COMP9331 Assignment 1

Implementation of Peer-to-Peer Network using Distributed Hash Table

Name：Yu Zhang

Student ID: z5238743

1. Environment

Development environment: Python 3.7 on mac

Test environment: Python 3.7 on CES VLab

2.Program Design

UDPClient Class contains Thread and UDP socket. The aim is to provide main function to use UDP socket to ping UDPServers.

TCPClient Class contains Thread and TCP socket. The aim is to provide main function to use UDP socket to ping TCPServers.

UDPServer Class contain Thread, UDP socket, time and 2 Protocol. The one is *“ping request”* and the other is “*ping respond”.* UDPServer are used for Part of step 1- Initialization.

class ValueError and class TypeError are provided for input error.

TCPServer Class contain Thread, UDP socket, time and 8 Protocol. The first protocol is *“REQUEST_LOST_PEER”.* The second protocol is *"QUIT".* The third protocol is *"Peer_Join".* The fourth protocol is *"REQUEST_PEER_CHANGE".* The fifth protocol is "*REQUEST_PEER_ACCEPTED*". The sixth protocol is *"REQUEST_STORE".* The seventh is *"RESPONSE_FILE".* The eighth is *"REQUEST_FILE".* Step 4 - Peer Departure(Graceful) use TCP and Step 5- Peer Departure(Abrupt) to check use TCP to check loss. Besides, Step 6 - Data Insertion and Step 7 - Data Retrieval use TCP to Store and Request. Step 3 – Peer Joining use TCP to join peer.

Peer Class provide 4 attributes, Peer’s ID, port of peer, IP address of peer and Peer’s TCPServer successful receiving sequence.

Initalization Class contain 8 main functions to execute program. *begin_show* function, *Peer_Joining* function, *ping_tcp_receiver* function, *ping_udp_receiver* function, *check_alive* function, p*ing_successors_thread* function and *check_input* function.

The main recall Initalization Class to execute program.

3.Program Description

The program, developed by Python 3.6, was designed to implement a simple p2p application. 8 class and a main function. Main function recall 8 class to complete step 1, step 2, step 3, step 4, step 5, step 6, step 7.

In step 1- Initialization, main function recall Initalization class to execute Peer class and use Peer’s ID, port of peer, IP address of peer to print Initiazation

In step 2-ping Successor, main function recall Initalization class to execute Peer class and peer initiate a UDPClient and send *"ping request" Protocol* to ping first successor's UDPServer and second successor's UDPServer. In UDPServer, when first successor's UDPServer and second successor's UDPServer receive *"ping request" Protocol* and record first predesuccessor and second predesuccessor, then send *"ping response" Protocol* send to peer, when peer receive *"ping response" Protocol and record the* successful receiving sequence and *print* ping response.

In step 3-Peer Join, main function recall part of Initalization class Peer to execute Peer class and peer initiate a TCPClient and send *"Peer_Join" Protocol* and judge which peer to join then in TCPServer, the peer found send *"REQUEST_PEER_CHANGE" Protocol and "REQUEST_PEER_ACCEPTED" Protocol* to the already-exist Peer and the already-exist change first successor and second successor and Peer Join

In step 4 and step 5, Peer Departure(Graceful) and Step 5- Peer Departure(Abrupt) both use TCPClient, in step 4, peer use *"QUIT" Protocol* to first successor and second successor then change it in first successor and second successor's TCPServer. In step 5, peer use "*REQUEST_LOST_PEER*" *Protocol*, then in TCPServer detect the successful receiving sequence. If it loss some sequences. It will be lose.

In step 6 and step 7, Peer both use TCPClient, in step 6, peer use "*REQUEST_STORE*" *Protocol* to ping first successor to choose which peer to store message
in step 7, peer use "*REQUEST_STORE" Protocol to ping* first successor to choose which peer to Request message and send *"RESPONSE_FILE" Protocol* to send message to the peer.

3.Message Formats:

Different message use different List contain different peer information.

Use pickle to decode information and encode information.

5. Design Tradeoffs

1. program don't contain STOP and WAIT
2. some burden loop

4.Possible Improvements

1. Process of program is Clear
2. Combining both TCP and UDP servers into on Thread will be a better method
3. Drease burden on this process