Exercise: Ad Hoc Networks
Winter 2007/2008

Due date: 6.12.2007, 23:59 hours

1 Introduction

A typical wireless sensor network is expected to work without human intervention, on battery power, for a long time period. Therefore, it is critically essential to properly manage energy consumption such that the lifetime of the network is maximized. From energy consumption perspective, a sensor node can be in one of the four primary states: transmitting, receiving, idle listening, and sleeping. Sensor nodes consume the least amount of energy in the sleeping state, while most amount of energy is consumed in the transmitting and receiving states. In idle listening state, a sensor node continuously listens to the medium to look for any possible traffic that may be received. It is well-known that for typical sensors, especially those equipped with short-range radio, the amount of energy consumed in idle listening state is of the same order as consumed in transmit and receive states. Thus, the battery life of a sensor node is increased if it spends more time in the sleeping state, rather than in idle listening state, when there is no data to transmit or receive.

In this exercise, you are asked to implement a part of *redundant nodes sleep management protocol* named as HICA [1]. A randomly deployed network tends to have more nodes than required for normal network operations, hence *Redundant nodes sleep management* deals with putting *redundant nodes* to sleeping state so that network lifetime could be increased by switching sleep cycles between different subsets of nodes. A network whose subset of nodes are in sleep state should continue to provide *connectivity* guarantees. Connectivity refers to the presence of a network backbone of active sensors, allowing a sensor in any part of the network to establish a multi-hop communication with any other sensor in the network through the backbone. A network is said to have a degree of connectivity k or simply a k-connected network, if the removal of any $k-1$ active nodes does not render the network disconnected; this is also equivalent to having k independent backbones. The presence of a backbone is determined by the existence of a connected dominating set, where a graph G is said to have a connected dominating set S if every vertex v in G is either in S or adjacent to some u in S and the subgraph induced by vertices set S is connected.

Rest of this document is organized as follows. Overview of HICA is presented in Section 2. HICA's coordinator eligibility rule and contention Resolution is explained in Section 3 and Section 4. Section 5 outline coordinators' switching mechanism. Finally, instructions for completing exercise is provided in Section 6.

Table 1. Information Table at node x

u_c	v
w_c	a
y	b_c
z	c_c

2 Overview

HICA is a distributed algorithm hence each node runs it locally using information available to it. The information required at a node, to run HICA locally, consists of following: (1) two-hops neighbors information: a node keep record of its two hops [1] neighbors (2) two hops coordinator information: a node maintains information that which nodes in its two hops neighborhood are coordinators. In short a node x may maintains information as shown in table 1. Here node u, v, are one-hop neighbors of node x, node w, y, z are one-hop neighbors of node u and node a, b, c are one-hop neighbors of node v. Furthermore, table 1 also indicates that node u, w, b and c are coordinators.

This information is gathered in a table by having each node locally broadcast its direct neighborhood information. Based on this two-hop neighborhood information, some nodes become part of a backbone or connected dominating set; such nodes are called *coordinators*. A node x considers itself a candidate to become a coordinator if it is eligible to be a coordinator according to coordinator eligibility rule (see Section 3). However, node x formally becomes a coordinator, after a carefully calculated delayed local advertisement of its coordinator status. The advertisement is necessary so that other nodes can evaluate their coordinator eligibility. Furthermore, this advertisement is delayed to avoid multiple nodes becoming coordinators nearly simultaneously for providing connectivity to the same set of node pairs. Each node calculates this delay locally, based on its importance to the overall network connectivity and should advertise to become a coordinator only if it is still eligible after the calculated delay. A node v, who is not eligible to become coordinator can switch to the sleeping state.

3 Coordinator Eligibly Criteria

A node uses the coordinator eligibility rule to evaluate whether or not it should become or remain a coordinator. This rule runs locally on each node and uses the two-hop neighborhood information, described in previous section. For clearly defining HICA coordinator eligibility rule we use the following notation.

$\eta_{i,p}^{x}$: A boolean value which is true if and only if node x has an immediate

[1] Note that, all the nodes in a node's transmission range are within one hop distance from it

neighbor node i and all nodes in set p are immediate neighbors of node i.

HICA Coordinator eligibility rule: A node x, with immediate neighbors y and z, should be a coordinator except when all y and z can reach each other either (i) directly, or (ii) via one or two other coordinators, or (iii) through any three coordinators $\{u, v, w\}$ such that $(\eta^x_{v,\{u,w\}} \& \eta^x_{y,\{u\}} \& \eta^x_{z,\{w\}})$ is true, for all coordinators u, v, and w.

Note that node u is searched among one-hop coordinators known to node x, whereas node v and w are searched only among one hop coordinators known to node y and z respectively. Therefore, considering that the number of coordinators known to these nodes will be very small, HICA's coordinator eligibility rule does not pose computation overhead.

4 Coordinator Contention Resolution

In case a node finds itself eligible for becoming a coordinator, it formally becomes one by locally advertising its status. However, additional nodes may, nearly simultaneously, advertise that they provide connectivity to the same set of node pairs. To resolve this advertisement contention, a node must delay its advertisement by an amount that should correspond to the importance of that node in providing network connectivity, such that a shorter delay reflects higher importance. After the backoff delay, a node reevaluates its eligibility based on the coordinator advertisements received, and should advertise to become a coordinator only if it is still eligible. For calculating the backoff delay, HICA uses the following information which is locally available at a node:

- $N_{avg}(x)$: A network-wide constant that represents the anticipated average number of nodes in one-hop neighborhood. For this exercise we fixed its value equals to 100.
- $C(x)$: Set of pairs that would be connected if node x becomes a coordinator.
- $E_r(x)$: Amount of energy remaining at node x.
- $E^m(x)$: Maximum energy of node x.
- $R_y(x)$: Random variable uniformly picked between (0,y].
- P_e: A priority based on percentage of energy left in a node that changes after every 20% decrease in energy. P_e initial value is 0 and later set equal to the maximum value that the previous equation with old priority value could have taken.

Using the above notations the HICA coordinator advertisement delay equation is as follows:

$$\left[\left\{P_e + \left(\frac{1}{|C(x)|}\right)\right\} \times N_{avg}(x)\right] + R_{0.5}(x) \tag{1}$$

Note that, in case $|C(x)|$ equals to zero, then $\frac{1}{|C(x)|}$ is set to 1.

5 Coordinator Switching

The coordinators should switch so that the responsibility of being a coordinator is distributed between all nodes. In HICA, a coordinator x announces the time, called *work time* $t_c(x)$, for which it wishes to remain a coordinator. Neighbors of node x listen to $t_c(x)$ announcement and sleep accordingly. These neighboring nodes awake after time $t_c(x)$ and rerun the eligibility algorithm. In case there are more than one coordinator in a neighborhood, the nodes adjust their sleep period to the minimum of their announced work-times. Hence, if a non-coordinator node y knows coordinators $\{x_1, x_2, ..., x_n\}$ then it decides to sleep for time $t_c^{min} := min(t_c(x_1), t_c(x_2), ..., t_c(x_n))$. Later after t_c^{min}, neighboring coordinators and nodes reevaluate their coordinator eligibly. HICA follows following simple work time mechanism and more sophisticated work time estimation is left for future research.

$$t_c(x) := max\left(50, 100 \times \frac{E_r(x)}{E^m(x)}\right) \tag{2}$$

Note that above time is in minutes.

6 Instructions For Exercise

In this section we provide some important instruction that how one should work to accomplish above mentioned assignment and how we will grade your assignment.

1. You should not change in TinyOS core functionality
2. Your code should be well commented and should have small sized modules. Note that a well designed and easy to understand code will get you better marks.
3. You should not try to implement all code yourself and should divide work in hand into small parts. Each student of a your group should work on separate part. You could define some standard output of each part and later could easily combine them.
4. You will not be able to complete the assignment if you will work alone or start working during the last week. Hence start working now and make every member of the group useful.
5. No late submissions are allowed hence try to submit ahead of deadline.
6. Grading policy is illustrated below:
   (a) Your code compiles 10% marks
   (b) Your design for local information table 30% marks. It's design should facilitate fast search for applying coordinator eligibility rule and should be memory efficient. Furthermore, it should contain right information required for HICA.
   (c) Implementation of coordinator eligibly rule, 30% marks
   (d) Coordinator contention resolution, 10% marks
   (e) Coordinator switching, 10% marks
   (f) Code readability, design, code commenting 10% marks

References

1. F. Aslam, A. Vater, C. Schindelhauer, and Z. A. Uzmi, "Hardware Independent Connectivity Algorithm for Wireless Sensor Networks," in *IEEE-ICC (http://cone.informatik.uni-freiburg.de/teaching/labcourse/Adhocnetworks-w07/wsn/hica-icc.pdf)*, 2008.