Programming Assignment 2: BitTorrent & Event-Driven Simulator (Part 2 - Implementation)

Nahian Tasnim: 10.5 hours. Started on March 18th

Description of the overall design:

My p2p event simulator contains all of the major components of a p2p network which can work together to maintain smooth event simulation in a BitTorrent protocol. The core of the simulator has event, event queue and simulator classes. Event class presented us with the scheduled events. There is a priority queue in the EventQueue class ensures that all the scheduled events are processed and managed in a chronological order. The Simulator class handles the global timer, coordinates the works of all the peers. It also schedules events and dispatches them. The network part consists of Client, Server, Peer and Message classes. In my system all the peers can act as client and server both. There is two types of peers. Seed has all the pieces of the file while leechers have some random pieces of the file. My system selects first peer as the seed and assign all the pieces of the file to it. Each peer can connect with other peers. The MessageType class has various types of message and Message class keeps track of all specific message types and executes next instructions according to those types. The Configs class parses the configuration files and provides access to the common parameters for the simulation. The FileHandler class is responsible for reading and writing the file pieces correctly to avoid error. The logging for peers happens within the P2PLog class which tracks their activities and gives an overview of the progress of the simulation. The Main class initializes the configuration, creates peers, sets up event queue and simulator. It also schedules initial events, runs the whole simulation, collects, saves and plots the results.

Execution flow:

After running the execution command in the prompt, the simulation executes in the following procedures one after another –

1. Simulator initializes the configuration using the Config class.
2. All the peers are created based on the parsed information (Both seed and leechers).
3. Then event queue is set up and the simulator initializes it.
4. The simulator class starts processing events in order.
5. Events are scheduled and dispatched to the peers and the peers handle them according to their condition.
6. The peers starts establishing connections with other peers using tit-for-tat and optimistic unchoke.
7. After connecting the peers find out the rarest piece and request for it to other connected peers.

8. Establishing connection with peers and piece exchange continues until all the peers get and downloads all the pieces.
9. This simulation continues until the queue generating events is empty. Then the simulation gets terminated.

Simulation Results and Analysis:

Methodology:

- I conducted the experiment with 5 peers, 7 peers, 9 peers and 10 peers. We can change the information in the peer_info.cfg file to vary the number of peers.
- I used SubLime Text to write my code and run the simulation. My SubLime text was already configured with all the plugins needed to use it like a python IDE. For common configuration piece size was 1024 and max connections was 5. The peer_info.cfg contains peer id, host name and port number parameter. I changed the values here to experiment with varied number of peers.
- Generative AI tools were used for assisting in debugging my code and analyzing errors to ensure accuracy and efficiency.

Graphs analysis:

I have experimented the simulation with 5 peers, 7 peers, 9 peers and 10 peers. I have included all the csv files and graphs in a separate folder.

5 peers result:

The time completion time vs total number of peer shows a relatively low completion time showcasing efficient peer interactions. The number of active peer gradually reduced to 1 from 5 in the last minute which shows that peers left after completing their task. The active peer becomes 0 at the time of the connection termination The average download speed was 19.25 while the average upload speed was 10.82. The completion time vs average speed graph visualize the impact of high download and upload speed on quick completion time.

7 peers result:

The time completion time vs total number of peer for 7 peers also shows a relatively low completion time however it’s higher than 5 peers completion time showcasing the impact of enhanced peer number. The graph of active peer vs time fluctuates a lot but gradually goes down to 1 at the last minute and becomes 0 when the simulation terminates. The average download and upload speed are 18 and 9 respectively.

9 peers result:

The file completion time for 9 peers is approximately 10.76 seconds which is less than 7 peers result even with higher peer number. The active peer vs time graph has some fluctuation. The

average speed vs time graph shows that due to having high downloading and uploading speed, even with high number of peers the completion time is still pretty low.

10 peers result:

The File colpetion time for 10 peers was 11.14 sec (approx.). The number of active peer started with 8 and the graph frequently fluctuated, but at least dropped to 0 before the simulation termination. The completion time vs speeds graph show a little lower downloading and uploading speed than 9 peers simulation resulting in higher completion time.

Overall, The increase in number of peers increases the completion time in most of the cases. However completion time can vary for other reasons also.