Computer Network

Lab assignment 1 Q2

Name-Amritansh Mishra
Roll no-IIT2018142

Q1) Libraries to be imported

```
from subprocess import check_output, STDOUT

from mininet.cli import CLI
from mininet.node import Host, Node
from mininet.link import Intf, Link, TCIntf
from mininet.log import debug, error, setLogLevel
from mininet.net import Mininet
from mininet.nodelib import LinuxBridge
from mininet.topo import Topo
from time import sleep
from tqdm import tqdm
import numpy
```

- Start 10 long lived TCP flows sending data from h2 to h1, and similarly 10 long lived TCP flows from h5 to h1.
- Start back-to-back ping trains from h2 to h1, and h5 to h1. Record the RTTs with ping 10 times a second.

For these two tasks I made a function which contained

```
        h2 = self.addHost('h2')
        h3 = self.addHost('h3')
        h4 = self.addHost('h4')
        h5 = self.addHost('h5')
```

```
       self.addLink(h1, s1, cls1=BasicIntf, cls2=PIEIntf, bw=100,
delay='10ms', max_queue_size=1000)
       self.addLink(h2, s1, cls1=BasicIntf, cls2=PIEIntf, bw=100,
delay='10ms', max_queue_size=1000)
       self.addLink(h3, s1, cls1=BasicIntf, cls2=PIEIntf, bw=100,
delay='10ms', max_queue_size=1000)

       self.addLink(h4, s2, cls1=BasicIntf, cls2=PIEIntf, bw=100,
delay='10ms', max_queue_size=1000)
       self.addLink(h5, s2, cls1=BasicIntf, cls2=PIEIntf, bw=100,
delay='10ms', max_queue_size=1000)
```

As you can see it has self.addLink(h2,s1,....),this is the function that actually adds a link causes the delay between them.

```
 print 'Start iperf connections h4->h3, h2->h1, and h5->h1'
   h43 = h4.popen("iperf -c %s -t 200 -P 10 -i 1| grep SUM > T43-b.out" %
(h3.IP()), shell = True)
   h21 = h2.popen("iperf -c %s -t 200 -P 10 -i 1| grep SUM > T21-b.out" %
(h1.IP()), shell = True)
   h51 = h5.popen("iperf -c %s -t 200 -P 10 -i 1| grep SUM > T51-b.out" %
(h1.IP()), shell = True)

   print "Starting ping trains h2->h1, h5->h1, h4->h3...."
   PT3 = h4.popen("ping -i 0.1 %s > RTT43-b.out" %( h3.IP()), shell =
True)
   PT2 = h5.popen("ping -i 0.1 %s > RTT51-b.out" %( h1.IP()), shell =
True)
   PT1 = h2.popen("ping -i 0.1 %s > RTT21-b.out" %( h1.IP()), shell =
True)
```

This piece of code actually does the desired task and starts the ping train.

```python
def getRTT(fileName):
   f = open(fileName, 'r')
   times = []

   while True:
       L =  f.readline()
       if L == "":
          break
       i = L.find("time=")
       if i != -1:
          z = L.find(" ms")
          n = L[i+5:z]
          times.append(float(n))
   return numpy.average(times)
```

This is a function that actually calculates the RTT.It uses a Python Library numpy.
In a similar way you can implement the Second Scenario.

Ans1)The congestion window h2->h1 and h5->h1 are equivalent
Because CNWD=rtt*throughput and rtt*throughput is almost same for both

Ans 2)S1->h1 has a higher drop rate.