CS 558: Computer Systems Lab

Assignment – 2: Network Simulation using Cooja

SUBMITED BY:

(1) Pinaki Das (204101042)
(2) Sanket Patil (204101041)
(3) Pankaj Panwar (204101040)
(4) Prabhat Singh (204101043)

Index

(1) Introduction

(1.1) What is Cooja ? …… 1

(2) Goal ……..…….1

(3) Basic Terminology ……..……2

(3.1) TSCH ……...…….……2

(3.2) 6TiSH ……3

(4) Problem statement ……..……..…….3

(4.1) Parameters Used ……...…….……3

(5) Setting Up Experiment …….…….……4

(5.1) Cooja Simulator Initialization ……..…….……4

(5.2) Creation of Motors …….……..……4

(6) Results ……...6

(7) Conclusion ….……10

Intorduction

Internet of Things in short IoT is a trending technological field that converts any electronic device into a smarter one. A lot of industries are beginning to adopt this technology on to their operations to increase their productivity and improve efficiency.

From washing machines to garage space in houses, IoT technology is bringing a large number of day-to-day objects into the digital fold to make them smarter. It is also evident that the IoT technology is going to transform into a multi-trillion dollar industry in the near future.

1.1 What is Cooja

A cross-layer java-based wireless sensor network simulator distributed with Contiki. It allows the simulation of different levels from physical to application layer, and also allows the emulation of the hardware of a set of sensor nodes.

2. Goal

Main Goal of this Assignment is to Evaluate the possible benefits in terms of real time requirements of implementing 6TiSCH based IoT network. To implement this evaluation we have created different simulation scenarios in an Emulator called “Cooja” . Cooja simulator is a part of the contiki-ng operating system . From running these different simulation scenarios we hope to show that by adjusting a certain parameter in the TSCH protocol certain real time requirements will respond in a predictable way to this. The parameter we will be adjusting in our simulations is the enhanced beacon (EB) period. The EBs are a certain message that are used within the TSCH protocol and these EBs are sent out periodically by each node within the TSCH network with the purpose of letting other nodes join the network. Having these sent out frequently would let other nodes join the network faster and in addition it should let the network be more resilient to disruptions caused by nodes moving in and out of the network due to mobility. Increasing the period between these EBs would reduce the total amount of data sent within the network and thereby letting the nodes use less of their assigned time slots for EBs and thereby reducing the delay.

TSCH: Time Slotted Channel Hopping (TSCH) is a MAC layer of IEEE 802.15.4-2015. TSCH was introduced in 2012.This is a Contiki-NG implementation of TSCH and the 6TiSCH so-called minimal configuration, which defines how to run a basic RPL+TSCH network. TSCH

aims at reducing the impact of the wireless medium unpredictability to enable the use of reliable low-power wireless networks. It is very good at saving the nodes energy because each node shares a schedule, allowing it to know in advance when to turn on or off its radio.

6TiSCH: The IETF IPv6 over the TSCH mode of IEEE802.15.4e (6TiSCH) are a set of standards and protocols being primarily developed by the 6TiSCH Working Group (WG) but it is also an open standard. 6TiSCH is being developed as an open standard which means that anyone can support the work. The 6TiSCH WG deems it necessary to develop 6TiSCH as an open standard for the reason that doing this will make it easier for the industry to adopt the protocol. 6TiSCH builds upon the IEEE802.15.4e Time Slotted Channel Hopping (TSCH) protocol and implements several other protocols on top of that to handle multi-hop routing and provide support for ip compatibility through ipv6. 6TiSCH provides solutions for several problems as specified by the 6TiSCH WG. These include solutions for zero-configuration network bootstrap of TSCH, network access authentication and parameter distribution, and management of the wireless medium through scheduling.

4 . Problem Statement :

In Time Slotted Channel Hopping (TSCH) and Routing over Low Power and Lossy Networks (RPL) based Low power and Lossy Networks (LLN), new nodes (pledges) join the network one by one. At the beginning, this kind of network formation is started by the RPL root node or TSCH coordinator node. The pledges need to receive network advertisement information carrying Enhanced Beacon (EB) frame to get synchronized with the TSCH network. Similarly, pledges complete their joining process once they receive routing information carrying RPL DIO packet. Pledges are called 6TiSCH joined node once they complete their joining process (i.e., receive both the EB and DIO packets successfully). Both the TSCH joining/ synchronization time and 6TiSCH joining time is very much important for resource constrained LLN devices. In this assignment, both the joining times (i.e., TSCH synchronization time and 6TiSCH joining time) need to be calculated using the Contiki-NG based Cooja Simulator

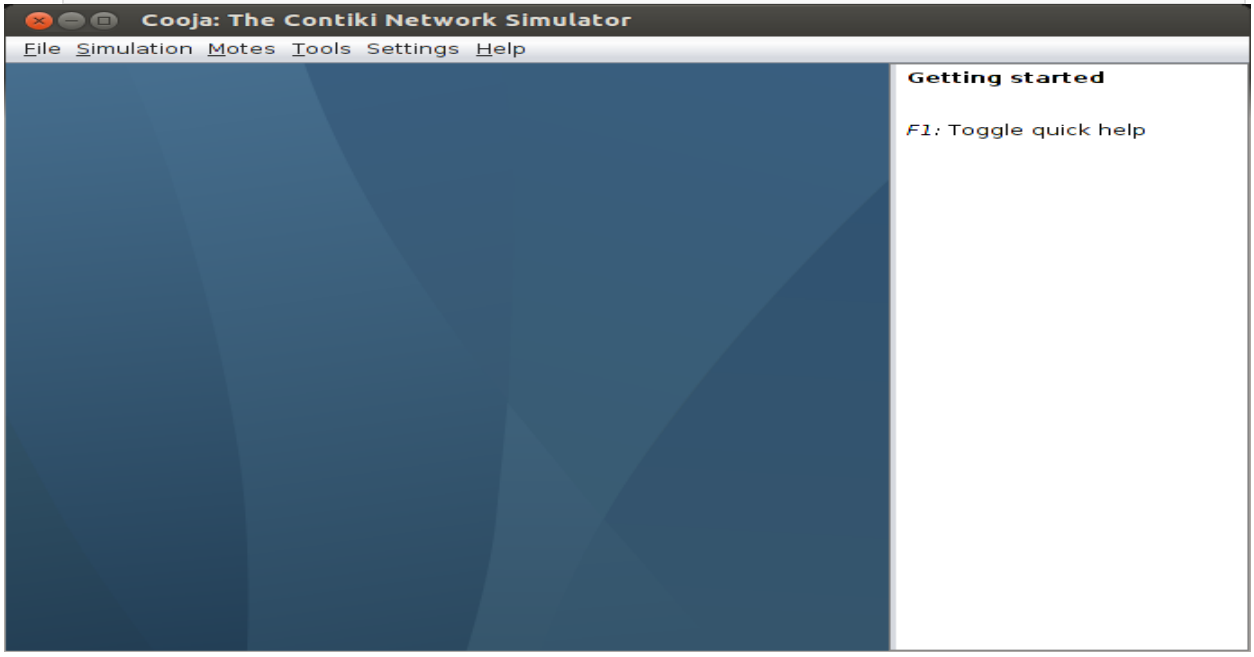
Paramters Used :

- Slotframe Length (SF) : : 49, 67
- Timeslot duration : : 10 ms
- RPL version : RPL Lite
- Number of Channels : 16
- Topology : 5 x 5 grid topology (25 nodes)
- Radio Distance between two nodes (X,Y) : 150
- Radio Range : 50 meter
- Interference Range : 75 meter
- Scheduling Function : 6TiSCH minimal configuration scheduling

(5) Setting Up Environment

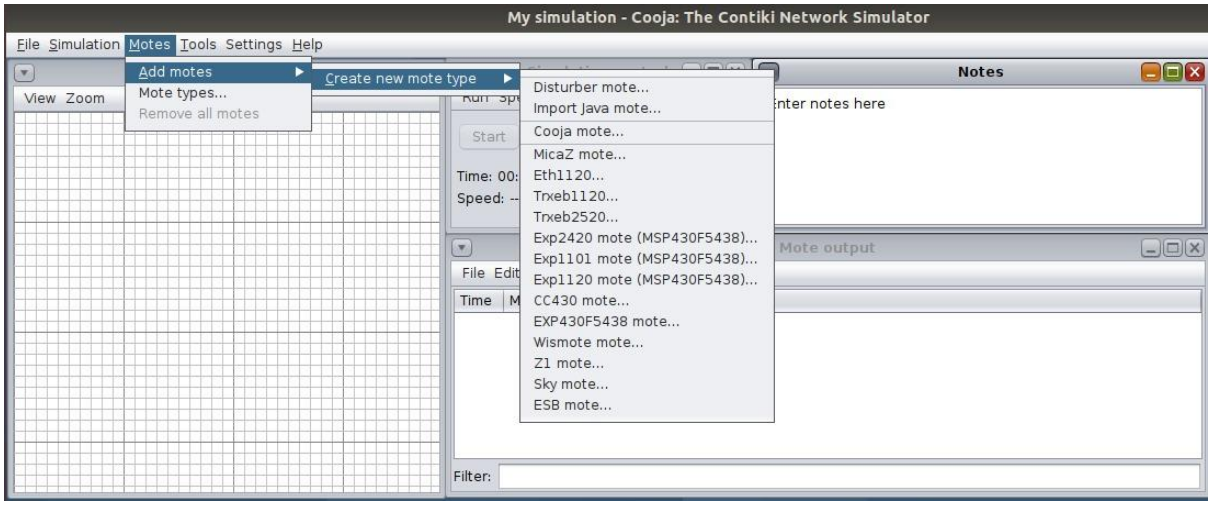
(5.1) Cooja Simulator Initialization

```
cd Contiki-ng/tools/cooja
ant run
```

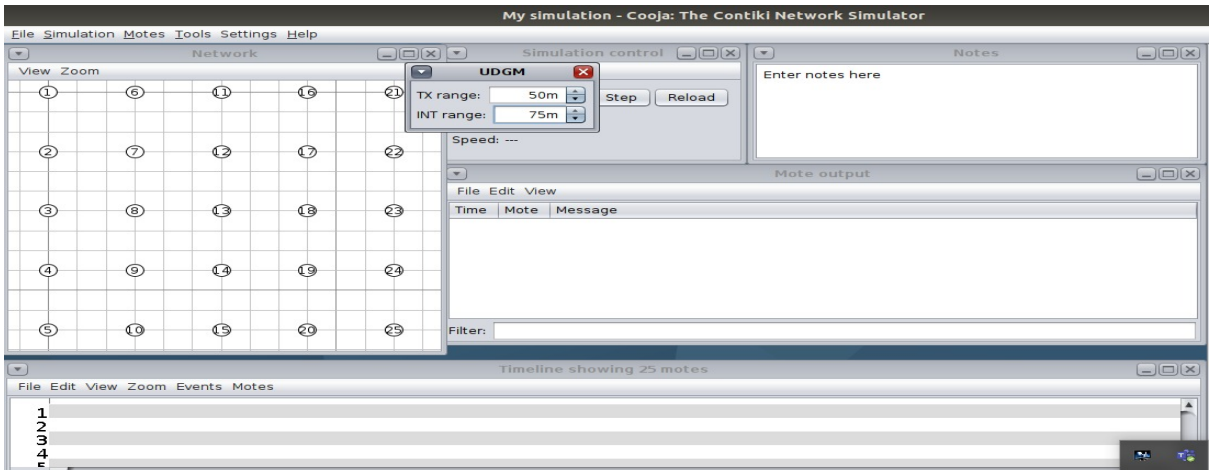


5.2] Creation of the Motes

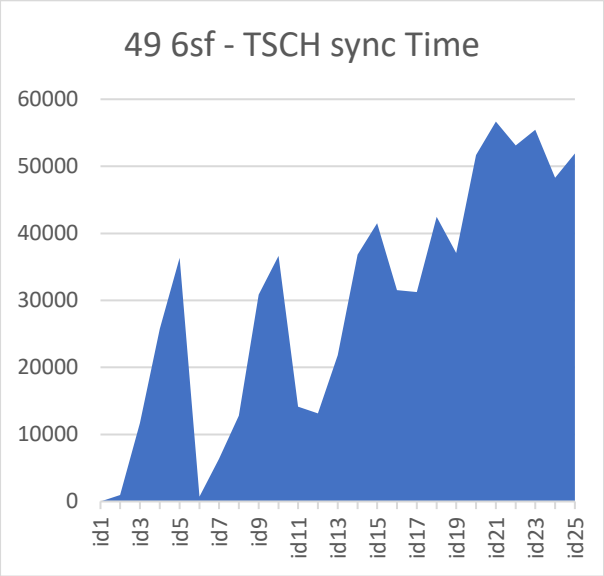
- Click on the file.
- Create new simulation.
- Now click on the add motes.
- Select the type of the motes as Cooja mote.

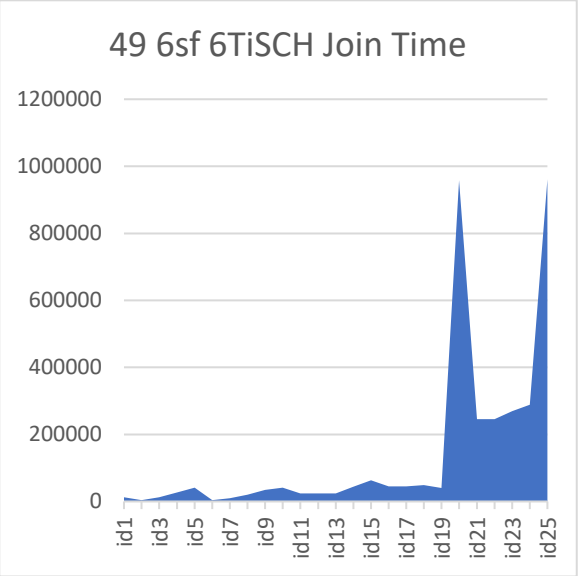


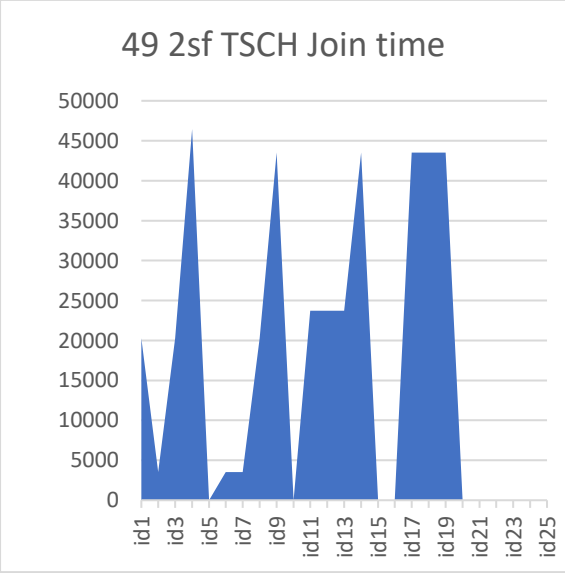
- Now browse the directory contiki-ng/examples/6TiSCH/simple-node
- Select the node.c file and open it.
- Clean it once and start compilation of the motes.
- After compiling click on the create button.
- Select the no of motes as 25 and linear positioning .
- Set the distance between two nodes as 150.

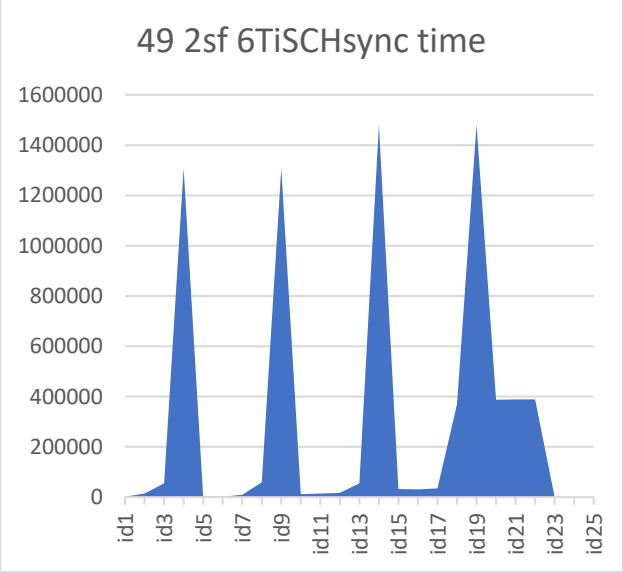


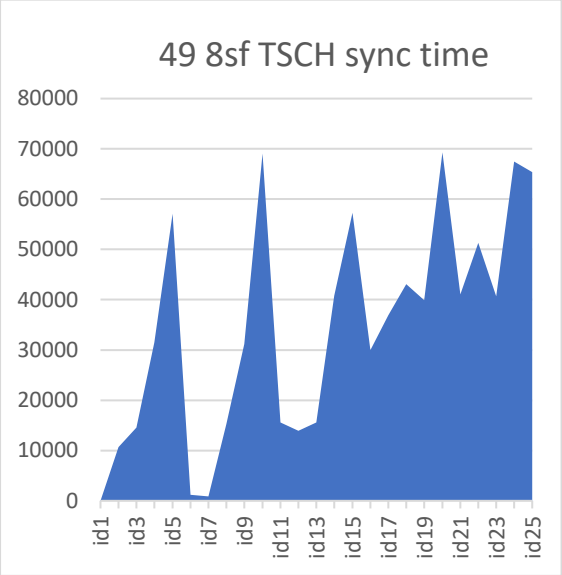
Results

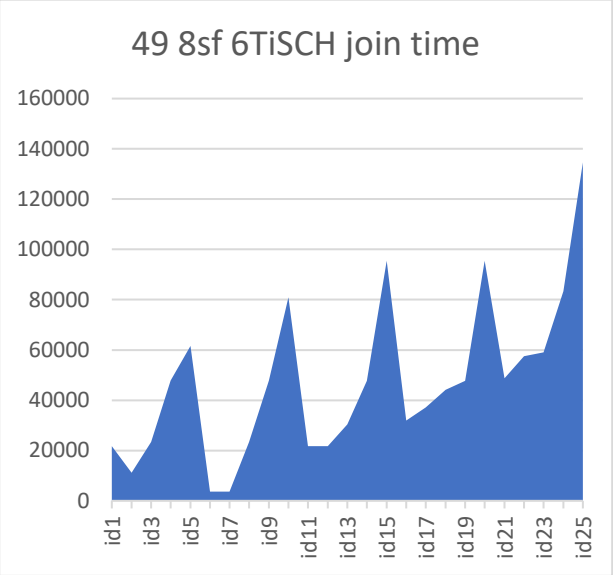

49 6sf - TSCH sync Time
60000
50000
40000
30000
20000
10000
0
id1
id3
id5
id7
id9
id11
id13
id15
id17
id19
id21
id23
id25

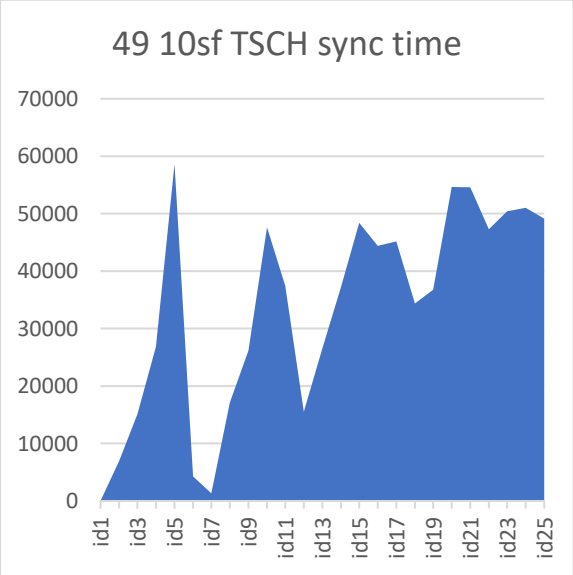

49 6sf 6TiSCH Join Time
1200000
1000000
800000
600000
400000
200000
0
id1
id3
id5
id7
id9
id11
id13
id15
id17
id19
id21
id23
id25

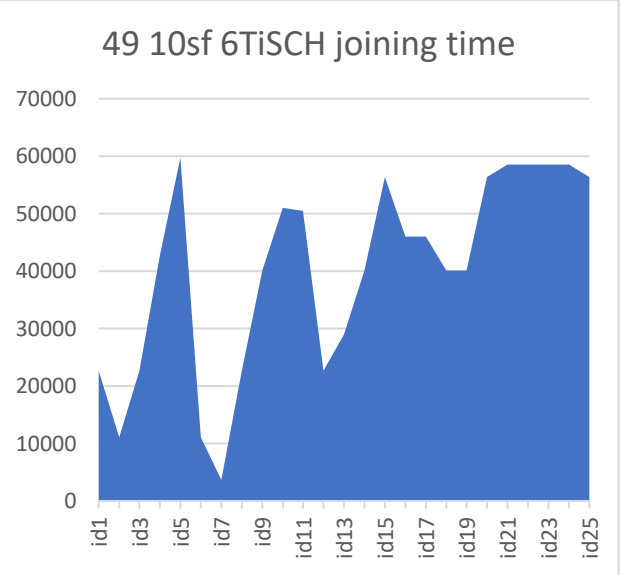

49 2sf TSCH Join time
50000
45000
40000
35000
30000
25000
20000
15000
10000
5000
0
id1
id3
id5
id7
id9
id11
id13
id15
id17
id19
id21
id23
id25

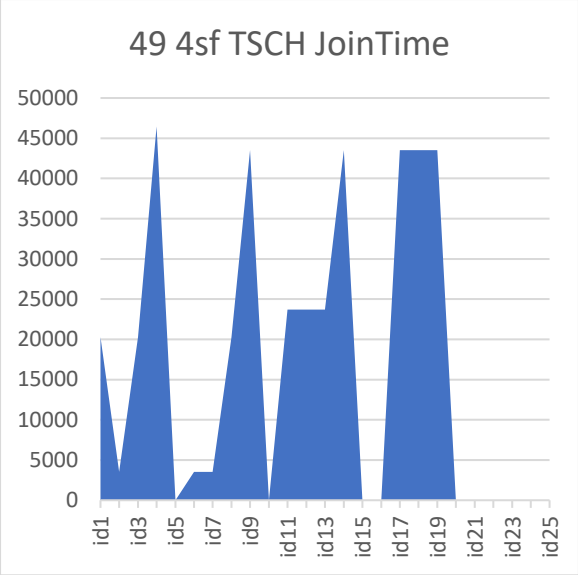

49 2sf 6TiSCHsync time
1600000
1400000
1200000
1000000
800000
600000
400000
200000
0
id1
id3
id5
id7
id9
id11
id13
id15
id17
id19
id21
id23
id25

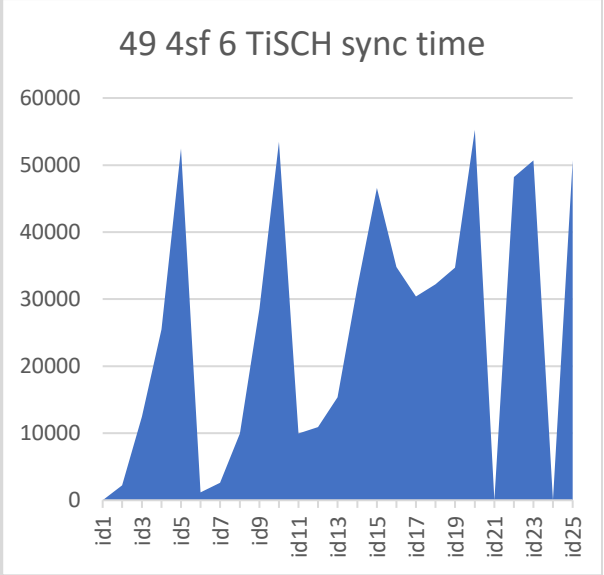

49 8sf TSCH sync time
80000
70000
60000
50000
40000
30000
20000
10000
0
id1
id3
id5
id7
id9
id11
id13
id15
id17
id19
id21
id23
id25

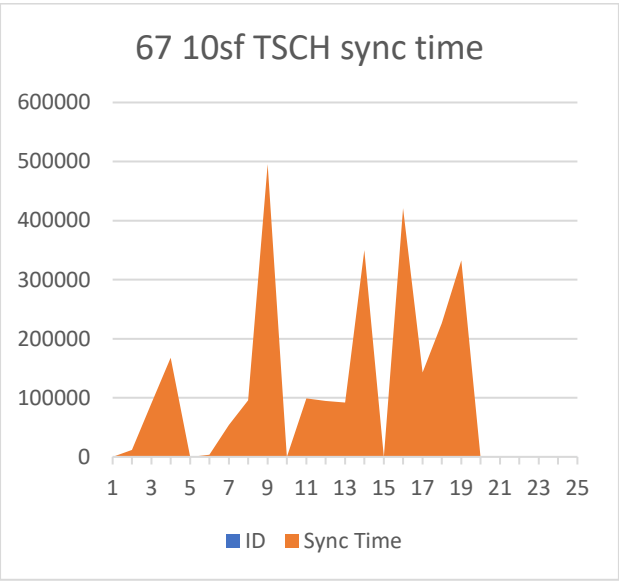

49 8sf 6TiSCH join time
160000
140000
120000
100000
80000
60000
40000
20000
0
id1
id3
id5
id7
id9
id11
id13
id15
id17
id19
id21
id23
id25

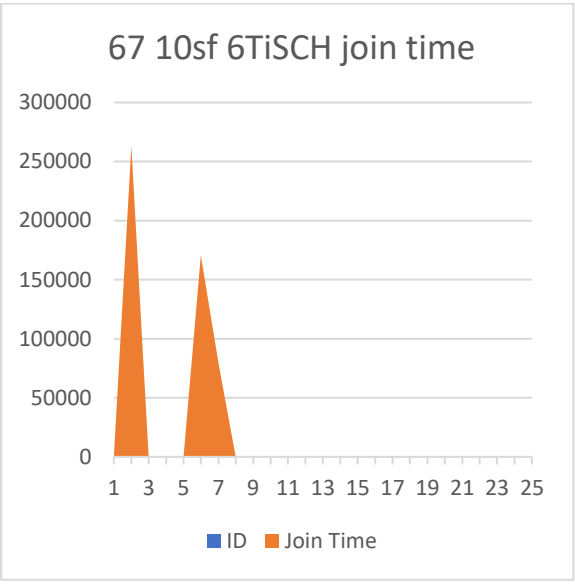

49 10sf TSCH sync time
70000
60000
50000
40000
30000
20000
10000
0
id1
id3
id5
id7
id9
id11
id13
id15
id17
id19
id21
id23
id25

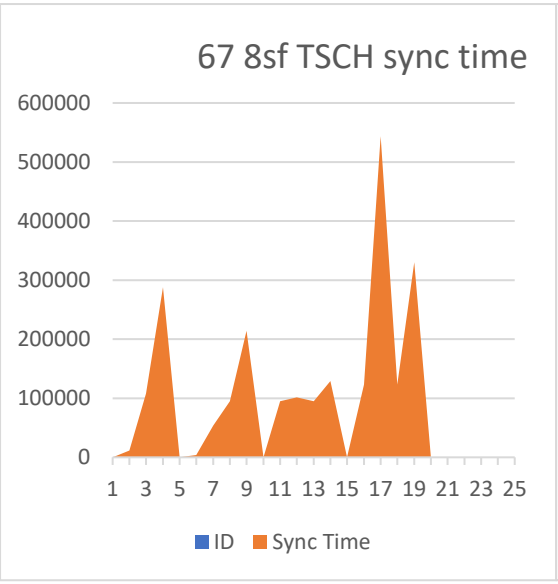

49 10sf 6TiSCH joining time
70000
60000
50000
40000
30000
20000
10000
0
id1
id3
id5
id7
id9
id11
id13
id15
id17
id19
id21
id23
id25

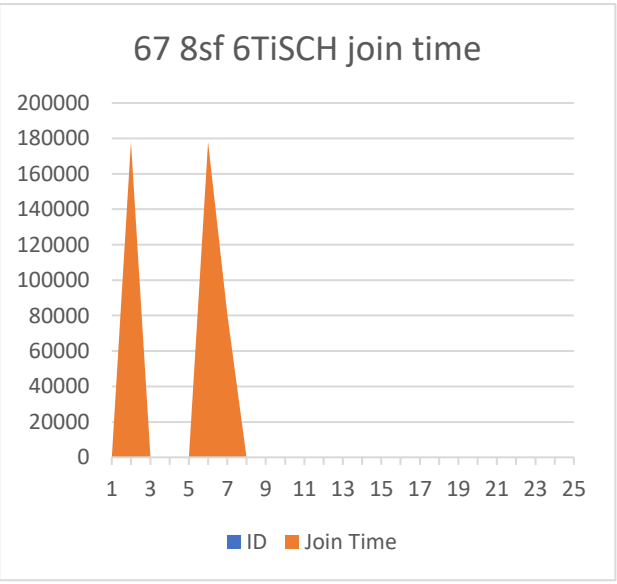

49 4sf TSCH JoinTime
50000
45000
40000
35000
30000
25000
20000
15000
10000
5000
0
id1 id3 id5 id7 id9 id11 id13 id15 id17 id19 id21 id23 id25

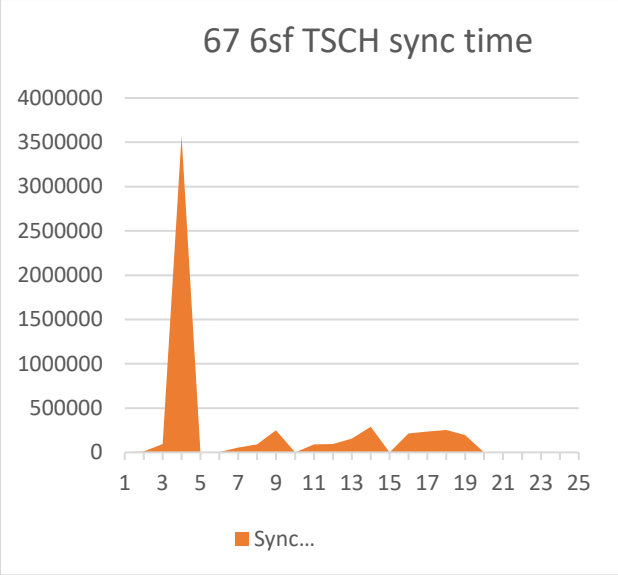

49 4sf 6 TiSCH sync time
60000
50000
40000
30000
20000
10000
0
id1 id3 id5 id7 id9 id11 id13 id15 id17 id19 id21 id23 id25

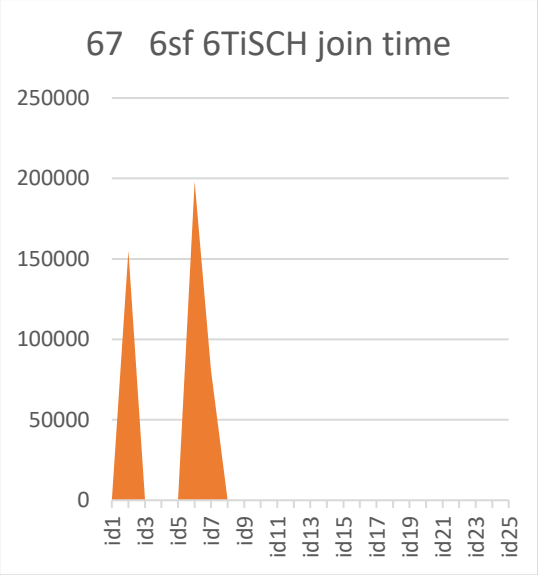

67 10sf TSCH sync time
600000
500000
400000
300000
200000
100000
0
1 3 5 7 9 11 13 15 17 19 21 23 25
ID
Sync Time

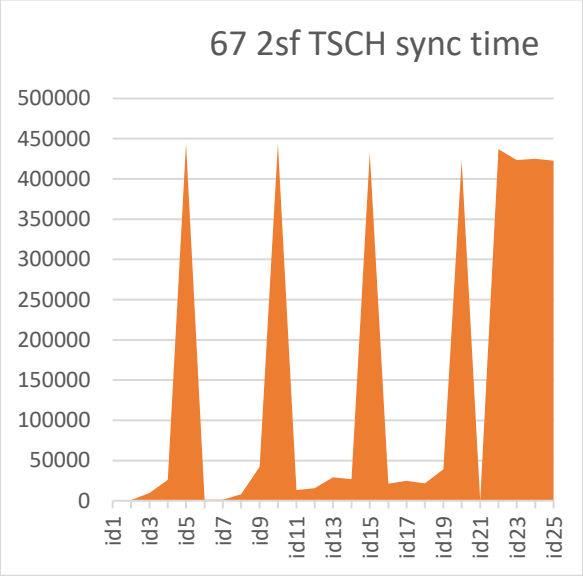

67 10sf 6TiSCH join time
300000
250000
200000
150000
100000
50000
0
1 3 5 7 9 11 13 15 17 19 21 23 25
ID
Join Time

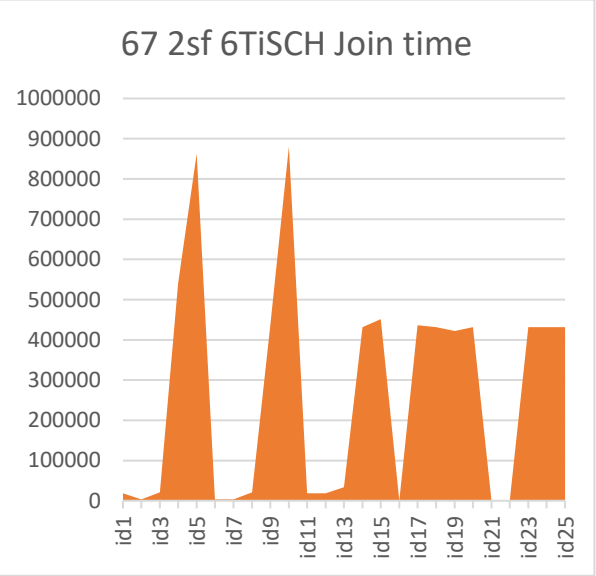

67 8sf TSCH sync time
600000
500000
400000
300000
200000
100000
0
1 3 5 7 9 11 13 15 17 19 21 23 25
ID
Sync Time

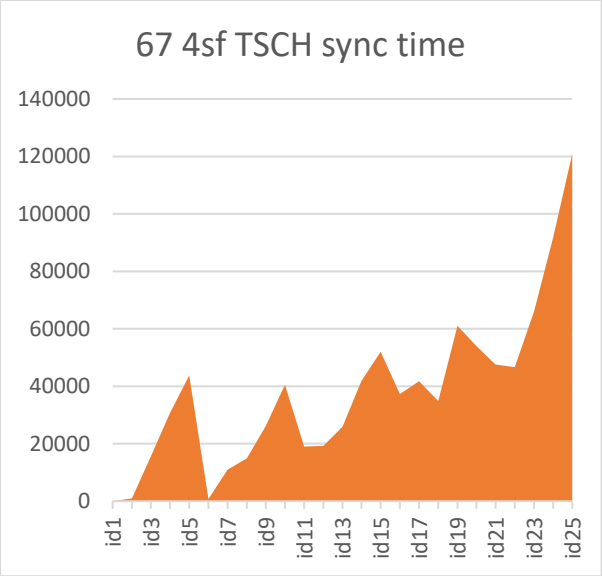

67 8sf 6TiSCH join time
200000
180000
160000
140000
120000
100000
80000
60000
40000
20000
0
1 3 5 7 9 11 13 15 17 19 21 23 25
ID
Join Time

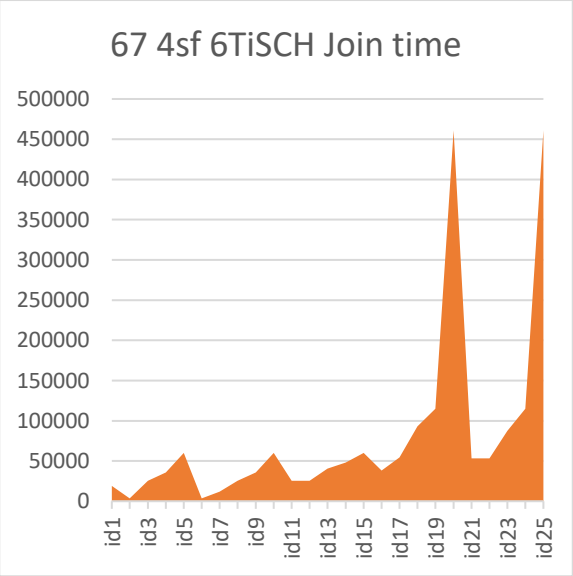

67 6sf TSCH sync time
4000000
3500000
3000000
2500000
2000000
1500000
1000000
500000
0
1 3 5 7 9 11 13 15 17 19 21 23 25
Sync...


67 6sf 6TiSCH join time
250000
200000
150000
100000
50000
0
id1 id3 id5 id7 id9 id11 id13 id15 id17 id19 id21 id23 id25


67 2sf TSCH sync time
500000
450000
400000
350000
300000
250000
200000
150000
100000
50000
0
id1 id3 id5 id7 id9 id11 id13 id15 id17 id19 id21 id23 id25


67 2sf 6TiSCH Join time
1000000
900000
800000
700000
600000
500000
400000
300000
200000
100000
0
id1 id3 id5 id7 id9 id11 id13 id15 id17 id19 id21 id23 id25


67 4sf TSCH sync time
140000
120000
100000
80000
60000
40000
20000
0
id1 id3 id5 id7 id9 id11 id13 id15 id17 id19 id21 id23 id25


67 4sf 6TiSCH Join time
500000
450000
400000
350000
300000
250000
200000
150000
100000
50000
0
id1 id3 id5 id7 id9 id11 id13 id15 id17 id19 id21 id23 id25

Conclusion

As the slot frame time increases the TSCH and 6TiSCH join time increases and there might be some cases in which the motes doesn’t get synchronized .

If motes are far from source then probability of synchronization is less as compared to motes located closer to source.