PROJECT: WIRELESS SENSOR NETWORKS

SOURCE ROUTING FOR DOWNWARD DATA TRAFFIC

July 5, 2017

Subhankar Roy
Telecommunication Engineering
University of Trento, Italy

Contents

Chapter 1

Introduction

As IoT gains momentum in today's world it gives us good reasons to study in great detail how wireless nodes relays information from the source to the sink and also vice-versa. A wireless sensor network is an agglomeration of sensor nodes, alternatively called as motes, which are deployed in the real world, primarily at different geographic locations within the communication reach. Nodes communicate with each other by sending messages which essentially contains data packets to be forwarded to other nodes or for its own interest. Generally, the sole purpose of the nodes is to gather information through its sensors, viz,. temperature, humidity, etc, and disseminate the garnered information to the *sink*.

The sink holds the responsibility to aggregate all the information from different nodes and perform necessary calculations or storing them for further offline analysis. Sometimes, the sink might want to send an acknowledgement to a particular node in the same network for the data received or the sink might want to actuate a node for physical operation. Hence, it is of utmost importance to the sink to learn the topology of the network so that it can carry out these desired activities. Once the sink or the root node has every knowledge to route packets across the network, it can embark upon *one-to-many* routing. In this report we are going to inform ourselves with the methodologies and the algorithms that will realize the objectives of the project, which is source routing of downward traffic data.

The rest of the report is organized as follows. Section 2 describes the methodology of tree building. In Section 3 the mechanism of building the source route table is elaborated. Section 4 explains how the source routing is implemented. Finally the report wraps up with the observations and a conclusion in sections 5 and 6 respectively.

Chapter 2

Tree Building

Before the sink can start building the routing table for one-to-many source routing, it needs to initiate the process of *tree building*. It is nothing but flooding the network with packets so that every node knows who its parent is. In other words, every node knows who to send data to in case some information needs to be passed towards the sink.

The process begins with the sink initiating the flooding, with packets containing a routing metric. Every node on receiving such message will compare the received metric with a previously received metric. In case it finds the metric to be healthier than the previously received metric then it adopts the sender as the parent and immediately broadcasts a data packet containing its own metric. The process ends with every node possessing a parent. Tree building process is periodic with a period of three minutes.

Interface *MyCollection.nc* provides a **buildtree** command to initiate the tree building process. Only the sink is permitted to trigger the above process.

Chapter 3

Building Source Routing Table

Once the tree has been built, the nodes, except the sink, can start sending data to the sink. The only information contained in the data packet is the *node id* of its parent. All the intermediate nodes will simply forward the packets, originated from other nodes, to its parent. As a result, the information from all the nodes climbs up the network and is accumulated in the sink node.

The sink node maintains a hashtable which contains a *key* and a *value*. The *key* is the ID of the node where the data packet originated and *value* is the ID of the parent. The table is populated with the information sent by the individual nodes. This hashtable will be utilized while source routing a data packet towards the destination node. Table 5.1 is an example of the source routing table with the entries bearing usual significance.

Event **notifyParent** is added to the interface *MyCollection.nc* to help the sink building routing hashtable for source routing.

Chapter 4

Source Routing

In source routing each data packet contains the complete routing information to reach the destination node. The intermediate nodes will play no part other than forwarding the packet as directed in the information header of the routing packet sent by the sink.

With all the necessary information at the disposal the root node can request to send data to any node in the network. The ordinary nodes will be notified when the packet destined to itself arrives.The root nodes fabricates a packet, which contains the list of forwarding nodes and a payload. The algorithm adopted is described as follows:

1. **At the root**: When the application at the root node **S** requests to send a packet to the destination node **D** the following algorithm should be executed:
   (a) Assign N := D
   (b) Search for node N in the routing table as constructed above to find N's parent P
   (c) If N is not found or a loop is detected, drop the packet
   (d) If P == root, transmit the packet to next-hop node N
   (e) Else add N to the source routing list of the packet, assign N := P, go to step (b).
2. **At the forwarders**: When an intermediate node receives a packet to be forwarded, it modifies the routing header by removing the next hop node address from the list and transmits the reduced packet with the payload to the next hop node. If a node receives a packet with empty forwarding list, it delivers the packet to the application. Same procedure is repeated until the packet reaches the destination.

Packet format: The sink nodes sends the packet in form of an array encapsulated by a struct. Due to the limitations of C allowing to make a dynamic array, the array is always

initialized to `MAX_PATH_LENGTH`(value 10). As a result the initial values of the packet are initialized with `-1` if the number of intermediate nodes are less than the defined constant. The payload always occupies the final cell of the array. The payload looks as below for the experimental set-up described in the following section.

-1	-1	-1	-1	2	3	4	5	6	7	Payload

As the array is received by non-destination nodes, the first non-negative entry is stripped off and replaced with a `-1`. When the packet arrives at the destination node, the payload is delivered and the node is notified. The packet is not forwarded any further.

Interface *OneToMany.nc*, containing a command **send** and an event **receive**, has been added and implemented by the component *OneToManyP.nc*. Appropriate wirings are also in place in the form of a configuration component.

Chapter 5

Experiment and Results

Out of the several experiments conducted with varied topology only one has been described in Figure 5.1. In this set up, seven nodes form a network with the assumption there are no network partitions.

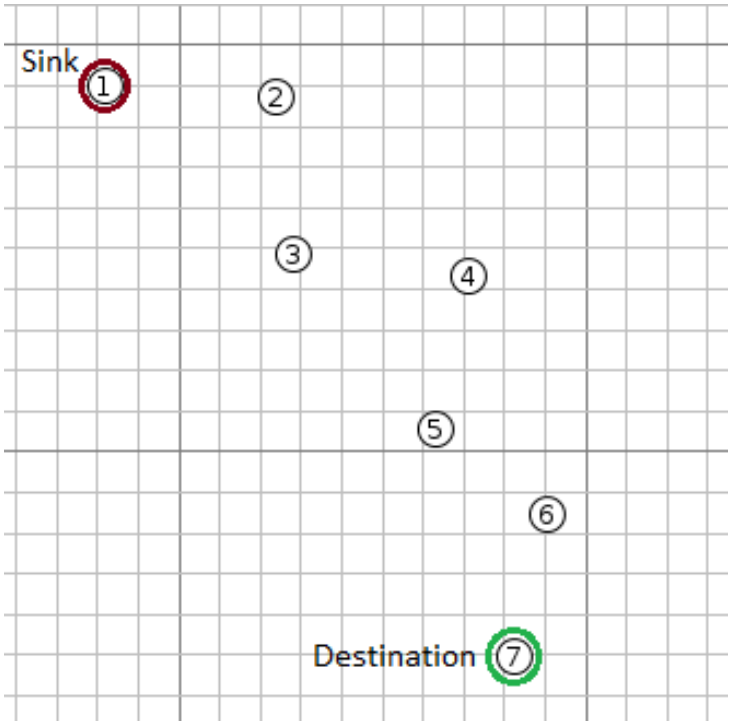

Figure 5.1: Network topology is shown with sink and destination nodes circled and labeled

After tree building, the nodes send necessary information to the sink to build the hashtable 5.1. This table is rebuilt once the tree has been rebuilt.

Once the table has been built at the sink, the sink can initiate the downward data transfer

Key(Child)	Value(Parent)
2	1(S)
3	2
5	4
4	3
7(D)	6
6	5

Table 5.1: The hashtable maintained at the Sink. Node 1 is the Sink and node 7 is the destination

to destination node 7 by a button click. Once the button click event is registered, it dispatches the array as described in the previous section. The payload consist a counter that increments with every message sent. It is recycled with a maximum value of 8. The path traced by the routed message is shown in figure 5.2. The experiment concluded with the Leds blinking at every button click, indicating the reception of a payload. The colour of the Led conforms with the payload dispatched from the sink.

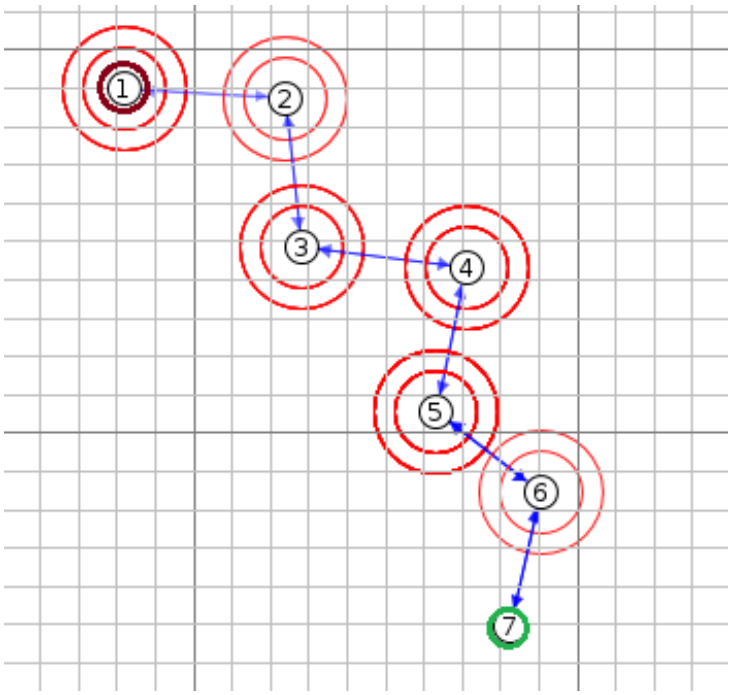

Figure 5.2: Downwards data flow from sink to destination

Chapter 6

Conclusion

Tree building is one of the most fundamental steps in a network of nodes to learn about the topology of the network. Once the topology is learned by the sink, it can build a data structure containing the essential routing information pertaining to every node. Then it can initiate a downward flow of data from the sink to the destination node

One of the advantages of source routing is that it relieves the nodes of maintaining routing information into a limited node memory. The sink, owing to generally a computer with higher memory, can afford to store all the routing informations.