Continuous Neighbor Discovery in Asynchronous Sensor Networks

Reuven Cohen and Boris Kapchits
Department of Computer Science
Technion
Haifa Israel

***Abstract*—In most sensor networks the nodes are static. Nevertheless, node connectivity is subject to changes because of disruptions in wireless communication, transmission power changes, or loss of synchronization between neighboring nodes. Hence, even after a sensor is aware of its immediate neighbors, it must continuously maintain its view, a process we call *continuous neighbor discovery*. In this work we distinguish between neighbor discovery during sensor network initialization and continuous neighbor discovery. We focus on the latter and view it as a joint task of all the nodes in every connected segment. Each sensor employs a simple protocol in a coordinate effort to reduce power consumption without increasing the time required to detect hidden sensors.**

I. Introduction

A sensor network may contain a huge number of simple sensor nodes that are deployed at some inspected site. In large areas, such a network usually has a mesh structure. In this case, some of the sensor nodes act as routers, forwarding messages from one of their neighbors to another. The nodes are configured to turn their communication hardware on and off to minimize energy consumption. Therefore, in order for two neighboring sensors to communicate, both must be in active mode.

In the sensor network model considered in this paper, the nodes are placed randomly over the area of interest and their first step is to detect their immediate neighbors – the nodes with which they have a direct wireless communication – and to establish routes to the gateway. In networks with continuously heavy traffic, the sensors need not invoke any special neighbor discovery protocol during normal operation. This is because any new node, or a node that has lost connectivity to its neighbors, can hear its neighbors simply by listening to the channel for a short time. However, for sensor networks with low and irregular traffic, a special neighbor discovery scheme should be used. This paper presents and analyzes such a scheme.

Despite the static nature of the sensors in many sensor networks, connectivity is still subject to changes even after the network has been established. The sensors must continuously look for new neighbors in order to accommodate the following situations:

1) Loss of local synchronization due to accumulated clock drifts.
2) Disruption of wireless connectivity between adjacent nodes by a temporary event, such as a passing car or animal, a dust storm, rain or fog. When these events are over, the hidden nodes must be rediscovered.
3) The ongoing addition of new nodes, in some networks to compensate for nodes which have ceased to function because their energy has been exhausted.
4) The increase in transmission power of some nodes, in response to certain events, such as detection of emergent situations.

For these reasons, detecting new links and nodes in sensor networks must be considered as an ongoing process. In the following discussion we distinguish between the detection of new links and nodes during initialization, i.e., when the node is in Init state, and their detection during normal operation, when the node is in Normal state. The former will be referred to as *initial neighbor discovery* whereas the latter will be referred to as *continuous neighbor discovery.* While previous works [1], [2], [3] address initial neighbor discovery and continuous neighbor discovery as similar tasks, to be performed by the same scheme, we claim that different schemes are required, for the following reasons:

- Initial neighbor discovery is usually performed when the sensor has no clue about the structure of its immediate surroundings. In such a case, the sensor cannot communicate with the gateway and is therefore very limited in performing its tasks. The immediate surroundings should be detected as soon as possible in order to establish a path to the gateway and contribute to the operation of the network. Hence, in this state, more extensive energy use is justified. In contrast, continuous neighbor discovery is performed when the sensor is already operational. This is a long-term process, whose optimization is crucial for increasing network lifetime.
- When the sensor performs continuous neighbor discovery, it is already aware of most of its immediate neighbors and can therefore perform it together with these neighbors in order to consume less energy. In contrast, initial neighbor discovery must be executed by each sensor separately.

Figure 1 shows a typical neighbor discovery protocol. In this protocol, a node becomes active according to its duty cycle. Let this duty cycle be α in Init state and β in Normal state. We want to have $\beta \ll \alpha$. When a node becomes active, it transmits periodical HELLO messages and listens for similar messages from possible neighbors. A node that receives a HELLO message immediately responds and the two nodes

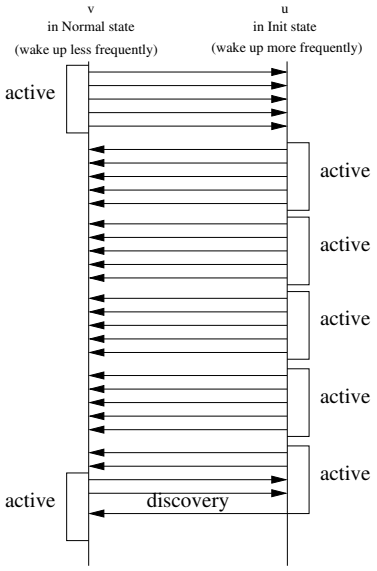


Fig. 1. The transmission of HELLO messages in Init and Normal states

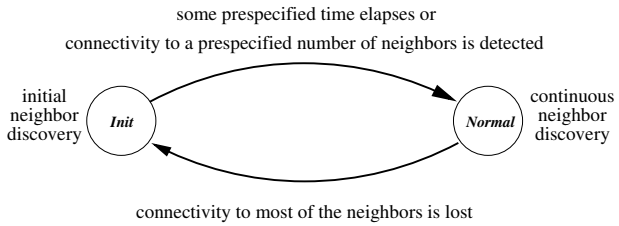


Fig. 2. Continuous neighbor discovery vs. initial neighbor discovery in sensor networks

can invoke another procedure to finalize the setup of their joint wireless link.

To summarize, in the Init state, a node has no information about its surroundings and therefore must remain active for a relatively long time in order to detect new neighbors. In contrast, in the Normal state the node must use a more efficient scheme. Such a scheme is the subject of our study. Figure 2 summarizes this idea. When node u is in the Init state, it performs initial neighbor discovery. After a certain time period, during which the node is expected, with high probability, to find most of its neighbors, the node moves to the Normal state, where continuous neighbor discovery is performed. A node in the Init state is also referred to in this paper as a hidden node and a node in the Normal state is referred to as a segment node.

The main idea behind the continuous neighbor discovery scheme we propose is that the task of finding a new node u is divided among all the nodes that can help v to detect u. These nodes are characterized as follows: (a) they are also neighbors of u; (b) they belong to a connected segment of nodes that have already detected each other; (c) node v also belongs to this segment. Let $deg_S(u)$ be the number of these nodes. This variable indicates the in-segment degree of a hidden neighbor u. In order to take advantage of the proposed discovery scheme, node v must estimate the value of $deg_S(u)$.

The rest of the paper is organized as follows. In Section II we present related work. Section III presents a basic scheme and problem definition. The core of the paper is Section IV, which presents three methods for estimating the in-segment degree of a hidden neighbor and analyzes their accuracy. Section V concentrates on a special case where the network nodes are uniformly distributed. For this case, we are able to find a numeric value for the accuracy of the three methods presented in IV. Section VI presents our continuous neighbor discovery scheme, which is based on our findings in Section IV. Section VII presents simulation results that demonstrate the scheme's efficiency. It also includes a discussion of problems that arise when two small segments have to detect one another. Finally, Section VIII concludes this work.

II. Related Work

In a WiFi network operating in centralized mode, a special node, called an access point, coordinates access to the shared medium. Messages are transmitted only to or from the access point. Therefore, neighbor discovery is the process of having a new node detected by the base station. Since energy consumption is not a concern for the base station, discovering new nodes is rather easy. The base station periodically broadcasts a special HELLO message[1]. A regular node that hears this message can initiate a registration process. The regular node can switch frequencies/channels in order to find the best HELLO message for its needs. Which message is the best might depend on the identity of the broadcasting base station, on security considerations, or on PHY layer quality (signal-to-noise ratio). Problems related to possible collisions of registration messages in such a network are addressed in [4]. Other works try to minimize neighbor discovery time by optimizing the broadcast rate of the HELLO messages [1], [5], [6], [7], [8]. The main differences between neighbor discovery in WiFi and in mesh sensor networks are that neighbor discovery in the former is performed only by the central node, for which energy consumption is not a concern. In addition, the hidden nodes are assumed to be able to hear the HELLO messages broadcast by the central node. In contrast, neighbor discovery in sensor networks is performed by every node, and hidden nodes cannot hear the HELLO messages when they sleep.

In mobile ad-hoc networks (MANETs), nodes usually do not switch to a special sleep state. Therefore, two neighboring nodes can send messages to each other whenever their physical distance allows communication. AODV [9] is a typical routing protocol for MANETs. In AODV, when a node wishes to send a message to another node, it broadcasts a special RREQ (route request) message. This message is then broadcast by every node that hears it for the first time. The same message is used for connectivity management, as part of an established route maintenance procedure, aside from which there is no special neighbor discovery protocol.

Minimizing energy consumption is an important target design in Bluetooth [10]. As in WiFi, the process of neighbor discovery in Bluetooth is also asymmetric. A node that wants

[1]The various systems and protocols that employ neighbor discovery use different names for their control message, such as BEACON or NEIGHBOR-DISCOVERY. For consistency, throughout this paper, we refer to all these control messages as HELLO.

to be discovered switches to an inquiry scan mode, whereas a node that wants to discover its neighbors enters the inquiry mode. In the inquiry scan mode, the node listens for a certain period on each of the 32 frequencies dedicated to neighbor discovery, while the discovering node passes through these frequencies one by one and broadcasts HELLO in each of them. This process is considered to be energy consuming and slow. A symmetric neighbor discovery scheme for Bluetooth is proposed in [11]. The idea is to allow each node to switch between the inquiry scan mode and the inquiry mode.

The 802.15.4 standard [12] proposes a rather simple scheme for neighbor discovery. It assumes that every coordinator node issues one special "beacon" message per frame, and a newly deployed node has only to scan the available frequencies for such a message. However, the standard also supports a beaconless mode of operation. Under this mode, a newly deployed node should transmit a beacon request on each available channel. A network coordinator that hears such a request should immediately answer with a beacon of its own. However, this scheme does not supply any bound on the hidden neighbor discovery time.

Neighbor discovery in wireless sensor networks is addressed in [2]. The authors propose a policy for determining the transmission power of every node, in order to guarantee that each node detects at least one of its neighbors using as little power as possible.

In [1], the authors study the problem of neighbor discovery in static wireless ad hoc networks with directional antennas. At each time slot, a sensor either transmits HELLO messages in a random direction, or listens for HELLO messages from other nodes. The goal is to determine the optimal rate of transmission and reception slots, and the pattern of transmission directions.

In [6], neighbor discovery is studied for general ad-hoc wireless networks. The authors propose a random HELLO protocol, inspired by ALOHA. Each node can be in one of two states: listening or talking. A node decides randomly when to initiate the transmission of a HELLO message. If its message does not collide with another HELLO, the node is considered to be discovered. The goal is to determine the HELLO transmission frequency, and the duration of the neighbor discovery process.

In [5], the sensor nodes are supposed to determine, for every time slot, whether to transmit HELLO, to listen, or to sleep. The optimal transition rate between the three states is determined using a priori knowledge of the maximum possible number of neighbors.

In [13], the $Disco$ algorithm is proposed for scheduling the wake-up times of two nodes that wish to find each other. For this algorithm, each node chooses a prime number; the choice depends on the required discovery time. Using the Chinese Remainders theorem, it is proved that the wake-up periods of the nodes will overlap within the required time. However, [13] does not discuss the problem of many sensors in the same segment collaborating to reduce the energy they expend for discovering hidden nodes.

As discussed in Section I, the sensor network nodes spend most of their time in sleep/idle mode, where they cannot receive or transmit messages. Therefore, the node's ability to discover a new neighbor is limited to periods when both are active. In [3], this neighbor discovery model is shown to be similar to the well-known "birthday paradox." In our work we use a similar analysis, in order to find the probability that a node will be discovered by one of its neighbors.

A novel low-power listening (LPL) technique, proposed in [14] to overcome sensor synchronization problems, is implemented by the B-MAC protocol [15]. The transmission of a packet is preceded by a special preamble. This preamble is long enough to be discovered if each node performs periodic channel sampling. However, this technique can usually not be used for initial neighbor discovery, and cannot be used at all for continuous neighbor discovery, because it actually requires the node to stay awake during the entire time it is searching for a new neighbor.

III. A Basic Scheme and Problem Definition

In the following discussion, two nodes are said to be neighboring nodes if they have direct wireless connectivity. We assume that all nodes have the same transmission range, which means that connectivity is always bidirectional. During some parts of our analysis, we also assume that the network is a unit disk graph; namely, any pair of nodes that are within transmission range are neighboring nodes. Two nodes are said to be *directly connected* if they have discovered each other and are aware of each other's wake-up times. Two nodes are said to be connected if there is a path of directly connected nodes between them. A set of connected nodes is referred to as a segment. Consider a pair of neighboring nodes that belong to the same segment but are not aware that they have direct wireless connectivity. See, for example, nodes a and c in Figure 4(a). These two nodes can learn about their hidden wireless link using the following simple scheme, which uses two message types: (a) SYNC messages for synchronization between all segment nodes, transmitted over known wireless links; (b) HELLO messages for detecting new neighbors.

Scheme 1 (detecting all hidden links inside a segment): This scheme is invoked when a new node is discovered by one of the segment nodes. The discovering node issues a special SYNC message to all segment members, asking them to wake up and periodically broadcast a bunch of HELLO messages. This SYNC message is distributed over the already known wireless links of the segment. Thus, it is guaranteed to be received by every segment node. By having all the nodes wake up "almost at the same time" for a short period, we can ensure that every wireless link between the segment's members will be detected. ■

To better understand the benefit of Scheme 1, we now compare its performance to the performance of a trivial algorithm where every node discovers its hidden neighbors independently. When Scheme 1 is used, a hidden node is discovered by all of its in-segment neighbors as soon as it is discovered by the first of them. In contrast, when Scheme 1 is not used, the hidden node is discovered by all of its in-segment neighbors only when it is discovered by the last of them. To analyze the time slots at which these nodes are discovered,

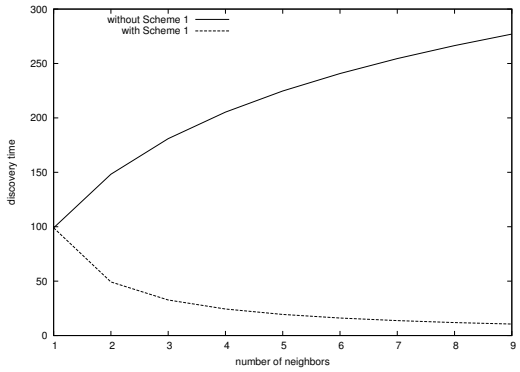


Fig. 3. Discovery delay of non-cooperative vs. cooperative schemes

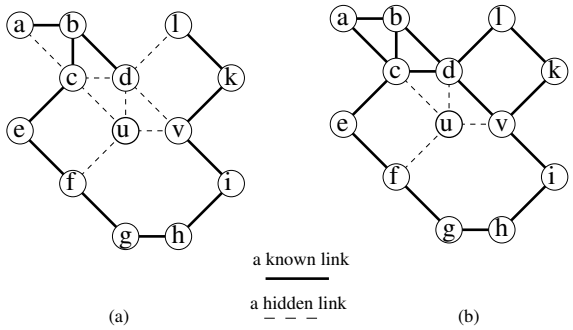


Fig. 4. Segments with hidden nodes and links

suppose that the time axis is divided into slots such that the probability that a node discovers a given hidden neighbor is p. Consider a node u with m in-segment hidden neighbors. The probability that u discovers its first in-segment hidden neighbor only at slot $k+1$ is

$$p_m(k) = (1-p)^{m^k}(1-(1-p)^m).$$

Since p_m has geometric distribution with probability of success equal to $p' = 1-(1-p)^m$, the expected time until the first discovery (first "success") is $E^m = (1-p')/p' = (1-p)^m/(1-(1-p)^m)$. If Scheme 1 is not used, node u discovers all its in-segment hidden neighbors one by one. The expected delay in this case is the expected delay until the first discovery in a set of m neighbors (E^m) plus the expected delay until the first discovery in a set of $m-1$ neighbors (E^{m-1}) and so on, namely, $\sum_{1\le i\le m} E^i$. Figure 3 shows a numerical comparison between the two cases when $p = 0.01$ and the number of in-segment neighbors ranges between 1 and 9.

Scheme 1 allows two neighboring nodes u and v to discover each other if they belong to a connected segment. However, as discussed in Section I, in order for two neighbors not yet connected to the same segment to detect each other, each node should also execute the following scheme:

Scheme 2 (detecting a hidden link outside a segment): Node u wakes up randomly, every $T(u)$ seconds on the average, for a fixed period of time H. During this time it broadcasts several HELLO messages, and listens for possible HELLO messages sent by new neighbors. The value of $T(u)$ is as follows:

- $T(u) = T_I$, if node u is in the Init state of Figure 2.
- $T(u) = T_N(u)$, if node u is in the Normal state of Figure 2, where $T_N(u)$ is computed according to the scheme presented in Section IV.

■

A random wake-up approach is used to minimize the possibility of repeating collisions between the HELLO messages of nodes in the same segment. Theoretically, another scheme may be used, where segment nodes coordinate their wake-up periods to prevent collisions and speed up the discovery of hidden nodes. However, finding an efficient time division is equivalent to the well-known node coloring problem, which is NP-hard and also cannot be well approximated. Since the time period during which every node wakes up is very short, and the HELLO transmission time is even shorter, the probability that two neighboring nodes will be active at the same time is practically 0. In the rare case of collisions, CSMA/CD can be used to schedule retransmissions.

By Scheme 1, the discovery of an individual node by any node in a segment leads to the discovery of this node by all of its neighbors that are part of this segment. Therefore, discovering a node that is not yet in the segment can be considered a joint task of all the neighbors of this node in the segment. As an example, consider Figure 4(a), which shows a segment S and a hidden node u. In this figure, a dashed line indicates a hidden wireless link, namely, a link between two nodes that have not yet discovered each other. A thick solid line indicates a known wireless link. After execution of Scheme 1, all hidden links in S are detected (see Figure 4(b)). The links connecting nodes in S to u are not detected because u does not belong to the segment. Node u has 4 hidden links to nodes in S. Hence, we say that the degree of u in S is $deg_S(u) = 4$. When u is discovered by one of its four neighbors in S, it will also be discovered by the rest of its neighbors in S as soon as Scheme 1 is reinvoked. Consider one of the four segment members that are within range of u, node v say. Although it may know about the segment members within its own transmission range, it does not know how many in-segment neighbors participate in discovering u.

In the next section we study three methods that allow v to estimate the value of $deg_S(u)$ for a hidden node u, and compare their accuracy and applicability.

IV. Estimating the In-segment Degree of a Hidden Neighbor

As already explained, we consider the discovery of hidden neighbors as a joint task to be performed by all segment nodes. To determine the discovery load to be imposed on every segment node, namely, how often such a node should become active and send HELLO messages, we need to estimate the number of in-segment neighbors of every hidden node u, denoted by $deg_S(u)$. In this section we present methods that can be used by node v in the Normal (continuous neighbor discovery) state to estimate this value. Node u is assumed to not yet be connected to the segment, and it is in the Init (initial neighbor discovery) state. Three methods are presented:

1) Node v measures the average in-segment degree of the segment's nodes, and uses this number as an estimate of the in-segment degree of u. The average in-segment degree of the segment's nodes can be calculated by the segment leader. To this end, it gets from every node in the segment a message indicating the in-segment degree of the sending node, which is known due to Scheme 1. We assume that the segment size is big enough for the received value to be considered equal to the expected number of neighbors of every node.
2) Node v discovers, using Scheme 1, the number of its in-segment neighbors, $deg_S(v)$, and views this number as an estimate of $deg_S(u)$. This approach is expected to yield better results than the previous one when the degrees of neighboring nodes are strongly correlated.
3) Node v uses the average in-segment degree of its segment's nodes and its own in-segment degree $deg_S(v)$ to estimate the number of node u's neighbors. This approach is expected to yield the best results if the correlation between the in-segment degrees of neighboring nodes is known. An interesting special case is when the in-segment nodes are uniformly distributed.

The in-segment degree of v and u depends on how the various nodes are distributed in the network. Let X be a random variable that indicates the degree $deg_S(v)$ of v, a uniform randomly chosen node in the segment S. Let Y be a random variable that indicates the degree $deg_S(u)$ of u, a uniform randomly chosen hidden neighbor of v, which we want to estimate. Note that u itself is not aware of the value of Y. Let Y' be the estimated value of Y. Clearly, we want Y' to be as close as possible to Y. We use the mean square error measure (*MSE*) to decide how good an estimate is. The *MSE* is defined as $E((Y-Y')^2)$. Since v and u are two random neighbors in the same graph, X and Y have the same distribution. Let us denote the correlation between X and Y, $\mathrm{corr}(X,Y)$, by $\mathcal{C}$. Throughout the section we assume that $deg_S(v)$ is small compared to the network size.

Denote the average graph degree by μ. Clearly, $E(X) = E(Y) = \mu$. Thus, for the first method the following holds:

$$\begin{aligned} \mathit{MSE}_1 &= E((Y-Y')^2) = E((Y-\mu)^2) \\ &= \mathrm{Var}(Y). \end{aligned} \quad (1)$$

For the second method, we have $Y' = X$. Hence,

$$\begin{aligned} \mathit{MSE}_2 &= E((Y-Y')^2) = E((Y-X)^2) \\ &= \sum_x \sum_y (y-x)^2 P(X=x, Y=y) \\ &= \sum_x \sum_y (y^2 - 2xy + x^2) P(X=x, Y=y) \\ &= E(X^2) + E(Y^2) - 2E(XY). \end{aligned} \quad (2)$$

By the correlation of random variables and the fact that $\mathrm{Var}(X) = \mathrm{Var}(Y)$, we get

$$\mathrm{corr}(X,Y) = \frac{\mathrm{cov}(X,Y)}{\mathrm{Var}(X)}.$$

Using the definition of covariance, we get

$$\begin{aligned} \mathrm{cov}(X,Y) &= E((X-E(X))(Y-E(Y)) \\ &= (XY - XE(Y) - YE(Y) + E(X)E(Y)) \\ &= (XY) - E(X)E(Y) - E(Y)E(X) + E(X)E(Y) \\ &= (XY) - E(X)E(Y). \end{aligned}$$

Hence,

$$\begin{aligned} E(XY) &= \mathrm{cov}(X,Y) + E(X)E(Y) \\ &= \mathrm{corr}(X,Y)\,\mathrm{Var}(X) + E(X)E(Y) \\ &= \mathcal{C}\,\mathrm{Var}(X) + E(X)E(Y). \end{aligned} \quad (3)$$

Substituting into Eq. 2 and keeping in mind that X and Y have the same distribution, we get

$$\begin{aligned} \mathit{MSE}_2 &= E(X^2) + E(Y^2) - 2(\mathcal{C}\,\mathrm{Var}(X) + E(X)E(Y)) \\ &= E(X^2) + E(X^2) - 2\mathcal{C}\,\mathrm{Var}(X) - 2E(X)E(X) \\ &= 2E(X^2) - 2E(X)^2 - 2\mathcal{C}\,\mathrm{Var}(X) \\ &= 2\,\mathrm{Var}(X) - 2\mathcal{C}\,\mathrm{Var}(X) \\ &= (2-2\mathcal{C})\,\mathrm{Var}(X). \end{aligned}$$

For the third estimation approach, we define a linear prediction problem. We seek the values of β and γ that minimize the *MSE* function $E((Y-Y')^2)$, where $Y' = \beta X + \gamma$. By differentiating the *MSE* with respect to γ, we get

$$\begin{aligned} \frac{\delta \mathit{MSE}}{\delta(\gamma)} &= \frac{\delta(E((Y'-Y)^2) = E((\beta X + \gamma - Y)^2))}{\delta(\gamma)} \\ &\cdots \\ &= 2\gamma + 2\beta\mu - 2\mu. \end{aligned}$$

Equating the result to 0 yields

$$\hat{\gamma} = \mu - \beta\mu. \quad (4)$$

In a similar way, differentiating the *MSE* with respect to β yields

$$\frac{\delta \mathit{MSE}}{\delta(\beta)} = 2\beta E(X^2) + 2\gamma\mu - 2E(XY).$$

We now replace γ with the value of $\hat{\gamma}$ from Eq. 4 and get:

$$\begin{aligned} \frac{\delta \mathit{MSE}}{\delta(\beta)} &= 2\beta E(X^2) + 2\gamma\mu - 2E(XY) \\ &= 2\beta E(X^2) + 2(\mu - \beta\mu)\mu \\ &\quad -2E(XY) \\ &= 2\beta E(X^2) + 2\mu^2 - 2\beta\mu^2 \\ &\quad -2E(XY). \end{aligned}$$

Therefore, the value of β that brings the *MSE* to its minimum is

$$\beta = \frac{\mu^2 - E(XY)}{\mu^2 - E(X^2)} = \frac{\mathrm{cov}(X,Y)}{\mathrm{Var}(X)}.$$

Since X and Y have similar distribution,

$$\mathrm{cov}(X,Y) = \mathrm{corr}(X,Y)\,\mathrm{Var}(X) = \mathcal{C}\,\mathrm{Var}(X). \quad (5)$$

Method	$deg_S(u)$	*MSE*
1	μ	$\mathrm{Var}(X)$
2	X	$(2-2\mathcal{C})\,\mathrm{Var}(X)$
3	$\mathcal{C}X+(1-\mathcal{C})\mu$	$(1-\mathcal{C}^2)\,\mathrm{Var}(X)$

Fig. 5. The three methods and their MSEs

We conclude that $\beta = \mathcal{C}\,\mathrm{Var}(X)/\,\mathrm{Var}(X) = \mathcal{C}$ and $\gamma = \mu - \beta\mu = (1-\mathcal{C})\mu$ are the values that minimize the *MSE*. Therefore, for the third estimation method, $deg_S(v)$ is estimated as:

$$Y' = \mathcal{C}X + (1-\mathcal{C})\mu,$$

and the value of MSE_3 is

$$\begin{aligned}
MSE_3 &= E((Y'-Y)^2)\\
&= E((\mathcal{C}X+(1-\mathcal{C})\mu - Y)^2)\\
&= E(\mathcal{C}^2X^2 + 2\mathcal{C}(1-\mathcal{C})X\mu\\
&\quad -2\mathcal{C}XY - 2(1-\mathcal{C})\mu Y + (1-\mathcal{C})^2\mu^2 + Y^2)\\
&= \mathcal{C}^2E(X^2) + 2\mathcal{C}(1-\mathcal{C})E(X)\mu - 2\mathcal{C}E(XY)\\
&\quad -2(1-\mathcal{C})\mu E(Y) + (1-\mathcal{C})^2\mu^2 + E(Y^2)\\
&= \mathcal{C}^2E(X^2) + 2\mathcal{C}(1-\mathcal{C})\mu^2 - 2\mathcal{C}E(XY)\\
&\quad -2(1-\mathcal{C})\mu^2 + (1-\mathcal{C})^2\mu^2 + E(Y^2)\\
&= \mathcal{C}^2E(X^2) + E(Y^2) - 2\mathcal{C}E(XY)\\
&\quad +(2\mathcal{C} - 2\mathcal{C}^2 - 2 + 2\mathcal{C} + 1 - 2\mathcal{C} + \mathcal{C}^2)\mu^2\\
&= \mathcal{C}^2E(X^2) + E(Y^2) + (2\mathcal{C} - \mathcal{C}^2 - 1)\mu^2\\
&\quad -2\mathcal{C}E(XY).
\end{aligned}$$

Using again the fact that X and Y have the same distribution and substituting the value of $E(XY)$ from Eq. 3 yields

$$\begin{aligned}
MSE_3 &= (\mathcal{C}^2+1)E(X^2) + (2\mathcal{C} - \mathcal{C}^2 - 1)\mu^2\\
&\quad -2\mathcal{C}(\mathcal{C}\,\mathrm{Var}(X) + \mu^2)\\
&= (\mathcal{C}^2+1)E(X^2) - (\mathcal{C}^2+1)\mu^2 - 2\mathcal{C}^2\,\mathrm{Var}(X)\\
&= (\mathcal{C}^2+1)(E(X^2) - \mu^2) - 2\mathcal{C}^2\,\mathrm{Var}(X)\\
&= (\mathcal{C}^2+1)\,\mathrm{Var}(X) - 2\mathcal{C}^2\,\mathrm{Var}(X)\\
&= (1-\mathcal{C}^2)\,\mathrm{Var}(X).
\end{aligned}$$

The table in Figure 5 summarizes the three methods discussed above and their MSEs. We see that the accuracy of the three methods depends on the correlation between the degrees of neighboring nodes and on the variance of node degree. For small values of $\mathcal{C}$, the first and the third estimation methods are more accurate than the second one. For greater values of $\mathcal{C} = corr(X,Y)$, the accuracy of the second and the third methods is closer to that of the first method. Moreover, for the same variance, the *MSE*s of these methods approach 0 when $\mathcal{C}$ approaches 1. An example of a sensor network with $\mathcal{C}$ close to 1 is a network whose sensors are spread around several spots of interest, because the correlation between node degrees is greater than when the nodes are uniformly spread all over the network.

V. The Uniform Distribution Special Case

We now examine a special case where the network nodes are uniformly distributed. The node degree has a binomial distribution, where the "probability of success" p is the probability that a node v is in the transmission range of another node u. This probability is equal to the ratio between the area covered by v and the area covered by the whole network. For this kind of distribution, the variance is known to be $np(1-p)$, where n is the number of nodes. Note that np is the expected node degree, denoted earlier by μ. For this special case we get

$$\begin{aligned}
\mathrm{corr}(X,Y) &= \frac{\mathrm{cov}(X,Y)}{\mathrm{Var}(X)}\\
&= \frac{E(XY)-\mu^2}{\mathrm{Var}(X)},
\end{aligned} \tag{6}$$

where

$$\begin{aligned}
E(XY) &= \sum_y\sum_x xyP(X=x,Y=y)\\
&= \sum_y\sum_x xyP(X=x|Y=y)P(Y=y)\\
&= \sum_y[yP(Y=y)\sum_x xP(X=x|Y=y)]\\
&= \sum_y[yP(Y=y)E(X|Y=y)].
\end{aligned} \tag{7}$$

We now show how to find $E(X|Y=y)$, namely, the expected number of neighbors of v given that the number of neighbors of u is known and equal to y. The set of neighbors of v can be divided into two subsets: subset A includes neighbors of v that are also neighbors of u; subset B includes neighbors of v that are not neighbors of u. In the same way, the set of neighbors of u can be divided into two sets: the same subset A, and subset B' of neighbors of u that are not neighbors of v. Theorem 1 shows the relationship between the neighbors of v and the neighbors of u:

Theorem 1: Let u, v and w be nodes in a geometric graph with the same transmission range, where nodes are distributed uniformly. If u is a neighbor of v and v is a neighbor of w, then the probability that u is also a neighbor of w is $\mathcal{R} = 1 - \frac{3}{4\pi}\sqrt{3} \approx 0.586503$.

The proof is presented in the appendix.

Following Theorem 1, we conclude that (a) the expected size of subset A is equal to the number of neighbors of u multiplied by $\mathcal{R}$, namely $E(|A|) = \mathcal{R}\cdot deg_S(u)$; (b) the expected size of subset B is equal to the average graph degree multiplied by $(1-\mathcal{R})$, where $(1-\mathcal{R})$ is the part of the area covered by v but not by u, as follows from Theorem 1. Since the degree of v is $|A|+|B|$, we get

$$E(X|Y=y) = \mathcal{R}y + (1-\mathcal{R})\mu.$$

Substituting this into Eq. 7 yields:

$$\begin{aligned} E(XY) &= \sum_y yP(Y=y)(\mathcal{R}y + (1-\mathcal{R})\mu) \\ &= \sum_y \mathcal{R}y^2 P(Y=y) \\ &\quad +(1-\mathcal{R})\mu \sum_y yP(Y=y) \\ &= \mathcal{R}E(Y^2) + (1-\mathcal{R})\mu^2. \end{aligned} \quad (8)$$

Substituting Eq. 8 into Eq. 6 yields

$$\begin{aligned} \mathrm{corr}(X,Y) &= \frac{E(XY)-\mu^2}{\mathrm{Var}(X)} \\ &= \frac{\mathcal{R}E(Y^2)+(1-\mathcal{R})\mu^2-\mu^2}{\mathrm{Var}(X)} \\ &= \frac{\mathcal{R}E(Y^2)-\mathcal{R}\mu^2+\mu^2-\mu^2}{\mathrm{Var}(X)} \\ &= \frac{\mathcal{R}\,\mathrm{Var}(Y)}{\mathrm{Var}(X)} \\ &= \frac{\mathcal{R}\,\mathrm{Var}(X)}{\mathrm{Var}(X)} \\ &= \mathcal{R}. \end{aligned}$$

Hence, for the uniform distribution special case, the three estimation approaches yield the following MSEs:

1) $MSE_1 = \mathrm{Var}(X)$,
2) $MSE_2 \approx 0.84\,\mathrm{Var}(X)$,
3) $MSE_3 \approx 0.66\,\mathrm{Var}(X)$.

We see that the third approach yields the smallest *MSE*. However, this approach, as well as the first one, requires some global information on the network topology, while the second approach requires only local information.

VI. An Efficient Continuous Neighbor Discovery Algorithm

In this section we present an algorithm for assigning HELLO message frequency to the nodes of the same segment. This algorithm is based on Scheme 1. Namely, if a hidden node is discovered by one of its segment neighbors, it is discovered by all its other segment neighbors after a very short time. Hence, the discovery of a new neighbor is viewed as a joint effort of the whole segment. One of the three methods presented in Section IV is used to estimate the number of nodes participating in this effort.

Suppose that node u is in initial neighbor discovery state, where it wakes up every T_I seconds for a period of time equal to H, and broadcasts HELLO messages. Suppose that the nodes of segment S should discover u within a time period T with probability P. Each node v in the segment S is in continuous neighbor discovery state, where it wakes up every $T_N(v)$ seconds for a period of time equal to H and broadcasts HELLO messages.

We assume that, in order to discover each other, nodes u and v should have an active period that overlaps by at least a portion δ, $0 < \delta < 1$, of their size H. Thus, if node u wakes up at time t for a period of H, node v should wake up between $t - H(1-\delta)$ and $t + H(1-\delta)$. The length of this valid time interval is $2H(1-\delta)$. Since the average time interval between two wake-up periods of v is $T_N(v)$, the probability that u and v discover each other during a specific HELLO interval of u is $\frac{2H(1-\delta)}{T_N(v)}$.

Let n be the number of in-segment neighbors of u. When u wakes up and sends HELLO messages, the probability that at least one of its n neighbors is awake during a sufficiently long time interval is $1-(1-\frac{2H(1-\delta)}{T_N(v)})^n$.

For the sake of our analysis, consider a division of the time axis of u into time slots of length H. The probability that u is awake in a given time slot is $\frac{H}{T_I}$, and the probability that u is discovered during this time slot is $P_1 = \frac{H}{T_I}(1-(1-\frac{2H(1-\delta)}{T_N(v)})^n)$. Denote by D the value of $\frac{T}{H}$. Then, the probability that u is discovered within at most D slots is $P_2 = 1-(1-P_1)^D$. Therefore, we seek the value of $T_N(v)$ that satisfies the following equation:

$$1-(1-P_1)^D \geq P,$$

which can also be stated as

$$P_1 \geq 1 - \sqrt[D]{1-P}.$$

Since $P_1 = \frac{H}{T_I}(1-(1-\frac{2H(1-\delta)}{T_N(v)})^n)$, we get

$$\frac{H}{T_I}(1-(1-\frac{2H(1-\delta)}{T_N(v)})^n) \geq 1 - \sqrt[D]{1-P},$$

and therefore

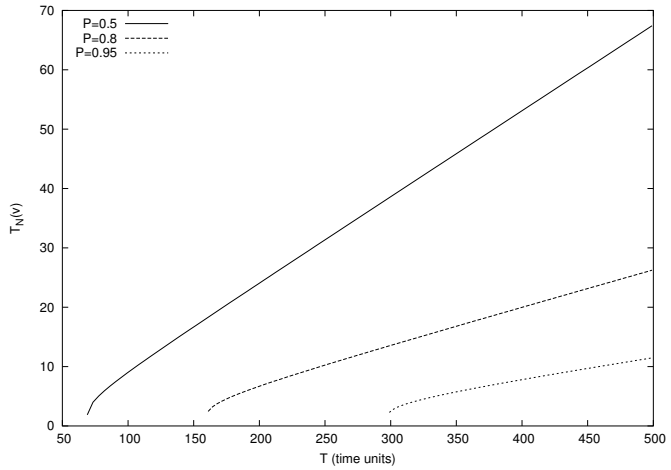
$$T_N(v) \leq \frac{2H(1-\delta)}{1-\sqrt[n]{1-\frac{T_I}{H}(1-\sqrt[D]{1-P})}}. \quad (9)$$

Since node v does not know the exact value of n, it can estimate it using the methods presented in Section IV.

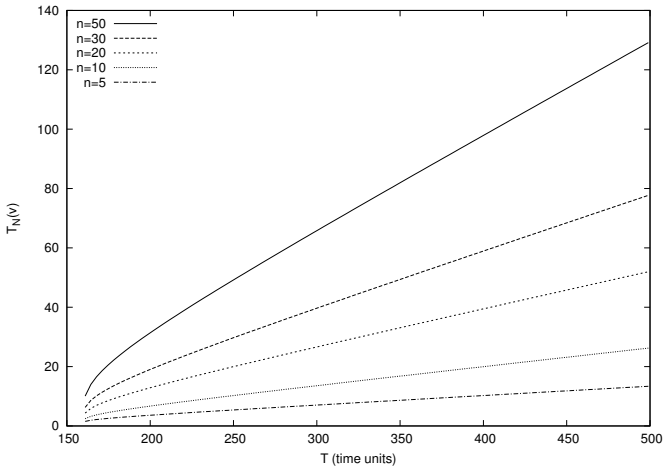
We now give a simple example for the proposed algorithm. Suppose that nodes in the Init state remain active until they enter the Normal state. Suppose also that the requirement is to discover a hidden node within 10 time units with probability 0.5. Consider a segment node in the Normal state, where continuous neighbor discovery is performed, that estimates the degree of its hidden neighbor as 1. Following our definitions, $D = 10$, $T_I = H = 1$ and n = 1. Substituting these values into Eq. 9 yields $T_N \approx 15$. Note that the intuitive value of $T_N = 20$ is wrong because it would yield a detection probability of $1-(1-\frac{1}{20})^{10} \approx 0.4$ to discover a hidden node within 10 time units.

If one needs to enforce not only the expected hidden neighbor discovery delay but also an upper bound on it, each node can be assigned a wake-up period according to the rules described in [13].

In Figure 6 we present two graphs that show the dependency between T and $T_N(v)$. We assume that a hidden node wakes up once every $100H$ time units on the average, and that $T_I = 100$, $H = 1$, and $\delta = 0.5$. In Figure 6(a) the estimated value of n is 10. The curves present the value of $T_N(v)$ as a function of the desired discovery time T for 3 different values of P: 0.5, 0.8 and 0.95. In Figure 6(b) P is set to 0.8 and n varies between 5 and 50. Again, $T_N(v)$ is calculated as a function of the desired discovery time. As expected, the nodes have to



(a) $T_N(v)$ as a function of T for $n = 10$



(b) $T_N(v)$ as a function of T for different values of n

Fig. 6. $T_N(v)$ as a function of maximum tolerated delay

work harder to achieve a greater discovery rate in less time, while the increase in the density of segment nodes allows to a greater $T_N(v)$ to be chosen. In both graphs the dependency between $T_N(v)$ and T is almost linear and, as we can see in Figure 6(b), the slope of the curves is almost linear in the value of n as well. This means that a node v can use linear approximation to compute the value of $T_N(v)$.

VII. Simulation Study

In this section we present a simulation study for the schemes presented in the paper. We simulate a large sensor network, with nodes distributed randomly and uniformly over the area of interest. We assume that the nodes have an equal and constant transmission range. Communication is always bi-directional. We also assume that most of the nodes discover each other and enter the continuous neighbor discovery state before the simulation begins.

Our simulation model consists of 2,000 sensor nodes, randomly placed over a 10,000 x 10,000 grid. The transmission range is set to r units. Any two nodes whose Euclidean distance is not greater than r are considered to have wireless connectivity. A portion of the nodes are randomly selected to be hidden. These nodes are uniformly distributed in the considered area. We set the algorithm parameters such that every hidden node will be detected with probability P within a predetermined period of time T. For the study reported in this section, r is chosen to be 300 (0.03 of the graph), the detection probability ranges between 0.3 and 0.7, and the target detection time is 100 time units.

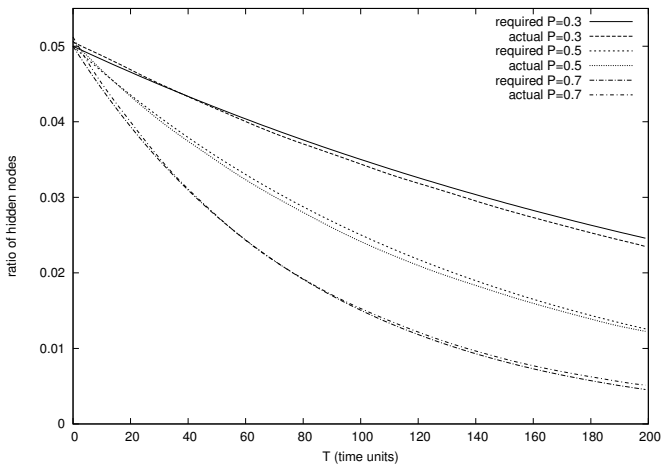
The hidden nodes are assumed to be in the initial neighbor discovery state, where they are supposed to wake up randomly, every T_I time units on the average, and to exchange HELLO messages with other nodes during a period of H time units. A non-hidden node v is assumed to be in the continuous neighbor discovery state, where it wakes up randomly, every $T_N(v)$ time units on the average for a period of H time units, in order to discover hidden nodes. For the study reported in what follows, $T_I = 20$, $H = 1$ and $\delta = 0.5$ are used. When a node is detected, it joins the segment and learns about its in-segment neighbors using Scheme 1. A hidden node that detects another hidden node remains in the initial neighbor discovery state.

Our simulations reveal that when the hidden nodes are uniformly distributed, the three algorithms proposed in Section IV yield very similar results. The reason for this similarity is that the degree estimation errors of the neighbors of every node cancel each other, and the mean estimation bias approaches 0. Because of this similarity, in most of the graphs we show only the results of one algorithm (Algorithm 3).

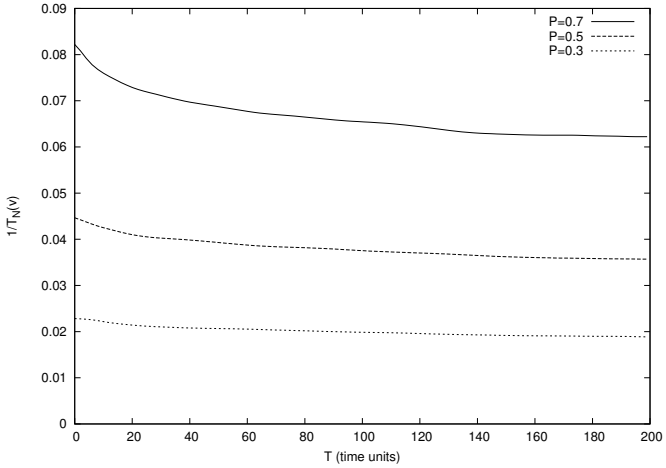
Figure 7(a) shows the ratios of hidden nodes to the total number of nodes as a function of time. The initial ratio is 0.05. We can see that after 100 time units, this ratio decreases to 0.035 for $P = 0.3$, to 0.025 for $P = 0.5$, and to 0.015 for $P = 0.7$. After 200 time units, the ratios of the hidden nodes are 0.025, 0.012 and 0.005 respectively. It is evident that these results are very close to the required ratios.

In the next simulation we start with 50% hidden nodes. Figure 7(b) shows the change in the average frequency of HELLO intervals of the segment nodes, as a function of time, for the same three values of P. We can see that for the smaller value of P (the lower curve), the frequency is almost 75% lower than the frequency for the larger value of P. We can also see that for a given value of P, the average frequency of HELLO intervals decreases with time. This is because as the segment grows, more nodes participate in the discovery process. Similar results are obtained for the case where the initial hidden node ratio was 0.05, but they can hardly be observed due to the small changes in the segment size during the simulation.

Another interesting case is when the hidden nodes are distributed non-uniformly in the area. To simulate this case, we randomly select some points as "dead areas," and assume that the probability of a node to be hidden increases when its distance to one of these points decreases. The rationale here is that bad weather, dust storms, or other environmental conditions may adversely affect wireless connectivity in some areas more than in others. Unlike the uniform distribution

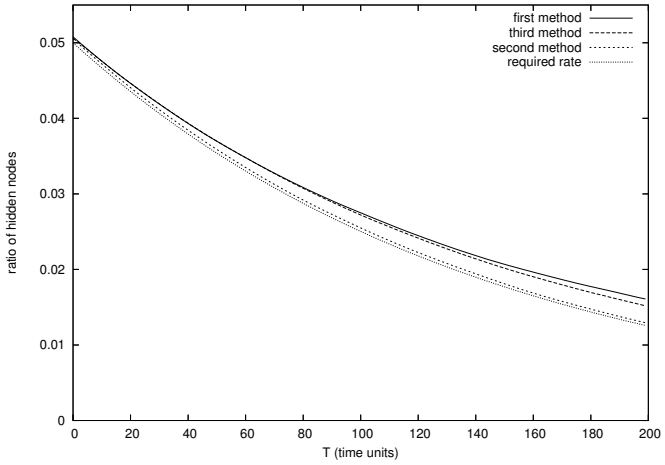


(a) Decrease in the ratio of hidden nodes

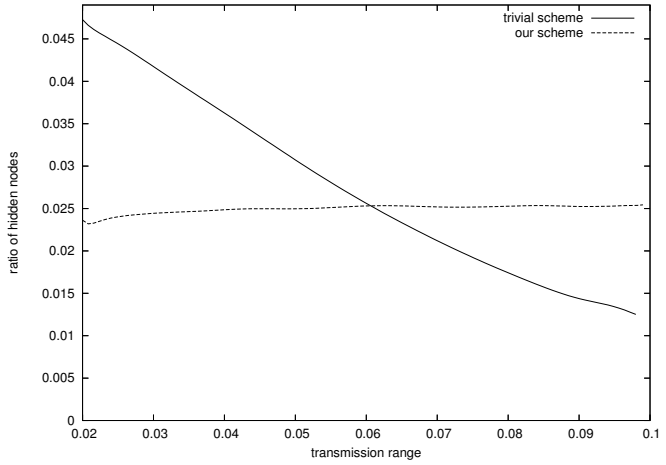


(b) The average $1/T_N(v)$ as a function of time

Fig. 7. Hidden neighbor detection for the case of uniform distribution



(a) Decrease in hidden node ratio



(b) Our scheme compared to a trivial scheme that does not adjust wake-up frequency to the network density

Fig. 8. Hidden neighbors detection with extreme point

case, here we do see differences between the three estimation algorithms presented in Section IV.

Figure 8(a) shows the percent of hidden nodes as a function of time for the three estimation algorithms and $P = 0.5$. Unlike in the uniform distribution case, here we can see some differences between the three algorithms: the second algorithm is the closest to the required rate (shown by a separate curve), where the first algorithm discovers the hidden nodes at a rate slower than the required one.

Figure 8(b) shows the ratio of hidden nodes after T for networks with different transmission ranges, and hence with different node average degrees. This graph reveals the flexibility of our scheme and its ability to adjust the wake-up frequency to the network density. We show this by comparing our scheme to a trivial scheme that does not take the network density into account. For the trivial scheme, all the nodes have the same wake-up frequency. The actual values, which depend on the wake-up frequency of the nodes, are not important. The comparison shows that the trivial scheme is too aggressive in dense networks and not aggressive enough in sparse ones. Recall that the goal of our scheme is not to discover nodes as quickly as possible, but to impose an upper bound on the discovery time while minimizing energy consumption. In light of this goal, we see that our scheme performs better because its discovery rate is fixed, and so is its overall expended energy.

The simulation starts with 5% hidden nodes, and each node in *Init* is configured with $P = 0.5$. For all transmission ranges, our scheme indeed guarantees that after T time units the percentage of hidden nodes will decrease by half, to 2.5%. Interestingly enough, the trivial scheme discovers half of the hidden nodes only when the transmission range is ≈ 0.06. When the transmission range is shorter, the trivial scheme discovers a smaller fraction of the hidden nodes. For instance, for a range of 0.03, the ratio of hidden nodes is reduced from 0.05 to 0.04. When the transmission range is greater than 0.06, the trivial scheme discovers more nodes during a time period of T. But this is, of course, with a much greater expense of energy than required in our scheme. We conclude that our

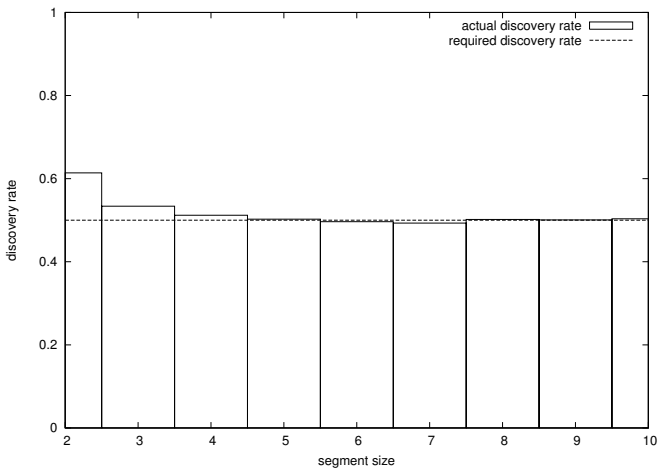


Fig. 9. The hidden neighbor discovery rate for a small detecting segment

algorithm can self-adjust to invest the minimum energy needed to guarantee the required discovery rate, whereas the trivial algorithm cannot.

So far we have assumed that the detecting node belongs to a big segment to which the detected node joins. However, it is still possible that two nodes in the *Init* state of Figure 2 will discover each other. These nodes can either stay in the *Init* state or switch to the *Normal* state. It is more efficient to switch to the *Normal* state because the overhead of detecting more neighbors is shared by all of the segment nodes. However, two of the assumptions made in Section IV are not valid when the detecting node is not a part of a big segment: the assumption that the expected in-segment degree of the segment's nodes is equal to the expected in-segment degree of the hidden node, and the (implicit) assumption that the segment's size is significantly bigger than the node's expected degree. For example, when a single node v in a small segment of size two detects another node, the expected in-segment degree of the nodes in the detecting segment is 1. In contrast, by Theorem 1, the expected degree of the hidden neighbor of v is about 1.58.

Figure 9 shows simulation results for the discovery by a small detecting segment. The transmission range is set to 0.3 of the graph, but similar results have been obtained for other transmission ranges. It is evident that the desired discovery rate is achieved for a segment of three or more nodes. For segments of two nodes, the discovery rate is faster than the desired rate. In such a segment, the in-segment degree of every node v and the in-segment degree of v's neighbor are both 1. Thus, every in-segment node v estimates the degree of a hidden neighbor u to be 1, while the actual expected degree of u is 1.58 as follows from Theorem 1. Our simulations reveal that for a 2-node segment, the in-segment degree of a hidden neighbor should be taken to be 1.4, in which case the target discovery rate is achieved, whereas for a larger segment Algorithm 3 should be used. On the basis of these results, we claim that our algorithms can be used for every segment size, despite our assumption during the analysis that the segment is "big enough."

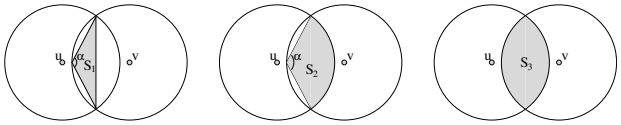


Fig. 10. Definitions for the proof of Theorem 1

VIII. Conclusions

We exposed a new problem in wireless sensor networks, referred to as ongoing continuous neighbor discovery. We argue that continuous neighbor discovery is crucial even if the sensor nodes are static. If the nodes in a connected segment work together on this task, hidden nodes are guaranteed to be detected within a certain probability P and a certain time period T, with reduced expended on the detection.

We showed that our scheme works well if every node connected to a segment estimates the in-segment degree of its possible hidden neighbors. To this end, we proposed three estimation algorithms and analyzed their mean square errors. We then presented a continuous neighbor discovery algorithm that determines the frequency with which every node enters the HELLO period. We simulated a sensor network to analyze our algorithms and showed that when the hidden nodes are uniformly distributed in the area, the simplest estimation algorithm is good enough. When the hidden nodes are concentrated around some dead areas, the third algorithm, which requires every node to take into account not only its own degree, but also the average degree of all the nodes in the segment, was shown to be the best.

Appendix

The proof of theorem 1:

Proof: Consider Figure 10. Node w is a neighbor of u only if it resides in the area marked as S_3. To find the probability that this is indeed the case, we have to find the ratio between S_3 and the unit circle area. Now, note that

$$S_1 = r\sin\tfrac{\alpha}{2}r\cos\tfrac{\alpha}{2}; S_2 = \pi r^2\tfrac{\alpha}{2\pi}; S_3 = 2(S_2 - S_1).$$

Since $\cos\frac{\alpha}{2} = \frac{x}{2r}$, where x is the distance between u and v, then $\frac{\alpha}{2} = \arccos\frac{x}{2r}$ holds. Hence, we can write:

$$S_1 = \tfrac{x}{2}r\sqrt{1-\tfrac{x^2}{4r^2}}; S_2 = \tfrac{r^2}{2}2\arccos\tfrac{x}{2r}.$$

Thus, $S_3 = 2r^2\arccos\frac{x}{2r} - xr\sqrt{1-\frac{x^2}{4r^2}}$.

A neighbor of v is also a neighbor of u only if it lies inside S_3. The probability for this is $\frac{S_3}{\pi r^2}$. Denote the probability for such an event as P_x, where x is the distance between u and v. Hence,

$$P_x = \frac{2}{\pi}\arccos\frac{x}{2r} - \frac{x}{\pi r}\sqrt{1-\frac{x^2}{4r^2}}.$$

In order to find the probability P that w is a neighbor of u, we should consider all possible values of x, from 0 to r,

while taking into account the density function:

$$\begin{aligned}
P &= \frac{1}{\pi r^2}\int_{x=0}^{r} 2\pi x P_x dx \\
&= \frac{2}{r^2}\int_{x=0}^{r} x\left(\frac{2}{\pi}\arccos\frac{x}{2r} - \frac{x}{\pi r}\sqrt{1-\frac{x^2}{4r^2}}\right)dx \\
&= \frac{2}{r^2}\left(\int_{x=0}^{r}\frac{2}{\pi}x\arccos\frac{d}{2r}dx\right) \\
&\quad -\frac{2\pi}{\pi r^2}\left(\int_{x=0}^{r}\frac{1}{\pi r}x^2\sqrt{1-\frac{x^2}{4r^2}}dx\right) \\
&= \frac{2}{r^2}\frac{2}{\pi}\left[-\frac{1}{2}rx\sqrt{1-\frac{x^2}{4r^2}}+\frac{1}{2}x^2\arccos\frac{x}{2r}\right]_0^r \\
&\quad +\frac{2}{r^2}\frac{2}{\pi}\left[r^2\arcsin\frac{x}{2r}\right]_0^r \\
&\quad -\frac{2}{r^2}\frac{1}{\pi r}\left[\frac{1}{8}x(-2r^2+x^2)\sqrt{4-\frac{x^2}{r^2}}\right]_0^r \\
&\quad -\frac{2}{r^2}\frac{1}{\pi r}\left[r^3\arcsin\frac{x}{2r}\right]_0^r.
\end{aligned}$$

Substituting the integration limits yields:

$$\begin{aligned}
P &= \frac{4}{\pi r^2}\left[-\frac{r^2}{2}\sqrt{1-\frac{1}{4}}+\frac{r^2}{2}\arccos\frac{1}{2}+r^2\arcsin\frac{1}{2}\right] \\
&\quad -\frac{2}{r^2}\left(\frac{1}{\pi r}\left[\frac{1}{8}r(-r^2)\sqrt{3}+r^3\arcsin\frac{1}{2}\right]\right) \\
&= \frac{4}{\pi r^2}\left[-\frac{1}{2}r^2\sqrt{\frac{3}{4}}+\frac{1}{2}r^2\frac{\pi}{3}+r^2\frac{\pi}{6}\right] \\
&\quad -\frac{2}{r^2}\left(\frac{1}{\pi r}\left[\frac{1}{8}r(-r^2)\sqrt{3}+r^3\frac{\pi}{6}\right]\right) \\
&= 2\left(\left[-\frac{1}{\pi 2}\sqrt{3}+\frac{1}{3}+\frac{1}{3}\right]-\left[-\frac{1}{8\pi}\sqrt{3}+\frac{1}{6}\right]\right) \\
&= -\frac{1}{\pi}\sqrt{3}+\frac{4}{3}+\frac{1}{4\pi}\sqrt{3}-\frac{1}{3} \\
&= 1-\frac{3}{4\pi}\sqrt{3}\approx 0.586503.
\end{aligned}$$

■

REFERENCES

[1] S. Vasudevan, J. Kurose, and D. Towsley, "On neighbor discovery in wireless networks with directional antennas," in *INFOCOM*, vol. 4, 2005, pp. 2502–2512.

[2] R. Madan and S. Lall, "An energy-optimal algorithm for neighbor discovery in wireless sensor networks," *Mob. Netw. Appl.*, vol. 11, no. 3, pp. 317–326, 2006.

[3] M. J. McGlynn and S. A. Borbash, "Birthday protocols for low energy deployment and flexible neighbor discovery in ad hoc wireless networks," in *MobiHoc: Proceedings of the 2nd ACM International Symposium on Mobile Ad Hoc Networking and Computing*. New York, NY, USA: ACM Press, 2001, pp. 137–145.

[4] D. Baker and A. Ephremides, "The architectural organization of a mobile radio network via a distributed algorithm," in *IEEE Transactions on Communications*, vol. 29, Nov. 1981, pp. 1694–1701.

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

[6] E. B. Hamida, G. Chelius, and E. Fleury, "Revisiting neighbor discovery with interferences consideration," in *PE-WASUN*, 2006, pp. 74–81.

[7] S. A. Borbash, "Design considerations in wireless sensor networks," Ph.D. dissertation, ISR, August 2004.

[8] G. Alonso, E. Kranakis, R. Wattenhofer, and P. Widmayer, "Probabilistic protocols for node discovery in ad-hoc, single broadcast channel networks," in *IPDPS*, 2003, p. 218.

[9] C. Perkins, E. Belding-Royer, and S. Das, "Ad hoc on-demand distance vector (AODV) routing," RFC 3561, July 2003.

[10] J. Haartsen, *Bluetooth Baseband Specification v. 1.0*.

[11] T. Salonidis, P. Bhagwat, L. Tassiulas, and R. O. LaMaire, "Distributed topology construction of bluetooth personal area networks," in *INFOCOM*, 2001, pp. 1577–1586.

[12] *IEEE 802.15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)*, IEEE 802.15 WPAN Task Group 4 (TG4), 2006.

[13] P. Dutta and D. Culler, "Practical asynchronous neighbor discovery and rendezvous for mobile sensing applications," in *SenSys: Proceedings of the 6th ACM conference on Embedded network sensor systems*. New York, NY: ACM Press, 2008, pp. 71–84.

[14] J. Hill and D. Culler, "A wireless embedded sensor architecture for system-level optimization," Technical report, U.C. Berkeley, 2001.

[15] R. Jurdak, P. Baldi, and C. V. Lopes, "Adaptive low power listening for wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 6, no. 8, pp. 988–1004, 2007.

PLACE PHOTO HERE

Reuven Cohen (M'93, SM'99) received his B.Sc., M.Sc. and Ph.D. degrees in Computer Science from the Technion – Israel Institute of Technology, completing his Ph.D. studies in 1991. From 1991 to 1993, he was with the IBM T.J. Watson Research Center, working on protocols for high speed networks. Since 1993, he has been a professor in the Department of Computer Science at the Technion. He has also been a consultant for numerous companies, mainly in the context of protocols and architectures for broadband access networks. Dr. Cohen has served as an editor of the IEEE/ACM Transactions on Networking and the ACM/Kluwer Journal on Wireless Networks (WINET). He is the technical program co-chair of Infocom'2010. Dr. Cohen is a senior member of the IEEE and heads the Israeli chapter of the IEEE Communications Society.

PLACE PHOTO HERE

Boris Kapchits received the B.Sc. in and M.Sc. in Computer Science from the Technion – Israel Institute of Technology, Haifa, Israel, in 2002 and 2005, respectively. Since 2005, he has been a Ph.D. student in Computer Science Department in the Technion, working on mesh sensor networks.