq a) \leq \inf_{s>0} e^{-sa} M_X(s)$$

Proceeding exactly in the same manner as before, it can be easily shown that

$$P(X \geq (1+\varepsilon)E[X]) \leq \left(\frac{1+\varepsilon}{e^{\varepsilon}}\right)^n, \quad \forall \varepsilon > 0$$

Since $1 + \varepsilon < e^{\varepsilon}$, the right hand side goes to 0 as $n \to \infty$. ■

APPENDIX D
CHERNOFF BOUNDS FOR ORDER STATISTICS OF EXPONENTIAL RANDOM VARIABLES

We first state an important theorem from [8][pp.18] that characterizes the distribution of the k-th order statistic from from n i.i.d exponential random variables.

Theorem 3: Let $X_1, \ldots, X_n$ denote a sequence of n i.i.d exponential random variables, each having mean $1/\beta$. Let the random variable $X^{(k)}$ denote the k-th order statistic. Then,

$$(X^{(k)}, k = 1, \ldots, n) \overset{d}{=} \frac{1}{\beta}\left(\sum_{\ell=1}^{k} \frac{Z_\ell}{n - \ell + 1}, k = 1, \ldots, n\right)$$

where the Z_ℓs are i.i.d exponential random variables with mean 1.

From Theorem 3, it immediately follows that

$$P\left(X^{(k)} \geq x\right) = P\left(\sum_{\ell=1}^{k} \tilde{Z}_\ell \geq \beta x\right)$$

where $\tilde{Z}_\ell$ is an exponential random variable with mean $1/(n - \ell + 1)$. Note that the $\tilde{Z}_\ell$s are independent, but not identically distributed exponential random variables. Applying Chernoff bound, we get

$$\begin{aligned} P\left(X^{(k)} \geq x\right) &\leq e^{-s\beta x} \prod_{\ell=1}^{k} E[e^{s\tilde{Z}_\ell}], \quad \forall s > 0 \\ &= e^{-s\beta x} \prod_{\ell=1}^{k} \left(1 - \frac{s}{n - \ell + 1}\right)^{-1} \end{aligned}$$

Note that the second inequality is only applicable when $s < n - k + 1$. Letting $s = 1$, we get

$$P\left(X^{(k)} \geq x\right) \leq e^{-\beta x} \frac{n}{n-k}$$

APPENDIX E
PROOF OF CORRECTNESS OF TERMINATION CONDITION

A. Notation

Throughout this section, S_r denotes the number of distinct nodes that successfully transmit at least once in the r-th phase. $\tilde{S}_r$ denotes the total number of successful transmissions in the r-th phase. Note that $S_r \leq \tilde{S}_r$. p_r denotes the probability of successful transmission in the r-th phase and is given by

$$p_r = \frac{n}{2^r}\left(1 - \frac{1}{2^r}\right)^{n-1} \leq \frac{n e^{-(n-1)/2^r}}{2^r}$$

t_r is the duration of the r-th phase and is given by

$$t_r = 2^{r+1} e \ln 2^r$$

B. Proof of Lemma 2

It is easy to verify that $\tilde{S}_r \sim$ Binomial(t_r, p_r). Since $r \leq \log\left(\frac{n-1}{2\ln n}\right)$ in *Stage 1* and p_r increases monotonically as r varies from $1 \ldots \log\left(\frac{n-1}{2\ln n}\right)$, it follows that

$$p_r \leq p \triangleq \frac{2\ln n}{n(n-1)}$$

Further,

$$t_r \leq t \triangleq \frac{2e(n-1)\ln\left(\frac{n-1}{2\ln n}\right)}{2\ln n}$$

Let $S \sim$ Binomial(t, p). Applying Chernoff bound [25], we get

$$P(S \geq (1+\varepsilon)tp) \leq \left(\frac{e^{\varepsilon}}{(1+\varepsilon)^{1+\varepsilon}}\right)^{tp}, \quad \forall \varepsilon > 0$$

Noting that $0 < tp = \frac{2e\ln\left(\frac{n-1}{2\ln n}\right)}{n} < 1$, $\forall n \geq 4$ and setting $\varepsilon = \frac{1}{tp} - 1$, we get

$$P(S \geq 1) \leq tpe^{1-tp} \leq tpe$$

Noting that $\tilde{S}_r$ is stochastically dominated by S, we get

$$P(\tilde{S}_r \geq 1) \leq tpe = \frac{2e^2 \ln\left(\frac{n-1}{2\ln n}\right)}{n} \leq \frac{2e^2 \log\left(\frac{n-1}{2\ln n}\right)}{n}$$

Application of the union bound yields

$$\begin{aligned} P\left(\bigcap_{r=1}^{\log\left(\frac{n-1}{2\ln n}\right)} \left\{\tilde{S}_r = 0\right\}\right) &\geq 1 - \frac{2e^2 \log^2\left(\frac{n-1}{2\ln n}\right)}{n} \\ &\to 1 \text{ as } n \to \infty \end{aligned}$$

Since $S_r \leq \tilde{S}_r$, Lemma 2 immediately follows.

C. *Proof of Lemma 3*

Applying Markov's inequality,

$$P(S_r \geq 2^{r-1}) \leq P(\tilde{S}_r \geq 2^{r-1}) \leq \frac{E[\tilde{S}_r]}{2^{r-1}} = \frac{t_r p_r}{2^{r-1}}$$

Further simplification of the right hand side yields

$$P(S_r \geq 2^{r-1}) \leq \frac{4e^2(n/2^r)\ln 2^r}{e^{n/2^r}} \leq \frac{4e^2 \ln 2^r}{e^{n/2^{r+1}}}$$

where the last inequality follows from the fact that $e^x \geq xe^{x/2}$, $\forall x > 0$ (see [24][pp.76]). Noting that $r \leq \log\left(\frac{n-1}{3\ln\ln n}\right)$ in *Stage 2*, we obtain

$$P(S_r \geq 2^{r-1}) \leq \frac{4e^3 \ln n}{\ln^{\frac{3}{2}} n} \leq \frac{4e^3}{\ln^{\frac{1}{2}} n}$$

Noting that there are at most $\log(2\ln n)$ phases in *Stage 2* and applying the union bound, we get

$$P\left(\bigcup_r \{S_r \geq 2^{r-1}\}\right) \leq \frac{4e^3 \log(2\ln n)}{\ln^{\frac{1}{2}} n} \to 0 \text{ as } n \to \infty$$

This proves Lemma 3.

D. *Proof of Lemma 4*

Here, we show that w.h.p $S_r > 2^{r-1}$ throughout *Stage 3*. This implies that $D_{i,r} \geq 2^{r-1}$ $\forall i$, which suffices to ensure that no node terminates the algorithm during *Stage 3*.

Let $Y_{i,r}$ be the time of first successful transmission by node i in phase r. Recall from Section III-C that $Y_{i,r}$ can be treated as an exponential random variable with distribution

$$P(Y_{i,r} \leq t) = 1 - e^{-tp_{i,r}}$$

where $p_{i,r}$ is the probability of a successful transmission by node i in the r-th phase and is given by

$$p_{i,r} = \frac{1}{2^r}\left(1 - \frac{1}{2^r}\right)^{n-1}$$

Further, the $Y_{i,r}$s, $i = 1, \ldots, n$, are i.i.d random variables. Let $Y_r^{(k)}$ denote the k-th order statistic. Then,

$$S_r > 2^{r-1} \iff Y_r^{(2^{r-1}+1)} \leq t_r$$

From Appendix D, we obtain

$$P\left(S_r \leq 2^{r-1}\right) \leq \frac{ne^{-p_{i,r}t_r}}{n - 2^{r-1} - 1}$$

Since $r \leq \log n$ in *Stage 3*, it follows that

$$P\left(S_r \leq 2^{r-1}\right) \leq \frac{ne^{-p_{i,r}t_r}}{n/2 - 1} \leq 4e^{-p_{i,r}t_r}, \ \forall n \geq 4 \qquad (20)$$

Now,

$$p_{i,r}t_r = 2e\left(1 - \frac{1}{2^r}\right)^{n-1} \ln 2^r$$

Noting that (see [25][pp.435])

$$\left(1 + \frac{x}{a}\right)^a \geq e^x\left(1 - \frac{x^2}{a}\right), \ \forall x, a \text{ such that } |x| \leq a, a \geq 1$$

we obtain

$$p_{i,r}t_r \geq 2e\left(\frac{1}{e}\left(1 - \frac{1}{2^r}\right)\right)^{\frac{n-1}{2^r}} \ln 2^r = 2e^{1-\frac{n-1}{2^r}} \ln 2^r$$

Since $r \geq \log\left(\frac{n-1}{3\ln\ln n}\right) + 2$ in *Stage 3*,

$$\begin{aligned} p_{i,r}t_r &\geq 2e^{1-\left(\frac{3\ln\ln n}{4}\right)} \ln\left(\frac{4(n-1)}{3\ln\ln n}\right) \\ &= \frac{2e}{\ln^{\frac{3}{4}} n} \ln\left(\frac{4(n-1)}{3\ln\ln n}\right) \\ &\geq 2e\left(\ln^{\frac{1}{4}} n - \frac{\ln(3\ln\ln n)}{\ln^{\frac{3}{4}} n}\right) \end{aligned}$$

Substituting the lower bound for $p_{i,r}t_r$ into (20) and noting that there are at most $\log(6\ln\ln n)$ phases in *Stage 3*, it can be verified using the union bound that

$$P\left(\bigcup_r \{S_r \leq 2^{r-1}\}\right) \to 0 \text{ as } n \to \infty$$

This concludes the proof of Lemma 4.