COEN 241 HW 3: Mininet & OpenFlow

Xiongsheng Yi

Task 1: Defining custom topologies

```
$ sudo mn --custom binary_tree.py --topo binary_tree
mininet> h1 ping h8
```

```
root@96e30fa9692e:~# mn --custom binary_tree.py --topo binary_tree
*** Error setting resource limits. Mininet's performance may be affected.
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4 h5 h6 h7 h8
*** Adding switches:
s1 s2 s3 s4 s5 s6 s7
*** Adding links:
(s1, s2) (s1, s5) (s2, s3) (s2, s4) (s3, h1) (s3, h2) (s4, h3) (s4, h4) (s5, s6) (s5, s7) (s6, h5) (s6, h
6) (s7, h7) (s7, h8)
*** Configuring hosts
h1 h2 h3 h4 h5 h6 h7 h8
*** Starting controller
c0
*** Starting 7 switches
s1 s2 s3 s4 s5 s6 s7 ...
*** Starting CLI:
mininet> hi ping h8
*** Unknown command: hi ping h8
mininet> h1 ping h8
PING 10.0.0.8 (10.0.0.8) 56(84) bytes of data.
64 bytes from 10.0.0.8: icmp_seq=1 ttl=64 time=9.31 ms
64 bytes from 10.0.0.8: icmp_seq=2 ttl=64 time=0.585 ms
64 bytes from 10.0.0.8: icmp_seq=3 ttl=64 time=0.061 ms
64 bytes from 10.0.0.8: icmp_seq=4 ttl=64 time=0.053 ms
64 bytes from 10.0.0.8: icmp_seq=5 ttl=64 time=0.062 ms
64 bytes from 10.0.0.8: icmp_seq=6 ttl=64 time=0.055 ms
64 bytes from 10.0.0.8: icmp_seq=7 ttl=64 time=0.052 ms
64 bytes from 10.0.0.8: icmp_seq=8 ttl=64 time=0.062 ms
64 bytes from 10.0.0.8: icmp_seq=9 ttl=64 time=0.058 ms
64 bytes from 10.0.0.8: icmp_seq=10 ttl=64 time=0.059 ms
64 bytes from 10.0.0.8: icmp_seq=11 ttl=64 time=0.068 ms
64 bytes from 10.0.0.8: icmp_seq=12 ttl=64 time=0.063 ms
64 bytes from 10.0.0.8: icmp_seq=13 ttl=64 time=0.053 ms
64 bytes from 10.0.0.8: icmp_seq=14 ttl=64 time=0.052 ms
64 bytes from 10.0.0.8: icmp_seq=15 ttl=64 time=0.060 ms
64 bytes from 10.0.0.8: icmp_seq=16 ttl=64 time=0.084 ms
64 bytes from 10.0.0.8: icmp_seq=17 ttl=64 time=0.065 ms
64 bytes from 10.0.0.8: icmp_seq=18 ttl=64 time=0.062 ms
64 bytes from 10.0.0.8: icmp_seq=19 ttl=64 time=0.066 ms
64 bytes from 10.0.0.8: icmp_seq=20 ttl=64 time=0.065 ms
64 bytes from 10.0.0.8: icmp_seq=21 ttl=64 time=0.061 ms
64 bytes from 10.0.0.8: icmp_seq=22 ttl=64 time=0.061 ms
64 bytes from 10.0.0.8: icmp_seq=23 ttl=64 time=0.049 ms
64 bytes from 10.0.0.8: icmp_seq=24 ttl=64 time=0.062 ms
64 bytes from 10.0.0.8: icmp_seq=25 ttl=64 time=0.061 ms
64 bytes from 10.0.0.8: icmp_seq=26 ttl=64 time=0.067 ms
64 bytes from 10.0.0.8: icmp_seq=27 ttl=64 time=0.061 ms
64 bytes from 10.0.0.8: icmp_seq=28 ttl=64 time=0.060 ms
^C
--- 10.0.0.8 ping statistics ---
28 packets transmitted, 28 received, 0% packet loss, time 27605ms
rtt min/avg/max/mdev = 0.049/0.410/9.314/1.716 ms
mininet>
```

Questions

1. What is the output of “nodes” and “net”

```
mininet> nodes
available nodes are:
c0 h1 h2 h3 h4 h5 h6 h7 h8 s1 s2 s3 s4 s5 s6 s7
mininet> net
h1 h1-eth0:s3-eth2
h2 h2-eth0:s3-eth3
h3 h3-eth0:s4-eth2
h4 h4-eth0:s4-eth3
h5 h5-eth0:s6-eth2
h6 h6-eth0:s6-eth3
h7 h7-eth0:s7-eth2
h8 h8-eth0:s7-eth3
s1 lo:  s1-eth1:s2-eth1 s1-eth2:s5-eth1
s2 lo:  s2-eth1:s1-eth1 s2-eth2:s3-eth1 s2-eth3:s4-eth1
s3 lo:  s3-eth1:s2-eth2 s3-eth2:h1-eth0 s3-eth3:h2-eth0
s4 lo:  s4-eth1:s2-eth3 s4-eth2:h3-eth0 s4-eth3:h4-eth0
s5 lo:  s5-eth1:s1-eth2 s5-eth2:s6-eth1 s5-eth3:s7-eth1
s6 lo:  s6-eth1:s5-eth2 s6-eth2:h5-eth0 s6-eth3:h6-eth0
s7 lo:  s7-eth1:s5-eth3 s7-eth2:h7-eth0 s7-eth3:h8-eth0
c0
mininet>
```

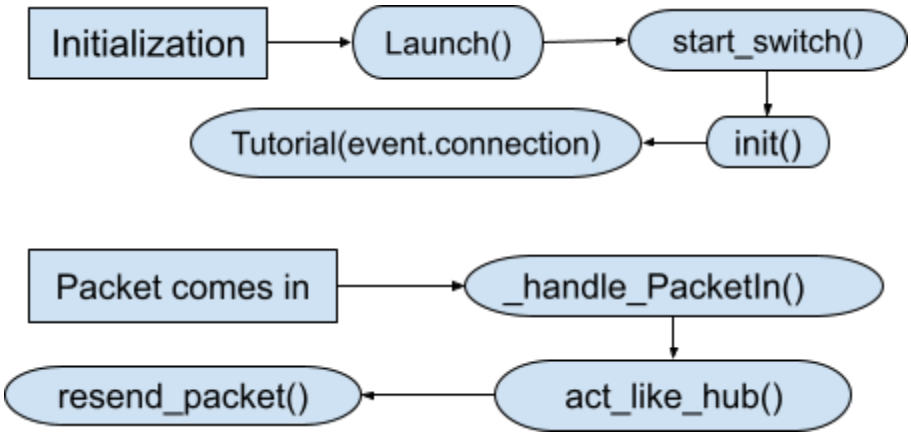
2. What is the output of “h7 ifconfig”

```
mininet> h7 ifconfig
h7-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.0.7  netmask 255.0.0.0  broadcast 10.255.255.255
        inet6 fe80::9829:8cff:feab:59f  prefixlen 64  scopeid 0x20<link>
        ether 9a:29:8c:ab:05:9f  txqueuelen 1000  (Ethernet)
        RX packets 78  bytes 5932 (5.9 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 13  bytes 1006 (1.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

mininet>
```

Task 2: Analyze the “of_tutorial’ controller

Questions

1. Draw the function call graph of this controller. For example, once a packet comes to the controller, which function is the first to be called, which one is the second, and so forth?



2. Have h1 ping h2, and h1 ping h8 for 100 times (e.g., h1 ping -c100 p2).

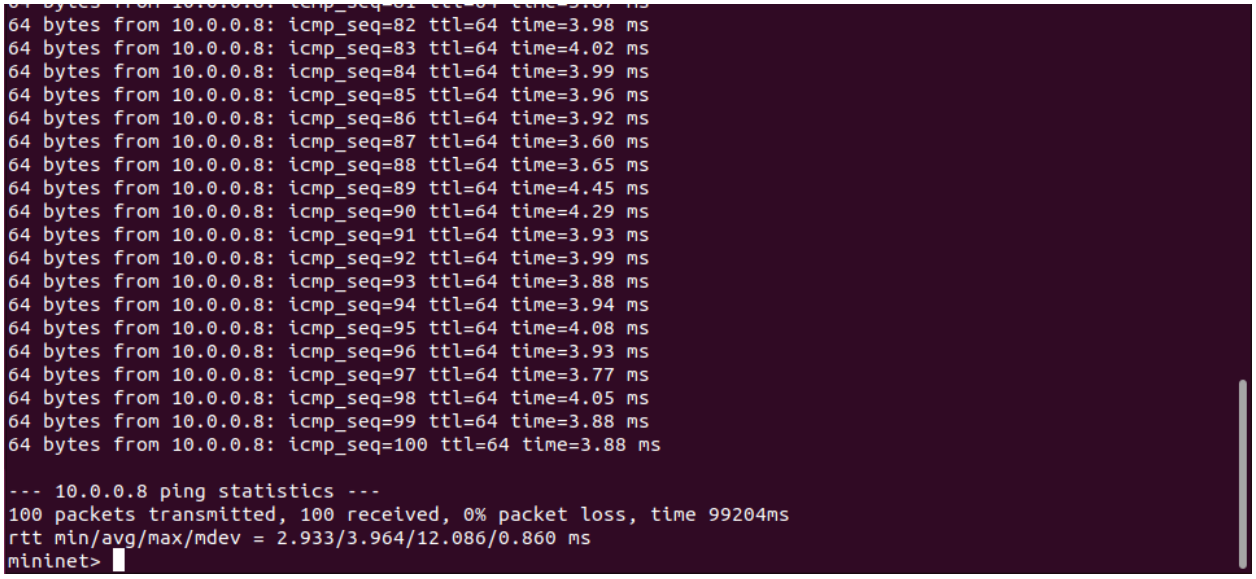
$ h1 ping -c100 p2

```
mininet> h1 ping -c100 h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=2.05 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=1.12 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=1.20 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=1.04 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=1.91 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=1.11 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=1.18 ms
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=1.09 ms
64 bytes from 10.0.0.2: icmp_seq=9 ttl=64 time=1.00 ms
64 bytes from 10.0.0.2: icmp_seq=10 ttl=64 time=0.995 ms
64 bytes from 10.0.0.2: icmp_seq=11 ttl=64 time=0.984 ms
64 bytes from 10.0.0.2: icmp_seq=12 ttl=64 time=1.18 ms
64 bytes from 10.0.0.2: icmp_seq=13 ttl=64 time=0.991 ms
64 bytes from 10.0.0.2: icmp_seq=14 ttl=64 time=1.09 ms
64 bytes from 10.0.0.2: icmp_seq=15 ttl=64 time=1.01 ms
64 bytes from 10.0.0.2: icmp_seq=16 ttl=64 time=1.16 ms
64 bytes from 10.0.0.2: icmp_seq=17 ttl=64 time=1.13 ms
64 bytes from 10.0.0.2: icmp_seq=18 ttl=64 time=0.960 ms
64 bytes from 10.0.0.2: icmp_seq=19 ttl=64 time=1.49 ms
64 bytes from 10.0.0.2: icmp_seq=20 ttl=64 time=0.990 ms
64 bytes from 10.0.0.2: icmp_seq=21 ttl=64 time=1.19 ms
64 bytes from 10.0.0.2: icmp_seq=22 ttl=64 time=1.38 ms
64 bytes from 10.0.0.2: icmp_seq=23 ttl=64 time=1.13 ms
64 bytes from 10.0.0.2: icmp_seq=24 ttl=64 time=0.953 ms
64 bytes from 10.0.0.2: icmp_seq=25 ttl=64 time=0.993 ms
64 bytes from 10.0.0.2: icmp_seq=26 ttl=64 time=1.05 ms
64 bytes from 10.0.0.2: icmp_seq=27 ttl=64 time=1.12 ms
64 bytes from 10.0.0.2: icmp_seq=28 ttl=64 time=1.19 ms
64 bytes from 10.0.0.2: icmp_seq=29 ttl=64 time=1.14 ms
64 bytes from 10.0.0.2: icmp_seq=30 ttl=64 time=1.27 ms
```

```
64 bytes from 10.0.0.2: icmp_seq=80 ttl=64 time=1.12 ms
64 bytes from 10.0.0.2: icmp_seq=81 ttl=64 time=1.08 ms
64 bytes from 10.0.0.2: icmp_seq=82 ttl=64 time=1.18 ms
64 bytes from 10.0.0.2: icmp_seq=83 ttl=64 time=1.06 ms
64 bytes from 10.0.0.2: icmp_seq=84 ttl=64 time=0.856 ms
64 bytes from 10.0.0.2: icmp_seq=85 ttl=64 time=1.11 ms
64 bytes from 10.0.0.2: icmp_seq=86 ttl=64 time=1.02 ms
64 bytes from 10.0.0.2: icmp_seq=87 ttl=64 time=1.18 ms
64 bytes from 10.0.0.2: icmp_seq=88 ttl=64 time=1.18 ms
64 bytes from 10.0.0.2: icmp_seq=89 ttl=64 time=1.11 ms
64 bytes from 10.0.0.2: icmp_seq=90 ttl=64 time=1.06 ms
64 bytes from 10.0.0.2: icmp_seq=91 ttl=64 time=0.866 ms
64 bytes from 10.0.0.2: icmp_seq=92 ttl=64 time=1.04 ms
64 bytes from 10.0.0.2: icmp_seq=93 ttl=64 time=0.968 ms
64 bytes from 10.0.0.2: icmp_seq=94 ttl=64 time=1.10 ms
64 bytes from 10.0.0.2: icmp_seq=95 ttl=64 time=1.32 ms
64 bytes from 10.0.0.2: icmp_seq=96 ttl=64 time=1.14 ms
64 bytes from 10.0.0.2: icmp_seq=97 ttl=64 time=1.17 ms
64 bytes from 10.0.0.2: icmp_seq=98 ttl=64 time=0.987 ms
64 bytes from 10.0.0.2: icmp_seq=99 ttl=64 time=1.09 ms
64 bytes from 10.0.0.2: icmp_seq=100 ttl=64 time=1.06 ms

--- 10.0.0.2 ping statistics ---
100 packets transmitted, 100 received, 0% packet loss, time 99235ms
rtt min/avg/max/mdev = 0.856/1.128/2.053/0.170 ms
mininet>
```

$ h1 ping -c100 p8

```
mininet> h1 ping -c100 h8
PING 10.0.0.8 (10.0.0.8) 56(84) bytes of data.
64 bytes from 10.0.0.8: icmp_seq=1 ttl=64 time=12.0 ms
64 bytes from 10.0.0.8: icmp_seq=2 ttl=64 time=3.72 ms
64 bytes from 10.0.0.8: icmp_seq=3 ttl=64 time=3.89 ms
64 bytes from 10.0.0.8: icmp_seq=4 ttl=64 time=3.55 ms
64 bytes from 10.0.0.8: icmp_seq=5 ttl=64 time=3.63 ms
64 bytes from 10.0.0.8: icmp_seq=6 ttl=64 time=4.63 ms
64 bytes from 10.0.0.8: icmp_seq=7 ttl=64 time=3.02 ms
64 bytes from 10.0.0.8: icmp_seq=8 ttl=64 time=3.42 ms
64 bytes from 10.0.0.8: icmp_seq=9 ttl=64 time=3.83 ms
64 bytes from 10.0.0.8: icmp_seq=10 ttl=64 time=3.96 ms
64 bytes from 10.0.0.8: icmp_seq=11 ttl=64 time=3.80 ms
64 bytes from 10.0.0.8: icmp_seq=12 ttl=64 time=3.80 ms
64 bytes from 10.0.0.8: icmp_seq=13 ttl=64 time=4.26 ms
64 bytes from 10.0.0.8: icmp_seq=14 ttl=64 time=3.72 ms
64 bytes from 10.0.0.8: icmp_seq=15 ttl=64 time=3.56 ms
64 bytes from 10.0.0.8: icmp_seq=16 ttl=64 time=3.63 ms
64 bytes from 10.0.0.8: icmp_seq=17 ttl=64 time=3.52 ms
64 bytes from 10.0.0.8: icmp_seq=18 ttl=64 time=3.61 ms
64 bytes from 10.0.0.8: icmp_seq=19 ttl=64 time=3.97 ms
64 bytes from 10.0.0.8: icmp_seq=20 ttl=64 time=3.89 ms
64 bytes from 10.0.0.8: icmp_seq=21 ttl=64 time=3.78 ms
64 bytes from 10.0.0.8: icmp_seq=22 ttl=64 time=4.04 ms
64 bytes from 10.0.0.8: icmp_seq=23 ttl=64 time=3.61 ms
64 bytes from 10.0.0.8: icmp_seq=24 ttl=64 time=4.48 ms
64 bytes from 10.0.0.8: icmp_seq=25 ttl=64 time=4.03 ms
```



a. How long does it take (on average) to ping for each case?

For h1 ping h2:

ave: 1.128ms

For h1 ping h8:

ave: 3.964ms

b. What is the minimum and maximum ping you have observed?

For h1 ping h2:

min: 0.856ms

max: 2.053ms

For h1 ping h8:

min: 2.933ms

max: 12.086ms

c. What is the difference, and why?

The difference is that h1ping h2 takes less time than h1 ping h8.

Because for h1 ping h2, packets only need to go through switch s3, but for h1 ping h8, packets need to go through switches s3 -> s2 -> s1 -> s5 -> s7.

3. Run “iperf h1 h2” and “iperf h1 h8”

```
mininet> iperf h1 h2
*** Iperf: testing TCP bandwidth between h1 and h2
*** Results: ['22.8 Mbits/sec', '25.4 Mbits/sec']
mininet>
```

```
mininet> iperf h1 h8
*** Iperf: testing TCP bandwidth between h1 and h8
*** Results: ['5.10 Mbits/sec', '5.64 Mbits/sec']
mininet>
```

a. What is “iperf” used for?

It is used for testing the maximum network bandwidth by measuring how much data can be transferred between two nodes with a given amount of time.

b. What is the throughput for each case?

For h1 - h2: 22.8 Mbits/sec
For h2 - h1: 25.4 Mbits/sec

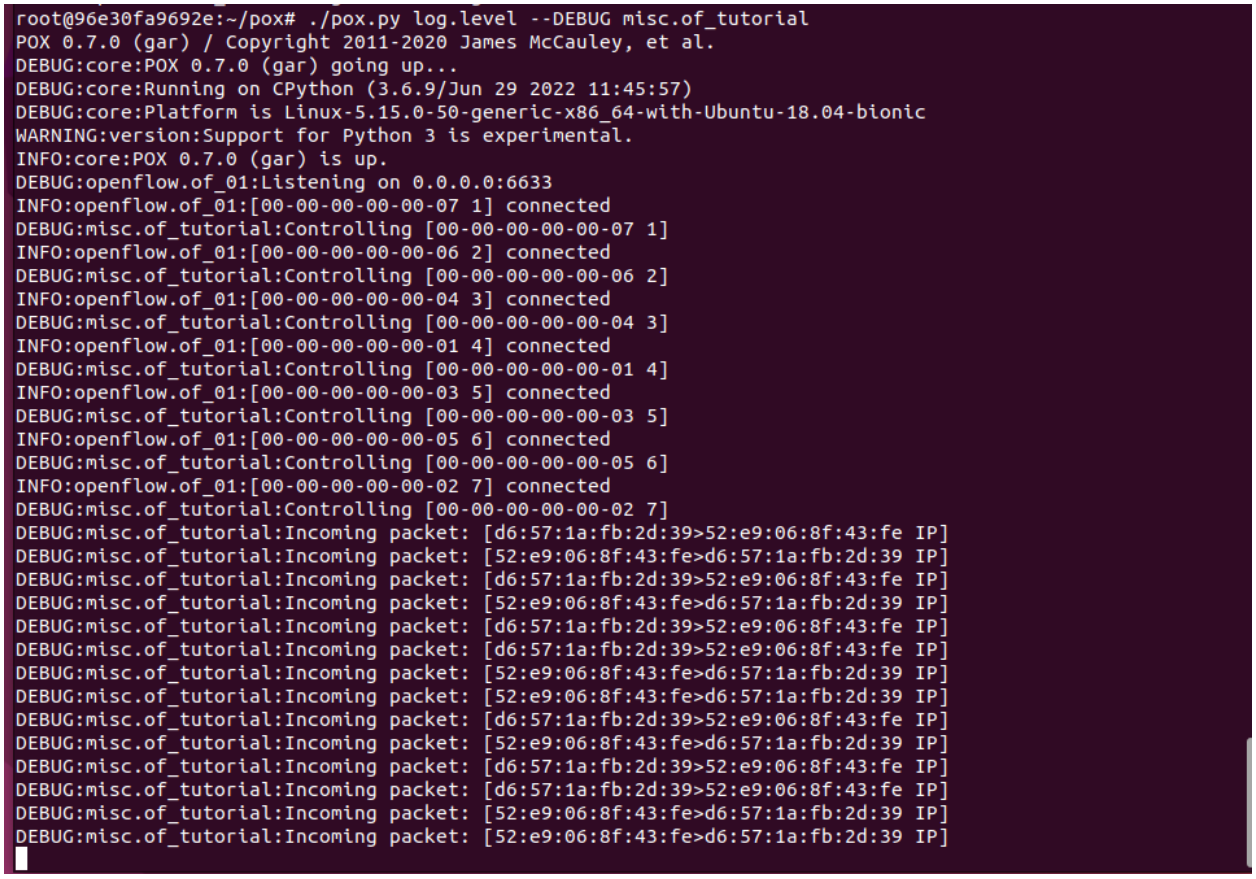
For h1 - h8: 5.10 Mbits/sec
For h8 - h1: 5.64 Mbits/sec

c. What is the difference, and explain the reasons for the difference.

The difference is that the throughput of h1 - h2 is higher than h1 - h8.
Because h1 - h2 only needs to go through switch s3, which takes less time to transfer for one packet and more packets can be transferred within a given amount of time.
But h1 - h8 needs to go through switches s3 -> s2 -> s1 -> s5 -> s7, which takes more time to transfer for one packet and less packets can be transferred within a given amount of time.

4. Which of the switches observe traffic? Please describe your way for observing such traffic on switches (e.g., adding some functions in the “of_tutorial” controller).

All the switches are observing traffic, because when a packet is sent, it is sent to all the switches and not just the ones between the two communicating hosts. If I want to observe this traffic, I can add a print statement in the _handle_PacketIn() function to show when the traffic goes through the switches.



Task 3: MAC Learning Controller

Questions

1. Describe how the above code works, such as how the "MAC to Port" map is established. You could use a ‘ping’ example to describe the establishment process (e.g., h1 ping h2).

In of_tutorial.py, there is a mac_to_port map to store mac to port key-value pairs, and the map is empty when this class is instantiated. When a packet comes in, the program will check if the source port and source mac are stored in mac_to_port map, if not, it will “learn”, which means to add this source mac-port key-value pair into the mac_to_port map. Then the program will check if destination mac has a related port stored in mac_to_port map, if so, send to this port, if not, send packet to everybody.

```
mininet> h1 ping -c1 h8
PING 10.0.0.8 (10.0.0.8) 56(84) bytes of data.
64 bytes from 10.0.0.8: icmp_seq=1 ttl=64 time=5.61 ms

--- 10.0.0.8 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 5.616/5.616/5.616/0.000 ms
mininet>
```

```
Learning that ca:b9:5c:ef:48:57 is attached at port 3
d6:57:1a:fb:2d:39 destination known. only send message to it
Src:  ca:b9:5c:ef:48:57 : 3 Dst: d6:57:1a:fb:2d:39
Learning that ca:b9:5c:ef:48:57 is attached at port 3
d6:57:1a:fb:2d:39 destination known. only send message to it
Src:  ca:b9:5c:ef:48:57 : 2 Dst: d6:57:1a:fb:2d:39
Learning that ca:b9:5c:ef:48:57 is attached at port 2
d6:57:1a:fb:2d:39 destination known. only send message to it
Src:  ca:b9:5c:ef:48:57 : 1 Dst: d6:57:1a:fb:2d:39
Learning that ca:b9:5c:ef:48:57 is attached at port 1
d6:57:1a:fb:2d:39 destination known. only send message to it
Src:  ca:b9:5c:ef:48:57 : 1 Dst: d6:57:1a:fb:2d:39
Learning that ca:b9:5c:ef:48:57 is attached at port 1
d6:57:1a:fb:2d:39 destination known. only send message to it
```

2. (Comment out all prints before doing this experiment) Have h1 ping h2, and h1 ping h8 for 100 times (e.g., h1 ping -c100 p2).

```
mininet> h1 ping -c100 h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=2.20 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=1.71 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=1.61 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=1.45 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=1.64 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=1.33 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=1.33 ms
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=1.18 ms
64 bytes from 10.0.0.2: icmp_seq=9 ttl=64 time=1.38 ms
64 bytes from 10.0.0.2: icmp_seq=10 ttl=64 time=1.33 ms
64 bytes from 10.0.0.2: icmp_seq=11 ttl=64 time=1.22 ms
64 bytes from 10.0.0.2: icmp_seq=12 ttl=64 time=1.47 ms
64 bytes from 10.0.0.2: icmp_seq=13 ttl=64 time=1.28 ms
64 bytes from 10.0.0.2: icmp_seq=14 ttl=64 time=1.37 ms
64 bytes from 10.0.0.2: icmp_seq=15 ttl=64 time=1.45 ms
64 bytes from 10.0.0.2: icmp_seq=16 ttl=64 time=1.28 ms
64 bytes from 10.0.0.2: icmp_seq=17 ttl=64 time=1.29 ms
64 bytes from 10.0.0.2: icmp_seq=18 ttl=64 time=1.25 ms
64 bytes from 10.0.0.2: icmp_seq=19 ttl=64 time=1.53 ms
64 bytes from 10.0.0.2: icmp_seq=20 ttl=64 time=1.44 ms
64 bytes from 10.0.0.2: icmp_seq=21 ttl=64 time=1.23 ms
```

```
64 bytes from 10.0.0.2: icmp_seq=85 ttl=64 time=1.20 ms
64 bytes from 10.0.0.2: icmp_seq=86 ttl=64 time=1.48 ms
64 bytes from 10.0.0.2: icmp_seq=87 ttl=64 time=1.32 ms
64 bytes from 10.0.0.2: icmp_seq=88 ttl=64 time=1.43 ms
64 bytes from 10.0.0.2: icmp_seq=89 ttl=64 time=1.43 ms
64 bytes from 10.0.0.2: icmp_seq=90 ttl=64 time=1.29 ms
64 bytes from 10.0.0.2: icmp_seq=91 ttl=64 time=1.38 ms
64 bytes from 10.0.0.2: icmp_seq=92 ttl=64 time=1.58 ms
64 bytes from 10.0.0.2: icmp_seq=93 ttl=64 time=1.39 ms
64 bytes from 10.0.0.2: icmp_seq=94 ttl=64 time=1.52 ms
64 bytes from 10.0.0.2: icmp_seq=95 ttl=64 time=1.41 ms
64 bytes from 10.0.0.2: icmp_seq=96 ttl=64 time=1.30 ms
64 bytes from 10.0.0.2: icmp_seq=97 ttl=64 time=1.11 ms
64 bytes from 10.0.0.2: icmp_seq=98 ttl=64 time=1.46 ms
64 bytes from 10.0.0.2: icmp_seq=99 ttl=64 time=1.37 ms
64 bytes from 10.0.0.2: icmp_seq=100 ttl=64 time=1.62 ms

--- 10.0.0.2 ping statistics ---
100 packets transmitted, 100 received, 0% packet loss, time 99201ms
rtt min/avg/max/mdev = 1.116/1.390/2.207/0.157 ms
```

```
mininet> h1 ping -c100 h8
PING 10.0.0.8 (10.0.0.8) 56(84) bytes of data.
64 bytes from 10.0.0.8: icmp_seq=1 ttl=64 time=6.14 ms
64 bytes from 10.0.0.8: icmp_seq=2 ttl=64 time=5.71 ms
64 bytes from 10.0.0.8: icmp_seq=3 ttl=64 time=4.58 ms
64 bytes from 10.0.0.8: icmp_seq=4 ttl=64 time=5.31 ms
64 bytes from 10.0.0.8: icmp_seq=5 ttl=64 time=4.64 ms
64 bytes from 10.0.0.8: icmp_seq=6 ttl=64 time=4.21 ms
64 bytes from 10.0.0.8: icmp_seq=7 ttl=64 time=4.96 ms
64 bytes from 10.0.0.8: icmp_seq=8 ttl=64 time=4.81 ms
64 bytes from 10.0.0.8: icmp_seq=9 ttl=64 time=4.65 ms
64 bytes from 10.0.0.8: icmp_seq=10 ttl=64 time=5.65 ms
64 bytes from 10.0.0.8: icmp_seq=11 ttl=64 time=4.15 ms
64 bytes from 10.0.0.8: icmp_seq=12 ttl=64 time=4.96 ms
64 bytes from 10.0.0.8: icmp_seq=13 ttl=64 time=5.22 ms
64 bytes from 10.0.0.8: icmp_seq=14 ttl=64 time=4.70 ms
64 bytes from 10.0.0.8: icmp_seq=15 ttl=64 time=4.14 ms
64 bytes from 10.0.0.8: icmp_seq=16 ttl=64 time=6.34 ms
64 bytes from 10.0.0.8: icmp_seq=17 ttl=64 time=6.45 ms
64 bytes from 10.0.0.8: icmp_seq=18 ttl=64 time=6.12 ms
64 bytes from 10.0.0.8: icmp_seq=19 ttl=64 time=6.04 ms
64 bytes from 10.0.0.8: icmp_seq=20 ttl=64 time=5.28 ms
64 bytes from 10.0.0.8: icmp_seq=21 ttl=64 time=5.39 ms
```

```
64 bytes from 10.0.0.8: icmp_seq=85 ttl=64 time=5.01 ms
64 bytes from 10.0.0.8: icmp_seq=86 ttl=64 time=5.06 ms
64 bytes from 10.0.0.8: icmp_seq=87 ttl=64 time=6.57 ms
64 bytes from 10.0.0.8: icmp_seq=88 ttl=64 time=5.07 ms
64 bytes from 10.0.0.8: icmp_seq=89 ttl=64 time=4.87 ms
64 bytes from 10.0.0.8: icmp_seq=90 ttl=64 time=4.75 ms
64 bytes from 10.0.0.8: icmp_seq=91 ttl=64 time=4.30 ms
64 bytes from 10.0.0.8: icmp_seq=92 ttl=64 time=5.01 ms
64 bytes from 10.0.0.8: icmp_seq=93 ttl=64 time=6.54 ms
64 bytes from 10.0.0.8: icmp_seq=94 ttl=64 time=5.58 ms
64 bytes from 10.0.0.8: icmp_seq=95 ttl=64 time=6.95 ms
64 bytes from 10.0.0.8: icmp_seq=96 ttl=64 time=3.72 ms
64 bytes from 10.0.0.8: icmp_seq=97 ttl=64 time=4.11 ms
64 bytes from 10.0.0.8: icmp_seq=98 ttl=64 time=3.98 ms
64 bytes from 10.0.0.8: icmp_seq=99 ttl=64 time=5.27 ms
64 bytes from 10.0.0.8: icmp_seq=100 ttl=64 time=4.43 ms

--- 10.0.0.8 ping statistics ---
100 packets transmitted, 100 received, 0% packet loss, time 99193ms
rtt min/avg/max/mdev = 3.728/4.988/7.172/0.679 ms
```

a. How long did it take (on average) to ping for each case?

For h1 ping h2:

ave: 1.390ms

For h1 ping h8:

ave: 4.988ms

b. What is the minimum and maximum ping you have observed?

For h1 ping h2:

min: 1.116ms

max: 2.207ms

For h1 ping h8:

min: 3.728ms

max: 7.172ms

c. Any difference from Task 2 and why do you think there is a change if there is?

Compared to Task2, the average ping time for h1-h2 and h1-h8 is slightly longer. Because it needs some time to check the mac_to_port map.

3. Q.3 Run “iperf h1 h2” and “iperf h1 h8”.

a. What is the throughput for each case?

```
mininet> iperf h1 h2
*** Iperf: testing TCP bandwidth between h1 and h2
*** Results: ['28.8 Mbits/sec', '31.3 Mbits/sec']
mininet> iperf h1 h8
*** Iperf: testing TCP bandwidth between h1 and h8
*** Results: ['3.40 Mbits/sec', '3.90 Mbits/sec']
mininet>
```

For h1 - h2: 28.8 Mbits/sec

For h2 - h1: 31.3 Mbits/sec

For h1 - h8: 3.40 Mbits/sec

For h8 - h1: 3.90 Mbits/sec

b. What is the difference from Task 2 and why do you think there is a change if there is?

The throughput for h1-h2 is larger as the destination port is known by the sender, so the data transfer is faster and more data can be sent within a given amount of time. But for h1-h8, there is now much change,

it could be because the network path still consists of five switches and the map might not have a big effect in the “iperf” measurement.