CS 470 Project 2: TCP Simulation

Brandon Ingli

April 14, 2020

Contents

List of Figures

1 The Code

1.1 tcpsim.sh

```
#!/bin/bash
ALPHA=0.125
BETA=0.25

declare -a SITE=()
declare -a ESTRTT=()
declare -a DEVRTT=()
declare -a TIMEOUT=()

for site in "$@"
do
        SITE+=("$site")
        ESTRTT+=(0)
        DEVRTT+=(0)
        TIMEOUT+=(1000)
done

((MAXINDEX = ${#SITE[@]} - 1))

TIME=1

rm -rf data

mkdir data
for site in "${SITE[@]}"
do
        mkdir "data/${site}"
done

echo "Round 1..."
for i in `seq 0 "$MAXINDEX"`
do
        echo "${SITE[$i]}"
        ESTRTT[$i]=`ping -c 1 "${SITE[$i]}" | tail -1 | awk -F
           ↪ '/' '{print $5}'`

        echo "$TIME ${ESTRTT[$i]}" >>
           ↪ "data/${SITE[$i]}/estrtt.dat"
        echo "$TIME ${ESTRTT[$i]}" >>
           ↪ "data/${SITE[$i]}/samplertt.dat"

done
echo -e "\n"
((TIME = TIME + 5))
```

```
42
43 for j in {2..100}
44 do
45         sleep 5s
46         echo "Round $j..."
47         for i in `seq 0 "$MAXINDEX"`
48         do
49                 echo "${SITE[$i]}"
50                 SAMPLERTT=`ping -c 1 "${SITE[$i]}" | tail -1 |
                       ↪ awk -F '/' '{print $5}'`
51                 ESTRTT[$i]=`echo "(1-$ALPHA) * ${ESTRTT[$i]} +
                       ↪ $ALPHA * $SAMPLERTT" | bc`
52                 SAMPLEDEV=`echo "$SAMPLERTT - ${ESTRTT[$i]}" | bc
                       ↪ | tr -d -`
53                 DEVRTT[$i]=`echo "(1-$BETA) * ${DEVRTT[$i]} +
                       ↪ $BETA * $SAMPLEDEV" | bc`
54                 TIMEOUT[$i]=`echo "${ESTRTT[$i]} + 4 *
                       ↪ ${DEVRTT[$i]}" | bc`
55
56                 echo "$TIME ${ESTRTT[$i]}" >>
                       ↪ "data/${SITE[$i]}/estrtt.dat"
57                 echo "$TIME $SAMPLERTT" >>
                       ↪ "data/${SITE[$i]}/samplertt.dat"
58                 echo "$TIME ${TIMEOUT[$i]}" >>
                       ↪ "data/${SITE[$i]}/timeout.dat"
59
60         done
61         echo -e "\n"
62         ((TIME = TIME + 5))
63 done
64
65 ./plotit.sh
```

- **Lines 2–3:** Set the α and β values for the formulas
- **Lines 5–8:** Set up data arrays for the site to ping and variables to keep track of
- **Lines 10–16:** For every site given as a parameter, add it to the `SITE` array, and initialize a spot in the other¿et a variable for the last index of the arrays
- **Line 18:** Establish the last index in the arrays
- **Line 20:** Start the timer at time = 1
- **Line 22:** Remove any previous data
- **Lines 24–28:** Set up the directory structure for the data
- **Lines 30–41:** Round 1 of pinging

  - **Line 31:** Loop from 0 to `MAXINDEX` inclusive, step by 1
  - **Line 34:** Ping and set the `ESTRTT` as the sample RTT
    * `ping -c 1 "$SITE[$i]"` sends one ping
    * `tail -1` gets the last line of output, which are the statistics
    * `awk -F '/' '{print $5}'` pulls out the average ping time in ms, which for one ping is the ping time
  - **Lines 36–37:** Write this sample RTT to the Est RTT and Sample RTT data files
  - **Line 41:** Increment the current time by 5

- **Lines 43–63:** Rounds 2 to 100 of pinging
  - **Line 43:** Loop from 2 to 100 inclusive, step by 1
  - **Line 45:** Sleep for 5 seconds first
  - **Line 47:** For every site in the array...
  - **Line 50:** Get the Sample RTT by pinging; see line 34's explanation
  - **Line 51:** Calculate the EstimatedRTT value; `bc` is used because there are floating point numbers involved, which bash cannot do natively
  - **Line 52:** Calculate the Sample Deviation using `bc`; Absolute value is done by removing the `-` character if it appears using `tr`
  - **Line 53:** Calculate the RTT Deviation using `bc`
  - **Line 54:** Calculate the timeout value using `bc`
  - **Lines 56–58:** Write the time/data pairs to the appropriate files
  - **Line 62:** Increment the time by 5 seconds

- **Line 65:** Call the `plotit.sh` script to render the plots. This is its own script to make it easier to re-plot with new graph settings without fetching new data.

1.2 `plotit.sh`

```
1  #!/bin/bash
2
3  WIDTH=700
4  HEIGHT=420
5
6  rm -rf plot
7  mkdir plot
8
9  for site in `ls data`
10 do
11
12   mkdir "plot/${site}"
13
14   gnuplot -e "set key off; set xlabel \"Time (seconds)\"; set
       ↪ ylabel \"Timeout (milliseconds)\"; \
```

```
15                  set terminal pngcairo size ${WIDTH},${HEIGHT}
                      ↪ enhanced font \"Times-Roman,10\"; \
16              set output \"plot/${site}/timeout.png\"; set title
                   ↪ \"${site} Timeout\";\
17              plot 'data/${site}/timeout.dat' with linespoints"
18
19    gnuplot -e "set key outside; set xlabel \"Time (seconds)\"; set
         ↪ ylabel \"RTT (milliseconds)\"; \
20              set terminal pngcairo size ${WIDTH},${HEIGHT}
                   ↪ enhanced font \"Times-Roman,10\"; \
21              set output \"plot/${site}/rtt.png\"; set title
                   ↪ \"${site} RTT\";\
22              plot 'data/${site}/samplertt.dat' with linespoints
                   ↪ title \"Sample RTT\", \
23              'data/${site}/estrtt.dat' with linespoints title
                   ↪ \"Est RTT\""
24  done
```

- **Lines 3–4:** Set the height and width in pixels of the resulting plot
- **Lines 6–7:** Remove existing plots and make the root directory for plots again
- **Line 9:** For every site that we have data for in `data/`...
- **Line 12:** Make the directory for that site's plots
- **Line 14:** For the timeout plot, no key, label the x and y axes
- **Line 15:** Setup the `PNG` output and set 10pt Times-Roman font
- **Line 16:** Set the output filename and title of graph accordingly
- **Line 17:** Plot the data
- **Line 19:** For the RTT plot, put the key outside the graph, label the x and y axes
- **Line 20:** Setup the `PNG` output and set 10pt Times-Roman font
- **Line 21:** Set the output filename and title of graph accordingly
- **Line 22:** Plot the sample rtt data and label appropriately
- **Line 23:** Plot the estimated rtt data abd label appropriately

2 The Data

The four servers I chose to ping were DNS servers, as they were more likely to not reject my frequent pings. They were...

- 8.8.8.8 (US)
- 101.110.40.225 (Japan)

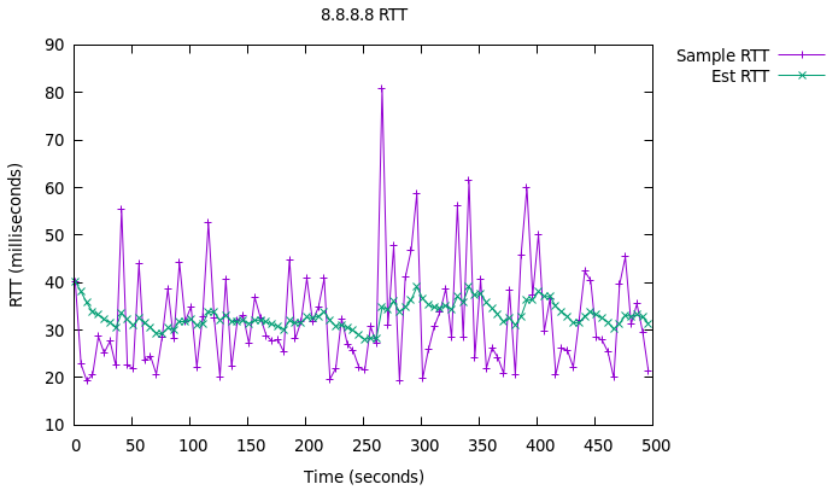
- 88.208.228.34 (UK)
- 185.203.224.102 (Spain)

The data collected consisted of 3 files for each site:

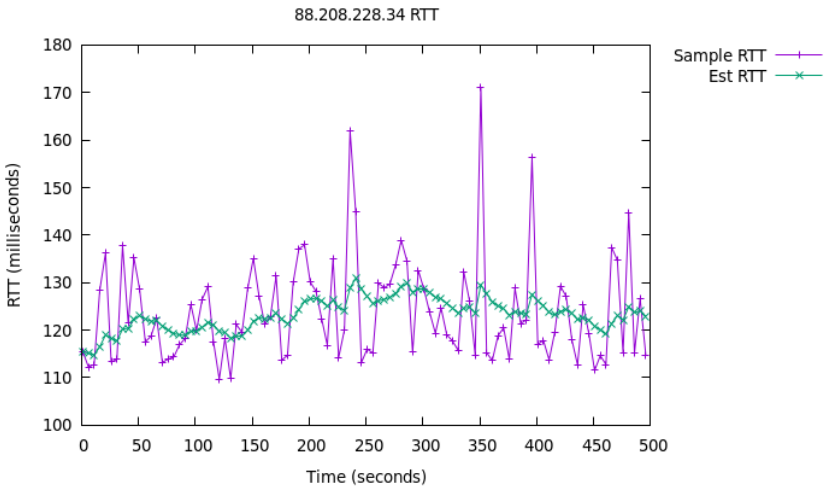
- `samplertt.dat` holds pairs of time t and ping time
- `estrtt.dat` holds pairs of time t and the calculated EstRTT
- `timeout.dat` holds pairs of time t and the calculated timeout value

Here's a short sampling of what the `samplertt.dat` file for 101.110.40.225 looks like:

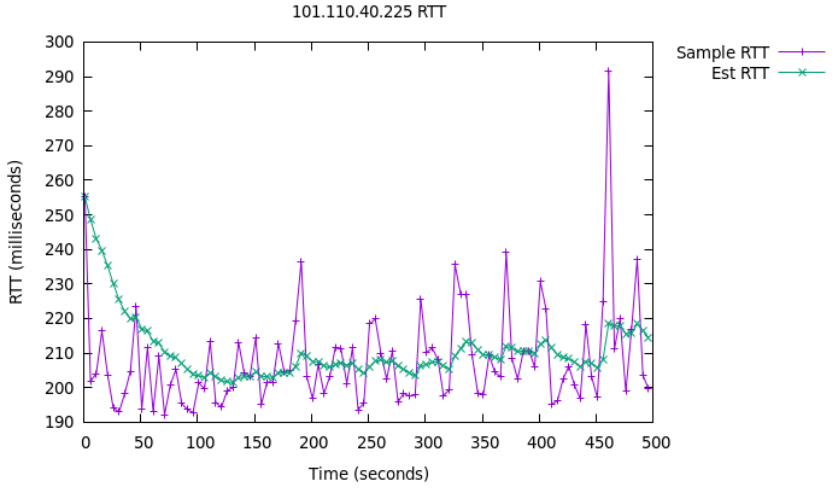
```
1 255.287
6 201.842
11 204.065
16 216.341
21 203.589
26 194.032
31 192.966
36 198.259
41 204.783
46 223.306
```



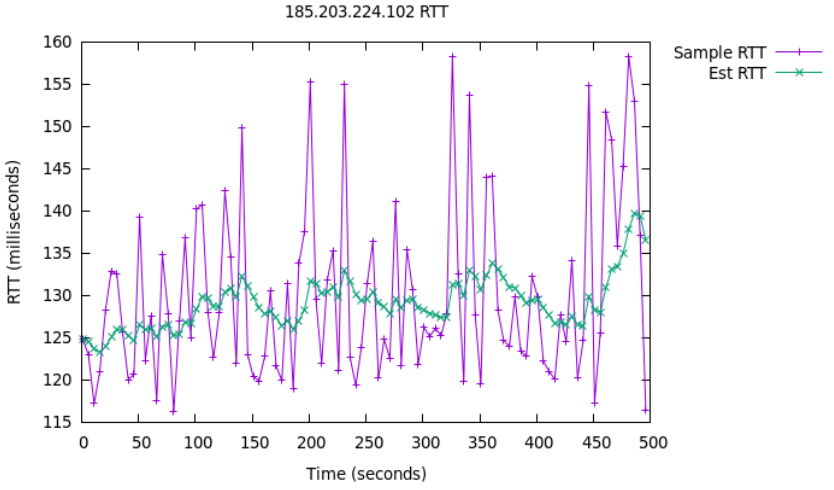
(a) 8.8.8.8 (US)



(b) 88.208.228.34 (UK)

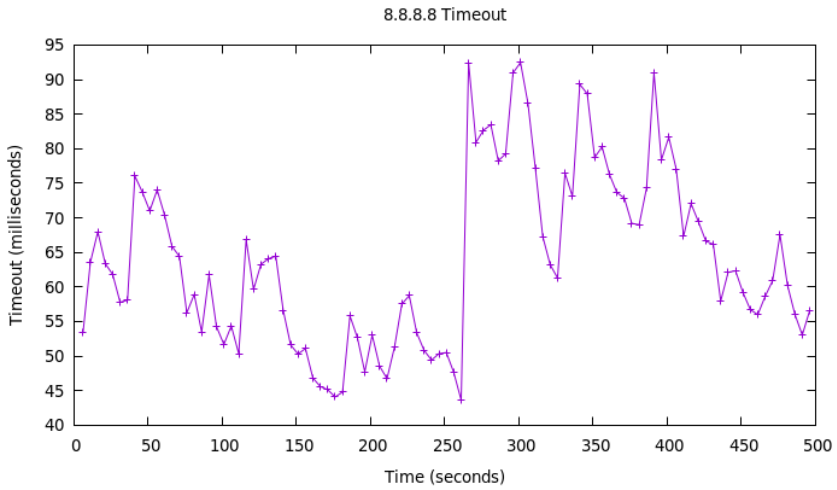


(c) 101.110.40.225 (Japan)

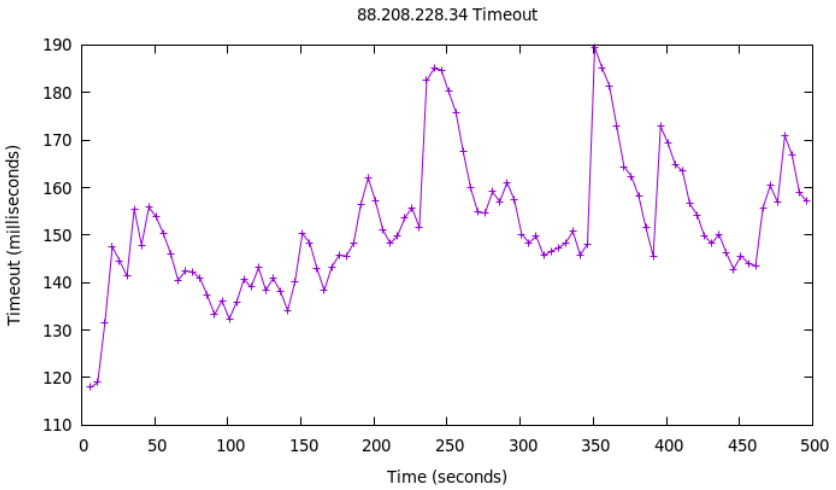


(d) 185.203.224.102 (Spain)

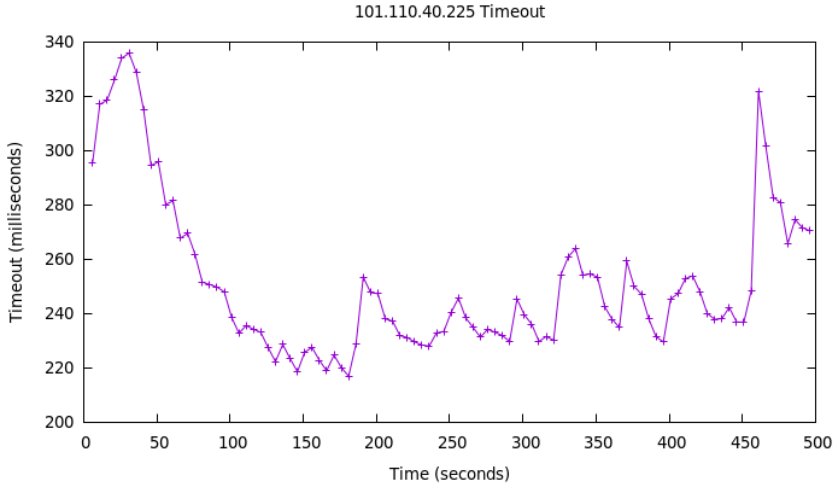
Figure 1: Estimated and Sample RTT Values



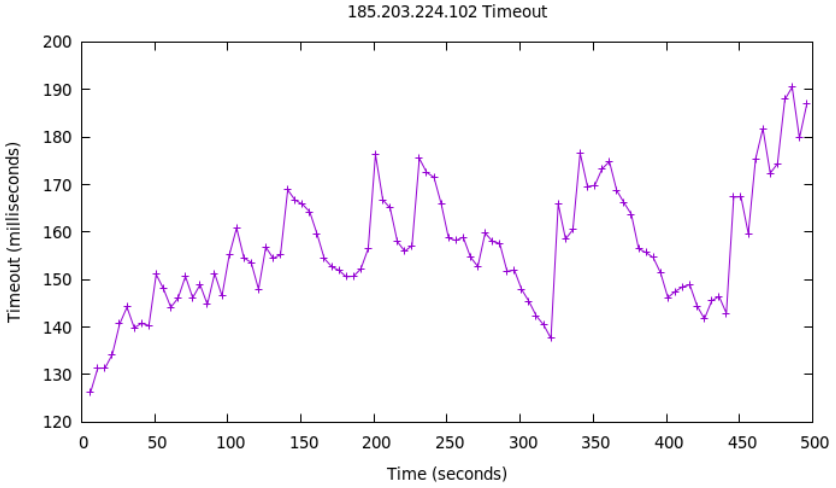
(a) 8.8.8.8 (US)



(b) 88.208.228.34 (UK)



(c) 101.110.40.225 (Japan)



(d) 185.203.224.102 (Spain)

Figure 2: Timeout Values

3 Discussion of Results

3.1 RTT

The graphs of my results for RTT are found in Figure 1 on page 7. The purple line represents the Sample RTT values, while the green line represents the calculated Estimated RTT value.

It is evident that the Sample RTT values change rapidly on all four graphs, with various unpredictable peaks and valleys. However, the Estimated RTT values remain relatively stable since it is a weighted average.

After a larger dip as the weighed average of the Estimated RTT is established, the estimated RTT for the US remained relatively stable. There are rises and falls along the way, but all values remain within 10 milliseconds or so of each other. Peaks and valleys on the Estimated RTT mirror those of the sample RTT, albeit slightly delayed and of a much smaller magnitude. Take, for example, the large jump from $\approx t = 265$ to $t = 270$. The sample RTT jumps significantly from about 26ms to more than 80ms, and immediately back near 30ms. This 54ms spike in sample RTT resulted in only about a 5ms jump in the estimated RTT, meaning the weighted average is stabilizing the values as it should.

The server in the UK is pretty similar. Large spikes, such as between $t = 230$ and $t = 245$, or $t = 350$ and $t = 360$, result in minor spikes in the Estimated RTT, which then quickly correct themselves to the nominal value.

The data out of Japan is consistently variable, or in other words the peaks and valleys are somewhat predictable. The two extremes average out to be relatively stable, as indicated by the Estimated RTT line, after an initial period of establishing the average from the initial sample RTT. We see only one large spike in the sample RTT near the end of the test period, which is again met with a smaller magnitude spike in Estimated RTT. If the test were to run longer, I would suspect the Estimated RTT line would return to a similar level as before the spike.

The data out of Spain is the most interesting of the four. We can see the bursty nature of the Internet through the sample RTT's periods of massive spikes and smaller changes. Nonetheless, the Estimated RTT stays relatively constant, and has an overall positive trend as the sample RTT spikes average larger and larger.

These tests were run at about 11:00pm CDT (5am in the UK, 6am in Spain, and 1pm in Japan). Since these were all DNS servers, it can be safe to conclude that in the UK and Spain, local DNS caches may have been flushed and obtaining new information, creating periods of congestion that are especially evident in Spain's data. Japan's latency is overall the largest, which can most likely be attributed to physical distance and the increased probability of loss as a result. On the contrary, the US server chosen, Google's DNS, is on the same continent and most likely connected through Google's Tier 1 network, ensuring lightning fast responses under 85ms.

3.2 Timeout

The graphs of my results for calculated timeout values are found in Figure 2 on page 8. The initial timeout of 1 second (1000 ms) is not shown in the data nor on the graphs so that the scaling is more suited for the data we care about.

On all four graphs, each graph starts with a large increase as the `EstRTT` and `DevRTT` get more data for their weighted averages, and the timeout formula adds in the safety margin of 4 `DevRTT`.

If we were to overlay a best-fit line on each graph once the Timeout value was established (for example, fitting a line starting around $t = 150$ in the Japan data once the initial spike is averaged out), we would see that all four would have a net positive slope. This means that the timeout value grows over time and potentially indicates increased congestion over the connection lifetime.

Because it is a main component of the formula for calculating timeout values, the graphs for timeout expectedly follow a similar trend to the estimated RTT graphs. The data from Japan shows this most clearly. The Estimated RTT starts rather high, as the first few RTT samples were found on a peak. Consequently, the timeout values start rather high as well. Then, as the estimated RTT dropped rather quickly, so too did the timeout values. Even the minute spikes in Estimated RTT are pretty closely replicated in the timeout graph in shape and duration.

We can also say something about the `DevRTT` from these graphs. Large spikes in timeout, like that from $\approx t = 265$ to $t = 270$ in the data from the US, indicates a large deviation in nearby RTTs that can also be seen in Figure 1a in that same time frame.

3.3 Overall Reaction

Overall, I'm surprised at the speed to which communication over the Internet can occur over large distances. Data can be sent from my home in St. Louis to Google's servers in Mountain View and back in less than half the average length of the blink of a human eye![1] Additionally, data can traverse undersea cables across the ocean and back in under two blinks of an eye, which is astonishing when put into perspective. This speed has really helped to globalize society in ways previously unimaginable, especially considering the current pandemic.

[1] 100ms per https://www.somatechnology.com/blog/thursday-thoughts/fast-average-blink/