**Using an Echo Application to Measure TCP Performance**

First, we'll set up the server. To begin, we will create a TCP socket object. Then we specify the reuse address, which allows us to run the server as many s as we need. On our local system, we bind the socket to the specified port. During the listening step, we use the backlog to the listen () function to ensure that we listen to numerous clients in a queue. Finally, we await the client's connection before sending some data to the server. When the data is received, the server sends it back to the client.

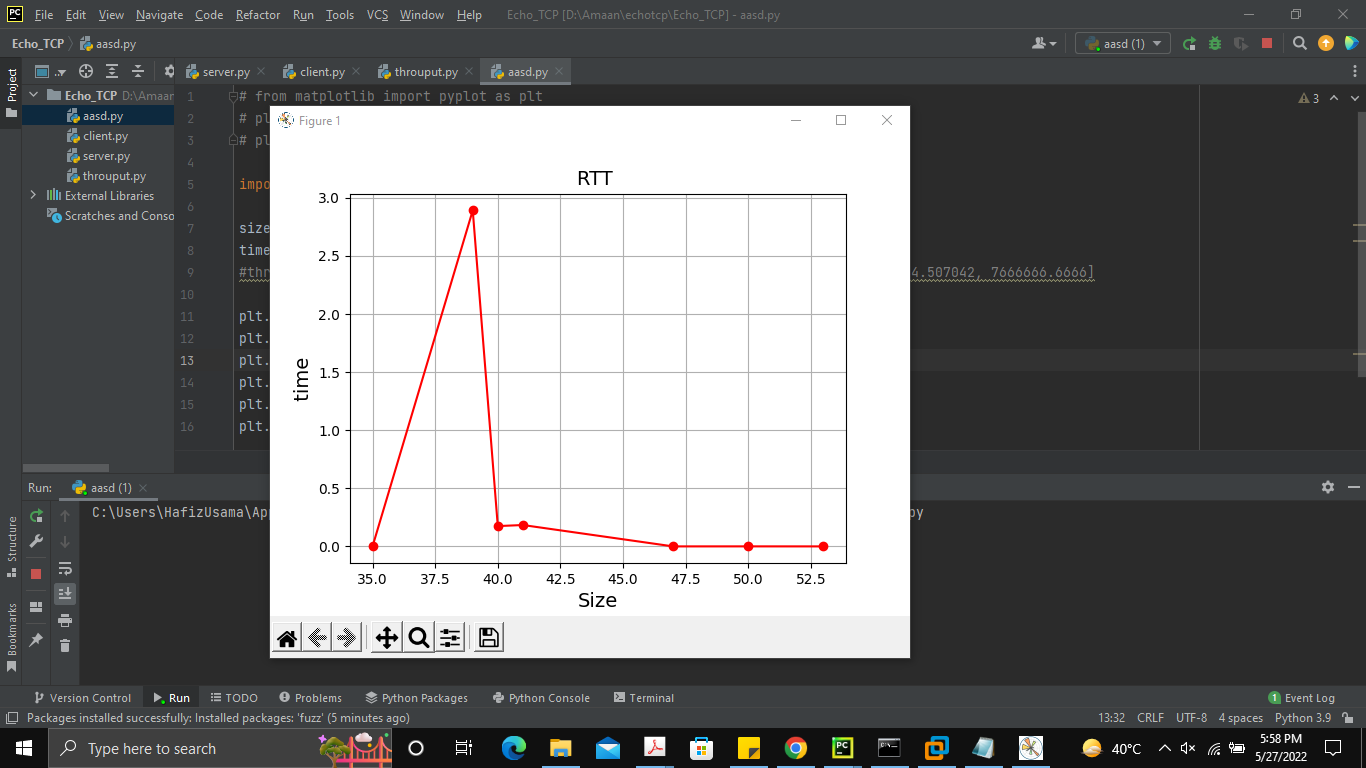
On the client side, we use the port parameter to build a client socket and connect to the server. The client then sends the message, **this class is awesome. TEST Message**. This is repeated to the server, and the client receives the message in a few parts instantly. Two try-except blocks are used here to capture any exceptions that occur during this interactive session.

**TCP’s round trip vs message size**

The round-trip (RTT) is the it takes from when a browser submits a request to when it response is received from a server, measured in milliseconds. It is a critical performance measure for online applications and, along with to First Byte

First I start a server and connect the client with server then start a Wireshark tool and find a packets which is send server to client or client to server make a pcap file and use a scapy tool and find a round trip from the pcap file of localhost and I try seven this process with deferent message size.

|  |  |
| --- | --- |
| **Round 1**  ('0.000025') size is 35 bytes  **Round 2**  ('2.891704') size is 39 bytes  **Round 3**  ('0.174795') size is 41 bytes    **Round 4**  ('0.183957') size is 40 bytes | **Round 5**  ('0.000057') size is 47 bytes  **Round 6**  ('0.000284') size is 50 bytes  **Round 7**  ('0.000015') size is 53 bytes |

**Graph of RTT**

**TCP throughput as a function of message size**

Network throughput in data transmission is the quantity of data successfully transmitted from one location to another in a particular period, and is commonly measured in bits per second (bps).

First I start a server and connect the client with server then start a Wireshark tool and find a packets which is send server to client or client to server make a pcap file and count the number of packets in one. Then go to Statistics find Packet Length and put in the formula to find a throughput.

Throughput (bits/sec) = ((number of successful packets) + (average pocket-size)) / Total sent in delivering that amount of data

|  |  |
| --- | --- |
| **Test one**  11 packets  96 size  0.000025  Throughput = 4,280,000  **Test two**  11 packets  107 size  Time 2.891704  Throughput=40.80638958897591  **Test three**  14 packets  100 size  Time 0.174795  Throughput=652.192568  **Test four**  10 packets  102 size  Time 0.183957  Throughput=608.837934 | **Test five**  10 packets  103.25 size  Time 0.000057  Throughput=1,986,842.10  **Test six**  10 packets  118.25 size  Time 0.000284  Throughput=451,584.507042  **Test seven**  9 packets  106 size  Time 0.000015  Throughput=7,666,666.6666 |

