CSE 589 PROJECT 1

CODE ORGANIZATION

IMPLEMENTATION DETAILS

I have implemented 1:1 download. I was not able to implement parallel download; however I have created two functions in the file client.c

- download() – This function downloads the file from one peer
- parallelDownload() – This function downloads file from all peers
    - I was not able to test the function "parallelDownload()" due to lack of time

CODE DETAILS:

The Program is split into multiple ".c" and ".h" files:

All source code is organized into two folders, namely: *"src"* and *"include"* :

1. **src** : has all the .c files source files
2. **include**: has all the .h files

There is a makefile named as *"makefile"* which shall create an executable of name "cse589_proj1_rajaramr"

Files inside "src" folder:

1. rajaramr_proj1.c -- The main program that shall get user input and branch out as client or server based on user input
    a. This file shall be the entry point of the application; it has the "main" function
    b. This file parses the user input and comprehends if it needs to run as a server or client by looking at the user input
2. client.c -- All of the client code resides here, right from opening a listening socket to accept peer connections to opening socket to register with server and get SERVER_IP_LIST
    a. Shall keep track of all connections made to and from the application when in client mode
    b. Has the implementation of following functions among others:
        i. client_shutdown() – corresponding to "EXIT" command. This function shuts down all sockets and closes all socket file descriptors
        ii. registerWithServer() – corresponding to "REGISTER" command, registers with the server and passes on the listening port of the application
        iii. connectToPeer() – corresponding to "CONNECT" command
        iv. download() – corresponding to "DOWNLOAD" command. ***This function was used in getting Data for analysis***
        v. parallelDownload()– This function also corresponds to "download", but shall attempt to download the file parallel from all peers
        vi. client_terminateConnection() – This function corresponds to the "TERMINATE" command, given a connection id this command tries to terminate the same
        vii. displayPeerList() – This function shall be used to display on the console the pee list sent by the server
        viii. getSocket() – this function returns a TCP socket file descriptor when called
3. server.c -- All the server code resides here, this is where the server shall setup shop to accept incoming registration request and also does send out serverIPList to registered clients

a. Shall keep track of all connections made to the application when in server mode
b. Has the implementation of following functions:
    i. server_shutdown() – Shuts down the server closes all current connections
    ii. server_terminate() – Terminates a connection, user provides details of the connection
    iii. sendListOfPeers() – Sends a list of the peers to the clients connected to the server

4. IPListOperations.c -- This file shall contain all the functions necessary to maintain a Linked list that shall have a given connections IP address, port number, hostname and socket file descriptors associated with the connection
    a. Has the implementation of the following functions:
        i. int addToIPList: Adds a connection entity to IPList maintained
        ii. removeFromIPList: Removes a connection entity from the IPList maintained
        iii. displayIPList: Displays the contents of the IPList (Hostname / IPAddress / Port)
        iv. freeIPList: Frees the IPList
        v. getMaxFD: When called shall return the value if the biggest Socket file descriptor among the connections that have been made
5. commandOperations.c -- This file shall contain modules that can handle commandline input, parse them and make sense of the same. Handles command-line requests responds appropriately
    a. commandMaster(): this function helps parse the command line inputs
    b. help(): This function displays the list of valid commands that the application shall accept from command line
    c. creator(): Prints the name, UBITname, and email address of RAJARAM RABINDRANATH

Files inside include folder:

The “include” folder only contains .h files, the following:

1. appMacros.h – has application level macros
    a. MAX_CONNECTIONS_CLIENT – Number of connections accepted by client
    b. MAX_CONNECTIONS_SERVER – Number of connections accepted by server
    c. SOCKET_REUSE_POLICY – Application’s socket reuse policy
    d. MST_MTU -- Maximum message size

2. error.h – has application level error macros
    a. TRUE
    b. FALSE
    c. SUCCESS
    d. FAILURE

3. globalvars.h – Holds certain global variables
    a. MYIP – Shall hold IP of the current machine
    b. MYPORT – Shall hold the Listening port of the current machine
    c. isServer – Shall hold a true/false info about the applicaton mode
    d. CommandArgs – for easy transfer of parsed STDIN input from user

4. client.h – Has client macros and one function prototype that client.c share with rest of the files in the application
5. commandOperations.h – holds all the function prototypes declared within the funcftion commandOperations.c
6. IPList.h -- holds the template for the structure IPList
7. IPListOperations.h -- holds the function prototypes of the IPListOperations.c file
8. server.h -- holds the function prototypes of the server.c file