NetPulse A2 report

System Architecture Overview

In our system, we employ a network of four devices, with one of them designated as the master client and the remaining three as slave clients. This system operates on a general algorithm in which all four clients receive lines of data from the Vayu server. The slave clients are responsible for relaying every received line to the master client. Once the master client has collected all the lines, it then distributes the complete dataset to the other three slave clients for submission. The master sends this line by line with a gap of 1ms so as to not overfill the client buffer.

Slave Client Threads

Within each slave client, there are two dedicated threads responsible for handling connections. The first thread is responsible for establishing a connection to the Vayu server, while the second thread focuses on connecting to the master client. These connections are established during the initialization phase.

Master Client Threads

On the master client, we have four threads in operation. Three of these threads are responsible for establishing connections with the slave clients, and the fourth thread is designated to connect to the Vayu server. These threads are essential for facilitating seamless communication within the system.

Initialization and Connection Phases

The system initialization follows a specific sequence. Initially, the master client waits for incoming connections from the slave clients. Meanwhile, each of the slave clients independently attempts to establish a connection with the master client. This process ensures that all devices are ready to engage in communication.

Collaborative Communication

Once all devices are connected, the four devices collectively begin their communication with the Vayu server. The lines are sent from the server to all clients. The slave clients forward every received line to the master client. The master client, in turn, collates all the lines and subsequently sends the complete dataset to the other three slave clients. This synchronized approach ensures that all clients have access to the complete dataset for final submission.

Exception handling

Case 1

If slave clients disconnects from the master client, the connected component continues to work as usual without the disconnected slave, and the disconnected slave client can seamlessly connect back to the same system and continues sending new lines to the master again.

Case 2

If any system disconnects from Vayu server, reconnection is possible, by exiting the terminal and re-establishing connection.

Instructions to run

For master client

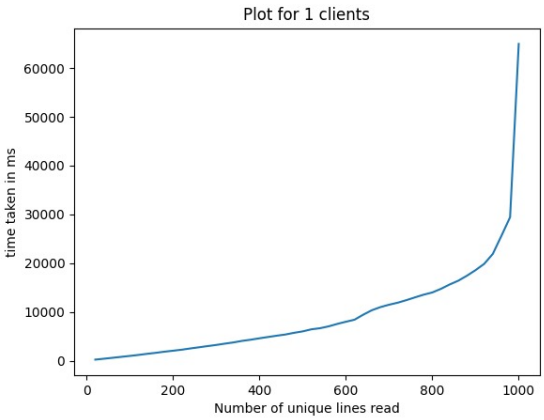
1. run Master.py.
2. As input, enter `act_server`. This will wait and listen for connections from slave clients.
3. After confirming that we got required number of slave clients, enter `connect_server` to start session with vayu server and collect lines.

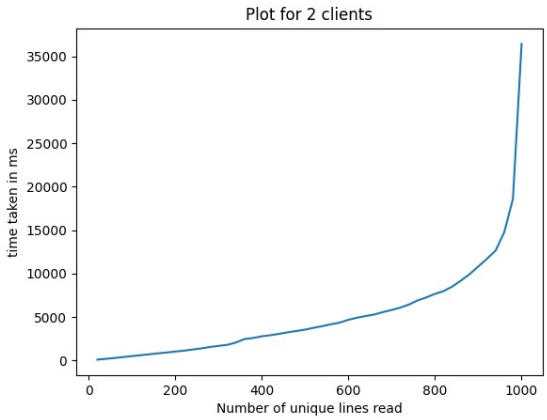
For slave client

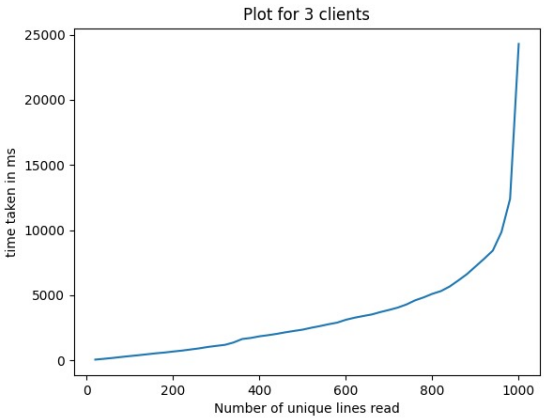
1. run Client.py
2. As input, enter `connect_client_server <master_IP_addr> <PORT>`. This establishes connection to the master client.
3. After required connections are made, enter `connect_server` for slave client to collect lines from vayu and send to master client.

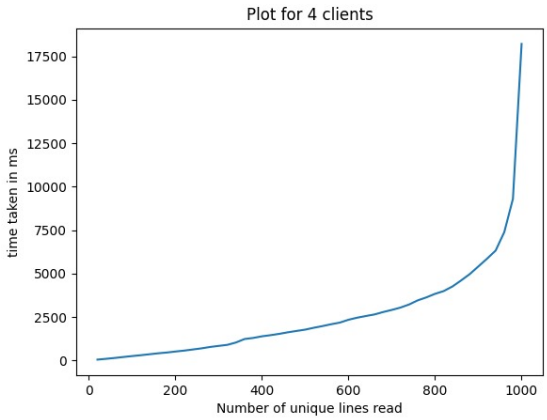
Performance graphs

For master client

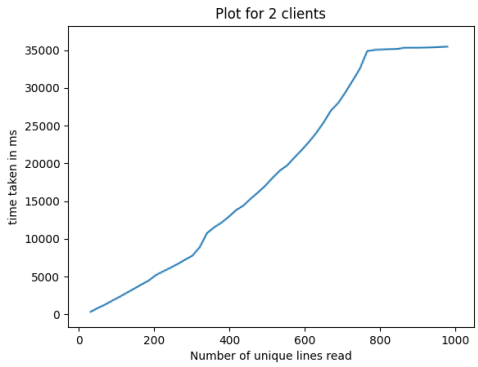


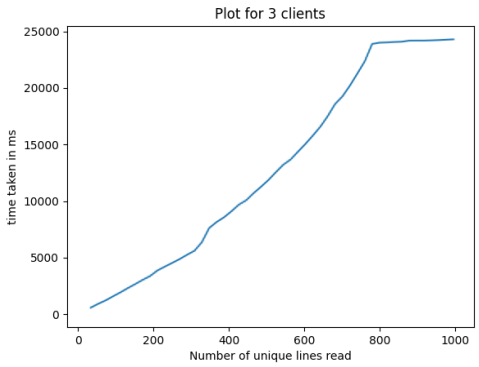


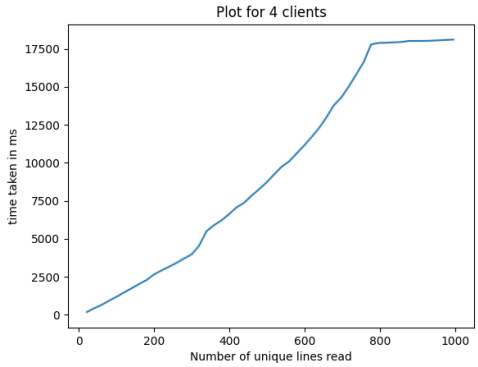




For slave clients







Observation from graphs

As number of clients increases, the total time taken reduces almost linearly, but when number of clients increase beyond 3, the rate of decrease of time slows down.

The plots for the slaves are as such because when the master is complete, it sends the full lines to all the slaves.

Team members

1. Bhavesh Gurnani - 2021CS50594
2. Piya Bundela - 2021CS10118
3. Aman Hassan - 2021CS50607
4. Brian Sajeev Kattikat - 2021CS50609