International Journal of Innovative Research in Advanced Engineering (IJIRAE) ISSN: 2349-2163
Issue 4, Volume 2 (April 2015) www.ijirae.com

FRIEND: A PREHANDSHAKING NEIGHBOR DISCOVERY PROTOCOL FOR WIRELESS AD HOC NETWORKS

[1] Ramya S, *Research scholar, M.Tech CNE (PT), NIE, Mysore*
[2] Dr. Phaneendra H D, *Professor, Dept. of ISE, NIE, Mysore*

ABSTRACT- Neighbor Discovery (ND) is a basic and crucial step for initializing wireless adhoc networks. A fast, precise, and energy-efficient ND protocol has significant importance to subsequent operations in wireless networks. However, various existing protocols have high probabilities to generate idle slots in their neighbor discovering processes, which extends the executing duration, and therefore compromises their performance. We proposed a novel randomized protocol FRIEND, a pre-handshaking neighbor discovery protocol, to initialize synchronous full duplex wireless adhoc networks. By introducing a pre-handshaking strategy to help each node be aware of activities of its neighbourhood, the probabilities of generating idle slots and collisions can be significantly reduced. Furthermore, with the development of single channel full duplex communication technology, we decrease the processing time needed in FRIEND, and construct the full duplex neighbor discovery protocol. Our simulation results show that FRIEND can decrease the duration of ND by some extent as compared to the classical ALOHA-like protocols. Additionally, we propose HD-FRIEND for half duplex networks and variants of FRIEND. Both theoretical analysis and simulation results show that FRIEND can adapt to several scenarios, and significantly decrease the duration of ND.

Keywords: Neighbor Discovery, Randomized protocol, Friend protocol.

1. INTRODUCTION

Communication is the primary factor which influenced the development of mankind. One of the primary goals of communication is exchanging information between two persons. Nowadays we have advanced technologies for communication. Communication can be between human beings or between machines. For the purpose of communication between machines we provided networks, normally connected by physical channels. Then, to avoid the difficulties with wired networks there come wireless networks.

Wireless ad hoc networks have attracted a lot of interest from both academia and industry due to their wide range of applications. In many situations, nodes are deployed without the support of pre-existing infrastructures for communication. Consequently, nodes in a wireless ad hoc network need to configure themselves through their own communication activities to form a reliable infrastructure during the initialization for further operations.

Adhoc network is a network formed without any central administration which consists of mobile nodes that use a wireless interface to send packet data. Since the nodes in a network of this kind can serve as routers and hosts, they can forward packets on be-half of other nodes and run user applications.

For each node, the knowledge of its one-hop neighbors (the nodes it can directly communicate) has significant importance to the upper layer protocols like MAC protocols, routing protocols, etc. So Neighbor Discovery (ND) is designed to discover a node's one-hop neighbors and thus is momentous and crucial for configuring wireless network. Randomized protocols are most commonly used to conduct ND process in wireless networks. In those protocols, each node transmits at different randomly chosen time instants to reduce the possibility of the collision with other nodes.

We 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 discovery message by probability p and listen by probability 1-p (the optimal value of p is proven to be 1=n). All the available protocols either take more time or create idle slots which lead to collision in the neighbour discovery as a result the time required to discover the neighbor is increased.

We propose a novel randomized protocol FRIEND, a pre-handshaking neighbor discovery protocol, to initialize synchronous full duplex wireless adhoc networks. By introducing a pre-handshaking strategy to help each node be aware of activities of its neighbourhood, we significantly reduce the probabilities of generating idle slots and collisions. Moreover, with the development of single channel full duplex communication technology [1, 2], we further decrease the processing time needed in FRIEND, and construct the first full duplex neighbour discovery protocol. After that we introduce various variants of FRIEND which are FRIEND-TR and FRIEND-HD.

In FRIEND-TR we are using GR part of algorithm many times recursively to decrease the extent of collision and FRIEND-HD we are using for nodes with half duplex radios.

2. Related Work:

A large number of works have 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] Compared with existing deterministic [11] and multi-user detection-based [12] protocols, randomized protocols are most commonly used to conduct ND process in wireless networks [3–8]. In those protocols, each node transmits at different randomly chosen time instants to reduce the possibility of the collision with other nodes. Commonly, researchers discuss ND protocols under a synchronous system, and they usually focus on a clique with n nodes.

There are two types of protocols available for wireless adhoc network

1. Deterministic.
2. Randomized.

2.1 Deterministic algorithms

Deterministic algorithms can be defined in terms of a state machine: a state describes what a machine is doing at a particular instant in time. State machines are passed in a discrete manner from one state to another. Just after we enter the input, the machine is in its initial state or start state. If the machine is deterministic, means that from this point onwards, its current state determines what will be its next state, its course through the set of states is predetermined. Note that a machine can be deterministic and still never stop or finish, and therefore fail to deliver a result.

In a deterministic neighbor discovery algorithms, 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. The downside of these algorithms is that usually they need increased running time and often the assumption that the number of neighbors must be known. For example disco protocol.

Disco Protocol: An example of a deterministic algorithm is Disco. The algorithm selects a pair of prime numbers such that the sum of their reciprocals is equal to the application's desired radio duty cycle. Each node increments a local counter with a globally agreed-upon period and, if this local counter is divisible by either of the primes, the node turns on its radio for one counter period. This protocol ensures that two nodes will have some overlapping radio on-time within a bounded number of periods, even if nodes independently set their own duty cycle. Other deterministic protocols include the Power Saving protocols, or the Asynchronous Wakeup which are used in wireless sensor network.

2.2 Randomized Algorithms

A randomized algorithm flips coins during its execution to determine what to do next. When considering a randomized algorithm, we generally care about its expected worst-case performance, which is the average amount of time it takes on the worst input of a given size. This average is computed over all the possible outcomes of the coin flips during the execution of the algorithm. We may also ask for a high-probability bound, showing that the algorithm doesn't consume too much resources most of the time. Formally, we think of a randomized algorithm as a machine M that computes M(x,r), where x is the problem input and r is the sequence of random bits. Our machine model is the usual random access machine or RAM model, where we have a memory space that is typically polynomial in the size of the input n, and in constant time we can read a memory location, write a memory location, or perform arithmetic operations on integers of up to O(logn) bit.

In a probabilistic neighbour discovery algorithm, each node transmits at randomly chosen times and yet discovers all its neighbours by a given time, with high probability. Examples of probabilistic protocols are the Aloha OMS protocol and birthday protocols.

Aloha oms: This protocol is a modified version of the random hello protocol. In this version, Operating Mode Shifting (OMS), which allows nodes to transfer among three working states, transmit (T), listen/idle (L) and sleep (S), is enabled, in order to save energy.

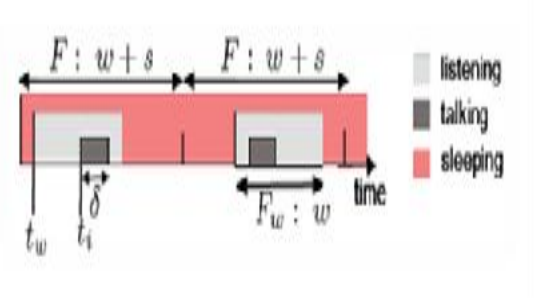


Figure 1.1 Aloha oms protocol

As shown in figure 1.1, each node can be in one of the three states: listening, talking or sleeping. These three states are performed inside a frame F, of size w + s. In each occurrence of F, a node picks randomly an instant tw, such that tw ϵ [0, s]. Then, a node picks randomly a second instant ti, such that ti ϵ [tw, tw + w – δ]. A node is in the talking state during [ti, ti + δ], in the listening state during [tw, tw + w] \[ti, ti + δ] and in the sleeping state the rest of the frame F. The hello message is transmitted at ti with duration of δ. In the sleeping state a node cannot receive any message.

Birthday protocols: The inspiration for this protocol is the birthday paradox, in which it's computed the probability that at least two people in a room have the same birthday. The birthday protocol suggests that the discovery of one node is fulfilled when at least two nodes have the "same birthday" in the same slot: one and only one of them is the sender, while the rest are all receivers. The discovery period is sliced in slots. Each slot lasts long enough Ts, to transmit one hello message. Ts ≥ Tl = lmsg / Rb, where lmsg and Rb stand for message length and transmission speed. A node can be in one of three states: transmit (T), listen (L), or sleep (S). In each slot, a node chooses independently state T with probability pt, L with probability pl and S with probability ps = 1-pl-pt.

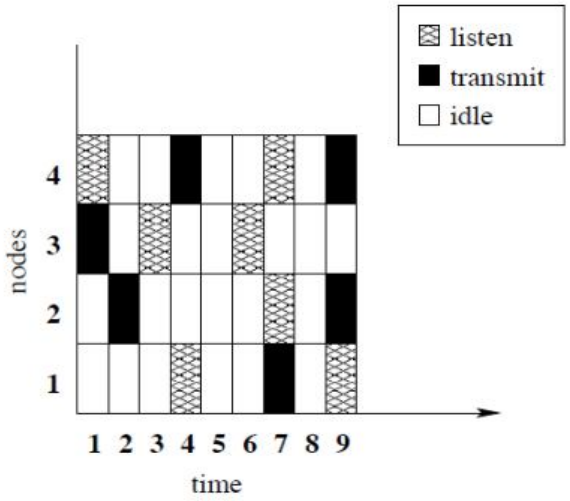


Figure 1.2 birthday protocol example

A simple example of the Birthday protocol can be observed in the figure 1.2. Four nodes, all in range of each other, perform discovery over 9 slots. We can observe that there are only four links discovered (3 →4, 4→1, 1→2, 1→4). No other links are discovered, due to no listeners (slot 2), no transmitter (slots 3 and 6), or no activity (slots 5 and 8).

Advantage

- First ever proposed ND protocol

Disadvantages

- This applies only for ideal assumptions and systems.
- This method is proposed only for synchronous systems.

3. Proposed System:

The proposed system is to introduce a pre-handshaking strategy to help each node be aware of activities of its neighborhood before normal transmissions, such that the system can have high probabilities to avoid collisions and idle slots. So to conduct this pre-handshaking, we add few tiny sub-slots before each normal slot. With the help of full duplex technology, at each sub-slot, every node will decide whether to transmit the discovery message in a normal slot by transmitting an anonymous election signal and catch its neighbor's signals simultaneously. With different transmitting-receiving scenarios, we design an effective approach 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 a sub-slot is added after the normal slot, and receivers will give feedback signals to transmitters in this sub-slot. In our design this overhead can be eliminated by using full duplex nodes. If a receiver finds that two or more nodes are transmitting simultaneously, it will transmit a warning message immediately to inform other transmitters the failure of their transmissions. The process is shown below in fig 3(b) wherein there is one sub slot in GR, while in fig 3(c) there are multiple sub-slots in GR.

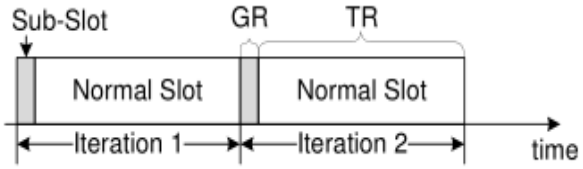


Fig 3 (b): Pre-handshaking with one sub-slot

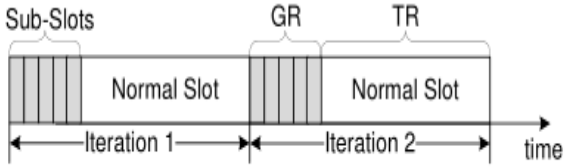
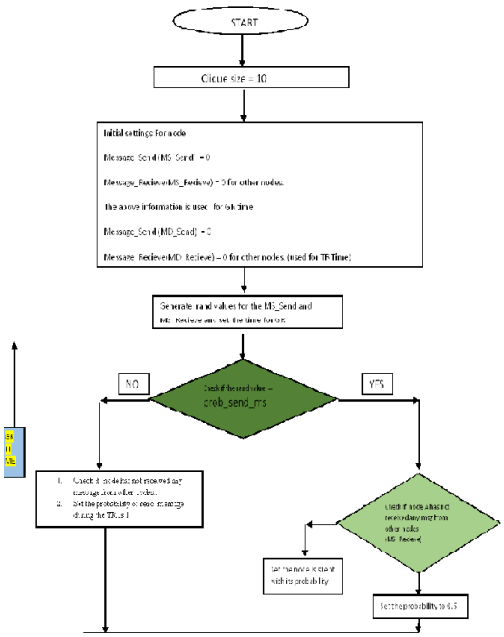


Fig 3(c): Pre-handshaking with multiple sub-slots

Merits of the proposed system

- Higher probabilities to avoid collisions and idle slots.
- Full duplex technology
- Overhead can be eliminated.
- Can decrease the ND duration by 48% when compared to ALOHA-like protocols.
- Applicable for multi-hop networks and duty cycled networks.



3.1 Algorithm Implementation

There are 6 algorithms that we are implementing in our project. We will discuss the working of all the algorithms now.

3.1.1 Pre-handshaking process FRIEND GR

Algorithm 1: FRIEND GR

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/An.

1) If A sends an M s (Line 5-10), it implies A hopes to send M d in TR.
a) At this moment, if A does not receive Ms during GR, it means A wins the election and will definitely send Md in the following TR.
b) If A receives Ms. It means there exist other candidates within A's direct communication range.Therefore A can only send M d by probability 1/2.

2) If A does not send M s (Line 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 no nodes decide to send M d in TR. A will reconsider sending M d by probability 1/An.
b) If A receives M s . It means that there are nodes intending to transmit and thus A will keep silent.

3.1.2 Neighbor discovery FRIEND TR

Algorithm 2: FRIEND TR
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

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

In FRIEND-TR, there are two scenarios:
1. If A sends Md, A will meanwhile check the existence of other signals (Line 1-7).
a) If A does not receive Md during TR, it means that A's transmission is successful. Consequently A will keep silent during the rest of ND process.
b) If A receives Md from other nodes, it means that the current transmission is failed.
2) If A does not send Md , A will check the number of transmitters (Line 8-18).
a) 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.
b) If A receives a single Md 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 An by 1.
c) If there is a collision at A, it means that the current transmission is failed.

3.1.3 FRIEND-tGR

Algorithm 3: FRIEND-tGR
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 sub-slots

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 an 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 an 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

To improve the successful transmission probability, we introduce more sub-slots in GR before TR in one iteration. The probability of idle slot generation is a bit more, so we add more sub-slots to reduce this probability. We now give FRIEND-tGR ($t \geq 2$) with t sub-slots in GR and describe it in Alg. 3. In FRIEND tGR, At is the local counter for each node to identify the current sub-slot in GR. Initially At = 0, and after one round of FRIEND tGR, A t will increase by 1. The maximum value of At is t. Because of the synchronization assumption, in each node the local At remains the same in each round. FRIEND-tGR is very similar to FRIEND-GR except in two aspects.

1. The first is from Line 1 to 5, in which we put t sub-slots in GR to achieve a higher probability of successful transmissions.
2. The other one is at Line 18, in which FRIEND-tGR invokes itself recursively to utilize the remaining sub-slots in GR.

3.1.4 HD FRIEND

Algorithm 4:HD FRIEND

1: if $A_f = 1$ then A has successfully sent M_d.
2: A will keep silent in TR (as well as 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 that the main difference in GR from the algorithm above. If a node intends to transmit in TR, it will send Ms 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 TR sub-slot, every node runs Alg. 5 to determine actions in FB sub slot.

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 failure of the transmission to transmitting nodes. After TR, nodes enter into FB sub-slot and the transmitting node will be aware of whether its transmission is successful. Each node runs Alg. 6 in FB sub-slots.

3.1.5 HD Friend TR

Algorithm 5: HD FRIEND TR

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

The algorithm 5 shown above represents the hd-friend-tr part. The basic difference from the friend tr is that this can be used for nodes with half duplex radios. For avoiding the collisions we are using feedback signals, which cannot be sent if we use the same algorithm of friend-tr. To overcome this we are using a friend feedback algorithm. To implement this we are dividing the normal slots into 3 sub-slots.

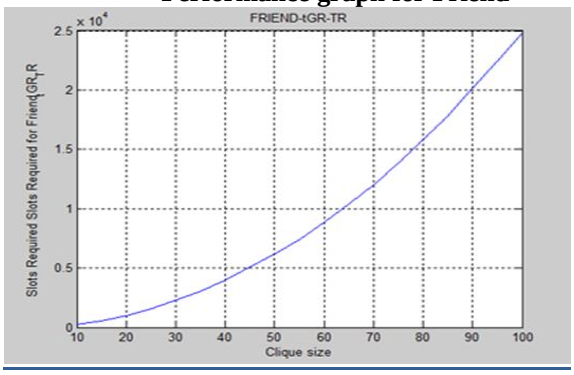
3.1.6 HD-FRIEND-FB

Algorithm 6: HD FRIEND FB

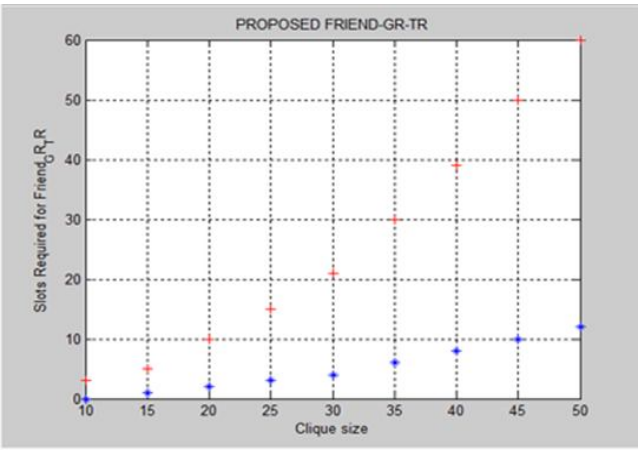
1: if A transmitted in TR then
2: A keeps listening in FB.
3: if A receives the feedback signal. then
4: Current iteration is invalid.
5: else
6: $A_f = 1$.
7: end if
8: else A received in TR
9: if A plans to send a feedback signal in FB then
10: Send the feedback signal.
11: end if
12: end if

In the FB sub-slot, if a receiving node detects collision, it will broadcast a feedback signal. As a result, transmitting nodes knows that their transmissions are 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 for it to keep silent during the remaining process of ND.

Performance graph for Friend



Comparison with aloha protocol



4. CONCLUSION AND FUTURE ENHANCEMENTS

4.1 CONCLUSION

In this paper, we proposed a pre-handshaking neighbor discovery protocol FRIEND by adding pre-handshaking sub-slots before the traditional slots. Furthermore, we applied the full duplex technology and used it to conduct pre-handshaking with new feedback mechanisms.

By simulation we compared the proposed method with ALOHA-like protocols. We tried to prove that both theoretical and simulation results of FRIEND will decrease the time to calculate neighbor discovery.

We analyzed the expected value and upper bound of ND processing time theoretically, and validated our analysis by simulation. Both theoretical analysis and simulations proved that FRIEND significantly decreases the time needed to finish the ND process.

We discussed some implementation issues and extensions of FRIEND, and showed that the half duplex HD-FRIEND, also significantly decreases time consumption.

4.2 FUTURE ENHANCEMENTS

In the future, we would like to evaluate the performance of FRIEND by test-bed experiments. We also want to consider more realistic models, e.g nodes with multi-packet reception techniques, nodes with low duty cycles and asynchronous models. Currently the algorithm works only for one-hop neighbors. We would like to add some more improvements so that it works in multi-hop networks too.

5. REFERENCES

[1]. J. I. Choi, M. Jain, K. Srinivasan, P. Levis, and S. Katti. “Achieving Single Channel, Full Duplex Wireless Communication”. In Proc. Of ACM MobiCom, 2010.
[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. of ACM MobiCom, 301-312, 2011.
[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. of ACM MobiHoc, 137-145, 2001.
[4]. S. Vasudevan, D. Towsley, D. Goeckel, and R. Khalili. “Neighbor Discovery in Wireless Networks and the Coupon Collector’s Problem”. In Proc. of ACM MobiCom, 181-192, 2009.
[5]. W. Zeng, X. Chen, A. Russell, S. Vasudevan, B. Wang, and W.Wei. “Neighbor Discovery in Wireless Networks with Multi packet Reception”. In Proc. of ACM MobiHoc, 3:1-10, 2011.
[6]. R. Khalili, D. Goeckel, D. Towsley, and A. Swami. “Neighbor Discovery with Reception Status Feedback to Transmitters”. In Proc. of IEEEINFOCOM, 2010.
[7]. S. A. Borbash, A. Ephremides, and M. J. McGlynn. “An Asynchronous Neighbor Discovery Algorithm for Wireless Sensor Networks”. Elsevier Ad Hoc Networks, 5(7): 998-1016, 2007.
[8]. L. You, Z. Yuan, P. Yang, and G. Chen. “ALOHA-Like Neighbor Discovery in Low-Duty-Cycle Wireless Sensor Networks”. In Proc. of IEEE WCNC, 749-754, 2011.
[9]. X. An, R. Venkatesha Prasad, and I. Niemegeers. “Impact of Antenna Pattern and Link Model on Directional Neighbor Discovery in 60 GHz Networks”. In IEEE TWC, (10)5:1435-1447, 2011.
[10]. R. Cohen and B. Kapchits. “Continuous Neighbor Discovery in Asynchronous Sensor Networks”. In IEEE TON, (19)1:69-79, 2011.
[11]. Keshavarzian and E. Uysal-Biyikoglu. “Energy-Efficient Link Assessment in Wireless Sensor Networks”. In Proc. of IEEE INFOCOM, 2004.

