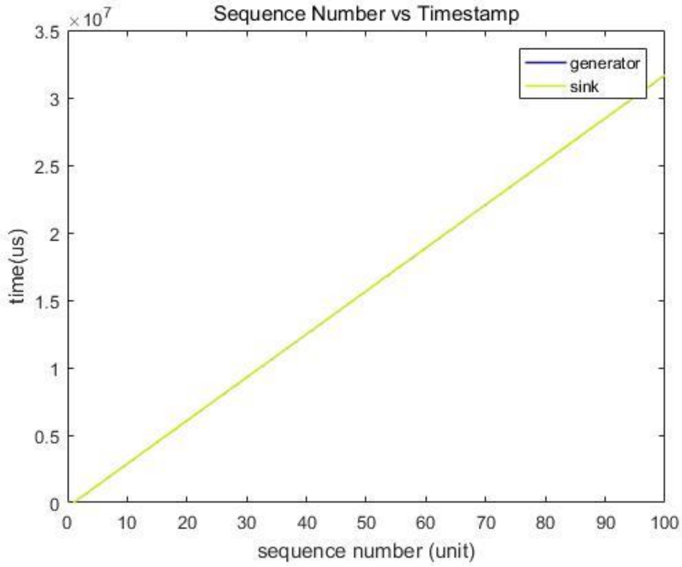
ECE 466 Lab 4

Xin Chen
1004391865

Part 1. Generating and Time-stamping Packet Trains

Exercise 1.5 Evaluation

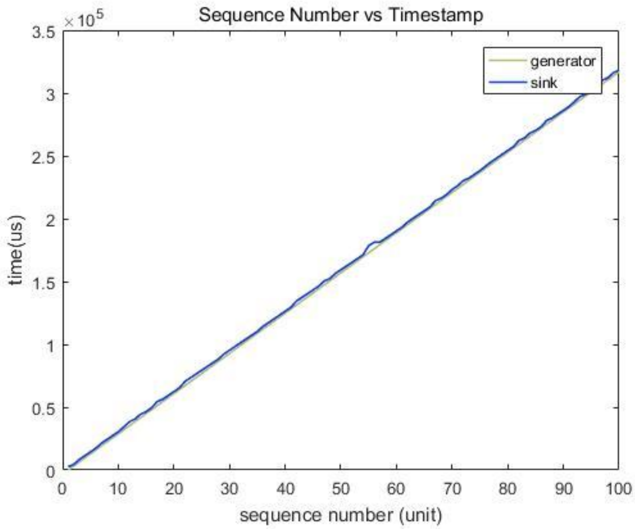
1. Packet train: N = 100, L = 400, r = 10



Plot1. Sequence Number vs. Timestamp

The above graph shows that two curves almost have the same value and close to each other. This represents that service curve has almost the same rate of arrival function. As the rate of arrival is not large, BlackBox is able to transmit the receiving packets efficiently.

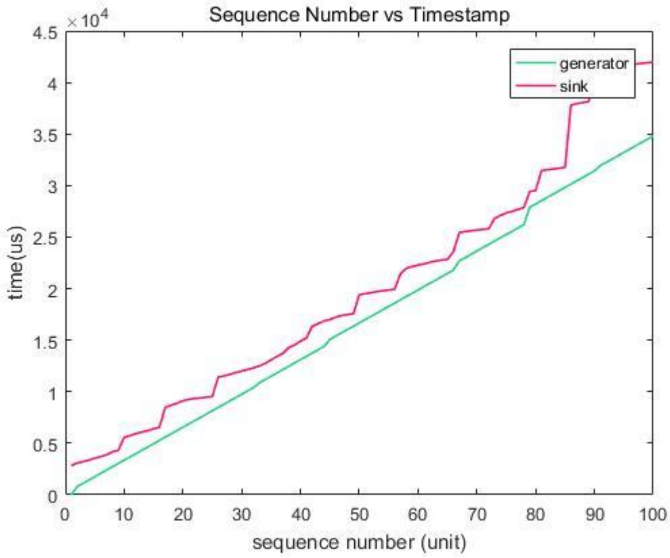
2. Packet train: N = 100, L = 400, r = 1000



Plot2. Sequence Number vs. Timestamp

The above graph shows that as the transmission rate increases, it causes a small backlog in the BlackBox in some time intervals. As the rate of BlackBox is infinite, and there is no delay of in the BlackBox, the backlog will be sent in the next timestamp and the number of packets will not keep increasing.

3. Packet train: N = 100, L = 400, r = 10000



Plot3. Sequence Number vs. Timestamp

The above graph shows that as the transmission rate increases, it almost has the same trend as previous graph. There are some jumps in the graph. This is due to the high sending rate in the BlackBox. BlackBox sends out the packet faster than arrival, causing there is some delay in the transmission.

4. Source Code

```

import java.io.*;
import java.net.*;
import java.util.StringTokenizer;
import java.util.Arrays;

public class TrafficEstimator {
    public static void main(String[] args) throws IOException {
        InetAddress addr = InetAddress.getByName(args[0]);
        int sendPort = 4444;
        int sinkPort = 4445;

        int N = Integer.parseInt(args[1]);
        int L = Integer.parseInt(args[2]);
        int r = Integer.parseInt(args[3]);
        //wait around 2ms then call the traffic generator
        long wait_time = 2;
        new trafficSink(sinkPort, N, L, r).start();

        long sys_pre = System.nanoTime();
        long sys_cur = System.nanoTime();

        while ((sys_cur - sys_pre) < wait_time*1000000) {
            sys_cur = System.nanoTime();
        }
        new trafficGenerator(addr, sendPort, sinkPort, N, L, r).start();

    }
}

```

Part 2. Bandwidth Estimation of Black Boxes

Exercise 2.1 Implement and Test the Probing Methodology

We found that if we set the R = 1000Mbps, it could cause some exception in the nanosecond timeout value out of range, so we decided to use R = 100kbps and 1Mbps for the probing methodology.

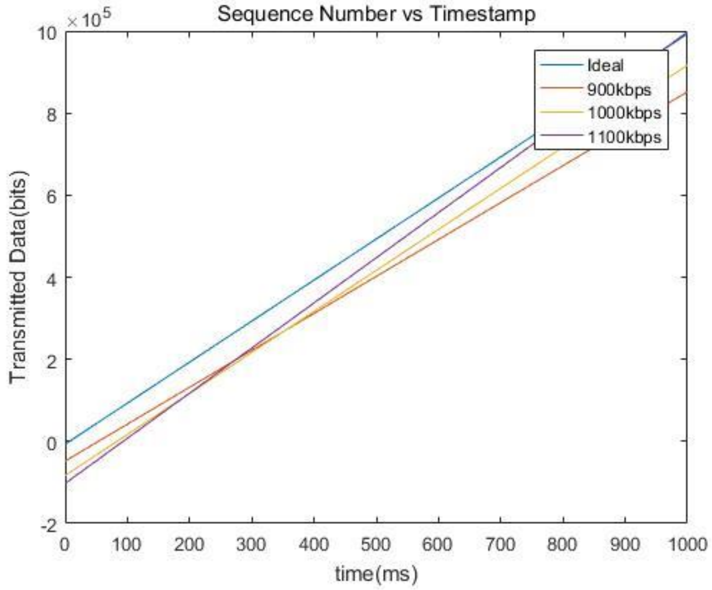
To implement the algorithm, I will first use a wide range. For example, for R = 1Mbps in the BlackBox, I will first choose the rate in the range of 500Kpbs and 1500Kbps and use the providing algorithm to output the most closely relevant service rate. (S(t) = max{r*t – Bmax}, and record the r which gives the maximum service curve). And then according to the r output, I will reduce the range of probing rate and the increasing interval until we find the final value.

The below graph is the Estimator for Part2.

```java

import java.io.*;
import java.net.*;
import java.util.StringTokenizer;
import java.util.Arrays;

public class TrafficEstimator1 {
    public static void main(String[] args) throws IOException {
        InetAddress addr = InetAddress.getByName(args[0]);
        int sendPort = 4444;
        int sinkPort = 4445;

        int N = Integer.parseInt(args[1]);
        int L = Integer.parseInt(args[2]);
        int r = Integer.parseInt(args[3]);
        int NoFile = 1;
        //wait around 2ms then call the traffic generator
        long wait_time = 2;
        new trafficSink1(sinkPort, N, L, r, NoFile).start();

        long sys_pre = System.nanoTime();
        long sys_cur = System.nanoTime();

        while ((sys_cur - sys_pre) < wait_time*1000000) {
            sys_cur = System.nanoTime();
        }
        new trafficGenerator1(addr, sendPort, sinkPort, N, L, r, NoFile).start();

    }
}

```

A. L = 400 Bytes and N = 100

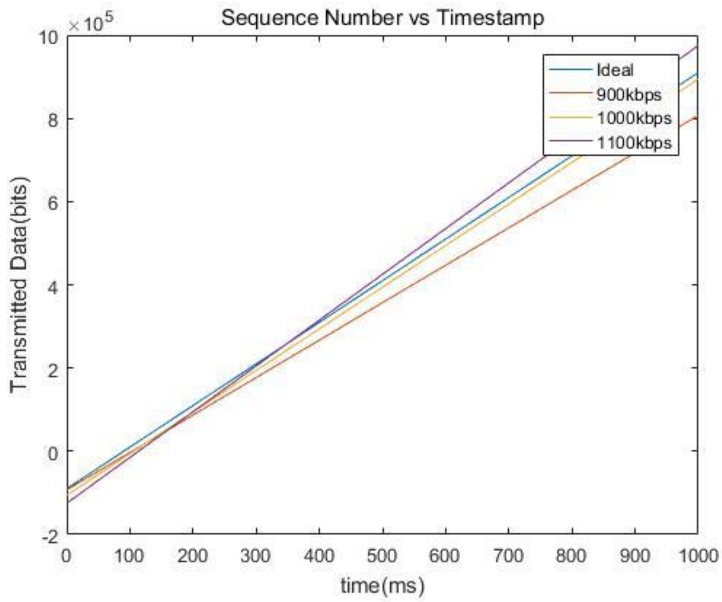
1. BlackBox1: b = L_{max}, R = 1Mbps, T = 10000



a. r = 900Kbps, S(t) = 900 * t - 48000
b. r = 1000Kbps, S(t) = 1000 * t - 83200
c. r = 1100Kbps, S(t) = 1100 * t – 102400

The above graph shows that the service curves are closely and overlap with each other. We can see that r = 1000Kbps almost has the same rate as the ideal one. But the burst size is a little bit off, this may happen as the error of measurement and the network environment.

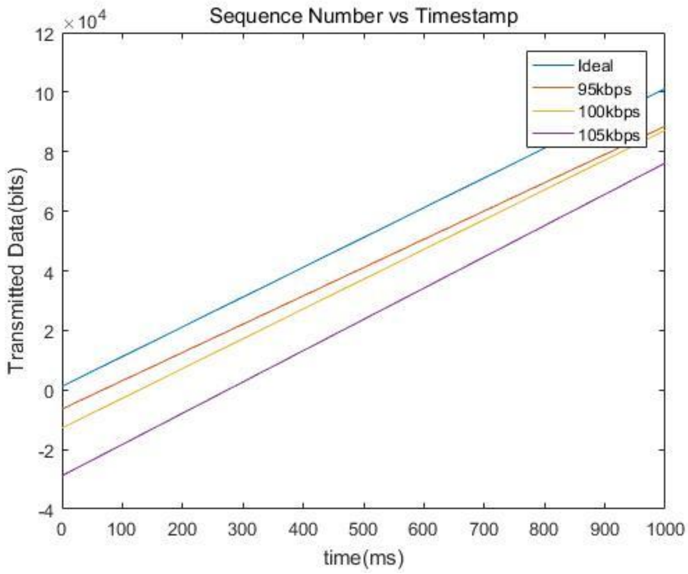
2. BlackBox2: b = 10000, R = 1Mbps, T = 100000



a. r = 900Kbps, S(t) = 900 * t - 92800
b. r = 1000Kbps, S(t) = 1000 * t - 105600
c. r = 1100Kbps, S(t) = 1100 * t – 124800

The above graph shows that the service curves are closely and overlap with each other. We can see that r = 1000Kbps almost has the same rate as the ideal one. Which shows the same result as the previous BlackBox. The burst size is also a little bit off, this may happen as the error of measurement and the network environment.

3. BlackBox3: b = L_{max}, R = 100Kbps, T = 20000

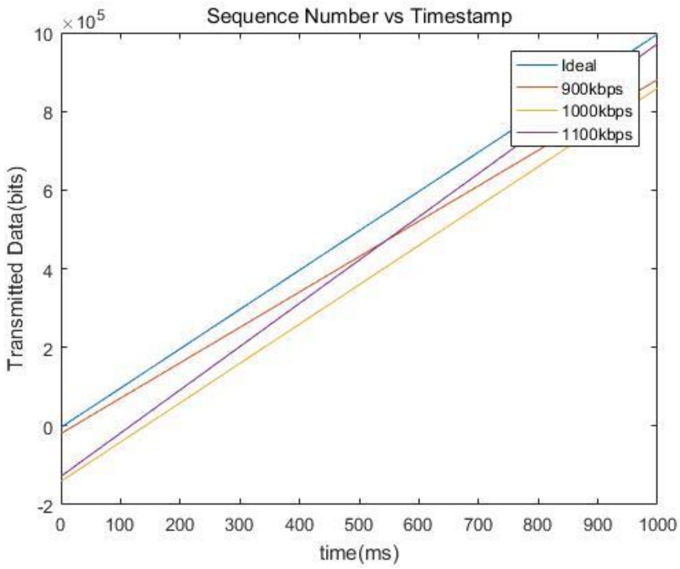


a. r = 95Kbps, S(t) = 95 * t - 6400
b. r = 100Kbps, S(t) = 100 * t - 12800
c. r = 105Kbps, S(t) = 105 * t – 28800

The above graph shows that the service curves are a little off from each other. We can see that r = 100Kbps almost has the same rate as the ideal one. This may happen as the error of measurement and the network environment.

B. L = 800 Bytes and N = 200

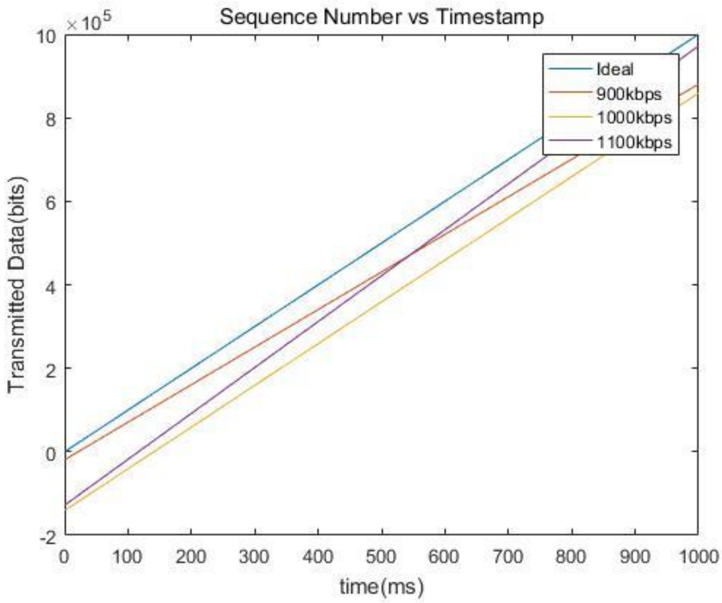
1. BlackBox1: b = L_{max}, R = 1Mbps, T = 10000



a. r = 900Kbps, S(t) = 900 * t - 19200
b. r = 1000Kbps, S(t) = 1000 * t - 140800
c. r = 1100Kbps, S(t) = 1100 * t – 128000

By repeating the experiment, with double packet trains and packet size, we get almost the same result. We can still see that r = 1000Kbps almost has the same rate as the ideal one.

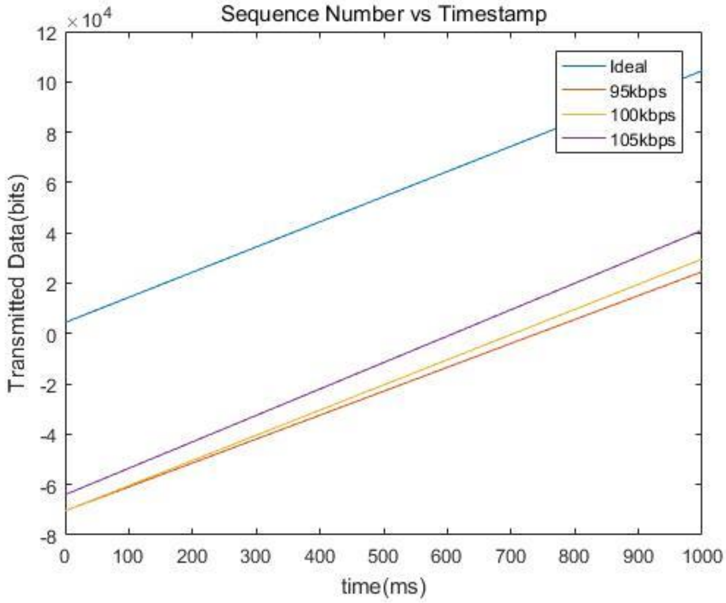
2. BlackBox1: b = 10000, R = 1Mbps, T = 100000



a. r = 900Kbps, S(t) = 900 * t - 19200
b. r = 1000Kbps, S(t) = 1000 * t - 140800
c. r = 1100Kbps, S(t) = 1100 * t – 128000

By repeating the experiment, with double packet trains and packet size, there is almost no change in the new service curve comparing with the previous result. The service curve is the same compare with the BlackBox with burst rate of 10000bits and delay of 10ms.

3. BlackBox1: $b = L_{max}$, R = 100Kbps, T = 20000

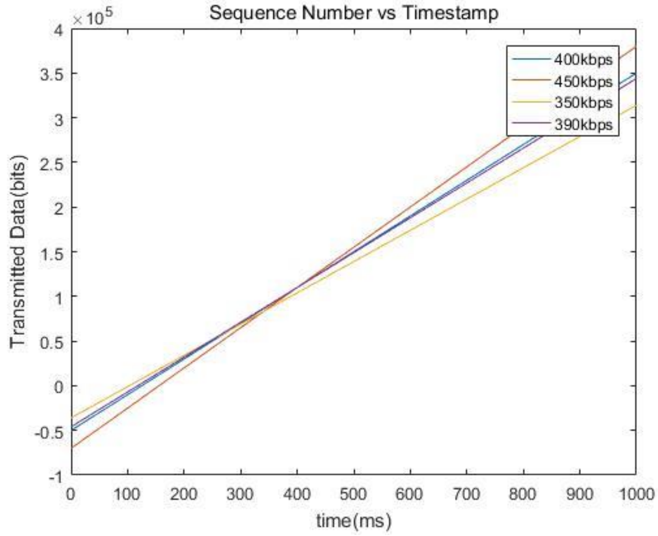


a. r = 95Kbps, S(t) = 95 * t - 70400
b. r = 100Kbps, S(t) = 100 * t - 70400
c. r = 105Kbps, S(t) = 105 * t – 64000

By repeating the experiment, with double packet trains and packet size, we do not get any improvement. We can see that r = 100Kbps curve has the same trend as the ideal service curve rate, but it shows an even larger burst rate.

<u>Exercise 2.2 Evaluation of BlackBox with unknown parameters</u>

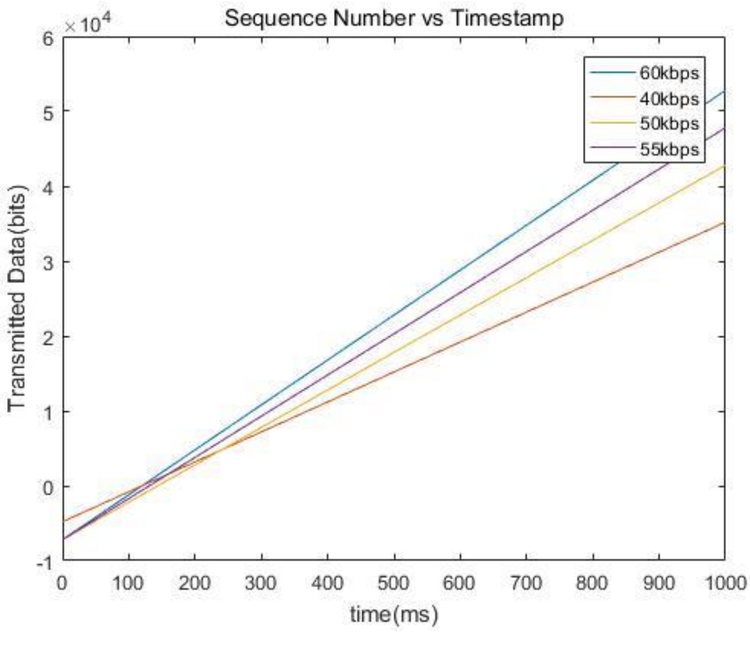
1. BlackBox1:



a. r = 400Kbps, S(t) = 400 * t - 50000
b. r = 450Kbps, S(t) = 450 * t - 70000
c. r = 350Kbps, S(t) = 350 * t - 36000
d. r = 390Kbps, S(t) = 390 * t – 46000

From the image shown above, we can show that the service curve with rate r = 390Kbps is the rate of the BlackBox1. From the output file, comparing the sink and generator's outputs, we can show that T = 100ms. And we have r = 390Kbps, T = 7000 bits.

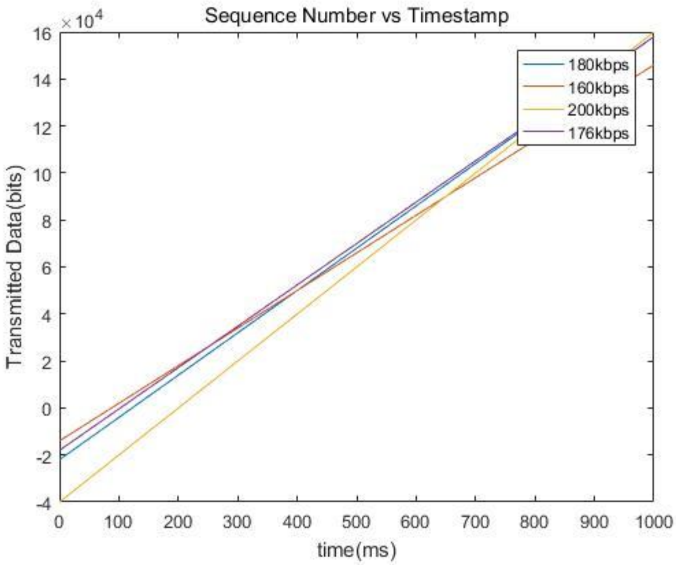
2. BlackBox2:



a. r = 60Kbps, S(t) = 60 * t - 7200
b. r = 40Kbps, S(t) = 40 * t - 4800
c. r = 50Kbps, S(t) = 50 * t - 7200
d. r = 55Kbps, S(t) = 55 * t – 7200

From the image shown above, we can show that the service curve with rate r = 55Kbps is the rate of the BlackBox2. From the output file, comparing the sink and generator's outputs, we can show that T = 0ms. And we have r = 50Kbps, T = 7200 bits.

3. BlackBox3:



a. r = 180Kbps, S(t) = 180 * t - 22000
b. r = 160Kbps, S(t) = 160 * t - 14000
c. r = 200Kbps, S(t) = 200 * t - 40000
d. r = 176Kbps, S(t) = 176 * t – 18000

From the image shown above, we can show that the service curve with rate r = 176Kbps is the rate of the BlackBox2. From the output file, comparing the sink and generator's outputs, we can show that T = 75ms. And we have r = 176Kbps, T = 4800 bits.

<u>Report your findings when you repeated the measurement experiments</u>

There is almost no improvement when repeating the experiments. Sometimes the burst size may change a little bit, it is really depending on the network environment.