IMPROVED CONTINUOUS NEIGHBORS DISCOVERY PROTOCOL FOR WIRELESS SENSOR NETWORKS

Mr .Prafulla L. Mehar*

Prof. R. K. Krishna**

A B S T R A C T

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 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 we discuss about the solution to the more power consumption during the neighbors discovery in wireless sensor network. Each sensor coordinate effort to reduce power consumption without increasing the time required to detect hidden sensors.

Keywords— neighbor discovery , wireless sensor network component.

* Dept. of Computer Technology, RGCERT, Chandrapur,Maharastra,India

** Dept. of Electronics Engineeering ,RGCERT, Chandrapur, Maharastra, India

I. INTRODUCTION

Despite the static nature of the sensor nodes, after the network has been established its connectivity is still subject to changes. In particular, even after a sensor node is aware of its immediate neighbors, it must continuously look for new ones 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 effects disappear, 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 (so-called dead nodes).
4. The increase in transmission power of some nodes, in some networks, in response to certain events, such as loss of connectivity with neighboring nodes or detection of important local happening.

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 and their detection during normal operation. The former will be referred to as initial neighbor discovery whereas the latter will be referred to as continuous neighbor discovery. While previous works [8, 3, 5] address initial neighbor discovery and continuous neighbor discovery as similar tasks, to be performed by the same protocol, we claim that they should be addressed by different protocols for the following reasons:

- Initial neighbor discovery is usually performed when the sensor has no clue about the structure of its immediate surroundings. In particular, the sensor cannot communicate with the gateway, and is therefore very limited in performing its tasks. Hence, in this state for the sake of quicker detection, more extensive energy use is justified. It is very important to detect the immediate surroundings as soon as possible in order to establish a path to the gateway and to contribute to the operation of the network. 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 the network life time.

- When the sensor performs continuous neighbor discovery, it is already aware of most of its immediate neighbors. It can therefore perform continuous neighbor discovery together with these neighbors in order to consume less energy. In contrast, initial neighbor discovery is an individual task, that must be executed by each sensor separately.

We now show, by means of an example, why an initial neighbor discovery protocol is inefficient for continuous neighbor discovery. Figure 1 presents a simple protocol. In this figure we assume that node u is in the initial neighbor discovery state, where its main task is to search for new neighbors. To this end, it periodically wakes up, at random times, and transmits a bunch of HELLO messages (the bunch size in the figure is 1). In the figure we see that the first 5 bunches of HELLO messages are transmitted when node v is sleeping, and therefore they cannot be received by v. The 6th bunch is transmitted when v is in active mode. Therefore, v is likely to receive at least one message of the 6th bunch, to which it responds with HELLO-ACK. From this time, the two nodes view each other as neighbors, and they maintain this relationship using periodic HELLO messages.

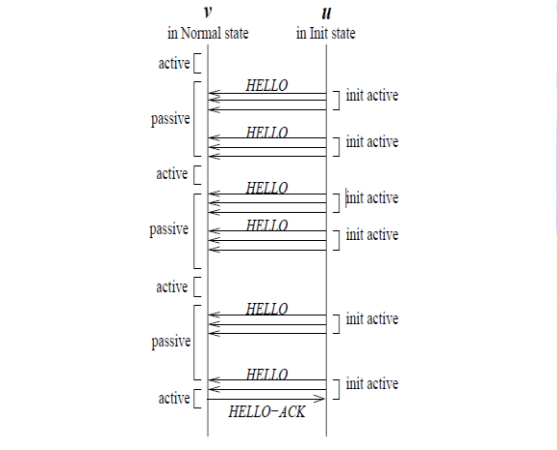

Figure 1: The transmission of HELLO control messages during neighbor discovery state

If a hidden node has duty cycle of 1%, we can assume that the discovering node v "hits" u when u is awake with probability of 1%. In this case, by the rules of geometric distribution, the discovery demands, in average, 100 bunches of HELLO messages. Hence, in order to guarantee the average discovery time of 10 seconds, u has to wake up every 0.1 second. Even if every wakeup lasts only 10msec, it gives us the duty cycle of 10 %, thereby expending a lot of its energy on finding its neighbors. Working with such a duty cycle might be reasonable only when node u is added to the

network, i.e., in the neighbor discovery state, but not as an ongoing algorithm for continuous neighbor discovery.

We distinguish between initial and continuous neighbor discovery in sensor network. Figure 5.2 summarizes this idea. When node u is initialized, it performs initial neighbor discovery. After a certain time period in the initial neighbor discovery state, during which the node is expected, with high probability, to find most of its neighbors, the node moves to the continuous neighbor discovery state. The main idea behind the continuous neighbor discovery scheme proposed in this chapter is that the task of finding a new node is divided among all the nodes in its vicinity.

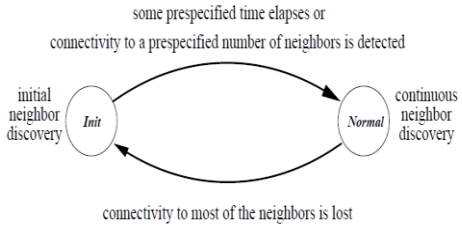

Figure 1.2: Continuous neighbor discovery vs. initial neighbor discovery in sensor networks

II. BASIC SCHEMES 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. For our analysis we also assume that the network is a unit disk graph; namely, any pair of nodes that are within the transmission range of each other are neighboring nodes. Two nodes are said to be directly connected if they have discovered each other and they are aware of the wake up times of each other. 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 2(a). These two nodes can learn about their hidden wireless link using the following simple scheme.

Scheme 1 (detecting a hidden link inside a segment) One of the segment nodes 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 node in the segment. By having all the nodes wake up "almost at the same time" for a short period, we can ensure that all the wireless links between the segment's members will be detected with minimum energy cost.

This scheme needs to be involved only when a new node is discovered by one of the segment nodes. The discovering node will also be the node, that triggers the protocol.

suppose that every node wakes up once a second in order to receive messages from its in-segment neighbors. Suppose also that the node stays active for about 10 milliseconds, thereby having a duty cycle of 0.1%. In this case, the SYNC message can reach every node in the segment within at most D seconds, where D is the distance between the segment leader and the farthest node. The SYNC message carries a WAKE-UP-TIME field, which is initialized to D and decremented by t(v;u) by every node v that transmits the SYNC, where t(v;u) is the interval between the time v receives the SYNC and the time it transmits it to its in-segment neighbor u. Since this scheme is not frequently involved, we can allow nodes to remain active for a relatively long period of time, compensating for possible synchronization inaccuracy.

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-ACK messages sent by new neighbors. The value of T(u) is as follows:

- T(u) = TI , if node u is in the initial neighbor discovery state of Figure 2.
- T(u) = TN(u), if node u is in the continuous neighbor discovery state of Figure 2, where TN(u) is computed according to the scheme presented in Section 4.

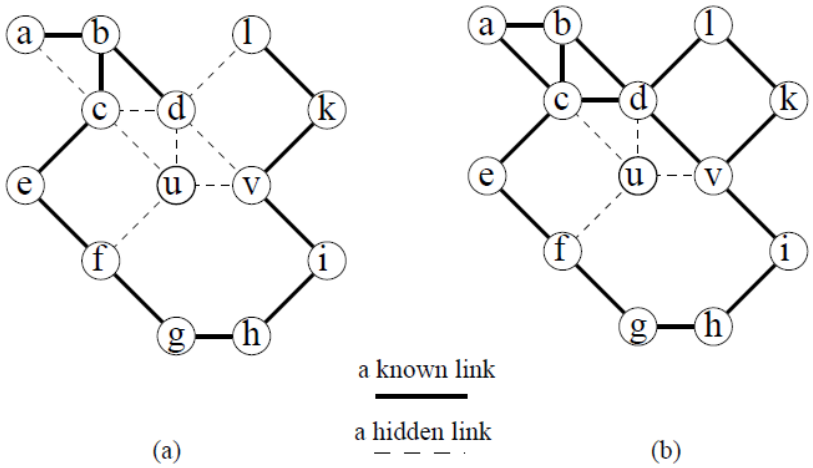

Figure 2: Segments with hidden nodes and links

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 2(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 2(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 degS (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 re-invoked. 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 segment neighbors participate in discovering u.

III. PROPOSED SYSTEM

For 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 neighbour discovery whereas the latter will be referred to as continuous neighbour discovery. While

previous works [1], [2], [3] address initial neighbour discovery and continuous neighbour discovery as similar tasks, to be performed by the same scheme, we claim that different schemes are required

A. ESTIMATING THE IN-SEGMENT DEGREE OF A HIDDEN NEIGHBOR

we consider the discovery of hidden neighbors as a common task to be performed by all segment nodes. To determine the discovery load to be imposed on every segment node, we need to estimate the number of in-segment nodes that are neighbors of every hidden node. That is the in-segment degree of the hidden neighbor, denoted by degS(u). In this section we present methods that allow node v in the continuous neighbor discovery state to estimate the number degS (u) of in-segment neighbors of its hidden neighbor u. Node u is assumed not to be connected to the segment yet, and it is in the initial neighbor discovery state. Three methods are presented:

- 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.
- Node v discovers, using Scheme 1, the number of its in-segment neighbors, degS(v), and views this number as an estimate of degS(u). This approach is expected to yield better results than the previous one when the degrees of neighboring nodes are strongly correlated.
- Node v uses the average in-segment degree of its segment's nodes and its own in-segment degree degS(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. A 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 in the segment S. Let Y be a random variable that indicates the degree $deg_S(u)$ of u in S. Note that u itself is not aware of the value of Y . Let Y' be the value of Y estimated by v. Clearly, we want Y' to be as close as

possible to Y . In what follows we analyze the three methods considered above and compare their accuracy and applicability. Since the in-segment degree of both the segment node (v) and the non-segment node (u) may have different values for different segment nodes, we use the mean square error measure (MSE) to decide how good the estimate is. The MSE is defined as $E((Y-Y')2)$. Since v and u are two random nodes in the same graph, we can claim that X and$Y$ have the same distribution. Let us denote the correlation between X and Y , corr($X; Y$), by C.

We assume that the node's average degree is small compared to the network size.

Let us denote the average graph degree by μ Clearly, $E(X) = E(Y) = \mu$ for the first method, the following holds:

$$MSE1 = E((Y- Y')^2) = E((Y - \mu)^2$$
$$= \mathrm{Var}(Y)$$

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

$$MSE2 = E((Y- Y')^2) = E((Y- X)^2)$$

$$\begin{aligned} MSE_2 &= E((Y-Y')^2) = E((Y-X)^2) \\ &= \sum_x \sum_y (y-x)^2 P(X=x, Y=y) \\ &= \sum_x \sum_y (y^2 - 2xy + x^2) P(X=x, Y=y) \\ &= E(X^2) + E(Y^2) - 2E(XY). \end{aligned}$$

$MSE2 = E(X^2) + E(Y^2) - 2(C\ \mathrm{Var}(X) + E(X)E(Y))$

$= E(X^2) + E(X^2) - 2C\ \mathrm{Var}(X) - 2E(X)E(X)$

$= 2E(X^2) - 2E(X)^2 - 2C\ \mathrm{Var}(X)$

$= 2\mathrm{Var}(X) - 2C\ \mathrm{Var}(X)$

$= (2 - 2C)\mathrm{Var}(X)$

for third method

$$MSE3 = E((Y' - Y)^2)$$
$$= E((CX + (1 - C)\ \mu - Y)^2)$$
$$= E(C^2 X^2 + 2C(1 - C)X\ \mu - 2CXY - 2(1 - C)\ \mu\ Y + (1 - C)^2\ \mu^2 + Y2)$$
$$= C^2\ E(X^2) + 2C(1 - C)E(X)\ \mu - 2CE(XY) - 2(1 - C)\ \mu\ E(Y) + (1 - C)^2\ \mu + E(Y^2)$$
$$= C^2\ E(X^2) + E(Y^2) + (2C - C^2 - 1)\ \mu^2 - 2CE(XY).$$

Using the fact that X and Y have the same distribution.

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

Hence, we have the following accuracy for the three estimation approaches:

1. Var(X)
2. (2 - 2C)Var(X)
3. (1 – C^2)Var(X)

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 $P = 1 - \frac{3}{4\pi}\sqrt{3} = 0.586503$.

if we assume uniform distribution of nodes, the three estimation approaches have the following accuracy.

1. Var(X)
2. 0:84Var(X)
3. 0:66Var(X)

We see that the third approach yields the best (smaller) *MSE*. However, note that this approach requires some global knowledge of the network topology, while the second approach requires only local knowledge.

B. AN EFFICIENT CONTINUOUS NEIGHBOR DISCOVERY ALGORITHM

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. Finally, suppose that 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 ∂, $0<\partial<1$ 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-$H(1-\partial)$ and t+$H(1-\partial)$. The length of this valid

time interval is $2H(1-\partial)$. Since the average time interval between two wake-up periods of v is $T_N(v)$, the robability that u and v discover each other during a specific HELLO interval of u is $\frac{2H(1-\partial)}{TN(v)}$

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

$1-(1-\frac{2H(1-\partial)}{TN(v)})^{n}$ 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 $\frac{H}{TI}$ and the probability that u is discovered during this time slot is $P_1 = \frac{H}{TI}(1-(1-\frac{2H(1-\partial)}{TN(v)})^{n}$ Denote by D the value of $\frac{T}{H}$ Then, the probability that u is discovered within at most D slots is $P2= 1-(1-P1)^{D}$. Therefore, we seek the value of $TN(v)$ 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 $P_1 = \frac{H}{TI}(1-(1-\frac{2H(1-\partial)}{TN(v)})^{n}$ we get

$$\frac{H}{TI}(1-(1-\frac{2H(1-\partial)}{TN(v)})^{n} \geq 1 - \sqrt[D]{1-P}$$

and therefore

$$T_N(v) \leq \frac{2H(1-\partial)}{1-\sqrt[n]{1-\frac{T}{H}(1-\sqrt[D]{1-P})}}$$

Since v does not know the exact value of n, it can be estimated.

IV. CONCLUSION

We presented an algorithm for determining the wake-up frequency of the nodes in a sensor network. This algorithm minimizes the energy consumption of the nodes and bounds the maximum delay on the routes from the nodes to the gateway. We simulated the algorithm over random sensor networks with different topologies and studied its impact on network energy consumption. This study revealed that the algorithm reduces the total energy consumption by 60-70% compared to energy consumption under equal assignment.

V. REFERENCES

[1] R. madan and S. Lall,"An energy optimal algorithm for neighbor discovery in wireless sensor network,"Mob. Netw Appl.,vol.11,no.3,pp.317-326.2006.

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

[3] M. J. McGlynn and S. A. Borbash, "Birthday protocols for low en-ergy deployment and exible neighbour 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] R. Madan, Department of Electrical Engineering, Stanford University, Stanford CA 94305-9505, U.S.A. ,An Energy-Optimal Algorithm for Neighbor Discovery in Wireless Sensor Networks.

[5] Wei Ye, John S. Heidemann, and Deborah Estrin. Medium access control with coordinated adaptive sleeping for wireless sensor networks. IEEE/ACM Trans. Netw., 12(3):493–506, 2004 .

[6] Gang Lu, Narayanan Sadagopan, Bhaskar Krishnamachari, and Ashish Goel. Delay efficient sleep scheduling in wireless sensor networks. In INFOCOM, pages 2470–2481, 2005.

[7] Gang Lu, Bhaskar Krishnamachari, and Cauligi S. Raghavendra. An adaptive energyefficient and low-latency mac for data gathering in wireless sensor networks. In IPDPS, 2004.

[8] Wei Lai and I. C. Paschalidis. Routing through noise and sleeping nodes in sensor networks: latency vs. energy trade-offs. In Proceedings of the 45th IEEE Conference on Decision and Control, pages 2716–2721, December 2006

[9] Abbas El Gamal and James Mammen. Optimal hopping in ad hoc wireless networks. In INFOCOM, pages 1–10, 2006.