CS655 PA1
Xingkun Yin
U44255956

Part 1

Logistics:

The part one implement a very simple client and server app where server echo back the message it get from client.

Server:

To start the server, go to pa1part1 and type in

python server.py <port number>

You can pick any port number you like. I usually pick 58989. The server will create a socket that bind to that port number and ip address of 0.0.0.0 which would bind to all interface. The server will constantly listening and reply back any message it receives.

Client:

python server.py <ip address> <port number>

client.py takes 2 command line argument for ip and port number. It will create a socket to connect to the ip and port and send a message. It will the receive the message. Socket errors are catched.

I run server on csa2:

```
[yinxingk@csa2 pa1part1]$ python server.py 58979
Listening on port  58979
Accepting on port  58979
Receive message:  A
Send message back
```

While client is run on a different machine, namely csa1:

```
[yinxingk@csa1 pa1part1]$ python client.py 128.197.11.36 58979
Message send:  A
Message received:  A
```

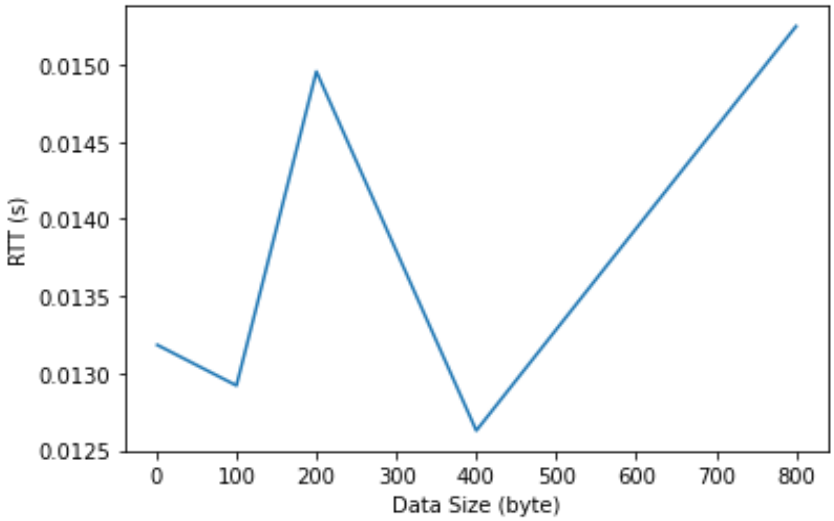
Part 2

This part consists of 3 phases, CSP, MP and CTP phase.

For all the experiments and graphs, there are 10 probes and the rtt and throughput is calculated as the average of the 10 probes.

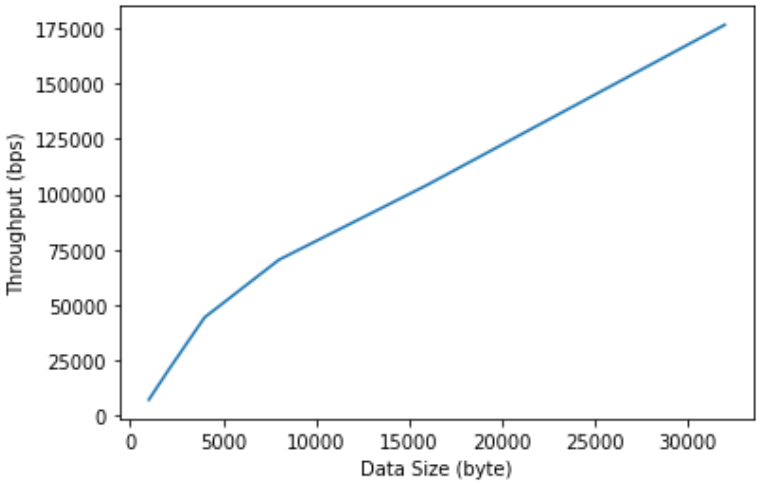
For the experiment, I use my own machine as the client and CSA2 as the server. I tried to use CSA1 as the client but the rtt is too small, probably due to the close distance between CSA1 and CSA2.

Latency vs. message size, own machine client to CSA2 server



Due to the small size of the message size, bandwidth is not the main issue. Neither is queueing delay. Propagation delay probably cause the main part of the latency resulting in the volatile shape. The RTT stay roughly the same.

Throughput vs. message size

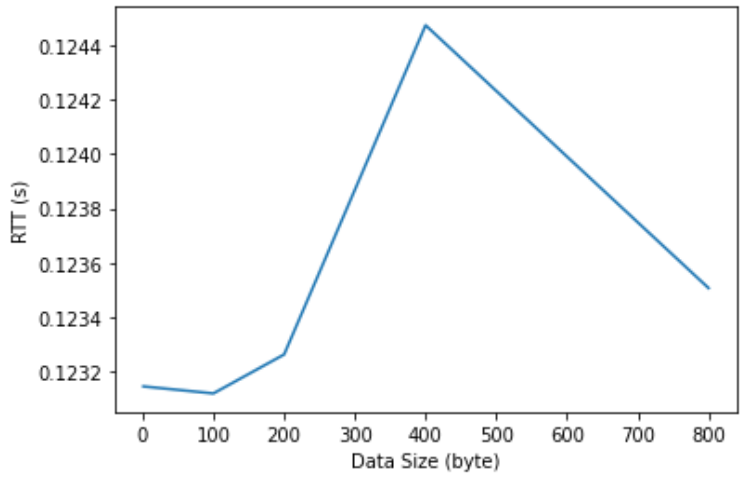


We can see a smooth grwoth in the throughput as the message size increase. This is probably due to the biggest data size here, 32K, is still not significant compare to the network capacity so the main delay is still propagation delay. The propagation delay is roughly the same for all message as the distance is fixed, and therefore as the message size grow, so does the throughput.

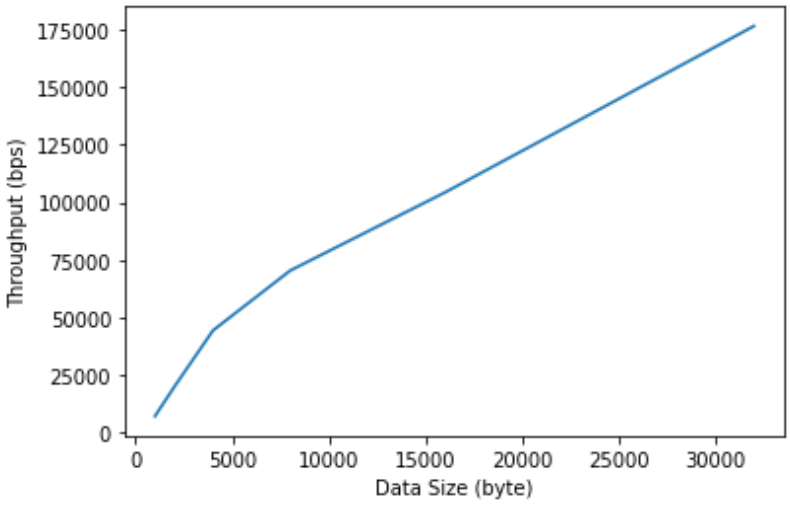
The same experiments are also conduct on a test server, I choose:

`pcvm1-24.instageni.utdallas.edu 50000`

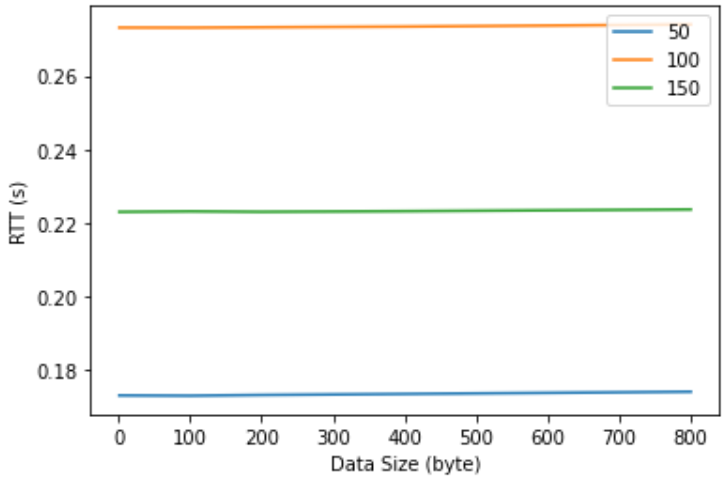
Latency vs. message size, pcvm1-24.instageni.utdallas.edu 50000

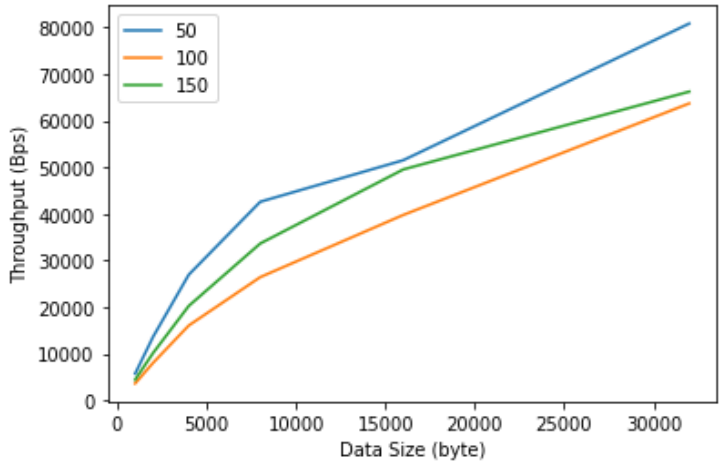


Throughput vs. message size



The result of the experiment towards the testing server show similar result. The RTT graph is smoother (although the shape look sharp but looking at the axis you will see that the standard deviation is much small than the one with own machine client to csa2 sever). This is probably due to the internet for CSA2 is much smoother than my own Wi-Fi. The throughput also show similar trend.





This is graphs using csa2 as client and pcvm1-24.instageni.utdallas.edu as server.
Server delay is set to 50, 100 and 150. The RTT seems consistent with the larger server delay, the higher RTT there is. The throughput has the same shape but seem to increase a little bit as the data size gets larger. We think the reasons are that system may cut messages into smaller packets and the larger the message is, the more packets there are. As the server delay is longer, there might be queueing delay as there are other people using the server as well.