Sensor Network Project

Team Members:
Gil Molina
Kaushik Agrawal
Mike Korenbaum

Goal:

Set up a shortest-path spanning tree over the nodes reachable from a central node. After the whole network converges to the static topology, the network should support the following features:

- Static Scenario: Each mote in the network displays its hop-distance from the central node on its LED. The LED would only allow a value up to 7 hops, but we are not going to test it on networks of larger diameter.
- Dynamic Scenario: Once you manually change the motes placement, say remove a certain mote, the network should recover to static state within a short time, say 1 minute. Related motes should update their number of hops on their LEDs.
- Path to root: Each mote is capable of sensing the TSR value. Once the value is below a threshold predefined in your program, the mote sends an alarm back to the central mote. And all motes along the route from this mote back to the central mote should turn on an LED, so that the route could be displayed.
- Network Size: Your code should work correctly on networks of up to 8 nodes and up to 4 hops.

Algorithms:

A) Minimum Spanning Tree:

Initialization: All the motes except the root mote are initialized to an infinite hop count, indicating that the mote does not know its parent.

Algorithm:

i) The root mote broadcasts its status (Hop count = 0, Parent Mote = 0) at intervals of 10 seconds.

ii) Whenever a mote receives a broadcast message, it tries to update it's current hop status to the root in the following manner:
* If the received hop count is less than the currently stored value, the sender of the message becomes the new parent of the node.
* Otherwise, the node silently ignores the broadcast message.

iii) The mote broadcasts its current hop status each time it accepts a new parent or after every fixed interval of 10 seconds.

iv) Since the node may move to a position where it would need to increase its hop counter, the hop count is set to infinity after a long period. This timeout is the same as a cache timeout. The mote times out only in the situation where it stops receiving the broadcast from its parent mote or if the parent starts broadcasting a different higher hop value.

B) TSR value:

Initialization: The threshold for the TSR value is set to 45. The initial TSR value is set to infinity.

Algorithm:

i) The TSR value is continuously read by the mote through the solar sensor on its circuit board.

ii) Whenever the TSR value drops below the pre-defined threshold, the following action is taken:

* The node sensing the drop triggers an alarm in the form of a message to its parent. As long as the mote keeps sensing the drop, the node continuously keeps broadcasting the alarm to the parent.

* The parent in turn sends the alarm to its own parent. This chain continues to the root.

iii) Each node in the chain turns off the alarm after a fixed interval. In effect, the alarm will be on as long as the node keeps receiving the alarm and gets shuts off when it stops receiving it.

Implementation:

Initialization: The basic unit of time for the implementation is 50 milliseconds. Consider this as a slot time. The yellow LED is kept ON to indicate that the mote does not have a parent.

Back-off mechanism: Whenever a mote is ready to transmit a packet, it chooses a random back-off interval. It chooses a random number from 0 to 9 and backs off for that much number of slots.

Implementation:

i) The current hop "x" for the mote is displayed as "x" number of blinks of green LED. The hop is displayed every 200 ms.

ii) Each mote is programmed to send and receive 2 kinds of messages:
   a. Hop Message: It contains the hop count and the address of parent mote.
   b. Alarm Message: It contains the address of the parent mote.

iii) Each mote broadcasts its hop status every 10 seconds for any new motes that are now in range. Also whenever a mote accepts a broadcast message from another mote, it in turn broadcasts its new status. These two steps help for the inclusion of new motes in the network dynamically.

iv) Before transmitting a hop message, a mote randomly backs-off using the back-off mechanism described above. On the completion of the back-off interval, the mote completes the broadcast of the hop message. This helps prevent collisions during broadcasts.

v) We periodically sample the TSR value every 1 second. Then we check the TSR value against the threshold every 1.3 second. If the TSR value drops below the threshold:
   * We turn the red LED on and broadcast the alarm message.
   * Otherwise we just throw away that sample.

   For all the other motes in the network, when they receive an alarm message, they check to see if the message contains its address. If so, it turns on the red LED and forwards the message.

   All the motes attempt to turn off the alarm LED every 2 seconds. However, if the mote keeps receiving alarm messages, the timer is reset and the red LED will remain on.

vi) Since a mote can move to a position where its hop count increases, every mote invalidates its hop count and parent every 45 seconds.