Computer Networks Course (CNT 5106C) Project
P2P File Sharing

Project Details
Project Members:
Project Group
1. Venkat Maneesh Reddy Konkala (UFID: 7637 - 9205)
2. Yallamandaiah Nandigam (UFID: 2945 - 1349)
3. Satya Naga Akhilesh Irrinki (UFID: 6473 - 0546)

Objective:

Objective of this project is to write a P2P file sharing software similar to BitTorrent. We have used Java Programming language for this project. We have used a choking and unchoking mechanism to distribute files between peers.

Protocol:
1. In this project, We have established connections between peers using reliable TCP protocol. File sharing will be done between the peers.
2. For initiation of file sharing. Peers first send a handshake message to each other. Handshake message consists of 18 byte Handshake Header, 10 byte Zero Bits and 4 byte Peer ID.
3. After handshaking, each peer can send a stream of actual messages which consists of 4 byte message length, 1 byte message type and message payload with variable size.
4. There are different types of payloads. The type of messages are have, bitfield, choke, unchoke, interested, not interested, request and piece.

Working:

1. In the PeerInfo config file, we have specified the order for peers. These are started by startRemotePeers in the order specified in Peerinfo file

2. Peer ID is the parameter in the peerProcess

3. If a peer is started then it should make TCP connection with the peer that already has been started earlier

4. Also, each peer reads the common configuration file, which provides information about the shared file and its size, choking and unchoking intervals, and no of preferred neighbors.

5. PeerInfo file bits 0 means it does not have any file details. PeerInfo.cfg file is modified with bit 1, once after the peer receives the entire file

6. For the initial peer that has been started, it would not be connected to any other peer as it is the first peer. It just listens to the port defined in the PeerInfo file

7. For every peer, if they establish a TCP connection to another peer or they change their preferred neighbor, we maintain log for each transaction

8. Log also will be maintained when peer change their optimistically unchoked neighbor or when they are choked or unchoked by another peer and also when they finished downloading a piece or the complete file.

File Sharing:

Whenever a peer in the network needs a file, the peer tries to search for the file among its neighboring peers using the filename if its available or either keyword.

1. To search for the file, initially the peer network requests for the file in the overlay network and searches for the file from the peers in the network which are at the distance of less than or equal to hop count and finally if the number of seconds are greater than hop count than the search request expires and duplicate searches are also not allowed at same time.
2. After that if the required file is found in any one of the peers then the peer containing the file sends a response back to the peer requested for the search. Once the peer that requested the file receives the response it can either consume the response or it can transfer the response to the peer it actually received the request from.
3. The requester would not consider and would completely ignore any more requests that are received after the request that is requested by the peer is expired.
4. After the response is received from the peer, the requested peer selects the peer which has the matching filename and index of the file and then tries to establish a TCP connection with that selected peer.
5. After the TCP connection is established, the requested peer then transfers the file into it’s directory and does all the required updates and finally once the job is done the TCP connection is terminated.
6. In case if the file is not found and the search request has failed, then the peer should try to search for the file again within the overlay network and this time the peer will increase the hop count by 1.
7. If the file is still not yet found then the hop count is again increased more either until the search request is found or till the hop count exceeds the pre-defined hop count number.

Steps to Run the project:
1. Add tree.jpg file in 1001 folder and add Common.cfg and PeerInfo.cfg as needed
2. Run make command to compile the code
2. run start remote peers and provide username and password to start peers in servers.

To Run process locally in intellij:
1. Go to peerProcess class and run main method
2. Open peerProcess configuration and in program arguments add some peer id
   and run peerProcess configuration

Contributions: Everyone has contributed equally to the project
Venkat Maneesh Reddy Konkala

- EstablishConnectionsWithPeers.java
- PeerConnector.java
- PeerProcess.java
- PeerUnchoker.java
- OptimisticPeerUnchoker.java
- FetchPeerInfo.java
- MessageExchanger.java
- MessageTypes.java

Yallamandaiah Nandigam

- Constants.java
- MessageTypes.java
- Logger.java
- Peer.java
- Makefile
- PeerUnchoker.java
- Props.java
- PeerProcess.java
- MessageExchanger.java
-

Satya Naga Akhilesh Irrinki

- FetchPeerInfo.java
- Messages.java

- PeerInfo.cfg
- Utils.java
- Setup.java
- PeerProcess.java
- OptimisticPeerUnchoker.java
- MessageExchanger.java
- MessageTypes.java

Video URL: