Q-SAND: a Quick Neighbor Discovery Protocol for Wireless Networks with Sectored Antennas

Nicolás Gammarano*, Javier Schandy† and Leonardo Steinfeld‡
Instituto de Ingeniería Eléctrica, Facultad de Ingeniería, Universidad de la República, Montevideo, Uruguay
*ngammarano@fing.edu.uy, †jschandy@fing.edu.uy, ‡leo@fing.edu.uy

***Abstract*—In this paper, we proposed Q-SAND (Quick Sectored-Antenna Neighbor Discovery), a neighbor discovery protocol for wireless networks with sectored antennas which enhances the state-of-the-art SAND protocol. Both SAND and Q-SAND protocols were successfully implemented in Contiki, an open source operating system for Wireless Sensor Networks and the Internet of Things, and extensively tested using Cooja simulator for Tmote Sky nodes with 6-sectored antennas. The neighbor discovery times were analyzed and analytical expressions were found showing that the time needed to discover all sensor nodes of a network is proportional to the number of nodes in that network. The proposed enhancements to SAND protocol speed up the discovery process up to K times per node, being K the number of sectors of the sectored antenna. Our experiments based on simulations show that for a 6-sectored antenna the time is reduced by 4 times per node, with a greater impact in time and power consumption in networks of increasing size. The Q-SAND protocol performance has been verified through simulations for different network topologies and sizes and compared with that of SAND.**

***Index Terms*—Wireless sensor networks, neighbor discovery, sectored antennas, directional antennas.**

I. Introduction

Directional antennas offer some advantages over omnidirectional antennas. In the first place, as the transmission beam is concentrated, the nodes can reach longer distances with the same transmission power or analogously, for a given transmission range, they require less transmission power. In second place, interference between neighbors is significantly reduced, allowing more simultaneous transmissions in the network, and thus increasing its capacity. Nevertheless, some simple aspects of nodes communication become challenging with directional antennas. It is the case of broadcast messages and neighbor discovery. The main challenges are: i) we have to repeat a message in every direction to ensure that every possible node receives it, ii) the receiver node also needs to be pointing to the transmitter node, iii) there could be many combinations of sectors discovered for a single pair of nodes.

SAND (Sectored-Antenna Neighbor Discovery) protocol [1] solves the neighbor discovery problem in wireless networks with sectored antennas by serializing the neighbor discovery process sequentially in time. One of the main disadvantages of SAND protocol is the time it takes to complete, which is increasingly linear with the number of nodes in the network.

The main contributions of the present work are: i) an open-source implementation of SAND protocol using Contiki OS [2], a popular operating system for the IoT, ii) a study of the neighbor discovery time taken by SAND protocol, iii) the proposal of enhancements to SAND protocol, resulting in Q-SAND (Quick SAND) protocol, and iv) a comparison between SAND and Q-SAND based on simulations.

The rest of this document is organized as follows. In Section II, we present the sectored antennas. In Section III we describe SAND protocol as was initially proposed by Felemban et al., proposing modifications in Section V, resulting in Q-SAND. In Section VI we explain some details of our implementation of both SAND and Q-SAND, showing then the simulation results in Section VII. Finally, we conclude the paper in Section VIII.

II. Sectored antennas

The originally proposed protocol SAND and our enhanced version are designed for sectored antennas. A K sectored antenna has K parasitic elements defining K sectors, each of which covers $\frac{360^\circ}{K}$ of the azimuth, as shown in Fig. 1. The K sectors together cover the entire azimuth. The parasitic elements can be electronically switched to operate as reflectors or directors in order to select a sector at will. An example of a 6-sectored antenna is the SPIDA (Sics Parasitic Interference Directional Antenna) designed in [3], evaluated in [4] and characterized in [5], showing that it has a very good directional antenna performance. A model of SPIDA antenna was used for the simulations in Cooja to assess the protocols. The antenna gain as a function of the angles follows a sinusoidal pattern, as proposed in [6]; $g_{dB}(\theta) = -1{,}565 + 5{,}835\cos(\theta - \theta_0)$, where θ_0 is the pointing direction.

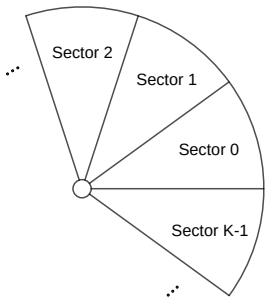


Fig. 1: K sectored-antenna.

III. Description of SAND Protocol

This section describes SAND protocol initially proposed by Felemban et al. [1] for the sake of completeness and in order

to ease the understanding of the proposed improvement of this work.

SAND is a neighbor discovery protocol for sensor nodes with sectored-antennas[1]. This protocol is based on the idea of serializing the neighbor discovery process sequentially in time. This means that there is always a single node discovering neighbors at a time. To achieve this, a token is used to identify the node that is discovering neighbors. The node that holds it is called the Token-Holder (TH) node.

The idea is not only to discover neighbor nodes, but also to find the sector combination between them that results in the best link quality. To do so, SAND proposes to explore all the K^2 sector combinations (K possible sectors of the TH node and K possible sectors of its neighbors). To carry out this exploration, the TH node and its neighbors need a mechanism to synchronize and know when to change sector (Hone-In Mechanism). Once synchronized, all the sector combinations are explored (Hello-Reply Mechanism). Finally, the TH node passes the token to a discovered neighbor node that has not discovered neighbors yet or, in case all its neighbors have already discovered neighbors, to the node from which it received the token initially (Token Passing Mechanism or Token Releasing Mechanism). If the TH has just finished discovering neighbors, it passes the token through the Token Passing Mechanism, otherwise through the Token Releasing Mechanism. The reason for this distinction is explained below, in Section III-D.

Summarizing, the protocol is formed by the three steps introduced before:

1) Hone-In Mechanism
2) Hello-Reply Mechanism
3) Token Passing Mechanism or Token Releasing Mechanism

Next, these steps are described in more detail.

A. Initialization

All nodes except the first TH node start in Fast Scan mode. In this mode, the nodes scan for activity in the network, switching its active sector every t_{switch}. It is important to note that initially the nodes are not synchronized.

B. Hone-In Mechanism

As mentioned before, the Hone-In Mechanism is necessary to synchronize the TH node with all its neighbors. The TH sends periodically every $t_{Hone\text{-}In}$ a number h of Hone-In messages per each sector. The message carries the information of the number of messages remaining (including the current message) before starting the next step: the Hello-Reply Mechanism. So the first Hone-In message will contain hK, the second $hK-1$, until the last one, that will contain 1 (in total, hK Hone-In messages will be sent). This mechanism is shown in Fig. 2 for a 4-sectored antenna and 3 nodes.

[1]The extended paper [7] proposes a modification to allow token recovery. This modification is not considered in this work since it is independent, and could be introduced later.

In turn, when receiving a Hone-In message containing the number m, the neighbor nodes (that are in Fast Scan mode) will set a timer that will expire in $mt_{Hone\text{-}In}$ and cease switching sectors. In this way, a local synchronization between the TH node and its neighbors is achieved, as the timer of all its neighbors will expire (ideally) at the same time. The total duration of the Hone-In Mechanism is $T_{HI} = hKt_{Hone\text{-}In}$.

In order to guarantee that all the neighbor nodes have the chance to receive messages in all its sectors, the switching time t_{switch} must be greater than $t_{Hone\text{-}In}$. t

To minimize the duration of the Hone-In Mechanism, the number of messages h must be also minimized. Nevertheless, h must be large enough to ensure that at least 1 message could be received for each one of the K^2 sector combinations, so h must satisfy $h > K\frac{t_{switch}}{t_{Hone\text{-}In}}$.

C. Hello-Reply Mechanism

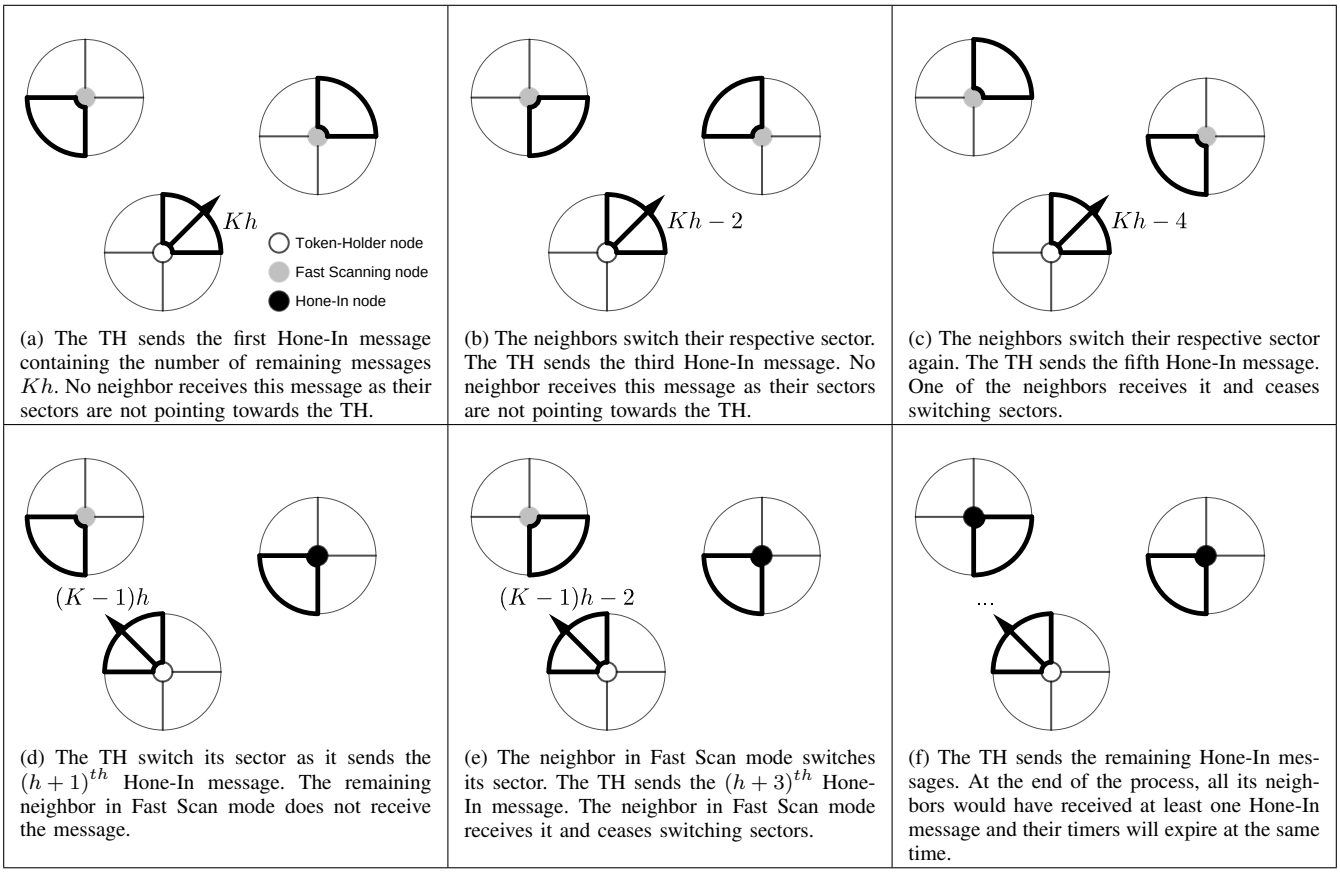
Once the TH node has sent the hK Hone-In messages, the Hello-Reply Mechanism starts. On the side of the neighbor nodes, such a mechanism starts when the timer set during the Hone-In Mechanism expires.

The idea is to explore the K^2 sector combinations in a predetermined way.

To accomplish this, the TH node sends Hello messages, to which the neighbor nodes that receive it respond with a Reply message containing its ID, its current sector and a bit indicating whether it has already done neighbor discovery or not (this is for the TH node to determine to which neighbor to pass the token after finishing the neighbor discovery process). When receiving a Reply message, the TH node adds to its neighbor discovery table the corresponding neighbor with its ID, current sector (both TH and neighbor current sectors), the neighbor discovery bit mentioned above and a link quality indicator, such as the RSSI.

Since many neighbors can respond to the same Hello message, there can be a collision between the Reply messages, causing those neighbors not to be discovered for that sector combination. To reduce the number of collisions, the time after a Hello message is divided into a number N_{slots} of time slots of duration t_{slots}. The neighbor nodes choose a time slot randomly to send their Reply message. To reduce the number of collisions even more, the TH repeats the Hello message N_{rounds} times per each sector combination, and adds to each Hello message the ID of every node that has already been discovered for the current sector combination (this type of message is called Long-Hello message). The nodes that recognize its ID in the Long-Hello message know they have already been discovered, so they do not respond with a Reply message.

At the end of the mechanism, the TH node will proceed to pass the token. The neighbor nodes will activate the sector in which the Hone-In message was initially received. The total duration of the Hello-Reply Mechanism is $T_{HR} = K^2 N_{rounds} N_{slots} t_{slots}$.



(a) The TH sends the first Hone-In message containing the number of remaining messages Kh. No neighbor receives this message as their sectors are not pointing towards the TH.

(b) The neighbors switch their respective sector. The TH sends the third Hone-In message. No neighbor receives this message as their sectors are not pointing towards the TH.

(c) The neighbors switch their respective sector again. The TH sends the fifth Hone-In message. One of the neighbors receives it and ceases switching sectors.

(d) The TH switch its sector as it sends the $(h+1)^{th}$ Hone-In message. The remaining neighbor in Fast Scan mode does not receive the message.

(e) The neighbor in Fast Scan mode switches its sector. The TH sends the $(h+3)^{th}$ Hone-In message. The neighbor in Fast Scan mode receives it and ceases switching sectors.

(f) The TH sends the remaining Hone-In messages. At the end of the process, all its neighbors would have received at least one Hone-In message and their timers will expire at the same time.

Fig. 2: Hone-In process for $K = 4$ and $t_{switch} = 2t_{Hone\text{-}In}$. In this case, h must be greater than $K\frac{t_{switch}}{t_{Hone\text{-}In}} = 4 \times 2 = 8$. At the beginning of the mechanism, all the neighbor nodes are in Fast Scan mode. At the end of the mechanism, all the neighbor nodes have received at least one Hone-In message from the TH and their respective active sectors are pointing towards it.

D. Token Passing and Token Releasing Mechanisms

Once the Hello-Reply Mechanism is completed, the TH node goes over its neighbor table, searching for the first node that has not done neighbor discovery yet (this is checked with the bit mentioned in Section III-C). If such a node is found, the TH node proceeds to pass the token to it. Otherwise (if such a node is not found), the TH node proceeds to pass the token to the node from which it originally received the token.

The token passage between nodes then forms a tree, whose root is the node that started the neighbor discovery process (the first TH node).

To pass the token, 2 different cases need to be distinguished.

1) Token Passing Mechanism: In case the TH node has just finished discovering neighbors, all its neighbors will point to a specific sector, directing its antenna to the TH node. It is necessary for the TH to send GoToFastScan messages signaling its neighbors to start Fast Scan mode (otherwise the neighbor nodes would stay with the same active sector forever). One single GoToFastScan message per sector ensures that all the neighbors receive at least one of such messages. Along with the GoToFastScan message, the ID of the node that will receive the token next is also sent. This is done for the next TH to know that it should not go to Fast Scan mode as it will receive the token.

Then, the TH node sends the token and waits for an acknowledgment. When the acknowledgment is received, the older TH node goes to Fast Scan mode.

The total duration of the Token Passing Mechanism is $T_{TP} = (K-1)\, t_{GoToFastScan} + t_{TokenAck}$, where $t_{TokenAck}$ is the time it takes to send the token and receive the acknowledgment.

2) Token Releasing Mechanism: In case the TH node had already discovered neighbors (it is not the first time it has the token), all its neighbors will be in Fast Scan mode. It is necessary for the TH to make a reduced version of the Hone-In Mechanism (we call it Mini-Hone-In), just for the node that will receive the token next.

Once enough Mini-Hone-In messages to guarantee the node is paying attention are sent, the TH node sends the token to it, and waits for an acknowledgment. When the acknowledgment is received, the older TH node goes to Fast Scan mode.

The total duration of the Token Releasing Mechanism is $T_{TR} = (h-1)\, t_{Hone\text{-}In} + t_{TokenAck}$.

So far, we described SAND protocol proposed by Felemban et al.; we will next find an analytical expression for the total

time taken by the protocol.

IV. Neighbor discovery time theoretical analysis

After the protocol ends, all n nodes in the network have discovered their neighbors forming a tree graph.

The tree graph is formed by the token passage and has n vertexes (because each node is represented by a vertex) and $n-1$ edges (because it is a tree). So there have been n Hone-In, Hello-Reply and Token-Passing mechanisms independently of the network topology, and $2\,(n-1)$ Token Passing and Token Releasing mechanisms (together). There have been then $2\,(n-1)-n=n-2$ Token Releasing mechanisms, independently of the network topology.

We have then

$$T_{SAND} = n\,(T_{HI}+T_{HR}+T_{TP})+(n-2)\,T_{TR}. \qquad (1)$$

The neighbor discovery time is independent of the network topology, and is increasingly linear with the number of nodes of the network.

V. Modifications to SAND: Q-SAND

One of the drawbacks of SAND protocol is that since the neighbor discovery time is increasingly linear with the number of nodes in the network, if the network is very big, so is the time the protocol needs to complete. Another issue is the reliability of the protocol that depends on the token (if the token is lost, the protocol fails), but the authors of SAND have already solved this issue by proposing a new version of the protocol [7] allowing token recovery when it is lost.

We then proposed an improvement to the protocol to reduce the neighbor discovery time. We call this modified protocol Q-SAND (for Quick SAND).

Q-SAND is capable of finding the "best" combination of sectors (that is the one with highest RSSI in our implementation) between a pair of neighbor nodes. It can be assumed in first place that the links are reciprocal (as shown by [8]) and in second place that a node has maximum link quality with a neighbor with its sector pointing to him and vice-versa. The last assumption is valid for line of sight and open environments. A mathematical way to express this is that a node has a maximum link quality indicator with a neighbor when their active sector is the one that contains the line that joins both nodes. Examples are shown in Fig. 3 for K even and in Fig. 4 for K odd.

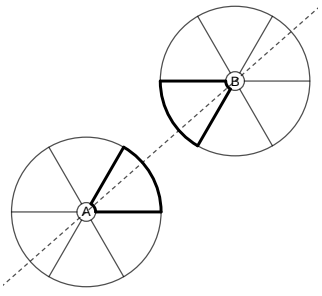
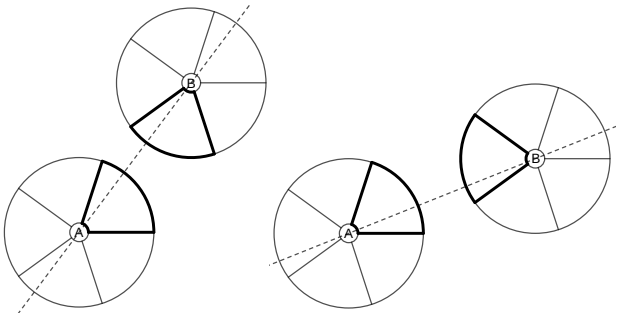


Fig. 3: Best sector combination between two nodes A and B for K even ($K=6$).



(a) (b)

Fig. 4: Best sector combination between two nodes A and B for K odd ($K=5$).

With that assumption in mind, it is clear that if the sector of node A that contains the line that joins both nodes is $S_A \in [0,...,K-1]$, then the sector of node B that contains the same line is S_B

- $\left(S_A+\frac{K}{2}\right) \bmod K$ if K is even, and
- $\left(S_A+\frac{K-1}{2}\right) \bmod K$ or $\left(S_A+\frac{K+1}{2}\right) \bmod K$ if K is odd.

We see that if K is even, we only need to try K sector combinations to find the best one (and $2K$ sector combinations if K is odd), compared to the K^2 sector combinations that are needed without this modification.

The main drawback is that we would only find the best sector combination, so this improvement is not applicable if we would like to obtain all the discovered sector combinations between two nodes and not just the best one.

An extra consideration compared to SAND is that the nodes need to be oriented in the same manner, for example, all nodes with sector 0 pointing to the north, so this improvement is not applicable if the nodes are free to rotate.

The advantage is that the new duration of the Hello-Reply Mechanism (which is the mechanism that commonly takes longer) is now

$$\begin{cases} KN_{rounds}N_{slots}t_{slots} = \frac{T_{HR}}{K} & \text{if } K \text{ is even} \\ 2KN_{rounds}N_{slots}t_{slots} = 2\frac{T_{HR}}{K} & \text{if } K \text{ is odd} \end{cases} \qquad (2)$$

Then, with Q-SAND, we have that the total time T_{Q-SAND} the protocol takes is

$$\begin{cases} n\left(T_{HI}+\frac{T_{HR}}{K}+T_{TP}\right)+(n-2)\,T_{TR} & \text{if } K \text{ is even} \\ n\left(T_{HI}+2\frac{T_{HR}}{K}+T_{TP}\right)+(n-2)\,T_{TR} & \text{if } K \text{ is odd} \end{cases} \qquad (3)$$

The neighbor discovery time in Q-SAND compared to SAND is greatly reduced when $T_{HR} \gg (T_{HI}+T_{TP}+T_{TR})\,K$ (up to K times if K is even and up to $\frac{K}{2}$ times if K is odd), and little reduced when $T_{HR} \ll T_{HI}+T_{HR}+T_{TP}$.

VI. Implementation

We implemented a basic and open-source version of both SAND and Q-SAND protocols in the programming language

C, using Contiki OS [2], a popular operating system for the IoT, with a plug-in for Tmote Sky [9] with 6-sectored SPIDA nodes, so $K = 6$ in this case. We adopted the 6LoWPAN stack protcol included with Contiki standard distribution. We used the C programming language preprocessor macro `#define` to define all the protocol parameters: t_{switch}, $t_{Hone\text{-}In}$, h, N_{slots}, t_{slots}, N_{rounds} and $t_{GoToFastScan}$ so they can be easily modified.

We defined several types of broadcast messages according to SAND and Q-SAND: Hone-In message, Hello message, Reply message, GoToFastScan message, Mini-Hone-In message, Token message and Acknowledgment message. These messages are used by the nodes to communicate using link-local addresses.

In our implementation, when the protocols (both SAND and Q-SAND) are initialized, the first TH is the one with ID = 1. In practice, this node usually is the border router of the network. Once the TH finishes discovering neighbors, it checks its neighbor table and passes the token to the neighbor with lowest ID that has not discovered neighbors yet.

Our implementation was based on a state machine, whose diagram is shown in Fig. 5. As we see, the three rows represent each one of the three steps that form the protocol (Hone-In Mechanism, Hello-Reply Mechanism, Token Passing and Token Releasing Mechanisms), and the two columns represent whether the node is acting as the TH (left) or as the neighbor node being discovered (right).

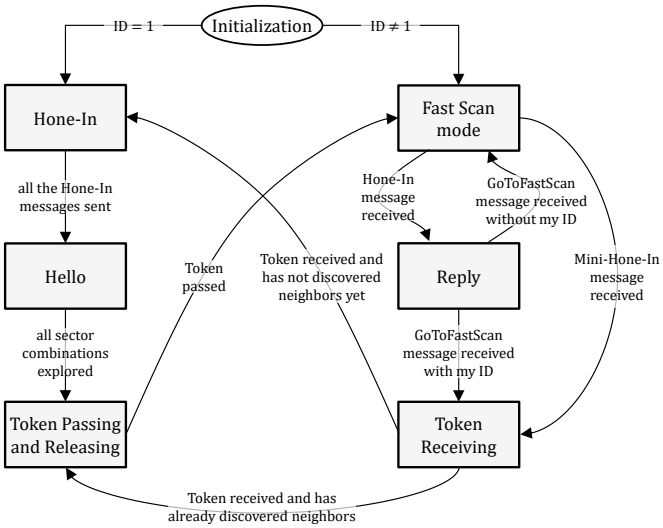


Fig. 5: State diagram of protocols SAND and Q-SAND.

The neighbor discovery table is saved in each mote's RAM. This table contains the node's sector, the neighbor ID, the neighbor sector, the RSSI, and a bit indicating whether the neighbor has already been a TH or not. This bit is used by the TH to know to which node to pass the token, as described in Section III. When the protocol ends and all the nodes have completed their respective neighbor discovery table, this bits should be all set.

To reduce the memory burden of a neighbor discovery table saving all the sector combination links, we decided to save only the sector combination with highest RSSI. This results in one single table entry per neighbor node.

Another simplification of the protocols implementation was that the neighbor discovery tables are not passed along with the token, so each node of the network only has access to its own neighbor discovery table. This simplification reduces the size of the messages exchanged, as well as the memory burden of each node, but does not allow to gather the neighborhood information in a centralized location. However, as each node does not receive the other nodes' tables, it has to update the neighbor discovery bit somehow. We did this by passing the IDs of all the nodes that have already discovered neighbors along with the token when it is passed upwards in the tree. When the token is passed downwards in the tree, the ID of the TH is passed along with the token, for the new TH to know to which neighbor pass the token when all its neighbors have completed the neighbor discovery process.

VII. Results

We simulated the implementation described in the previous section with Cooja (a network simulation tool including an instruction-level emulator for the node microcontroller and radio propagation models) for different networks. The parameters of both SAND and Q-SAND protocols used in the simulations are shown in Table I.

TABLE I: Parameters of SAND and Q-SAND protocols used in the simulations

Protocol mechanism	Parameter	Value
Fast Scan mode	t_{switch}	31,25 ms
Hone-In Mechanism	$t_{Hone\text{-}In}$	15,625 ms
	h	12
Hello-Reply Mechanism	N_{slots}	5
	t_{slots}	15,625 ms
	N_{rounds}	5
Token Passing Mechanism	$t_{GoToFastScan}$	15,625 ms

First, we simulated different random networks with the same number of nodes but different topologies, in order to verify the results of the theoretical analysis in Section IV. We obtained the exact same neighbor discovery time for networks with the same number of nodes, confirming the independence of the neighbor discovery time with respect to the network topology for both SAND and Q-SAND protocols.

We then proceeded to simulate a random network with 16 nodes for both protocols. Fig. 6 shows the simulated network. The colored node is the border router of the network with ID = 1, and is the node that starts with the discovery process. The tree formed by the token passage is shown with continuous lines. The links between nodes are shown both in continuous and dotted lines.

The resulting neighbor discovery tables were the same for both protocols: the exact same best combination of sectors was discovered for each pair of nodes.

Table II shows the neighbor discovery table of node 1. We see that -as we stated in Section V-, if the best sector

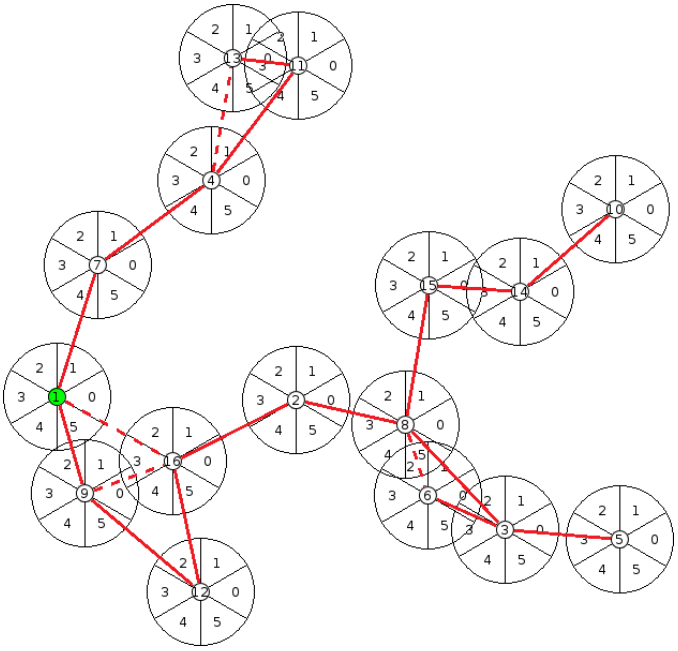


Fig. 6: Simulated network with 16 nodes.

TABLE II: Neighbor Discovery Table of Node 1

Node 1 sector	Neighbor ID	Neighbor sector	RSSI
1	7	4	-42
5	9	2	-39
0	16	3	-44

combination between node 1 and a neighbor is achieved for node 1 sector S_1, then it is achieved for its neighbor sector $\left(S_1 + \frac{K}{2}\right) \bmod K$. We can also see that the RSSI decreases with distance.

In the table of each of all nodes, we verified that each one of the best sector combination $(sectorA, sectorB)$ was achieved by one of the following combinations: $(0,3)$, $(1,4)$, $(2,5)$, $(3,0)$, $(4,1)$ or $(5,2)$.

We simulated then SAND and Q-SAND protocols for different random networks with up to 16 nodes. In all cases, the best sector combinations discovered by both protocols had the same RSSI, showing that Q-SAND effectively finds best sector combinations. Fig. 7 shows the total time taken by both protocols, both theoretical results calculated through equations 1 and 3 with the parameters shown in Table I and simulations results. We can see that simulation results match very well the theoretical equations for both protocols. The total time taken is increasingly linear with the number of nodes in the network for both protocols, taking 15,5 s per node for SAND and 3,781 s for Q-SAND. We see that with the set of parameters used, Q-SAND is more than 4 times faster than SAND in discovering neighbors.

VIII. Conclusion

A fully directional neighbor discovery protocol for wireless networks with sectored antennas has been proposed. A functional version of both SAND and Q-SAND protocols has been

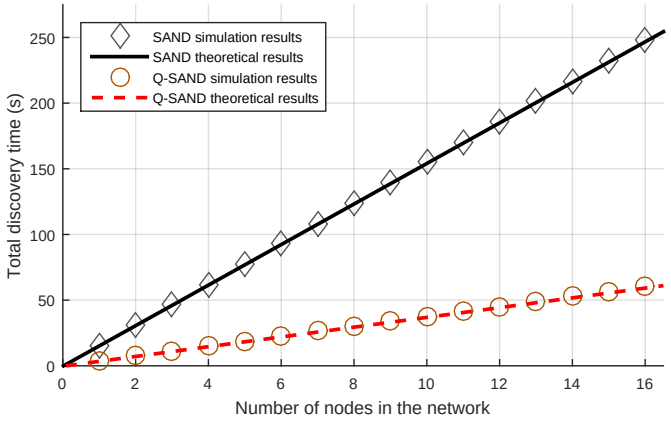


Fig. 7: Neighbor discovery time vs. number of nodes in the network.

analyzed, implemented and simulated. Both protocols have been compared through theoretical analysis and simulation results, showing that Q-SAND outperforms SAND in protocol duration. This is particularly useful for networks with a large number of nodes or with K-sectored antennas with a sizeable number of sectors K (i.e. $K = 12$).

Acknowledgment

The authors would like to thank Fondo María Viñas for supporting this project (FMV_1_2014_1_104872).

References

[1] E. Felemban, R. Murawski, E. Ekici, S. Park, K. Lee, J. Park, and Z. Hameed, "SAND: Sectored-Antenna Neighbor Discovery Protocol for Wireless Networks," in *2010 7th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, June 2010, pp. 1–9.

[2] A. Dunkels, B. Gronvall, and T. Voigt, "Contiki - a lightweight and flexible operating system for tiny networked sensors," in *29th Annual IEEE International Conference on Local Computer Networks*, Nov 2004, pp. 455–462.

[3] M. Nilsson, "SPIDA: A Direction-Finding Antenna for Wireless Sensor Networks," in *Real-World Wireless Sensor Networks*, P. J. Marron, T. Voigt, P. Corke, and L. Mottola, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 138–145.

[4] E. Öström, L. Mottola, and T. Voigt, "Evaluation of an Electronically Switched Directional Antenna for Real-World Low-Power Wireless Networks," in *Real-World Wireless Sensor Networks*, P. J. Marron, T. Voigt, P. Corke, and L. Mottola, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 113–125.

[5] B. Rodríguez, J. Schandy, J. P. González, L. Steinfeld, and F. Silveira, "Fabrication and characterization of a directional SPIDA antenna for wireless sensor networks," in *2017 IEEE URUCON*, Oct 2017.

[6] B. Wei, A. Varshney, N. Patwari, W. Hu, T. Voigt, and C. T. Chou, "dRTI: Directional Radio Tomographic Imaging," in *Proceedings of the 14th International Conference on Information Processing in Sensor Networks*, ser. IPSN '15. New York, NY, USA: ACM, 2015, pp. 166–177.

[7] R. Murawski, E. Felemban, E. Ekici, S. Park, S. Yoo, K. Lee, J. Park, and Z. Hameed Mir, "Neighbor discovery in wireless networks with sectored antennas," *Ad Hoc Networks*, vol. 10, no. 1, pp. 1 – 18, 2012.

[8] A. Varshney, T. Voigt, and L. Mottola, *Using Directional Transmissions and Receptions to Reduce Contention in Wireless Sensor Networks*. Cham: Springer International Publishing, 2014, pp. 205–213. [Online]. Available: https://doi.org/10.1007/978-3-319-03071-5_21

[9] *Tmote Sky Datasheet*, Moteiv Corporation, June 2006, rev. 1.0.2.