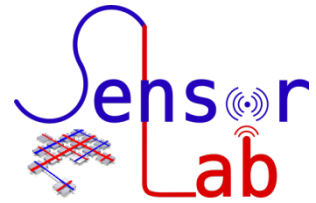




GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN



Georg-August-Universität Göttingen
Faculty of Mathematics and Computer Science
Institute of Computer Science
Telematics Group
Sensorlab

Lab 4 –TinyOS Networking

Chencheng Liang – University of Goettingen – 11603960

Introduction

In this lab the questions get me to know about hidden and exposed problem in WSN. Trickle algorithm is described in question 4.3, which can keep all nodes in the WSN get the newest information to reach consistent state. And in question 4.4 two network protocols Dissemination and Collection are described.

In assignment 4.1, an application is implement, in which all motes broadcast their message, and receive message from each other, then do the corresponding actions(turn on or off the LEDs). And in the base station, it listen to all message passed by.

In assignment 4.2, the programs can be installed to the motes remotely from the base station by Deluge.

First store the program to the base station, then transmit the program to the motes, and send a signal to reboot the motes and tell the motes to run which program.

Question 4.1 [1]

In Figure 1, A cannot hear B and B cannot hear A directly. When A sends message to C, B do not know that A is sending message to C, and B will also send message to C. Then the collision occurs in C.

There are multiple way to solve this in following:

Increase transmitting power of nodes: By to do so, the radio range of the nodes are increased, so the nodes can hear each other.

Move the node: provide the mobility features to the nodes.

Using pooling and token passing strategy .

Use omni directional antennas.

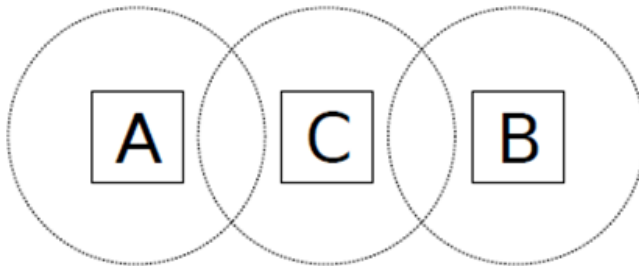


Figure 1

Question 4.2

In Figure 2, if B is sending message to A, and C want to send message to D, but C knows B is sending and B is in its range, so C will not send message to D. But actually, C can send message to D, and will not interfere A to receive the message from B. So C is exposed by B [1].

Expose problem will cause unnecessary delay.

IEEE 802.11 RTS/CTS mechanism helps to solve this problem. When a node hears an RTS(request to send) from a neighboring node, but not the corresponding CTS(clear to send), that node can deduce that it is an exposed node and is permitted to transmit to other neighboring nodes [2].

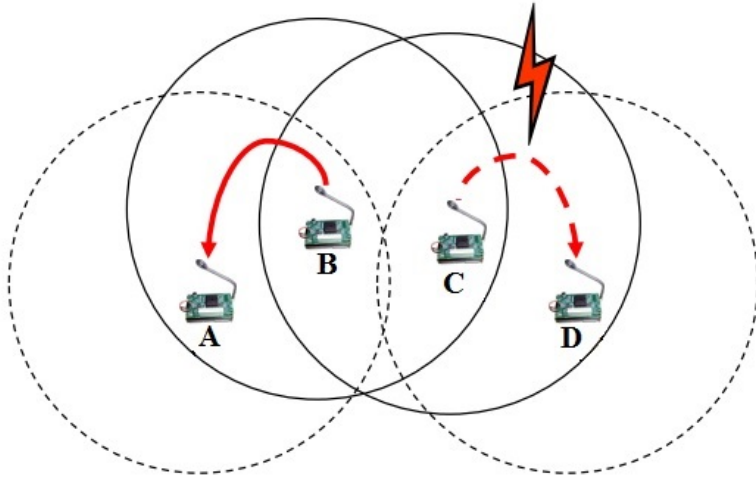


Figure 2 [1]

Question 4.3 [3]

If the new data is introduced to a node, this node broadcast the newest information to other nodes. The sending rate has now been dynamically changed, minimised to a small value so that the node broadcasts that it has new data frequently. The nodes received this inform know that their data are outdated. Then these nodes will ask the broadcaster to send them newest data. Then the broadcaster will send the newest data to other nodes. And repeat this step, to all nodes in the network keep the newest information.

The purpose of this algorithm is to maintain the WSN by propagating the newest code and bring all nodes to a consistent state.

Question 4.4[4]

Dissemination: It is used to deliver a piece of data to other nodes in the network. It allows administrators to reconfigure, query, and reprogram a network. Reliability is important because it makes the operation robust to temporary disconnections or high packet loss.

Collection: It is the complementary operation to disseminating and it collects the data generated in the network on a base station. The general approach used is to build one or more collection trees, each of which is rooted at a base station. When a node has data which needs to be collected, it sends the data up the tree, and it forwards collection data that other nodes send to it.

Assignment 4.1

In `Mts400TesterC.nc`, add `AMReceiverC()` component to make the nodes can receive the message from each other. And wire it.

In `DataMsg.h`, add "`nx_unit16_t Reference`, `nx_unit16_t MyLED`, `nx_unit16_t Sensor_ID`" to store the message. If the `Reference=1`, which means the node is the reference node. `MyLED` is about to tell the base station, the state of the LEDs on the node, which we can verify the results. `Sensor ID` will tell the nodes the source id of that message.

In `Mts400TesterP.nc`, add the interfaces (`Receive`, `Packet`, `AMPacket`) for receiving message and get the information of nodes, such as node ID and source ID.

Under `InfraredLight.readDone()`, use "`datamsg_t* packet = (datamsg_t*)(call`

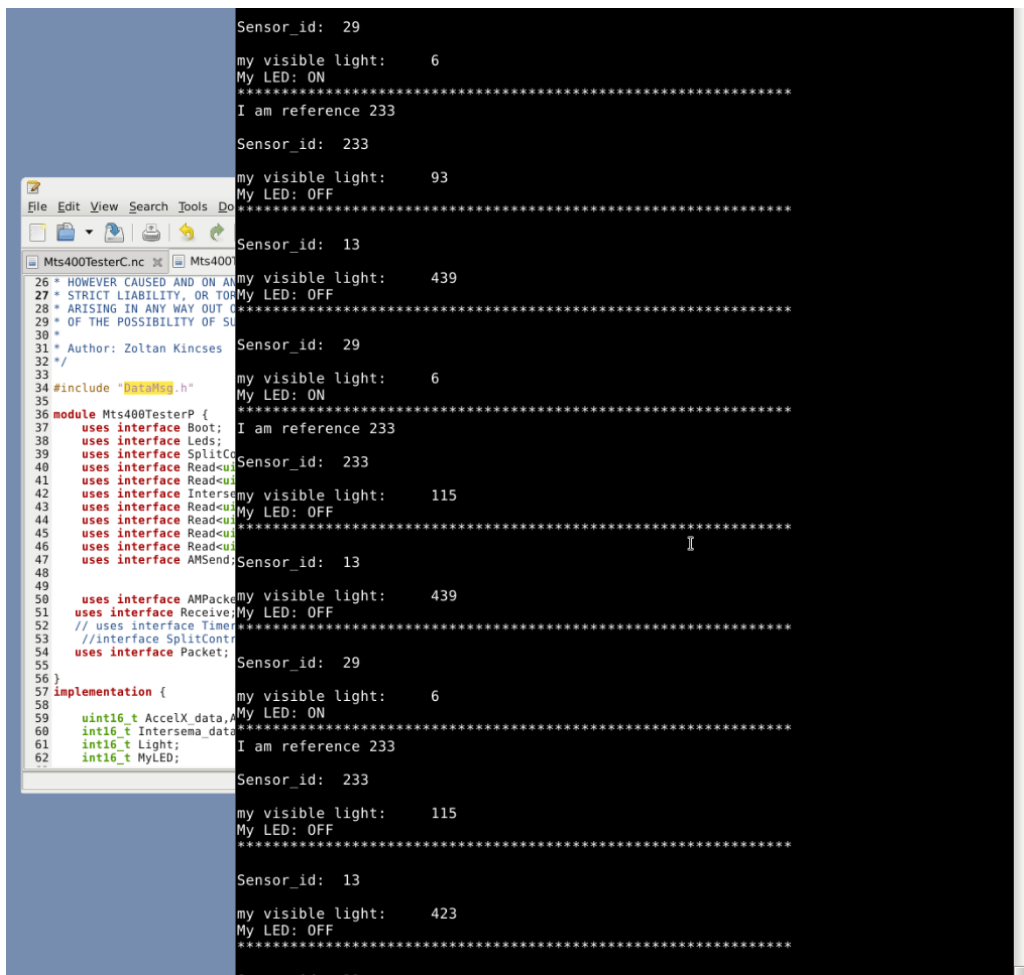
AMSend.getPayload(&message, sizeof(datamsg_t));" to get the package into payload, then fill in the message, which including if this node is the reference(I set node 233 as the reference node. If this node id is 233 then the variable Reference will be set to 1), and light data. Then use " call AMSend.send(AM_BROADCAST_ADDR, &message, sizeof(datamsg_t));" to send the message.

Under event message_t* Receive.receive(), the information about reference light, source node id, and if it is a reference will be get off from payload, then I can process these information, to implement the requirement of this assignment.

In Mts400Tester.java, use " get_Sensor_ID()", " get_VisLight_data()" and" get_MyLED()" to get information from the message from other nodes. And print them on the screen.

Then install program to the nodes, and use base station to listen to the message.

The result is as following:



```

Sensor_id: 29
my visible light: 6
My LED: ON
*****
I am reference 233
Sensor_id: 233
my visible light: 93
My LED: OFF
*****
Sensor_id: 13
my visible light: 439
My LED: OFF
*****
Sensor_id: 29
my visible light: 6
My LED: ON
*****
I am reference 233
Sensor_id: 233
my visible light: 115
My LED: OFF
*****
Sensor_id: 13
my visible light: 439
My LED: OFF
*****
Sensor_id: 29
my visible light: 6
My LED: ON
*****
I am reference 233
Sensor_id: 233
my visible light: 115
My LED: OFF
*****
Sensor_id: 13
my visible light: 423
My LED: OFF
*****

```

In this result, we can see that if the light of other nodes darker than the reference node 233, then they will turn on their LEDs. If the light of other nodes brighter than reference node 233, they will turn off their LEDs

Assignment 4.2

First install DelugeBasestation to a mote(as usual use " make iris install,0 mib520,/dev/ttyUSB0 "). Install GoldenImage to other 3 motes, program GoldenImage will not light the LEDs, so when we turn on the motes, there is no response.

Then I use the mote with DelugeBasestation program to send a new program(Blink) to other motes remotely.

First go to Blink directory, use command " `tos-deluge serial@/dev/ttyUSB1:57600 -i 1 build/iris/tos_image.xml` " to store Blink program in the base station. Then use command " `tos-deluge serial@/dev/ttyUSB1:57600 -d 1` " to send Blink program to other motes. After sending done, use command " `tos-deluge serial@/dev/ttyUSB1:57600 -dr 1` " to reflash the remote nodes to install Blink program in their first slot. And I see that after rebooting, the other 3 nodes are blinking, which means the program Blink is installed successfully remotely.

And use the same method, I install the GoldenImage program to the slot 2 of the 3 nodes. Then by using command " `tos-deluge serial@/dev/ttyUSB1:57600 -dr 1` " and " `tos-deluge serial@/dev/ttyUSB1:57600 -dr 2`", I can switch the programs(GoldenImage and Blink) running on the 3 nodes.

Lessons Learned & Conclusion

In this lesson, I learned how to write a small application, which seems useful. And I can use base station to monitor all messages. In know how to send and receive messages from both motes and base station. There are many problems I met in this lab, for examples, I forgot to add head file to my code, I forgot to reboot the motes, the variables should be declare before other sentences. I debug my code and check my steps carefully, then eventually I got these results.

I learned how to use Deluge base station to send and install programs to the motes. By doing so, I no longer need to install the program to the motes one by one with USB programming board, and remove the motes from USB programming board, then give them batteries and get them to the work place then turn on them. With Deluge, I save a lot of time to do that repeated work. Especially when I test my code, I just need to write a script to compile everything and send my program to the motes and reboot them. And I thing for some scenario, Deluge is absolutely necessary, for example, if there are hundreds of motes need to be programmed and placed to the work place, we need deluge to manage this thing, otherwise, it is too time-consuming. And if there are millions of motes, it is impossible without using Deluge.

I will use Deluge to help my work on other labs.

In the questions, I learn what are hidden and exposed problems, I used to encounter these questions when I was leaning ad-hoc network. I learned trickle algorithm, so I understood how the motes can reach the consistent state.

References

- [1] <http://er.yuvayana.org/hidden-terminal-and-exposed-terminal-problem-and-its-solution/>
- [2] https://en.wikipedia.org/wiki/Exposed_node_problem
- [3] <https://www.scss.tcd.ie/publications/tech-reports/reports.10/TCD-CS-2010-22.pdf>
- [4] http://tinyos.stanford.edu/tinyos-wiki/index.php/Network_Protocols