Department of Electrical Engineering

University of Cape Town

EEE4121F MODULE-A

LAB 2: Investigation of Mobility Models and Device Association Control Using Mininet-WiFi

7 March 2025

Due Date: 14 March 2025

Contents

1. Overview
2. Objectives
3. Investigation of mobility models in wireless networks
4. Investigation of device association control in homogeneous wireless networks
5. Report
6. References

1. Overview

This lab focuses on mobility models and network selection (or association) mechanisms in wireless networks.

2. Objectives

By the end of this lab, students should be able to:
1. Understand different mobility models and configure network mobility models in Mininet-WiFi.
2. Investigate device association control mechanisms in a homogeneous wireless network.

3. Investigation of mobility models in wireless networks

In mobile networks, mobility models characterize the movements of users with respect to their location, velocity, and direction over a period of time. Mobility models try to mimic the mobility of persons, vehicles or any other objects that are mobile in the network. Mobility models play an important role in the design of mobile networks. Mininet-WiFi supports the following mobility models: (1) RandomWalk, (2) TruncatedLevyWalk, (3) RandomDirection, (4) RandomWayPoint, (5) GaussMarkov, (6) ReferencePoint, and (7) TimeVariantCommunity.

3.1 Configuration of mobility models in Mininet-WiFi

The mobility models can be configured with
net.setMobilityModel()

For example:
net.setMobilityModel(time=0, model='RandomDirection', max_x=100, max_y=100, seed=20)
where
time = time (in seconds) in which the mobility is started
model = mobility model
max_x = maximum x value
min_x = minimum x value
seed = defines a starting value for a pseudo random sequence.

In addition, you may want to consider some parameters that can be used when you add some nodes. To do this, two mobility models namely RandomDirection and RandomWayPoint will be considered.

3.2 **RandomDirection** mobility model

When adding a node such as sta1, using RandomDirection, the following can be used:
sta1 = net.addStation(..., min_x=10, max_x=20, min_y=10, max_y=20, min_v=1, max_v=2)
where
min_x = Minimum X position # default=0
max_x = Maximum X position # default=100
min_y = Minimum Y position # default=0
max_y = Maximum Y position # default=100
min_v = Minimum value for node velocity
max_v = Maximum value for node velocity

3.3 **RandomWayPoint** mobility model

When adding a node such as sta1, using RandomWayPoint, the following can be used:
sta1 = net.addStation(..., min_x=10, max_x=20, min_y=10, max_y=20, min_v=1, max_v=2, min_wt=0, max_wt=100)
where
min_x = Minimum X position # default=0
max_x = Maximum X position # default=100
min_y = Minimum Y position # default=0
max_y = Maximum Y position # default=100
min_v = Minimum value for node velocity # default=5
max_v = Maximum value for node velocity # default=5
min_wt = Minimum wait time # default=0
max_wt = Maximum wait time # default=100

Lab Question 1 [20 marks]

Use the <**mobilityModel.py**> file in the Mininet-WiFi to investigate mobility models. The file contains the code for RandomDirection mobility model.
Please note that the password to be used is **wifi**

Question 1.1 (RandomDirection) [10 marks]

Run the following script and observe how the nodes behave when configured with the RandomDirection mobility model.
~/mininet-wifi$ sudo python examples/mobilityModel.py

After running the script for 5 minutes, do the following: (i) take a screenshot of the model (ii) measure the distance between sta1 and sta2, (iii) measure the TCP bandwidth between sta1 and sta2, and stop the script (**mininet-wifi> stop**).

Question 1.2 (RandomWayPoint) [10 marks]

In the <**mobilityModel.py**> file, replace model='RandomDirection' with model='RandomWayPoint'.
For sta1, replace net.addStation(..., min_x=10, max_x=30, min_y=50, max_y=70, min_v=5, max_v=10) with net.addStation(..., min_x=10, max_x=20, min_y=10, max_y=20, min_v=1, max_v=2, min_wt=0, max_wt=100)

For sta2, replace net.addStation(..., min_x=60, max_x=70, min_y=10, max_y=20, min_v=1, max_v=5) with net.addStation(..., min_x=60, max_x=70, min_y=50, max_y=70, min_v=5, max_v=10, min_wt=30, max_wt=70)

Rename the file as **mobilityModel2.py** and run the script. After running the script for 5 minutes, do the following: (i) take a screenshot of the model (ii) measure the distance between sta1 and sta2, (iii) measure the TCP bandwidth between sta1 and sta2, and stop the script (**mininet-wifi> stop**).

4. Investigation of device association control in a homogeneous wireless network

In wireless networks, network selection (or association) decisions can be made based on a single criterion or multiple criteria. This section focuses on two single-criterion strategies for making network association decisions in wireless networks. The two strategies are strongest signal first (SSF) and the least-loaded first (LLF). The SSF always selects the access point (or base station) with the strongest signal strength for a device while the LLF selects the least-loaded access point for a device.

To investigate the two association control strategies, use the **<associationControl.py>** file in Mininet-WiFi. Make the following modifications to the file:

(a) Change the x-axis position of node n from **(45 + n)** to **(10*n)**
(b) Change the position of ap1 from **'70, 50, 0'** to **'50, 50, 0'** and change the position of ap2 from **'90, 50, 0'** to **'110, 50, 0'**.
(c) Change the propagation model value from **(Model="logDistance" , exp=5)** to **(Model="logDistance" , exp=4)**

Save the modified file as **<associationControl2.py>**

Lab Question 2 [30 marks]

Question 2.1 (strongest signal first) [10 marks]

The default device association method in the file **<associationControl2.py>** is strongest signal first (ac_method='ssf')
Run the script file as follows:

~/mininet-wifi$ sudo python examples/associationControl2.py

Use the distance command to view the distance between sta1 and ap1.

mininet-wifi> distance sta1 ap1

Also view the distance between sta1 and ap2.
Then check the ap to which the sta1 is connected using the **iw** command below.

mininet-wifi> sta1 iw dev sta1-wlan0 link

Record the values obtained in the following table and repeat the process for all the stas. For example, check the ap to which sta2 is connected using the following:

mininet-wifi> sta2 iw dev sta2-wlan0 link

Node	Distance to ap1(meters)	Distance to ap2 (meters)	ap to which node is connected
sta1			
sta2			
sta3			
sta4			
sta5			
sta6			
sta7			
sta8			
sta9			
sta10			

Question 2.2 [5 marks]

Explain the results obtained in Question 2.1

Question 2.3 (least-loaded first) [10 marks]

Change the device association method in the file **<associationControl2.py>** to least-loaded first (ac_method='llf')

Run the script file as follows:

~/mininet-wifi$ sudo python examples/associationControl2.py

Check the ap to which the sta is connected using the **iw** command.
Record the values obtained in the following table.

Node	ap to which node is connected
Sta1	
Sta2	
Sta3	
Sta4	
Sta5	
Sta6	
Sta7	
Sta8	
Sta9	
Sta10	

Question 2.4 [5 marks]

Explain the results obtained in Question 2.3

5. Report

Write and submit a report containing your answers to the lab questions. The report should be submitted on Amathuba.

6. References

(1) https://github.com/intrig-unicamp/mininet-wifi
(2) https://usermanual.wiki/Pdf/mininetwifidraftmanual.297704656.pdf
(3) R. Fontes and C. Rothenberg, "Wireless Network Emulation with Mininet-WiFi," https://intrig.dca.fee.unicamp.br/mininetwifi/

This lab has been adapted from References 1-3.