Name: Suyash Sardar

USC ID: 7087370255

Socket Project

EE - 450

Section: 30500

<u>Assignment:</u>

I’ve designed a distributed recommendation system. So given some data (countries containing users and their current friends) the system can recommend a new friend for a given user in that country.

The way I’ve designed the system, it consists of:

1. Two backend servers: **serverA** and **serverB** - which have the data/information about the current relationships between users. They are also responsible for running the recommendation algorithm that will suggest a new friend for a given user based on common neighbours principle.
2. **Main Server**: This server has an endpoint facing the client, from where the clients can request for a recommendation. It then talks to the backend servers in order to get the recommendation and return it back to the client. It is mainly responsible for handling multiple requests from clients and routing them to the correct backend server which might have the relevant information required for the recommendation algorithm to work.
3. **Clients** which can hit the main server endpoint to get recommendation of a friend for a given user in a given country.

Files:

serverA.cpp	Backend serverA which reads from data1.txt, creates a graph of all users in mentioned countries and their friends. It listens to the main server for recommendation queries and runs the common neighbour algorithm to suggest new friends.
serverB.cpp	Same as ServerA, except that it reads from data2.txt
servermain.cpp	Main server which has a client facing endpoint listening for recommendation queries, it is responsible for routing the queries to the correct backend server.
client.cpp	Client program for creating a recommendation query.

Port Numbers used:

ServerA	Listens and Talks to MainServer on port **30255** using UDP
ServerB	Listens and Talks to MainServer on port **31255** using UDP
MainServer	Listens and Talks to clients on port **33255** using TCP and to ServerA, ServerB on port **32255** using UDP.
Client	Uses a dynamic port to talk and listen to MainServer over TCP.

<u>Format of messages exchanged:</u>

Client → Main Server	User_id and Country serialised as a space separated string
Main Server → Backend Server	User_id and Country serialised as a space separated string
Backend Server → Main Server	Returns an integer: - **user_id** of the recommended user if a recommendation can be made. - **-1** if the user in query is already connected to all other users. - **-2** if the user in query is not present in the given country.
Main Server → Client	Returns an integer: - **user_id** of the recommended user if a recommendation can be made. - **-1** if the user in query is already connected to all other users. - **-2** if the user in query is not present in the given country. - **-3** if the country is not present

<u>Messages printed on screen:</u>

<u>ServerA:</u>

Initial Boot Up	The server A is up and running using UDP on port 30255
After sending country list	The server A has sent a country list to Main Server
On receiving a query from main server	The server A has received request for finding possible friends of User <uid> in <c> The server A is searching possible friends for User <uid>... Here are the results: <uid_ans> The server A has sent the results to Main Server
If <uid> is not present in country	User <uid> does not show up in <c> The server A has sent 'User <uid> not found' to Main Server
If <uid> is already connected to all	User <uid> is already connected to all other users, no new recommendation The server A has sent 'User <uid> is already connected to all users in <c>' to Main Server
If <c> is not present	-

ServerB:

Initial Boot Up	The server B is up and running using UDP on port 31255
After sending country list	The server B has sent a country list to Main Server
Rest of the messages:	*Same as for the table for ServerA*

MainServer:

Initial Boot Up	The Main Server is up and running
After receiving country lists	The Main Server has received the country list from server A using UDP over port 32255 The Main Server has received the country list from server B using UDP over port 32255 Server A | Server B <country1> | <country2> ..
On receiving a query from client	The Main Server has received the request for User <uid> in Country <c> from a client using TCP over port 33255 <c> shows up in serverA The Main Server has sent request for User <uid> to serverA/B using UDP over port 30255

On receiving recommended user from backend server	Main Server has received searching result of User <uid> from serverA/B The Main Server has sent searching result to client using TCP over port 33255
If <uid> is not present in country	Main Server has received 'User <uid> not found in <c>' The Main Server has sent searching result to client using TCP over port 33255
If <uid> is already connected to all	Main Server has received 'User <uid> is already connected to all users in <c>' The Main Server has sent searching result to client using TCP over port 33255
If <c> is not present	<c> does not show up in server A&B The Main Server has sent ‘<c>: Not found' to client using TCP over port 33255

Client:

Initial Boot Up	Client is up and running Please enter the User ID: <uid> Please enter the country name: <c>
After sending <uid> and <c> to mainserver	Client has sent User <uid> and Country <c> to Main Server using TCP
On receiving the recommendation	Client has received results from Main Server: <uid> is possible friend of <uid> in <c>
If <uid> is not present in country	User <uid>not found in <c>
If <uid> is already connected to all	User <uid> is already connected to all users in <c>
If <c> is not present	Country <c> not found

Idiosyncrasies:

Main Server cannot handle multiple concurrent requests i.e two clients making the request at the exact same time. Did not resolve this issue as TA told me this scenario won't be tested and it's out of the scope of the project.

Reused code:

Used code for UDP, TCP communication from Beej's Tutorial (http://www.beej.us/guide/bgnet/pdf/bgnet_usl_c_1.pdf), page 31-38.

Have inserted comments like '// Reused code from Beej's Tutorial ' in the code files wherever relevant.

Following are the functions that contain reused code in the given files:

serverA.cpp	udp_listen_on, udp_talk_on
serverB.cpp	udp_listen_on, udp_talk_on
servermain.cpp	sigchld_handler, get_socket_fd, udp_talk_to, udp_listen_on
client.cpp	get_socket_fd