Neighbor Discovery in Peer-to-Peer Wireless Networks with Multi-Channel MPR Capability

Lizhao You[†], Xiaojun Zhu[†], Guihai Chen[†§]
[†]State Key Laboratory for Novel Software Technology, Nanjing University, China
[§]Department of Computer Science and Engineering, Shanghai Key Laboratory of Scalable Computing and Systems, Shanghai Jiao Tong University, China
Email: lzyou@smail.nju.edu.cn, gxjzhu@gmail.com, gchen@nju.edu.cn

***Abstract*—We study the time duration of neighbor discovery in peer-to-peer wireless networks with multi-channel multi-packet reception capability. Radios with this capability, at any time slot, can either transmit messages in one channel or receive messages from all channels. This capability is provided by emerging CDMA/OFDMA techniques. Neighbor discovery in this scenario is different from other multi-packet reception scenarios like MIMO since collision can still happen in single channel. To discover neighbors with such capability in single-hop networks, we prove that the expected running time of all randomized algorithms is $\Theta(n/k)$ where n is the number of nodes and k is the number of channels. We provide a Las Vegas algorithm that finds all neighbors with variable running time. We prove that its running time is $\Theta(n/k \ln n)$ with high probability. We also give a Monte Carlo algorithm that terminates in $\Theta(n/k)$ time. We show that it finds all neighbors with high probability. Both algorithms are validated by simulation.**

I. Introduction

Neighbor discovery is a fundamental step for the initialization of wireless networks. The knowledge of neighbors is essential for further operations, e.g., topology control and routing algorithms. Many algorithms [1]–[7] have been proposed and analyzed for neighbor discovery, most of which are randomized algorithms [2]–[6] due to easy implementation. For example, previous work [3] uses Aloha-like protocols and transforms the running time analysis to the *Coupon Collector's Problem* in wireless networks. Later work [4] extends the analysis to low-duty-cycle wireless sensor networks. Both works [3], [4] assume single-channel transmission and reception.

However, multi-packet reception (MPR) capability, provided by advanced physical layer techniques like CDMA (code division multiple access), MIMO (multiple-input and multiple-output) and OFDMA (orthogonal frequency-division multiple access), breaks the assumption of single-channel communication, and brings us a new paradigm: single-channel transmission multi-channel reception. An example of this kind of real system is FlashLinQ [9], which is being built by Qualcomm using OFDM technique. More importantly, FlashLinQ has a synchronous peer-to-peer ad hoc wireless network architecture, and adopts OFDMA for neighbor discovery [11]. In FlashLinQ, the whole bandwidth is first divided into several channels. Each user acquires a channel to send neighbor discovery packets, but receives discovery packets from all channels simultaneously. We call it *multi-channel* MPR model. Note that this model also describes CDMA networks. In this paper, we will focus on this kind of networks.

Recent work [5] shows the improvement of discovery time order bound for MPR wireless networks compared with single-channel networks. However, we differ from Zeng et al. [5] in the communication model. We consider *multi-channel* MPR model, while Zeng et al. [5] use *multi-antenna* MPR model. For k-capability MPR, the *multi-antenna* MPR model assumes that if there are no more than k transmissions at the same time, the receiver can decode those packets successfully. This assumption does not hold in our scenario, since we are channel-based. If more than two users transmit at the same channel, there is a collision. This key feature makes us different from Zeng et al. [5]. Then a natural question arises: does the result developed in *multi-antenna* MPR networks still hold in *multi-channel* MPR networks?

In this paper, we give a positive answer. We believe this to be the first work to consider neighbor discovery running time in *multi-channel* MPR networks. This paper has the following contributions:

- We are the first to prove the lower bound of expected neighbor discovery time for all randomized algorithms to be $\Theta(\frac{n}{k})$, where n is the number of nodes in clique and k is the number of channels;
- We give two randomized algorithms based on Zeng et al. [5], and prove that the running time are $\Theta(\frac{n}{k} \ln n)$ and $\Theta(\frac{n}{k})$ respectively. The results turn out to be the same with *multi-antenna* MPR networks, which enriches the discussion of neighbor discovery in MPR networks.

The rest of the paper is organized as follows. We summarize the related work in Section II, and present the assumptions and formal model in Section III. Then we propose and analyze two randomized algorithms in Section IV. Section V gives some discussions, and Section VI concludes the paper.

II. Related Work

Our work is motivated by FlashLinQ system [9], a wireless peer-to-peer network. In this system, control packets are transmitted in single channel, while reception happens in all channels. Moreover, all time slots can be synchronized by CDMA/GSM cellular timing, DVBH timing, GPS timing,

This work was supported in part by China NSF grants (60825205, 61073152, 61133006) and China 973 project (2012CB316200).

978-1-4577-2053-6/12/$31.00 ©2012 IEEE

along with in-band timing [9]. Ni et al. [11] first consider the neighbor discovery problem in this kind of network, but focus on channel assignment strategy to minimize the distance of two nodes with same transmission channel. Instead, we consider the discovery time problem given randomized channel usage, and give theoretical analysis.

Neighbor discovery algorithms can be classified into three categories: deterministic [1], randomized [2]–[6], and multi-user detection (MUD) [7], [8]. Deterministic algorithms [1] often come with centralized scheduling, and the distributed implementation is hard to design due to the lack of network parameters. Hence in this paper we focus on randomized algorithms. The MUD approaches [7], [8] focus on the enabling signal processing techniques, while we focus on the the neighbor discovery time analysis. The neighbor discovery time bound problem is first proposed and analyzed by Vasudevan et al. [3], who consider the single-channel communication model. Many works follow. For example, You et al. [4] consider this problem in low-duty-cycle wireless sensor networks. Mittal et al. [6] design and analyze randomized distributed algorithms for multi-channel cognitive radio networks. Karowski et al. [12] consider beacon-based multi-channel wireless sensor networks. Instead, we consider a single-channel transmission multi-channel reception MPR model. Neighbor discovery time bound problem for *multi-antenna* MPR model is studied by Zeng et al. [5]. We use similar technique to analyze our *multi-channel* MPR model, but have some improvements. We prove the lower bound for all randomized algorithms. Also, we do not make approximations in asymptotic analysis and do not assume feedback mechanism that can be impractical due to ACK collisions.

III. Preliminary

A. Network Model and Assumptions

We consider a static wireless network modeled as a graph $G(V, E)$, where $V = \{1, 2, ..., n\}$ is the node set, and E is the edge set. An edge $(u, v) \in E$ iff. v is in the transmission range of u. The system is time-slotted, and slot boundary is aligned. This assumption can be guaranteed in FlashLinQ system, where nodes are synchronized using CDMA/GSM cellular timing, DVBH timing, GPS timing, along with in-band timing [9]. Furthermore, the synchronization accuracy can also be improved by enlarging cyclic prefix (CP) in OFDM physical layer [10].

The mission of neighbor discovery is to discover the ID of its neighbors for each node. In other words, the goal is to discover all links in a network. The communication is modeled by single-channel transmission multi-channel reception. There are overall k channels, labeled $\{1, ..., k\}$, where k is fixed, and is known by all nodes beforehand. Also, $k > 1$. At each time slot, nodes choose to transmit neighbor discovery control packets in a channel, or receive packets simultaneously in all channels. The ID information is embedded in control packets.

If more than two nodes select the same channel to transmit, there is a collision in this channel. Otherwise, the channel is perfect. In other words, packet error due to random fading effect is negligible. This assumption is widely adopted in literature [3]–[5], and is quite accurate when control packets are always well protected by physical layer (e.g., 1/2 convolutional codes and BPSK modulation in IEEE 802.11a/g standard).

Then we classify two kinds of network scenarios: 1) single-hop; 2) multi-hop. We first give results on single-hop *multi-channel* MPR networks, i.e., clique, and then multi-hop networks. Most works [3]–[5] assume single-hop networks, and our results are comparable with those works. Also, in some scenario like FlashLinQ [9], a single-hop peer-to-peer network is possible, because the communication range is hundreds of meters if it is designed for mobile phones.

B. Problem Statement

A link (i, j) is discovered when i transmits in a single channel without collision and j listens. We call j finds its neighbor i. Neighbor discovery for a network terminates when all links are discovered. The overall time is called neighbor discovery time, denoted by T. We want to minimize T. Also, for a given algorithm, we want to analyze T.

Before we analyze neighbor discovery time, we first give a formal definition on an algorithm for neighbor discovery.

Definition 1. *An algorithm for neighbor discovery in* multi-channel MPR *networks is defined as follows:*

- *Input: number of channels k;*
- *Algorithm: at each time slot, nodes transmit in channel c, or listen in all k channels;*
- *Output: set of neighbor nodes*

A deterministic algorithm is an algorithm that has deterministic channel c at each time slot. A randomized algorithm is defined over all deterministic algorithms with a given distribution, which can be achieved by tossing coins to determine channel at each time slot. We say the output of an algorithm is correct if the algorithm finds all the neighbors for each node. If the output of a randomized algorithm is correct but its running time is a random variable, we call it Las Vegas randomized algorithm. Otherwise, if the output has some error but its running time is fixed, we call it Monte Carlo randomized algorithm. Obviously, given a randomized algorithm, *the neighbor discovery time is equal to the running time of this algorithm*. Then, we give a theorem about the lower bound of running time for all randomized algorithms in single-hop networks.

Theorem 1. *The lower bound of expected neighbor discovery time for all randomized algorithms is $\Theta(\frac{n}{k})$ time slots in single-hop networks of n nodes with k orthogonal channels.*

Proof: We prove this theorem using *Yao's minmax principle* [13], which states that the expected running time of the optimal (Las Vegas) randomized algorithm is larger than the expected running time of the optimal deterministic algorithm on the input generated by a distribution P. Here we consider the running time of the optimal deterministic algorithm for an input n and k. For simplicity, we assume $n = k * m$, where m is an integer. For any deterministic algorithm, each node has to transmit at least one time. So at least $\frac{n}{k}$ time slots are needed, i.e., the lower bound. Then we propose a deterministic algorithm with running time $\frac{2n}{k}$, which means the running time

of optimal deterministic algorithm is upper bounded by $\frac{2n}{k}$. Thus, the running time of the optimal deterministic algorithm is $\Theta(\frac{n}{k})$. In other words, the expected running time of all Las Vegas randomized algorithms is at least $\Theta(\frac{n}{k})$.

The proposed deterministic algorithm runs as follows: first, we use $m = n/k$ time slots to make k nodes transmit at each time slot, since we have k channels. Then we can make k nodes transmitting at the same time slot into a group, since only the group member does not discover each other. Note that there are m groups, and then we need k slots to schedule group discovery since a group has k members. Because we allow k transmissions at each time slot, we can select k groups from m groups to transmit at each time slot. In other words, we need to schedule m/k phases consequently, and each phase needs k time slots. Therefore, the running time is $k\frac{m}{k} = m = \frac{n}{k}$. The overall running time is $\frac{n}{k} + \frac{n}{k} = \frac{2n}{k}$. ■

IV. Proposed Randomized Algorithms

Consider Theorem 1, the lower bound is achieved under centralized scheduling. Here we propose two randomized algorithms, and consider the performance in single-hop networks. We first assume that the number of nodes in network n is known beforehand. Then we relax the assumption and discuss the distributed implementation in Section V.

A. Las Vegas Algorithm

Here we give a simple Las Vegas algorithm (named Algorithm 1): at each time slot, nodes choose a channel to transmit with probability p, and listens with probability $1 - kp$. Note that there are k channels. The algorithm terminates when all $n(n-1)$ links are discovered. Then we have a theorem:

Theorem 2. *Algorithm 1 guarantees that the neighbor discovery time is $\Theta(\frac{n}{k}\ln n)$ time slots with high probability[1] in single-hop networks.*

Proof: Consider two arbitrary nodes i and j. Let p_s denote the probability that i is discovered by j in a given time slot. Note that the discovery can happen in any channel. Therefore the link discovery probability

$$p_s = kp(1-kp)(1-p)^{n-2} = \frac{k}{n}(1-\frac{k}{n})(1-\frac{1}{n})^{n-2}. \quad (1)$$

Let T be the neighbor discovery time, and T_{ij} denote the time that node i is discovered by j. We next show that $\exists d_1$, $\exists d_2$ and $d_1 > d_2 > 0$, $Pr(T > d_1\frac{n\ln n}{k}) \to 0$, and $Pr(T > d_2\frac{n\ln n}{k}) \to 1$. In other words, $T = \Theta(\frac{n\ln n}{k})$ with high probability.

1) Upper Bound: Let T_i be the time that node i finds all neighbors. Then $T = \max_i\{T_i\}$ and $T_i = \max_j\{T_{ij}\}$. Because all nodes are symmetric, by *union bound*, we have

$$Pr(T > t) \le n^2 Pr(T_{ij} > t).$$

The probability that link (i,j) is not discovered by time slot t is given by

$$\Pr(T_{ij} > t) = (1-p_s)^t = (1 - \frac{k}{n}(1-\frac{k}{n})(1-\frac{1}{n})^{n-2})^t.$$

[1]In this paper, an event happens with high probability (w.h.p.) means that the probability tends to 1 as n tends to infinity.

Let $t = d_1\frac{n\ln n}{k}$, where $d_1 = e\beta_1$ ($\beta_1 > 2$). Because $1 + x \le e^x$, we have

$$\Pr(T > \frac{d_1 n\ln n}{k}) \le n^2 e^{-(1-k/n)(1-1/n)^{n-2}d_1\ln n}.$$

Let $\alpha = \frac{n-k}{n-1}$. Because $\beta_1 > 2$ and $(1-1/n)^{n-1} \ge 1/e$ [3], we have

$$\Pr(T > \frac{d_1 n\ln n}{k}) \le n^2 e^{-\beta_1\alpha\ln n} = n^{2-\alpha\beta_1} \to 0$$

as $n \to \infty$ and $d_1 > 2e$. In other words, $T = O(\frac{n\ln n}{k})$ with high probability.

2) Lower Bound: Let T' denotes the time when all nodes have transmitted at least once. Then it is trivial that $T \ge T'$, and $\Pr(T > t) \ge \Pr(T' > t)$. Because $(1-x)^t \le e^{-tx}$, the probability that each node has transmitted at least once by time slot t is given by

$$\Pr(T' > t) = (1-(1-kp)^t)^n \ge (1-e^{\frac{-kt}{n}})^n,$$

where $1-(1-kp)^t$ is the probability that a node transmits at least once by time slot t. Let $t = d_2\frac{n\ln n}{k}$, where $d_2 > 1$. So

$$\Pr(T' > t) \ge (1-e^{-d_2\ln n})^n = (1-n^{-d_2})^n.$$

For $d > 1$, $\lim_{n\to\infty}(1-n^{-d})^n = 1$. In other words, $\Pr(T > \frac{d_2 n\ln n}{k}) \to 1$ as $n \to \infty$ and $d_2 > 1$. Because $d_1 > 2e$, it is easy to set $d_1 > d_2$. Therefore, $T = \Theta(\frac{n\ln n}{k})$ with high probability. The proof is complete. ■

Our result is the same with *multi-antenna* MPR model [5], which means that even with single-channel collision constraint, the performance is not degraded with k-MPR capability. Compared with the result of $\Theta(n\ln n)$ for single-channel networks [3], MPR models get a better result due to the improved link discovery probability p_s from $\frac{1}{ne}$ to $\frac{k}{ne}$. In other words, the improvement is from k-MPR capability.

B. Monte Carlo Algorithm

Compared with the lower bound for all randomized algorithms, there still has a gap $\Theta(\ln n)$. The design of better Las Vegas algorithm is left for future work. However, if we allow some error of algorithm output, we can approach the lower bound. In this section, we give a Monte Carlo algorithm, which has the same order with lower bound for all algorithms.

It is worth mentioning that we do not assume feedback model, because 1) it is hard to implement collision detection in wireless networks due to the attenuated wireless signal with distance; 2) for control packets, which are always broadcast packets, it is hard to decide who should be responsible for replying ACK. If all reply, ACKs are collided. However, we can still utilize some mechanisms to improve the performance, e.g., 1) normal control packets embedding neighbor information can serve as implicit ACKs; 2) MPR capability allows several simultaneously ACKs. This area is left for future work.

Our Monte Carlo algorithm is similar to Zeng et al. [5]. First we refer to a node that only listens as passive, otherwise active. At the beginning, all nodes are active. Algorithm 2 is defined as follows: time is divided into phases; In phase i, there are n_i nodes, and the time slot is $W_i = \eta\frac{n_i}{k} = \Theta(\frac{n_i}{k})$ ($\eta > 17$);

the transmission probability in each channel is $p_i = 1/n_i$. At the end of a phase, nodes that transmit three times turn into passive. The phases continue until there are at most $\frac{n}{\ln n}$ active nodes left. Then we run Algorithm 1 defined in last section $d\frac{\frac{n}{\ln n}}{k}\ln\frac{n}{\ln n} = O(\frac{n}{k})$ $(1 < d < 2e)$ time slots.

We will prove that with proper W_i and p_i, at least half of the active nodes in a phase will become passive at the end of this phase. The result is shown in Lemma 1. Because all phases are independent and similar, for simplicity, we only consider a single phase and omit the phase notation of all symbols.

Lemma 1. *Let S denote the set of passive nodes at the end of a phase, and n denote the initial number of nodes. Let $W = \eta n/k$ and $p = 1/n$. When $\eta > 17$, we have $\Pr(|S| < n/2) < e^{-n/k}$.*

Due to space limitation, we only give the key idea of the proof. The details can be found elsewhere [5], [16]. The key idea is that: $|S|$ is associated with a truncated number of successful transmissions Z within W time slots, while Z can be constructed using a Doob Martingale [14] with respect to the increased number of successful transmissions in each time slot. We first bound $E[Z]$ and then use *Method of Bounded Difference* [14] to get the final result.

Then we will show that Algorithm 2 has correct output with high probability, which is stated as follows.

Lemma 2. *Algorithm 2 gives correct output with high probability.*

Proof: Our algorithm has two sections. For the second section, according to Theorem 2, Algorithm 1 has correct output within $O(\frac{n}{k})$ with high probability. Thus we focus on section one. Remember that we assume the definition of three times as threshold. We will prove that it guarantees the network neighbor discovery with high probability.

Consider a single node, the probability that there exists nodes who never find this transmission node all three time is given by $(kp)^3$ where p is the probability for transmission in single channel, which means that such node also transmits all three times, no matter in the same channel or not. In our algorithm, $p = 1/n$. Therefore, the probability that all nodes discover this node is larger than $1-n(kp)^3 = 1-k^3/n^2$ due to the *union bound*. Given fixed k, this event happens with high probability. Note that we adopt this standard for all nodes. When all nodes have transmitted three times, the neighbor discovery finishes with probability $(1 - O(\frac{1}{n^2}))^n$, still with high probability, because $\lim_{n\to\infty}(1 - n^{-2})^n = 1$.

Because each stage decreases the number of active nodes at least half, we have m stages from n active nodes to at most $\frac{n}{\ln n}$ active nodes, i.e., $m \le \log\ln n$. The correct probability is given by $P = \prod_{i=1}^{m}(1 - e^{-n_i/k})$. For all i, $n_i \ge n/\ln n$, so

$$P \ge (1 - e^{-\frac{n}{k\ln n}})^{\log\ln n} \ge (1 - e^{-\frac{n}{k\ln n}})^{\frac{\ln\ln n}{\ln 2}}.$$

Because $\lim_{n\to\infty}(\ln\ln n)\ln(1 - e^{-n/(k*\ln n)}) = 0$, $P \to 1$. The proof is accomplished. ∎

Now we analyze the running time of Algorithm 2. Because section 1 has at most $m \le \log\ln n$ rounds, the overall running time of Algorithm 2 is $\sum_{i=0}^{m-1}\Theta(\frac{n}{2^i k}) + \Theta(\frac{\frac{n}{\ln n}}{k}\ln\frac{n}{\ln n}) = \Theta(\frac{n}{k})$. Combining Lemma 1 and 2, we have a randomized algorithm with running time $\Theta(\frac{n}{k})$, which has correct output with high probability. Therefore, we have:

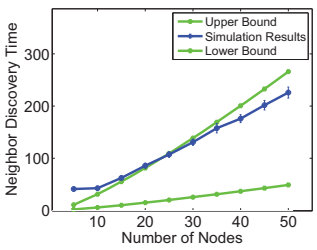

Fig. 1. Evaluation of Algorithm 1

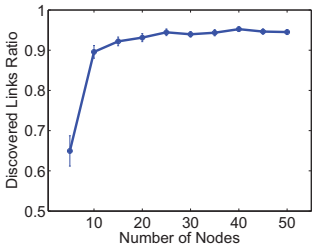

Fig. 2. Evaluation of Algorithm 2

Theorem 3. *Algorithm 2 guarantees the neighbor discovery time is $\Theta(\frac{n}{k})$ time slots in single-hop networks, and the output is correct with high probability.*

C. Simulation Evaluation

In order to validate our theoretical results, we run simulations in single-hop networks. We assume n is known beforehand, and $k = 5$. The effect of algorithms under unknown scenarios is validated in [3], [4], so we omit the verifications. The results are shown in Figures 1 and 2. For algorithm 1, we measure the neighbor discovery time, and compared it with derived upper bound and lower bound in Theorem 1, where $d_1 = 1$ and $d_2 = 2e$. For algorithm 2, the metric is the ratio of discovered link over all links, and $\eta = 20$. The number of nodes increase from 5 to 50. Also we repeat the simulations 100 times, and draw 95% confidence intervals around the average. As we can see from Figure 1, running time is between the upper bound and lower bound, closer to upper bound, especially when n is large. For Figure 2, the discovery ratio turns out to be very close to 1 when n is not small. Note that the theoretical results are not accurate when n is small. This is predicable since our results are derived by asymptotic analysis. The simulation results match the theoretical results especially when the number of nodes increases.

V. Discussions

A. Distributed Implementation

We first discuss the distributed implementation of Las Vegas algorithm. Currently, we use $p = 1/n$, where n is the global information. A standard approach [3] can be used to handle unknown n case, which behaves like this:

- Run algorithm in stages. In stage i, $p = \frac{1}{2^i}$ and the running time is $\frac{d2^i\ln 2^i}{k}$. There must be a stage t satisfying $2^t \ge n$. Then at stage t, the above analysis works.
- We also need a stopping rule. Otherwise it will run all the time. The stopping rule can be: stop at stage $i+1$, when $N^i \ge 2^{i-1}$ and $N^{i+1} < 2^i$, where N^i is the number of nodes found in stage i.

We call this algorithm as Algorithm 3.

Then it can be proved that the upper bound of the overall time is a factor of two slot-down [3], [5]. The insight is that the sum of time slots before stage t is actually less than stage t. The discussion also extends to Algorithm 2 , because in each phase, we can use the same technique. We also prove that the stopping rule is sufficient. Assume $n = 2^m + k$, where $0 < k \leq 2^m$. Stage $m+2$ satisfies the stopping rule with high probability, because at stage $m+1$, the transmission probability $p_s \approx \frac{k}{2^{m+1}}(1-\frac{1}{2^{m+1}})^{2^{m+1}-1} \geq \frac{k}{2^{m+1}e} \geq \frac{k}{ne}$. Then $\{N^{m+1} = n \geq 2^m\}$ happens with high probability. For stage $m+2$, $p_s \approx \frac{k}{2^{m+2}}(1-\frac{1}{2^{m+2}})^{2^m+k-1} \geq \frac{k}{2^{m+2}e} \geq \frac{k}{ne}$. Then $\{N^{m+1} = n \leq 2^{m+1}\}$ happens with high probability. So the claim is proved. For stage 1 to m, we cannot guarantee $p_s \geq \frac{k}{ne}$, so the condition cannot be satisfied with high probability.

Note that Algorithm 3 can be applied to the multi-hop case, since it is totally distributed and decreases the probability geometrically until entering the desired time slot. In multi-hop scenario, the desired time slot is the slot t where $2^t \geq \Delta$ and Δ is the maximal node degree of the network. Therefore, the time bound is $O(\frac{\Delta}{k} ln\Delta)$. Previous works [3], [5] demonstrate that the clique analysis results apply in multi-hop networks through simulations. Hence we omit the validation here.

B. Comparison with Other Models

Fixed Channel Usage Approach: A natural idea is to use fixed channel to transmit neighbor discovery packets, while the channels are selected beforehand. Consider uniform deterministic usage, which can be achieved by hashing ID to channel or greedy selection, we can have $\Theta(\frac{n}{k}\ln\frac{n}{k})$ by applying above analysis. If we assume random selection and fixed usage, we can prove that the lower bound is still $\Theta(\frac{n}{k}\ln\frac{n}{k})$. We first classify k sets, where set i includes nodes choosing channel i. The neighbor discovery time for the clique is larger than the time for any set, i.e., $\Pr(T > t) \geq \Pr(T' > t)$, where T' is the discovery time for any set. Consider the maximal set where the numbers of nodes are larger than n/k, the lower bound turns out to be $\Omega(\frac{n}{k}\ln\frac{n}{k})$ by applying above analysis. The worst case for this strategy is that all nodes select one channel, then the result is $O(\frac{n}{k}\ln n)$. Therefore, the discovery time for this strategy is $\Omega(\frac{n}{k}\ln\frac{n}{k})$ and $O(\frac{n}{k}\ln n)$ w.h.p..

Multi-Channel Wireless Networks: Multi-channel wireless networks mean that the reception only happens in single-channel, i.e., no MPR capability. It is common in multi-channel sensor networks [15] and cognitive radio networks [6]. Nodes first have to choose a channel, then decide to transmit or listen. The probability selecting a channel is given by $1/k$ for k channels. Then the probability to transmit is p. Let p_s be the link discovery probability. Then $p_s = k\frac{p}{k}\frac{1}{k}(1-p)(1-\frac{p}{k})^{n-2}$ for single-hop networks. We can make an approximation that $p_s \approx 1/ne$. Applying above analysis, we have the result that the neighbor discovery time is upper bounded by $O(n\ln n)$, and lower bounded by $\Omega(\frac{n}{k}\ln n)$ with high probability.

Channel Unreliability: Note that we use link-centric approach instead of node-centric approach used in [3], [4] to define p_s, because in our *multi-channel* MPR scenario a node has to transmit multiple times to be discovered by all its neighbors. One of the advantages of link-centric model is that it can handle channel unreliability case since channel unreliability means several transmissions for a link. In particular, we can integrate a variable p_l into Equation (1) to form new p_s. It just adds a factor, and does not affect the result of the analysis. If links have diverse propagation characteristics, i.e., each link has different p_l, we can still integrate link probability in each p_s. In that case, we need a more elaborate analysis.

VI. Conclusion

In this paper, we revisit the neighbor discovery problem in *multi-channel* MPR networks, and analyze the time bound required to discovery all neighbors. We first show the lower bound for all randomized algorithms to be $\Theta(\frac{n}{k})$. For *multi-channel* MPR single-hop network of n nodes and k channel, a Las Vegas algorithm is proposed, and its running time is $\Theta(\frac{n}{k}\ln n)$ if n is known beforehand. In order to approach the lower bound, we give a $\Theta(\frac{n}{k})$ Monte Carlo algorithm, whose output is correct with high probability. Then we extend both algorithms to distributed and multi-hop scenarios, and make comparisons with other models. In the future, we want to complete our work by considering implicit feedback scenario and integrating heterogeneous link loss probability.

References

[1] A. Keshavarzian, E. Uysal-Biyikoglu, F. Herrmann, and A. Manjeshwar, "Energy-efficient Link Assessment in Wireless Sensor Networks", in Proc. INFOCOM, 2004.

[2] M.J. McGlynn and S.A. Borbash, "Birthday protocols for low energy deployment and flexible neighbor discovery in ad hoc wireless networks", in Proc. MobiHoc, 2001, pp.137-145.

[3] S. Vasudevan, D.F. Towsley, D. Goeckel, and R. Khalili, "Neighbor discovery in wireless networks and the coupon collector's problem", in Proc. MOBICOM, 2009, pp.181-192.

[4] L. You, Z. Yuan, P. Yang, and G. Chen, "ALOHA-like Neighbor Discovery for Low-Duty-Cycled Wireless Sensor Networks", in WCNC, 2011.

[5] W. Zeng, X. Chen, A. Russell, S. Vasudevan, B. Wang and W. Wei, "Neighbor Discovery in Wireless Networks with Multipacket Reception", in MobiHoc, 2011.

[6] N. Mittal, Y. Zeng, S. Venkatesan, and R. Chandrasekaran, "Randomized Distributed Algorithms for Neighbor Discovery in Multi-Hop Multi-Channel Heterogeneous Wireless Networks", in ICDCS, 2011.

[7] D. Angelosante, E. Biglieri, and M. Lops, "Neighbor discovery in wireless networks: A multiuser-detection approach", Physical Communication, vol. 3, no. 1, pp. 28-36, 2010.

[8] L. Zhang, J. Luo, and D. Guo, Compressed neighbor discovery for wireless networks, preprint, http://arxiv.org/abs/1012.1007.

[9] X. Wu, S. Tavildar, S. Shakkottai, T. Richardson, J. Li, R. Laroia and A. Jovicic, FlashLinQ: A Synchronous Distributed Scheduler for Peer-to-Peer Ad Hoc Networks, in Allerton, 2010.

[10] K. Tan, J. Fang, Y. Zhang, S. Chen, L. Shi, J. Zhang, and Y. Zhang, "Fine-grained channel access in wireless LAN", in Proc. SIGCOMM, 2010, pp.147-158.

[11] J. Ni, R. Srikant, and X. Wu, "Coloring spatial point processes with applications to peer discovery in large wireless networks", in Proc. SIGMETRICS, 2010, pp.167-178.

[12] N. Karowski, A. C. Viana, and A. Wolisz, "Optimized Asychronous Multi-channel Neighbor Discovery", in Proc. Infocom, 2011.

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

[14] M. Mitzenmacher and E. Upfal. Probability and Computing: Randomized Algorithms and Probabilistic Analysis. Cambridge University Press, 2005.

[15] L. Tang, Y. Sun, O. Gurewitz, and D. B. Johnson, "EM-MAC: A Dynamic Multichannel Energy-Efficient MAC Protocol", in Proc. MobiHoc, 2011.

[16] L. You, X. Zhu and G. Chen, Neighbor Discovery in Peer-to-Peer Wireless Networks with Multi-Channel Multi-Packet Reception Capability, Technical Report, Nanjing University, 2011.