Lab 1 Report

Martin Sanchez and Ben Jacobson

2/1/17

1 Two Nodes

In this first section we explored a simple network consisting of two nodes and one bidirectional link. We simulated the following scenarios:

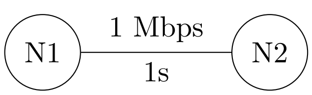
1. Set the bandwidth of the links to 1 Mbps, with a propagation delay of 1 second. Send one packet with 1000 bytes from n1 to n2 at time 0.

2. Set the bandwidth of the links to 100 bps, with a propagation delay of 10 ms. Send one packet witih 1000 bytes from n1 to n2 at time 0.

3. Set the bandwidth of the links to 1 Mbps, with a propagation delay of 10 ms. Send three packets from n1 to n2 at time 0 seconds, then one packet at time 2 seconds. All packets should have 1000 bytes.

After running the simulations we will show our network configuration, the output of the simulation and the calculations we used to verify that the output was correct.

The results of running the simulator for each of the scenarios are below:

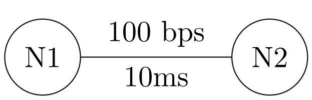
1. 

   The output of the simulation was:

```
    Created:0   ID:1  Received:  1.008
```

   These are our verifying calculations:

$$\begin{aligned} d &= d_{trans} + d_{prop} \\ &= (1000 * 8)/1000000 + 1 \\ &= 1.008 \end{aligned}$$

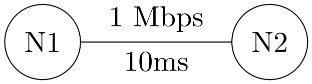
2. 

   The output of the simulation was:

```
        Created:0   ID:1  Received:  80.01
```

   These are our verifying calculations:

$$\begin{aligned} d &= d_{trans} + d_{prop} \\ &= (1000 * 8)/100 + 0.01 \\ &= 80.01 \end{aligned}$$

3. 

The output of the simulation was:

```
Created:0   ID:1   Received:0.018000000000000002
Created:0   ID:2   Received:0.026000000000000002
Created:0   ID:3   Received:0.034
Created:2.0   ID:4   Received:2.018
```

These are our verifying calculations:

$$\begin{aligned} d1 &= d1_{trans} + d1_{prop} \\ &= (1000 * 8)/1000000 + 0.01 \\ &= 0.018 \end{aligned}$$

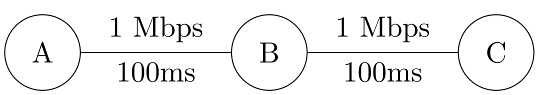
$$\begin{aligned} d2 &= d1_{trans} + d2_{trans} + d2_{prop} \\ &= 0.018 + (1000 * 8)/1000000 + 0.01 \\ &= 0.026 \end{aligned}$$

$$\begin{aligned} d3 &= d2_{trans} + d3_{trans} + d3_{prop} \\ &= 0.026 + (1000 * 8)/1000000 + 0.01 \\ &= 0.034 \end{aligned}$$

$$\begin{aligned} d4 &= 2.0 + d4_{trans} + d4_{prop} \\ &= 2.0 + (1000 * 8)/1000000 + 0.01 \\ &= 2.018 \end{aligned}$$

2 Three Nodes

In this section we will use the simulator to setup a network consisting of three nodes and two links. We will test two fast links and one fast link with a slow link. For each scenario we will show our network configuration, the last 5 lines of the simulation ouput and the calculations we used to verify the output. The results are as follows:

1. 

Our data for the two fast links is below:

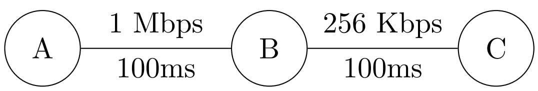
```
1Mbps
Created:0   ID:995   Received:8.176000000000007
Created:0   ID:996   Received:8.184000000000006
Created:0   ID:997   Received:8.192000000000007
Created:0   ID:998   Received:8.200000000000006
Created:0   ID:999   Received:8.208000000000007

1Gbps
Created:0   ID:995   Received:0.2079760000000001
Created:0   ID:996   Received:0.2079840000000001
Created:0   ID:997   Received:0.20799200000000012
Created:0   ID:998   Received:0.20800000000000013
Created:0   ID:999   Received:0.20800800000000014
```

After examining our output we determined that the transmission delay dominated.
These are our verifying calculations:

$$
\begin{aligned}
d &= d_{trans1} + d_{prop1} + (1000 * d_{trans2}) + d_{prop2} \\
&= ((1000 * 8)/1000000) + .1 + (1000 * (1000 * 8)/1000000) + 0.1 \\
&= 0.008 + 0.1 + (1000 * 0.008) + 0.1 \\
&= 8.208
\end{aligned}
$$

$$
\begin{aligned}
d &= d_{trans1} + d_{prop1} + (1000 * d_{trans2}) + d_{prop2} \\
&= ((1000 * 8)/1000000000) + .1 + (1000 * (1000 * 8)/1000000000) + 0.1 \\
&= 0.000008 + 0.1 + (1000 * 0.000008) + 0.1 \\
&= 0.208008
\end{aligned}
$$

2. 

Our data for the one fast link and one slow link is below:

```
Created:0  ID:995  Received:31.333000000000002
Created:0  ID:996  Received:31.364250000000002
Created:0  ID:997  Received:31.395500000000002
Created:0  ID:998  Received:31.426750000000002
Created:0  ID:999  Received:31.458000000000002
```

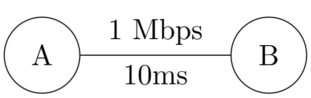
These are our verifying calculations:

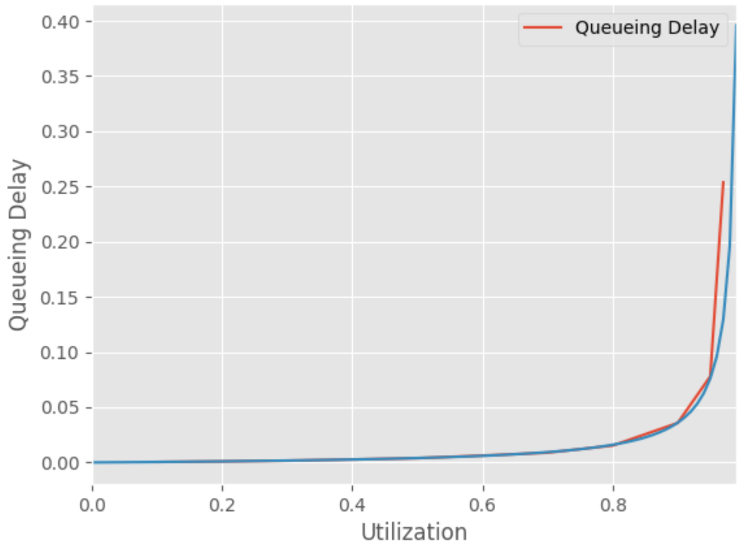
$$
\begin{aligned}
d &= d_{trans1} + d_{prop1} + (1000 * d_{trans2}) + d_{prop2} \\
&= ((1000 * 8)/1000000) + .1 + (1000 * (256 * 8)/1000000) + 0.1 \\
&= 0.008 + 0.1 + (1000 * 0.03125) + 0.1 \\
&= 31.458
\end{aligned}
$$

3 Queueing Theory

In this section we explore whether or not the simulator can validate basic queueing theory. We use the same configuration as the twonodes3.txt.

It is shown below:





This is the graph of our queue theory output. As you can see the red is the experimental queueing delay data. The Blue line is the theoretical data. It is apparent that the simulator does indeed validate basic queueing theory. However, like any experimental model the simulator is inperfect and there are slight differences above the 80% usage mark.

The load we used to generate cover the full range of values from 0 to 1. We used extra points in the range from 0.9 to 1.0 to get more data points in this critical range. The data is found in the following table:

Rate	Load
0.1	12.5
0.2	25.0
0.3	37.5
0.4	50.0
0.5	62.5
0.6	75.0
0.7	87.5
0.8	100.0
0.9	112.5
0.95	118.75
0.97	121.25

The data we collected follows the theoretical exponential curve proving that the bene simulator can correctly simulate queueing theory.

4 Summary

In this lab we successfully explored some of the basic functionality of the bene simulator. We were able to simulate a two node basic network with various bandwidths and propagation delays. We then even sent multiple packets of data across our networks and the simulator responded well. See section on two nodes.

In the section on three nodes, we explored a slightly more complicated network consisting of three nodes and two links. The first scenario contained two links with the same exact bandwidth and propagation

delay. While the second scenario included a link that was about 75% of the speed as the first link. The simulator also responded well to and gave us the same answers as we got in our calulations by hand. See section on three nodes.
Lastly, we were able to prove that the Bene simulator can correctly validate basic queueing theory with a good margin of error. See graph from section on queueing theory.