Time-Efficient Protocols for Neighbor Discovery in Wireless Ad Hoc Networks

Guobao Sun, *Student Member, IEEE*, Fan Wu, *Member, IEEE*, Xiaofeng Gao, *Member, IEEE*, Guihai Chen, *Member, IEEE*, and Wei Wang, *Member, IEEE*

Abstract—Neighbor discovery (ND) is a basic and crucial step for initializing wireless ad hoc networks. A fast, precise, and energy-efficient ND protocol has significant importance to subsequent operations in wireless networks. However, many existing protocols have a high probability of generating idle slots in their neighbor discovering processes, which prolongs the executing duration, thus compromising their performance. In this paper, we propose a novel randomized protocol FRIEND, which is a prehandshaking ND protocol, to initialize synchronous full-duplex wireless ad hoc networks. By introducing a prehandshaking strategy to help each node be aware of activities of its neighborhood, we significantly reduce the probabilities of generating idle slots and collisions. Moreover, with the development of single-channel full-duplex communication technology, we further decrease the processing time needed in FRIEND and construct the first full-duplex ND protocol. Our theoretical analysis proves that FRIEND can decrease the duration of ND by up to 48% in comparison with classical ALOHA-like protocols. In addition, we propose HD-FRIEND for half-duplex networks and variants of FRIEND for multihop and duty-cycled networks. Both theoretical analysis and simulation results show that FRIEND can adapt to various scenarios and significantly decrease the duration of ND.

Index Terms—Full-duplex technology, neighbor discovery (ND), randomized algorithm, wireless ad hoc networks.

I. INTRODUCTION

WIRELESS ad hoc networks have attracted a lot of interest from both academia and industry due to their wide range of applications. In many scenarios, nodes are deployed without the support of preexisting infrastructures for communication. As a result, nodes in a wireless ad hoc network need to configure themselves through their own communication

activities to form a reliable infrastructure during initialization for further operations. For each node, knowledge of its one-hop neighbors (the nodes with which can directly communicate) has significant importance to the upper layer protocols such as MAC protocols, routing protocols, etc. Consequently, *neighbor discovery* (ND) is designed to discover a node's one-hop neighbors and, thus, is momentous and crucial for configuring wireless networks.

Compared with existing deterministic [11] and multiuser detection-based [12] protocols, randomized protocols are most commonly used to conduct an ND process in wireless networks [3]–[8]. In those protocols, each node transmits at different randomly chosen time instants to reduce the possibility of collision with other nodes. Usually, researchers discuss ND protocols under a synchronous system and focus on a clique with n nodes, e.g., the famous birthday protocols [3]. In birthday protocols, at each single slot, every node independently chooses to transmit a discovery message by probability p and listen by probability $1 - p$. (The optimal value of p is proven to be $1/n$.) By reducing the ND problem to *coupon collector's problem* [16], Vasudevan *et al.* [4] proved that the upper bound of the expected time of a birthday protocol is neH_n, where H_n is the nth harmonic number. Much subsequent research on ND are based on birthday protocols. For example, Vasudevan *et al.* [4] proposed solutions to scenarios for unknown neighbor numbers, asynchronous systems, and systems with reception status feedback mechanisms. Zeng *et al.* [5] discussed the performance of birthday protocols with multipacket reception (MPR). You *et al.* [8] discussed discovery time's upper bound when nodes have a low duty cycle by reducing the problem to k *coupon collector's problem*.

However, the family of birthday protocols has a vital drawback. The probability of generating an idle slot is given by

$$p_0 = \left(1 - \frac{1}{n}\right)^n.$$

When $n = 10$, $p_0 \approx 0.349$. When $n \to +\infty$, $p_0 \to 1/e \approx 0.368$. Therefore, when the number of nodes is large, the probability that no node transmits in a slot is about 37%. We must point out that the probability that there is only one node transmitting is

$$p_1 = \binom{n}{1} \frac{1}{n} \left(1 - \frac{1}{n}\right)^{n-1} \geq \frac{1}{e}.$$

The last inequality comes from Lemma 2, which we will present in the succeeding sections. We can see that the probability of idle slots is as large as the probability that a node

successfully transmits its discovery message, and they contribute about 73% to all possible scenarios. Furthermore, the probability of collisions also increases the iterations running in the protocols. For instance, two nodes simultaneously transmitting in a slot has a probability of $1/(2e) \approx 0.184$, and three nodes simultaneously transmitting has a probability of $1/(6e) \approx 0.06$.[1] Comparing the relatively small probability of collisions, the idle-slot probability is unacceptably high. If we can effectively reduce the probability of idle slots, ND time will be tremendously reduced. Fortunately, with the development of full-duplex wireless communication technology [1], [2], we can design more time-efficient protocols, i.e., protocols that consume less time, to cope with this issue if nodes can simultaneously transmit and receive in a single slot.

Our key idea is twofold. On one hand, we introduce a prehandshaking strategy to help each node be aware of activities of its neighborhood before normal transmissions, such that the system can have higher probabilities to avoid collisions and idle slots. To conduct this prehandshaking, we add some tiny subslots before each normal slot. With the help of full-duplex technology, at each subslot, every node will decide whether to transmit the discovery message in a normal slot by transmitting an anonymous election signal and catch its neighbors' signals simultaneously. With different transmitting–receiving scenarios, we design an effective strategy for each node to determine how to behave in normal slots. Correspondingly, we assign the behaviors of each node in the normal slots to complete the ND process. On the other hand, the reception status feedback mechanism is ameliorated by using full-duplex wireless radios. Originally, in [6], a subslot is added after the normal slot, and receivers will give feedback signals to transmitters in this subslot. In our design, this overhead can be eliminated by using full-duplex nodes. If a receiver finds that two or more nodes are simultaneously transmitting, it will transmit a warning message immediately to inform other transmitters of the failure in their transmissions.

Our contributions in this paper are listed as follows.

- We propose a novel ND protocol named FRIEND, which is a protocol based on prehandshaking activities, in which prehandshaking activities are inserted before each normal slot. In FRIEND, we avoid the vital drawback of traditional birthday protocols and reduce the probabilities of collisions and idle slots. Other existing protocols based on birthday protocols can be easily ameliorated with our design, such as the protocols proposed in [5] and [8].
- To the best of our knowledge, we are the first to consider the issue of ND with full-duplex technology. For such a long time, research on the ND problem in wireless networks is based on half-duplex nodes. Full-duplex technology enables nodes to simultaneously transmit and receive, which can be utilized to accelerate the ND process. Along with the emergence of full-duplex technology, we can optimistically predict the transition from half-duplex to full-duplex nodes, which implicates the significance of our design.

- We extend the discussion of FRIEND to multihop networks and show that FRIEND still performs better than the ALOHA-like protocol.
- Furthermore, we propose another strategy named HD-FRIEND for nodes with half-duplex radios. Our theoretical analysis shows that HD-FRIEND decreases time for the ND process by approximately 36.7%.
- Finally, we discuss how to improve the ND process in low-duty-cycle networks and propose methods to handle the problem of when to start and terminate ND when n is unknown to nodes.

The rest of this paper is organized as follows: Section II describes our model and assumptions. Section II introduces FRIEND and its theoretical analysis. Section V gives some discussion and extensions, and in Section VI, we evaluate FRIEND by simulations. In Section VII, we present related works. This paper concludes with our future works in Section VIII.

II. NETWORK MODEL AND ASSUMPTIONS

Here, we introduce the network model and several assumptions, under which we will present our FRIEND protocol and corresponding analysis. These assumptions are reasonable in the research on ND, and many former works, including the traditional ALOHA-like protocols, are also based on similar assumptions [3]–[5], [8]. Our assumptions are listed as follows.

- Each node has a unique ID (e.g., the MAC address).
- Time is identically slotted, and nodes are synchronized on slot boundaries. The synchronization can be achieved by different techniques, and many works have focused on this problem (e.g., [19], [20], and [27]).
- All nodes are in a clique of size n.
- n is known to all nodes in the clique. Typically, n can be preconfigured on nodes before deploying or calculated based on the density of the network. The preconfigured or calculated result does not need to be exactly accurate, because a small difference only has little influence on node decisions about transmission probabilities and can be normally ignored.
- Nodes use omnidirectional antennas, and all nodes have the same transmission range.
- No MPR technique is used, i.e., for a node that is receiving, a collision occurs when two or more nodes simultaneously transmit packets to it in a slot.
- Nodes can listen and transmit on the same channel simultaneously.
- Nodes can distinguish between collisions and idle slots.

We also neglect possible errors caused by fading. Hence, for two nodes A and B, if A transmits without collisions in a slot and B is within the transmission range of A, then B can receive the packet with no error.

III. FRIEND: PREHANDSHAKING PROTOCOL

Here, we present our novel protocol FRIEND based on the assumptions in Section II and analyze its performance theoretically. First, in Section III-A, we add one tiny subslot before each

---

[1]Lemma 1 in Section III points out that the probability of n nodes transmitting in a slot is $1/(n!e)$.
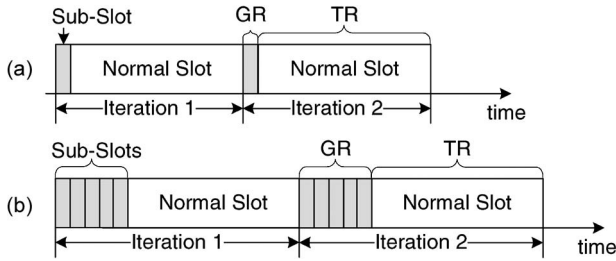
Fig. 1. Description of an iteration. GR is used for prehandshaking, and TR is used for transmitting discovery messages. In (a), there is one subslot in GR, whereas in (b), there are multiple subslots in GR.

normal slot and complete our design for the prehandshaking process. Next, in Section III-B, we extend our idea for the prehandshaking process by introducing more subslots before the normal slot and design the corresponding variation of FRIEND. Moreover, in Section III-C, we discuss in detail how many subslots should be used for the prehandshaking process to achieve the best performance with the least overhead.

A. FRIEND With Single Subslot for Prehandshaking

As mentioned in Section I, for each normal slot, we insert a subslot before it to perform the prehandshaking process. We name this combination as an *iteration*. (It can be also considered as a "big slot.") Let GR be the greeting process and TR be the transmission process in one iteration. Note that the length of a subslot can be as short as 1 bit, since we do not care what a node transmits and only need to know whether the signals exist or not. Vasudevan *et al.* [4] also adopted this assumption. Let M_s be such kind of messages, which means an anonymous election signal with short duration. The normal slot is used to exchange discovery messages, which may contain nodes' IDs or MAC addresses. The size of a subslot is significantly smaller than that of a normal slot (It is mentioned in [18] that the size of a slot can be about 10 bytes.) Thus, the overhead caused by subslots is almost negligible. We define this kind of discovery messages as M_d.

Fig. 1 shows the combination of subslots and normal slots. In Fig. 1(a), we insert one subslot for one normal slot, whereas in Fig. 1(b), we insert multiple subslots before one normal slot to further increase the probability of successful transmissions, as will be discussed in Section III-B.

We are now ready to present our FRIEND protocol to determine the action of a node in a slot. FRIEND is a distributed protocol, and for each node, the target is to discover all its neighbors after finite iterations. Assume that we are considering a clique of n nodes. We divide FRIEND into two subroutines: FRIEND-GR and FRIEND-TR.

Let us describe the main idea of FRIEND-GR: the prehandshaking process. At the beginning of a subslot, each node should determine its action in the following normal slot. The purpose is to find a subset of nodes in the network to send M_d without collisions. Algorithm 1 describes the details of FRIEND-GR. Note that each node should run a copy of FRIEND-GR. To simplify our description, assume that we run FRIEND-GR on node A. Recall that M_s is the election signal

and that M_d is the discovery message. Define A_f as a flag variable to indicate whether A has successfully sent M_d. If $A_f = 0$, then A has to send M_d successfully in one of the following iterations; otherwise, A will keep silent and only receive messages. Initially, $A_f = 0$. Define A_n as the number of undiscovered neighbors of A. Initially, A_n should be $n - 1$, and we let $A_n = n$ for simplicity of later discussion.

Algorithm 1 FRIEND-GR (Prehandshaking)
1: If $A_f = 1$ **then**   ▷ A has successfully sent M_d.
2:   A will keep silent in TR and exit.
3: **end if**
4: Node A decides to send M_s by probability $1/A_n$ and keep listening by probability $1 - 1/A_n$.
5: **if** A sends M_s **then**   ▷ A hopes to send M_d in TR.
6:   **if** A does not receive M_s during GR **then**
7:     A will transmit M_d in TR;
8:   **else**   ▷ A receives M_s from other nodes
9:     A will transmit M_d in TR by probability $1/2$.
10:   **end if**
11: **else**   ▷ A does not send M_s
12:   **if** A does not receive M_s during GR **then**
13:     A will transmit M_d in TR by probability $1/A_n$;
14:   **else**   ▷ A receives M_s from other nodes
15:     A will keep silent in TR.
16:   **end if**
17: **end if**

In FRIEND-GR, each node decides to send M_s by probability $1/A_n$ or keep silent by probability $1 - 1/A_n$. (The values of probabilities are chosen to be optimal according to [3].) Next, we face two cases as follows.

1) If A sends M_s (lines 5–10), it implies that A hopes to send M_d in TR.
   a) At this moment, if A does not receive M_s during GR, it means that A wins the election and will definitely send M_d in the following TR.
   b) If A receives M_s, it means that there exist other candidates within A's direct communication range. Therefore, A can only send M_d by probability $1/2$. (We will explain the reason for setting probability $1/2$ after the proof of Lemma 1.)
2) If A does not send M_s (lines 11–17), it implies that A hopes to keep silent in the following TR.
   a) At this moment, if A does not receive M_s in GR, it means that no nodes decide to send M_d in TR. A will reconsider sending M_d by probability $1/A_n$.
   b) If A receives M_s, it means that there are nodes intending to transmit, and thus, A will keep silent.

When FRIEND-GR is finished, we enter the TR and start the process of neighbor discovering. Next, we run FRIEND-TR, i.e., the neighbor discovering process, and the detailed description is shown in Algorithm 2.

Algorithm 2 FRIEND-TR (Neighbor Discovering)

1: **if** A plans to send M_d **then**
2:   A sends M_d and monitors the channel meanwhile.
3:   **if** A does not receive M_d during TR **then**
4:    $A_f = 1$.   ▷ A will keep silent from now on
5:   **else**   ▷ A receives M_d from other nodes
6:    Current iteration is invalid.
7:   **end if**
8: **else**   ▷ A does not plan to send M_d
9:   A keeps listening.
10:   **if** A does not receive M_d during TR **then**
11:    Current iteration is invalid.
12:   **else if** A receives a single M_d **then**
13:    Record the ID in M_d.
14:    $A_n = A_n - 1$.   ▷ A records one of its neighbors.
15:   **else**   ▷ There is a collision at A
16:    Current iteration is invalid.
17:   **end if**
18: **end if**

In FRIEND-TR, there are two scenarios as follows.

1) If A does not receive M_d during TR, it means that A's transmission is successful. Consequently, A will keep silent during the rest of the ND process.
   a) If A receives M_d from other nodes, it means that the current transmission failed.
   b) If A does not send M_d, A will check the number of transmitters (lines 8–18).
2) If A does not receive M_d during TR, it implies that no nodes send M_d in TR. Therefore, the current iteration is invalid.
   a) If A receives a single M_d during TR, it means that there is one node successfully transmitting its M_d. A will record the ID in M_d and decrease the value of A_n by 1.
   b) If there is a collision at A, it means that the current transmission failed.

We will keep running FRIEND-GR and FRIEND-TR in turn until $A_n = 1$. Now, we finish the description of FRIEND and start the discussion about the performance of this protocol.

We denote the probability that a node successfully transmits its M_d without collisions in TR by P_1. We now begin to analyze the expected time needed to discover all nodes with high probability with two lemmas.

Lemma 1: When all nodes independently transmit by probability $1/n$, the probability that k nodes simultaneously transmit in a single slot is given by $p_k = (1/k!e)$ while $n \to +\infty$.

Proof: Since nodes choose their actions independently, the probability that k nodes simultaneously transmit in a slot with clique size n is given by $p_{n,k} = \binom{n}{k}(1/n)^k(1 - (1/n))^{n-k}$. When $n \to +\infty$, we use Poisson distribution to replace binomial distribution. Hence, $p_k = \lim_{n \to +\infty} p_{n,k} = (e^{-\lambda}\lambda^k/k!)$ with $\lambda = n \cdot (1/n) = 1$. Thus, the result holds. ∎

From Lemma 1, we can see that the probability that three or more nodes simultaneously transmit in a subslot is so small that it is acceptable to ignore it and assume that there are only two

nodes transmitting when collision occurs to simplify the design of FRIEND since it is hard and also unnecessary to infer the exact number of transmitting nodes, which explains line 9 in Algorithm 1.

Lemma 2: $(1 - (1/n))^{n-1} \geq (1/e) \forall n = 2, 3, \ldots$

This lemma is just the same as [4, Lemma 1].

We then use these two lemmas to evaluate the probability of a successful discovery in an iteration.

Theorem 1: When there are n nodes in a clique and all nodes run FRIEND, the probability that a node successfully transmits M_d in TR is bounded by

$$P_1 \geq \frac{1}{e^2}\left(1 - \frac{1}{n}\right) + \frac{5}{4e}.$$

Furthermore, when $n \to +\infty$

$$P_1 \geq \frac{1}{e^2} + \frac{5}{4e}. \tag{1}$$

Proof: We analyze different events that may occur in GR. If no one sends M_s in GR, all nodes will reconsider their actions. The successful event's (only one node transmits in TR) probability is

$$p_0 = \left(1 - \frac{1}{n}\right)^n \binom{n}{1}\frac{1}{n}\left(1 - \frac{1}{n}\right)^{n-1} = \left(1 - \frac{1}{n}\right)^{2n-1}.$$

If there is exactly one node sending a signal in GR, no collisions will occur in TR. Therefore, the probability is

$$p_1 = \binom{n}{1}\frac{1}{n}\left(1 - \frac{1}{n}\right)^{n-1} = \left(1 - \frac{1}{n}\right)^{n-1}.$$

If there are at least two nodes transmitting signals in GR, each node will transmit its M_d with probability $1/2$. Thus, the successful event's probability is

$$p_2 = \sum_{k=2}^{n}\binom{n}{k}\left(\frac{1}{n}\right)^k\left(1 - \frac{1}{n}\right)^{n-k} \cdot k \cdot \frac{1}{2}\left(1 - \frac{1}{2}\right)^{k-1}$$

$$= \sum_{k=2}^{n}\binom{n}{k}\left(\frac{1}{n}\right)^k\left(1 - \frac{1}{n}\right)^{n-k}\frac{k}{2^k}.$$

Obviously, $P_1 = p_0 + p_1 + p_2$. Together with Lemma 2, we can get the following inequalities:

$$p_0 = \left(1 - \frac{1}{n}\right)^{2n-2}\left(1 - \frac{1}{n}\right) \geq \frac{1}{e^2}\left(1 - \frac{1}{n}\right) \quad p_1 \geq \frac{1}{e} \tag{2}$$

$$p_2 \geq \binom{n}{2}\left(\frac{1}{n}\right)^2\left(1 - \frac{1}{n}\right)^{n-2}\frac{2}{2^2} = \frac{1}{4}\left(1 - \frac{1}{n}\right)^{n-1} \geq \frac{1}{4e}. \tag{3}$$

As a result, the theorem holds. The derivation of inequality (1) is trivial; hence, we omit it. ∎

According to Theorem 1, $P_1 \geq 0.572$ when $n = 10$ and when $n = 20$, $P_1 \geq 0.584$. Note that $(1/e^2) + (5/4e) \approx 0.595$. For simplicity, we will regard inequality (1) as an equation in our later discussion, i.e., $P_1 = 0.595$.

We can see that the probability is significantly improved in comparison with probability $1/e$ derived in [4].

B. Recursive Protocol: FRIEND-tGR

To further improve the successful transmission probability, we introduce more subslots in GR before TR in one iteration.

In Section III-A, the probability of an idle slot is $(1 - (1/n))^{2n} \approx (1/e^2) \approx 0.135$. It is still too high in practice, although we have significantly reduced it. Thus, we add more subslots to reduce this probability. We now give FRIEND-tGR ($t \geq 2$) with t subslots in GR and describe it in Algorithm 3.

Algorithm 3 FRIEND-tGR (Multiple Prehandshaking)
1: **if** $\{A_t = t\}$ **then**   ▷ FRIEND-tGR has run t times.
2:   A will keep silent in TR and exit.
3: **else**   ▷ Still processing in t subslots
4:   $A_t = A_t + 1$.
5: **end if**
6: **if** $\{A_f = 1\}$ **then**   ▷ A has successfully sent M_d before.
7:   A will keep silent in TR and exit.
8: **end if**
9: A decides to send M_s by probability $1/A_n$.
10: **if** A sends M_s **then**
11:   **if** A does not receive M_s during GR **then**
12:     A will transmit M_d in TR;
13:   **else**   ▷ A receives M_s from other nodes
14:     A will transmit M_d in TR by probability $1/2$.
15:   **end if**
16: **else**   ▷ A does not send M_s
17:   **if** A does not receive M_s during GR **then**
18:     Call FRIEND-tGR and exit.
19:   **else**   ▷ A receives M_s from other nodes
20:     A will keep silent in TR.
21:   **end if**
22: **end if**

In FRIEND-tGR, A_t is the local counter for each node to identify the current subslot in GR. Initially, $A_t = 0$, and after one round of FRIEND-tGR, A_t will increase by 1. The maximum value of A_t is t. Because of the synchronization assumption, in each node, the local A_t remains the same in each round.

FRIEND-tGR is very similar to FRIEND-GR, except in two aspects. The first aspect is at lines 1–5, in which we put t subslots in GR to achieve a higher probability of successful transmissions. The other aspect is at line 18, in which FRIEND-tGR invokes itself recursively to utilize the remaining subslots in GR. By using this recursive strategy, we can further reduce the probability of idle slots.

We denote the successful event's occurrence in FRIEND-tGR by P_t. Now, we analyze the performance of FRIEND-tGR.

Theorem 2: P_{t+1} is bounded by

$$P_{t+1} \geq \frac{P_t}{e}\left(1 - \frac{1}{n}\right) + \frac{5}{4e}$$

where P_1 is given by Theorem 1.

Proof: If there are $t + 1$ subslots in GR, we again analyze different events that may occur in GR. If no one sends a signal

in GR, all nodes will invoke Algorithm 3 recursively. Thus, the successful event's probability is

$$p_0 = \left(1 - \frac{1}{n}\right)^n \cdot P_t \geq \frac{P_t}{e}\left(1 - \frac{1}{n}\right). \tag{4}$$

The other two scenarios are just the same as the proof in Theorem 1. According to inequalities (2)–(4)

$$P_{t+1} \geq \frac{P_t}{e}\left(1 - \frac{1}{n}\right) + \frac{5}{4e}.$$

Similarly, for simplicity, we get

$$P_{t+1} = \frac{P_t}{e} + \frac{5}{4e} \tag{5}$$

as $n \to +\infty$.

We then point out the upper bound of P_t.   ∎

Theorem 3: $\lim_{t \to +\infty} P_t = (5/4(e - 1)) \approx 0.727$.

This result can be derived by using (5) trivially; hence, we omit the proof.

We can see that the probability of a successful transmission in a slot is increased by approximately 98% compared with probability 0.368 in the algorithm proposed in [4].

C. Proper Number of Subslots

We have proved that the probability of a successful transmission can be significantly increased if there are sufficient subslots for nodes to detect other nodes' actions. Nevertheless, it is impossible to introduce infinite subslots in GR. We now discuss how to select a proper number of subslots in GR.

Let us consider Algorithm FRIEND-3GR. We can get the lower bound of P_3 due to Theorems 1 and 2 as follows:

$$P_3 \geq \frac{1}{e^4}\left(1 - \frac{1}{k}\right)^3 + \frac{5}{4e^3}\left(1 - \frac{1}{k}\right)^2 + \frac{5}{4e^2}\left(1 - \frac{1}{k}\right) + \frac{5}{4e}$$

where k stands for the number of nodes to be discovered at the current iteration. We can get $\lim_{k \to +\infty} P_3 \approx 0.710$. It is quite close to the optimal value; hence, it is feasible to introduce only three subslots before TR.

We can also compute the probability with other numbers of subslots in GR. Thus

$$\lim_{k \to +\infty} P_2 = \frac{1}{e^3} + \frac{5}{4e^2} + \frac{5}{4e} \approx 0.679$$
$$\lim_{k \to +\infty} P_4 = \frac{1}{e^5} + \frac{5}{4e^4} + \frac{5}{4e^3} + \frac{5}{4e^2} + \frac{5}{4e} \approx 0.720.$$

It is obvious that more subslots used for GR require more accurate synchronization techniques. To make the tradeoff, it is feasible to determine that there are three subslots in GR. Our simulation for different numbers of subslots in GR also proved this. In Fig. 2, we can see that FRIEND-3GR has almost the same performance as FRIEND-4GR but has less overhead and fewer requirements for synchronizing techniques.

Now, we discuss the expected value and the upper bound of slots needed to discover all n nodes by FRIEND-3GR.

Theorem 4: By using FRIEND-3GR and FRIEND-TR, the expected value of slots needed to discover all nodes with high probability is $1.5n$.
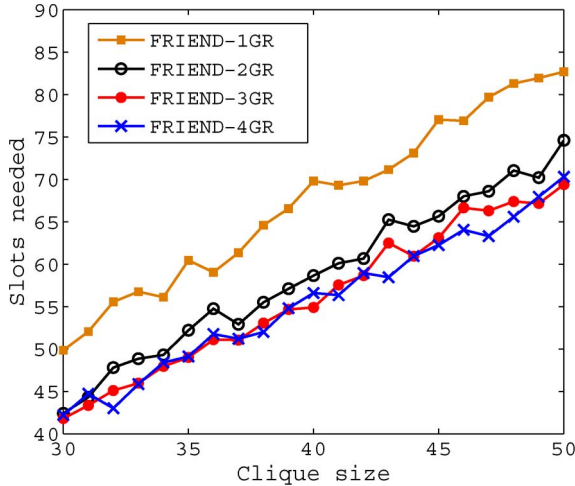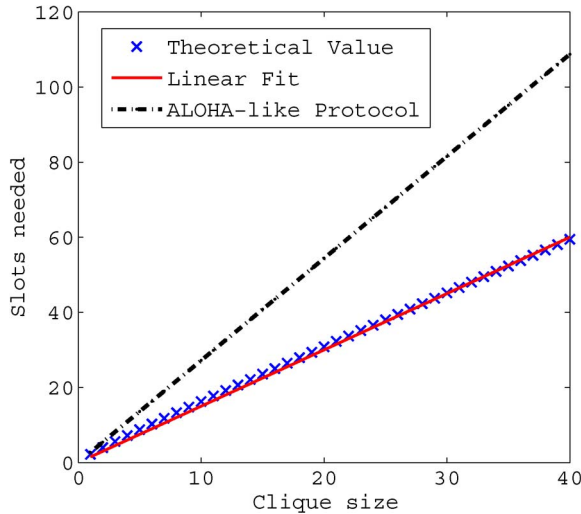
Fig. 2. Comparison of FRIEND-tGR.

Fig. 3. Comparison among the exact values of FRIEND, the linear fitting, and the performance in [4] (ALOHA-like protocol).

Proof: We assume that the discovery process is divided into epochs, and each epoch consists of at least one slot. Epoch i starts when the ith node is discovered and terminates when the $(i+1)$th node is discovered. Let T_i denote the number of slots of epoch i, and T_i is a geometrically distributed variable with parameter P_3 with $k = n - i$. (There are $n - i$ nodes to be discovered in epoch i.) Hence

$$E[T] = \sum_{k=1}^{n} E[T_k] = \sum_{k=1}^{n} \frac{1}{P_3} \approx 1.5n \qquad (6)$$

where the last approximation comes from the result of the linear fitting since it is nontrivial to derive an exact upper bound of the summation. ∎

Fig. 3 shows the expected values of time slots needed to discover all nodes in different sizes of cliques in FRIEND. We can see that the linear fitting is quite close to the theoretical values of FRIEND, and the time used is significantly decreased in comparison with [4].

Next, we point out the upper bound of the time slots needed to discover all nodes with high probability.

Theorem 5: By using FRIEND-3GR and FRIEND-TR, all nodes can be discovered in $3n$ slots with high probability.

Proof: Since P_3 varies little as k changes, we regard P_3 as a constant $1/1.5 = 2/3$ for simplicity, according to (6). Thus, T is a sum of n independent and identically distributed geometric random variables, and this distribution's parameter is $p = 2/3$. As a result, T is a negative binomial random variable with parameters n and $p = 2/3$.

The probability mass function is

$$P(T = t) = \binom{t-1}{n-1} p^n (1-p)^{t-n}, \qquad t = n, n+1, \ldots.$$

On the other hand, the following equation holds:

$$P(T > t) = P(X < n), \qquad X \sim \text{Binomial } (t, p).$$

Furthermore, Chernoff bounds point out that

$$P\left(X < (1-\delta)tp\right) < e^{-tp\delta^2/2}, \qquad 0 < \delta \leq 1. \qquad (7)$$

The formal proof of this inequality can be found in [16]. Then, we substitute $\delta = 1 - n/tp$ into (7), i.e.,

$$P(T > t) = P(X < n) < e^{\frac{-tp}{2}\left(1 - \frac{n}{tp}\right)^2}.$$

Therefore, we can get $P(T > 3n) < e^{-(n/4)}$. It is clear that $e^{-(n/4)} \to 0$ for sufficiently large n. Hence, the ND process can be finished in $3n$ slots with high probability. ∎

IV. HD-FRIEND: FRIEND FOR HALF-DUPLEX RADIOS

For nodes with half-duplex radios, although nodes cannot be made aware of other nodes' actions during their own transmissions, we can still use the similar strategy to reduce the probability of generating idle slots. We call it *HD-FRIEND*, which means the half-duplex counterpart of FRIEND.

Similarly, there should be at least two subslots in one iteration, which is the same as the case in FRIEND. However, there should be one more subslot that is used for transmitting feedback signals, because radios are half duplex and cannot notify collisions during reception. As a result, there are three subslots in one iteration. The first subslot is used to conduct the greeting process (*GR* subslot), and the second subslot is used for transmission of the discovery message (*TR* subslot). Feedback signals are transmitted in the third subslot (*FB* subslot). We then assign different actions for different nodes that may choose to transmit or receive in one iteration. Initial settings are the same as they are in Section III; hence, we omit them.

In GR subslots, each node runs Algorithm 4 to determine nodes that may transmit in the TR subslot.

Algorithm 4 HD-FRIEND-GR (Half Duplex)
1: **if** $A_f = 1$ **then**   ▷ A has successfully sent M_d.
2:   A will keep silent in TR (and FB) and exit.
3: **end if**
4: Node A decides to send M_s by probability $1/A_n$ and keep listening by probability $1 - 1/A_n$.
5: **if** A sends M_s **then**   ▷ A hopes to send M_d in TR.
6:   A will transmit M_d in TR;

7: **else**　　▷ A does not send M_s
8: **if** A does not receive M_s during GR **then**
9:　　A will transmit M_d in TR by probability $1/A_n$;
10: **else**　　▷ A receives M_s from other nodes
11:　A will keep silent in TR.
12:　**end if**
13: **end if**

We can see the main difference in GR from the given algorithm. If a node intends to transmit in TR, it will send M_s in GR to notify other nodes and send M_d in TR, regardless of other nodes' actions. Receiving nodes behave the same way as they are in FRIEND.

Then, in the TR subslot, every node runs Algorithm 5 to determine actions in the FB subslot.

Algorithm 5 HD-FRIEND-TR (Half Duplex)
1: **if** A plans to send M_d **then**
2:　A sends M_d.
3:　A will keep listening in FB.
4: **else**　　▷ A does not plan to send M_d
5:　A keeps listening.
6:　**if** A does not receive M_d during TR **then**
7:　　Current iteration is invalid.
8:　**else if** A receives a single M_d **then**
9:　　Record the ID in M_d.
10:　　$A_n = A_n - 1$.　　▷ A records one of its neighbors.
11:　**else**　　▷ There is a collision at A
12:　　A will send a feedback signal in FB.
13:　**end if**
14: **end if**

For a transmitting node, it will send its discovery message during TR and keep listening in FB to get feedback. While for a receiving node, it will keep listening in TR to determine whether to send a feedback signal to notify the transmitting nodes of the failure in transmission.

After TR, nodes enter the FB subslot, and the transmitting node will be aware of whether its transmission is successful. Each node runs Algorithm 6 in FB subslots.

Algorithm 6 HD-FRIEND-FB (Half Duplex)
1: **if** A transmitted in TR **then**
2:　A keeps listening in FB.
3:　**if** A receives the feedback signal. **then**
4:　　Current iteration is invalid.
5:　**else**
6:　　$A_f = 1$.
7:　**end if**
8: **else**　　▷ A received in TR
9:　**if** A plans to send a feedback signal in FB **then**
10:　　Send the feedback signal.
11:　**end if**
12: **end if**

In the FB subslot, if a receiving node detects collision, it will broadcast a feedback signal. As a result, transmitting nodes know that their transmissions failed. On the other hand, if the transmitting node does not receive the feedback signal, it knows that the transmission is successful, and it is time to keep silent during the remaining process of ND.

We can use a similar method to analyze the performance of HD-FRIEND. A theorem is as follows.

Theorem 6: When there are n nodes in a clique and all nodes run HD-FRIEND, the probability Q_1 that a node successfully transmits M_d in TR is bounded by

$$Q_1 \geq \frac{1}{e} + \frac{1}{e^2}\left(1 - \frac{1}{n}\right) \approx 0.503.$$

Proof: There are two cases where there is one and only one node transmitting its M_d in TR. If there is only one node transmitting M_s in GR, then the node will surely be discovered. We denote this probability by p'_1, and it is easy to see that

$$p'_1 = \binom{n}{1}\frac{1}{n}\left(1 - \frac{1}{n}\right)^{n-1} = \left(1 - \frac{1}{n}\right)^{n-1} \geq \frac{1}{e}.$$

If there are no nodes transmitting M_s in GR, all nodes will reconsider their actions. We denote the probability that there is one node transmitting in TR in this case by p'_2, and we can similarly get

$$p'_2 = \left(1 - \frac{1}{n}\right)^n \cdot p'_1 \geq \frac{1}{e^2}\left(1 - \frac{1}{n}\right).$$

According to the equation

$$Q_1 = p'_1 + p'_2$$

the theorem holds.　　　　　　　　　　　　　　　　　　　■

Furthermore, we can insert more subslots in GR and extend HD-FRIEND-GR to HD-FRIEND-tGR by using the same method as FRIEND-tGR. We denote the probability that a node successfully transmits in TR by Q_t for the case where there are t subslots in GR. The proof of the following theorem is similar as it is in Section III, and thus, we omit it.

Theorem 7: Q_{t+1} is bounded by

$$Q_{t+1} \geq \frac{Q_t}{e}\left(1 - \frac{1}{n}\right) + \frac{1}{e}.$$

We can also get the upper bound of Q_t.
Theorem 8: $\lim_{t \to +\infty} Q_t = (1/e - 1) \approx 0.582$.

The results show that our strategy can still decrease ND time by up to 36.7%, although nodes only have half-duplex radios. In Section VI, we will present the simulation result for HD-FRIEND.

V. DISCUSSION

Here, we discuss some issues related to FRIEND and their corresponding analysis results. Although the following discussions are based on FRIEND, they still hold for HD-FRIEND, due to the fact that FRIEND and HD-FRIEND are based on the similar prehandshaking strategy.

A. Unknown Number of Neighbors

Here, we discuss the situation when n is unknown to nodes. The basic idea is similar to [4].

We divide the process of discovery into phases. In phase i, each node runs the protocol with parameter $n = 2^i$, which means that we assume that there are 2^i nodes in phase i. This phase lasts $\lceil 1.5 \cdot 2^i \rceil$ slots. As a result, in the $\lceil \log_2 n \rceil$th phase, each node regards the number of nodes as n, and this phase lasts about $1.5n$ slots. This is just the expected value that we have derived in Section III-C.

Now, we determine the expected time needed in this case; the total time is given by

$$E[T] = \sum_{m=1}^{\lceil \log_2 n \rceil} 1.5 \cdot 2^m.$$

Since we know that $\sum_{m=1}^{\lceil \log_2 n \rceil} 2^m = 2n - 2$, the total time is

$$E[T] \approx 3(n - 1).$$

Comparing the result with Theorem 4, we can observe that lack of knowledge of n results in about a factor of 2 slowdown when n is relatively large. We will show the simulation results of the unknown n case in Section VI.

B. Multihop Networks

There is an obvious obstacle that we must face when extending FRIEND to the multihop case. In a clique, node i can determine whether its transmission is successful or not by itself, by still listening during its transmission. Nevertheless, the potential hidden terminals may cause for not all nodes within i's transmission range to receive i's message correctly.

To make sure that transmitting nodes can get correct feedback signals, we can modify FRIEND to let receivers take the responsibility. Now, transmitters do not detect collisions themselves, and nodes that are receiving will detect collisions. If a receiver has detected the collision in TR, it will send a feedback signal immediately. This way, transmitters will know their transmissions failed.

The existence of hidden terminals has a great impact on FRIEND's performance. This is predictable because each node belongs to several different cliques, and a node's action in a clique may affect nodes in other cliques. However, according to simulation, we show that FRIEND still has much better performance than the ALOHA-like protocol, which we will see in Section VI.

C. Low-Duty-Cycle Scenario

The basic drawback of the ALOHA-like protocol also exists in low-duty-cycle wireless networks. In [8], the performance of ALOHA-like protocol is analyzed when nodes' duty cycle is smaller than 1. In this case, the probability of generating an idle slot is still given by

$$p_0 = \left(1 - \frac{1}{n}\right)^n \approx \frac{1}{e}$$

which indicates that the basic idea of our FRIEND can be also utilized here. Since we have conducted extensive analysis of FRIEND, we will omit further discussion here.

We note that the performance of FRIEND can be optimized furthermore. When nodes all choose to be silent in GR, there may be some number of nodes that are not in receiving mode but are in dormant mode. As a result, each node can decrease its estimation of n (e.g., set $n = (1/2)n$ after each round) and, thus, improve the probability of transmitting. However, in this paper, we do not focus on designing the strategy of estimating n and leave it to our future work.

D. Initiating and Terminating ND Process

Here, we will discuss the case where nodes start the process of ND at different time instants as compared with the previous discussion, which is based on the assumption that nodes start at the same start instant. In addition, we will address the issue of when to terminate the ND process when n is unknown to nodes.

1) Initialization: To simplify the discussion, we assume that the maximum offset between any two nodes in the clique is δ. As a result, if the node, which is the earliest to start ND, begins at time slot t, all nodes will begin at $[t, t + \delta]$.

In comparison with Section V-A, we add δ slots to each phase, i.e., now, phase i lasts $\lceil 1.5 \cdot 2^i \rceil + \delta$ slots. Hence, all nodes will stay in the ith phase for at least $\lceil 1.5 \cdot 2^i \rceil$ time slots, and all nodes can be discovered with high probability at the $\lceil \log_2 n \rceil$th phase.

2) Termination: We use the similar strategy in [4] to determine when to terminate the ND process. An extra time slot is added to the end of each phase, i.e., phase i lasts $\lceil 1.5 \cdot 2^i \rceil + 1$ slots. In the last slot of a phase, nodes that have not successfully transmitted their discovery messages will broadcast signals in this slot. Therefore, the process of ND can be terminated once no signals can be detected in the last slot of a phase.

VI. PERFORMANCE EVALUATION

A. Simulation Setup

Simulations of the performance comparison were implemented using MATLAB. We simulate the random actions that nodes may choose to take in a slot, according to the corresponding probabilities. Our simulations include various settings of the sizes of the cliques. In a clique where all nodes are within communication range of each other, we simulate the discovery process in a clique of 3–100 nodes, considering the usual settings of wireless networks. It can be seen from the previous sections that the more nodes are deployed in a clique, the better FRIEND's performance will be. In terms of the multihop case, we put 200 nodes to a 300 m × 300 m 2-D plane. Nodes are put into the plane according to a uniform distribution, and they all have the same transmission range 50 m. We can know that the average number of neighbors for a certain node is about 18.

We compare FRIEND-3GR with the ALOHA-like protocol with the feedback mechanism proposed in [4]. Furthermore,
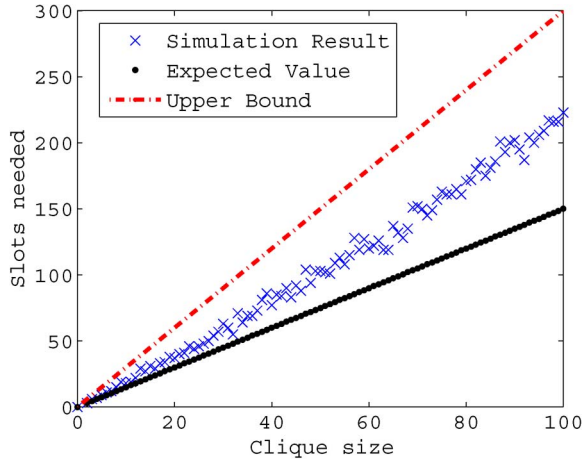
Fig. 4.   ND time in clique for FRIEND-3GR.

we show the simulation results of HD-FRIEND-3GR and the results for the unknown n scenario. The advantage of having a feedback mechanism has been already shown in [4] and [6]. Thus, we will not compare FRIEND-3GR with protocols that do not have such mechanisms. Each data point in the figures stands for an average result over 20 runs for accuracy.

B. Simulation Results

1) Validation of Theoretical Upper Bound: We now use simulation to validate the theorems stating that the expected value of time slots needed is $1.5n$ and that the upper bound is $3n$.

Fig. 4 shows the number of slots needed to discover all nodes in different sizes of cliques. Three kinds of values are compared: the simulation results, the expected values, and the upper bounds. We can see that the simulation results are larger than the corresponding expected values. This is mainly because when we simulate the discovery process, we regard a value as an output only when all nodes can be discovered in the time given in 20 simulation runs. Nevertheless, the simulation results are still smaller than the upper bounds we derived, which proves the correctness of our derivation.

2) Comparison in Clique: Similarly, we analyze the performance of FRIEND-3GR with the ALOHA-like protocol. (In this paper, we regard the ALOHA-like protocol as the birthday protocol with a feedback mechanism proposed in [4].) For a certain clique size, a time threshold can be regarded as an exact value only when all nodes are discovered in 20 consecutive runs.

Fig. 5 shows the comparison between two protocols with different sizes of cliques. We can see that FRIEND-3GR significantly reduces the processing time, as well as the upper bound estimation. When there are 100 nodes in a clique, it takes more than 600 slots to finish the ND process by the ALOHA-like protocol, whereas FRIEND-3GR only uses 300 slots to finish the process.

We must point out that the definitions of a slot are slightly different in these two protocols. In FRIEND-3GR, there are three tiny subslots before the normal slot, whereas in [4], there is one subslot after the normal slot. Because the duration of
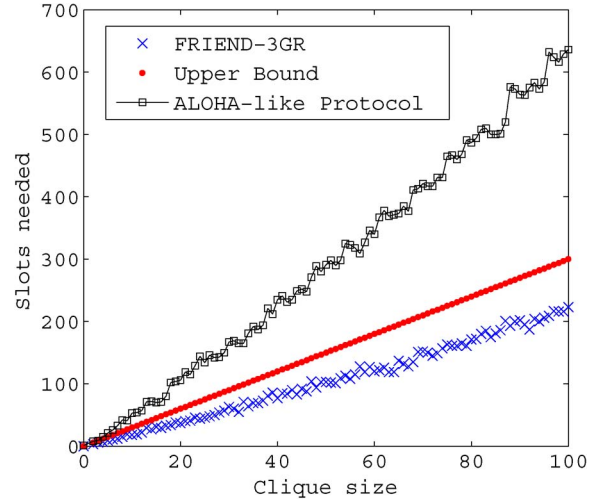

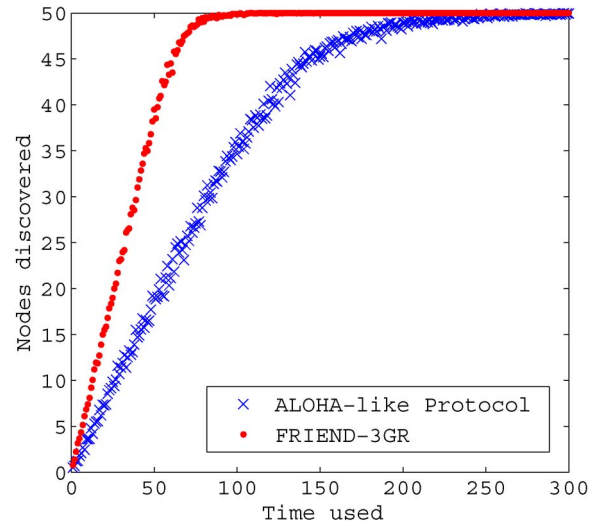Fig. 5.   Comparison of ND time in clique.

Fig. 6.   Comparison of discovered node numbers in clique.

subslots is really short, as mentioned in Section III, we can still compare the performance of two protocols by comparing their consumption of time slots.

Fig. 6 shows the trend of the number of discovered nodes in a clique with increasing number of iterations. We can see that ND is almost finished after 100 slots in FRIEND-3GR, whereas it costs about 200 slots in the ALOHA-like protocol. These observations can be also found in Figs. 4 and 5.

3) Validation of n Unknown Case: The nodes discovery ratio of different clique sizes when FRIEND-3GR is deployed without knowledge of n is shown in Fig. 7. All nodes can be discovered in the given time with high probability, which proves the correctness and effectiveness of the estimation mechanism proposed in Section V.

It is worth noticing that this time, the discovery process can be finished in the expected values with high probability, which is different from the previous simulation for FRIEND. This is mainly because in the estimation mechanism, the $\lceil \log_2 n \rceil$th phase is actually conducting the ND process with desired configuration. However, phases before the $\lceil \log_2 n \rceil$th phase
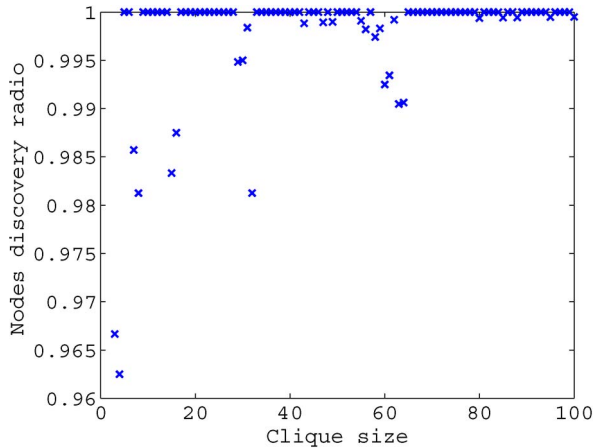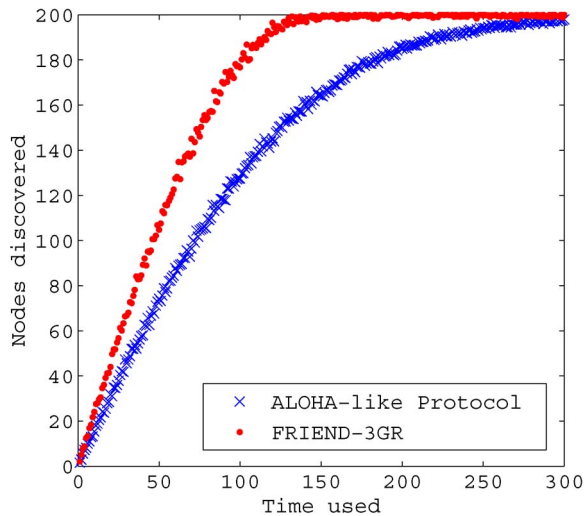
Fig. 7. Nodes discovery ratio of FRIEND-3GR (unknown n) in clique.

Fig. 9. Comparison of neighbor discovery time in clique (HD-FRIEND-3GR).

Fig. 8. Comparison of discovered node numbers in network.

will also contribute to the discovering process. Consequently, here, the discovery process can be finished in the corresponding expected values.

4) Comparison in Multihop Networks: Fig. 8 shows that the number of discovered nodes in a network is increasing with the number of iterations. We can know that after 150 slots, almost all 200 nodes are discovered by FRIEND-3GR, whereas it takes about 300 slots to discover all nodes by the ALOHA-like protocol.

5) Half-Duplex Networks: Due to the similar reason in Section III-C, we choose HD-FRIEND-3GR to simulate and compare its performance with the ALOHA-like protocol with the feedback mechanism. In Fig. 9, we can see that even if nodes use half-duplex radios, HD-FRIEND-3GR still decreases time tremendously.

The definitions of a slot are different in HD-FRIEND-3GR and [4]. HD-FRIEND-3GR has three more subslots before the slot that is described in [4]. Nevertheless, since the length of subslots is short, as we have mentioned in Section III, we can compare the performance of two protocols by comparing the number of slots.
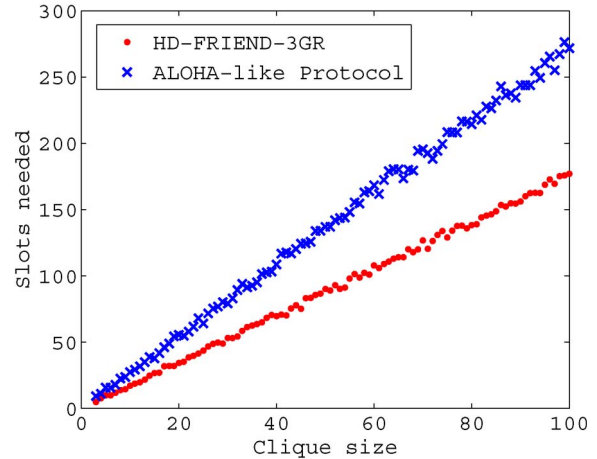
VII. RELATED WORK

A large amount of research has focused on the problem of accelerating the process of ND in wireless networks, and various protocols have been proposed to adapt to different situations [3]–[15]. Due to space limitations, we mainly introduce several works with a close relationship with FRIEND. Birthday protocols in [3] use a randomized strategy for nodes in a synchronous system to choose their actions in a slot independently and randomly. The authors proved that for a clique with n nodes, the optimal probability that a node transmits is $1/n$.

Vasudevan *et al.* [4] later pointed out that the expected time slots needed to finish the ND process by using the birthday protocol in [3] is neH_n, where H_n is the nth harmonic number. The authors also proposed protocols for more realistic situations, where the size of a clique is unknown to nodes, a feedback mechanism is introduced into the system, and the clocks of nodes are not identical, i.e., the system is asynchronous [7]. Basically, a factor of 2 slowdown is brought in if the size of a clique is unknown, whereas a factor of $\ln n$ slowdown is brought in if there are no feedback mechanisms.

With the development of MPR [23], Zeng *et al.* [5] extended the result of [4] to the MPR situation where no collision occurs if and only if there are no more than $k(k \geq 2)$ nodes simultaneously transmitting and proved that the expected time needed to discover all nodes is $\Theta(n \ln n/k)$. Ideally, if $k \geq n$, the discover time is shortened to $\Theta(\ln n)$. Similarly, the authors designed protocols for realistic situations in [4] and analyzed the upper bounds, respectively. You *et al.* [26] proposed the similar result in peer-to-peer MPR networks.

You *et al.* [8] extended the result of [4] to the situation when the duty cycle of nodes is not 1, i.e., some nodes may be dormant at a certain time instant. By reducing the problem to the generalization of the classical coupon collector's problem [17], it is proven that when the duty cycle is $1/2$, the upper bound is $ne(\log_2 n + (3\log_2 n - 1)\log_2 \log_2 n + c)$ with a constant c, and lack of knowledge of n results in a factor of 2 slowdown as well in a clique. Sun *et al.* [24] proposed a prehandshaking strategy, which significantly reduces the probability of idle slots, for full-duplex wireless ad hoc networks. In [25], Sun *et al.* also discussed how the ALOHA-like

protocol works in low-duty-cycle sensor networks with MPR technique.

Many papers have focused on the feasibility of designing a practical full-duplex wireless radio. Choi *et al.* [1] proposed a method named antenna cancelation to avoid self-interference. However, this technique requires three antennas, which makes it unattractive in comparison with a three-antenna MIMO system with higher throughput. This method also suffers from the constraint of bandwidth seriously, which makes it not feasible for wideband signals such as WiFi.

Jain *et al.* [2] overcame the drawbacks of [1] and proposed a novel mechanism, which is called balun cancelation, in which a balun circuit is used to create inverse signals to achieve the cancelation of self-interference. This method requires only two antennas and has no bandwidth constraints theoretically. Furthermore, Choi *et al.* [1] and Jain *et al.* [2] made an experimental device, which supports single-channel full-duplex communication. Although some realistic conditions make the device not as perfect as it is in theory, it is still safe to say that the single-channel full-duplex technology is promising, and thus, our work utilizes it to accelerate the ND process.

In addition to works aimed at accelerating the process of ND, there are also many other researches that discuss other issues about ND. For instance, Poturalskim *et al.* [21] and Stoleru *et al.* [28] proposed the formal system definition of secure ND when there are adversary nodes in the environment. Furthermore, they proposed a secure ND protocol when the system model satisfies certain conditions. Kohvakka *et al.* [22] discussed the issue of energy consumption of the ND process. Since our main concern is how to finish the ND process in the shortest time, we will not introduce them in detail.

VIII. CONCLUSION AND FUTURE WORK

In this paper, we have proposed a prehandshaking ND protocol FRIEND by adding prehandshaking subslots before the traditional slots. Furthermore, we applied full-duplex technology and used it to conduct prehandshaking with new feedback mechanisms. We analyzed the expected value and the upper bound of ND processing time theoretically and validated our analysis by simulation compared with the ALOHA-like protocol proposed in [4]. Both theoretical analysis and simulations proved that FRIEND significantly decreases the time needed to finish the ND process. Furthermore, we discussed some implementation issues and extensions of FRIEND and showed that the half-duplex counterpart of FRIEND, i.e., HD-FRIEND, also significantly decreases time consumption.

In the future, we would like to evaluate the performance of FRIEND by testbed experiments. We also want to consider more realistic models, e.g., nodes with MPR techniques, nodes with low duty cycles, and asynchronous models.

REFERENCES

[1] J. I. Choi, M. Jain, K. Srinivasan, P. Levis, and S. Katti, "Achieving single channel, full duplex wireless communication," in *Proc. ACM MobiCom*, 2010, pp. 1–12.

[2] M. Jain, J. I. Choi, T. M. Kim, D. Bharadia, S. Seth, K. Srinivasan, P. Levis, S. Katti, and P. Sinha, "Practical, real-time, full duplex wireless," in *Proc. ACM MobiCom*, 2011, pp. 301–312.

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

[4] S. Vasudevan, D. Towsley, D. Goeckel, and R. Khalili, "Neighbor discovery in wireless networks and the coupon collector's problem," in *Proc. ACM MobiCom*, 2009, pp. 181–192.

[5] W. Zeng, X. Chen, A. Russell, S. Vasudevan, B. Wang, and W. Wei, "Neighbor discovery in wireless networks with multipacket reception," in *Proc. ACM MobiHoc*, 2011, p. 3.

[6] R. Khalili, D. Goeckel, D. Towsley, and A. Swami, "Neighbor discovery with reception status feedback to transmitters," in *Proc. IEEE INFOCOM*, 2010, pp. 1–9.

[7] S. A. Borbash, A. Ephremides, and M. J. McGlynn, "An asynchronous neighbor discovery algorithm for wireless sensor networks," *Ad Hoc Netw.*, vol. 5, no. 7, pp. 998–1016, Sep. 2007.

[8] L. You, Z. Yuan, P. Yang, and G. Chen, "ALOHA-like neighbor discovery in low-duty-cycle wireless sensor networks," in *Proc. IEEE WCNC*, 2011, pp. 749–754.

[9] X. An, R. Venkatesha Prasad, and I. Niemegeers, "Impact of antenna pattern and link model on directional neighbor discovery in 60 GHz networks," *IEEE Trans. Wireless Commun.*, vol. 10, no. 5, pp. 1435–1447, May 2011.

[10] R. Cohen and B. Kapchits, "Continuous neighbor discovery in asynchronous sensor networks," *IEEE/ACM Trans. Netw.*, vol. 19, no. 1, pp. 69–79, Feb. 2011.

[11] A. Keshavarzian and E. Uysal-Biyikoglu, "Energy-efficient link assessment in wireless sensor networks," in *Proc. IEEE INFOCOM*, 2004, pp. 1751–1761.

[12] D. Angelosante, E. Biglieri, and M. Lops, "Neighbor discovery in wireless networks: A multiuser-detection approach," in *Proc. Inf. Theory Appl. Workshop*, 2007, pp. 46–53.

[13] Z. Zhang and B. Li, "Neighbor discovery in mobile ad hoc self-configuring networks with directional antennas: Algorithms and comparisons," *IEEE Trans. Wireless Commun.*, vol. 7, no. 5, pp. 1540–1549, May 2008.

[14] N. Karowski, A. Viana, and A. Wolisz, "Optimized asynchronous multichannel neighbor discovery," in *Proc. IEEE INFOCOM*, 2011, pp. 536–540.

[15] G. Jakllari, W. Luo, and V. Krishnamurthy, "An integrated neighbor discovery and MAC protocol for ad hoc networks using directional antennas," *IEEE Trans. Wireless Commun.*, vol. 6, no. 3, pp. 1114–1024, Mar. 2007.

[16] R. Motwani and P. Raghavan, *Randomized Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 1995.

[17] P. Erdös and A. Rényi, "On a classical problem of probability theory," *Magyar Tud. Akad. Mat. Kutato Int. Kozl*, vol. 6A, pp. 215–220, 1961.

[18] L. Kong, L. Fu, X. Liu, and M. Wu, "Accelerating initialization for sensor networks," in *Proc. IEEE GLOBECOM*, 2009, pp. 1–6.

[19] S. Yoon, C. Veerarittiphan, and M. L. Sichitiu, "Tiny-sync: Tight time synchronization for wireless sensor networks," *ACM Trans. Sens. Netw.*, vol. 3, no. 2, p. 8, Jun. 2007.

[20] K. B. Rasmussen, S. Capkun, and M. Cagalj, "SecNav: Secure broadcast localization and time synchronization in wireless networks," in *Proc. ACM MobiCom*, 2007, pp. 310–313.

[21] M. Poturalskim, P. Papadimitratos, and J. Hubaux, "Secure neighbor discovery in wireless networks: Formal investigation of possibility," in *Proc. ASIACCS*, 2008, pp. 189–200.

[22] M. Kohvakka, J. Suhonen, M. Kuorilehto, V. Kaseva, M. Hännikäinen, and T. D. Hämäläinen, "Energy-efficient neighbor discovery protocol for mobile wireless sensor networks," *Ad Hoc Netw.*, vol. 7, no. 1, pp. 24–41, Jan. 2009.

[23] J. Jeon and A. Ephremides, "Neighbor discovery in a wireless sensor network: Multipacket reception capability and physical-layer signal processing," in *Proc. 48th Annu. Allerton Conf. Commun., Control, Comput.*, 2010, pp. 310–317.

[24] G. Sun, F. Wu, X. Gao, and G. Chen, "PHED: Pre-handshaking neighbor discovery protocols in full duplex wireless ad hoc networks," presented at the IEEE Global Commun. Conf., Anaheim, CA, USA, 2012.

[25] G. Sun, F. Wu, and G. Chen, "Neighbor discovery in low-duty-cycle wireless sensor networks with multipacket reception," in *Proc. IEEE ICPADS*, 2012, pp. 490–497.

[26] L. You, X. Zhu, and G. Chen, "Neighbor discovery in peer-to-peer wireless networks with multi-channel MPR capability," in *Proc. IEEE ICC*, 2012, pp. 4975–4979.

[27] Y. Wang, G. Pan, and Z. Huang, "Direct multi-hop time synchronization with constructive interference," in *Proc. IPSN*, 2012, pp. 115–116.

[28] R. Stoleru, H. Wu, and H. Chenji, "Secure neighbor discovery and wormhole localization in mobile ad hoc networks," *Ad Hoc Netw.*, vol. 10, no. 7, pp. 1179–1190, Sep. 2012.

Guobao Sun (S'12) is currently pursuing the B.S. degree with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China.

His research interests include wireless networking, social networks, and wireless security.

Mr. Sun is a Student Member of China Computer Federation and the Association for Computing Machinery.

Fan Wu (M'06) received the B.S. degree in computer science from Nanjing University, Nanjing, China, in 2004 and the Ph.D. degree in computer science and engineering from the University at Buffalo, The State University of New York, Buffalo, NY, USA, in 2009.

He is an Associate Professor with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China. He was a Postdoctoral Research Associate with the University of Illinois at Urbana–Champaign, Champaign, IL, USA. His research interests include wireless networking, economic incentives for cooperation, and peer-to-peer computing.

Dr. Wu received the Excellent Young Scholar Award from Shanghai Jiao Tong University in 2011 and the Pujiang Scholar award in 2012. He is a Member of the Association for Computing Machinery. (For more information, please visit http://www.cs.sjtu.edu.cn/~fwu/.)

Xiaofeng Gao (M'09) received the B.S. degree in information and computational science from Nankai University, Tianjin, China, in 2004; the M.S. degree in operations research and control theory from Tsinghua University, Beijing, China, in 2006; and the Ph.D. degree in computer science from The University of Texas at Dallas, Richardson, TX, USA, in 2010.

She is an Associate Professor with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China. Her research interests include data engineering, wireless communications, and combinatorial optimizations.

Guihai Chen (M'10) received the B.S. degree from Nanjing University, Nanjing, China; the M.Eng. degree from Southeast University, Nanjing; and the Ph.D. degree from The University of Hong Kong, Pokfulam, Hong Kong.

In 1998, he was a Research Fellow with Kyushu Institute of Technology, Kitakyushu, Japan. In 2000, he was a Visiting Professor with The University of Queensland, Brisbane, Australia. From September 2001 to August 2003, he was a Visiting Professor with Wayne State University, Detroit, MI, USA. He is a Distinguished Professor and Deputy Chair with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China. He has published more than 200 papers in peer-reviewed journals and refereed conference proceedings. His research interests include wireless sensor networks, high-performance computer architecture, peer-to-peer computing, and performance evaluation.

Dr. Chen has served on the technical program committees of numerous international conferences. He is a Member of the IEEE Computer Society.

Wei Wang (M'09) received the Ph.D. degree in computer software and theory from Tongji University, Shanghai, China.

He is an Associate Professor with the Department of Computer Science and Technology, Tongji University. His research interests include parallel and distributed computing and information security.

Dr. Wang is a member of the Association for Computing Machinery. He received the IBM Ph.D. Fellowship in 2007.