Networks Notes

jr2fh

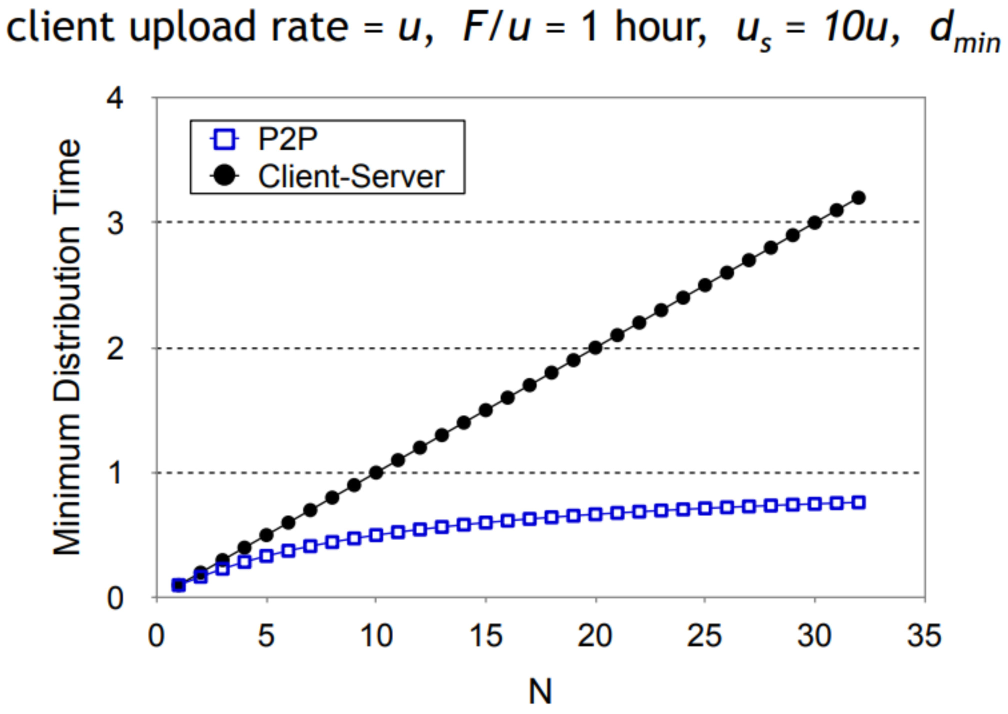
April 7th 2020

1 Peer to Peer Architecture

- no server with 100% uptime.
- peers request and provide service to other peers
- as a result of the lack of a central server p2p is self-scaling
- peers don't have constant ip addresses, which makes management and tracking a task
- ex: bittorrent, skype, kankan

2 P2P v.s. client-server

- Client Server:
- file transfer in client-server is limited by server upload capacity, and client download capacity
- c-s time to distribute: $D_{c-s} \geq \max\{NF/u_s, F/d_{min}\}$
- increases linearly with the number of clients
- Peer to Peer:
- P2P file transfer is limited by the peer download and upload capacity
- Server need only upload one copy
- p2p time to distribute: $D_{p2p} \geq \max\{F/u_s, F/d_{min}, NF/(u_s + \Sigma u_i)\}$



3 Distributed Hash Table

- distributes data over millions of peers evenly
- any peer can query the database
- peers know only a small number of other peers
- robust to peers coming and going
- peers query their connected peers to resolve query.
- assign the data to the peer with the closes hash id

3.1 Circular DHT

- each peer is only connected to its immediate successor and predecessor
- O(n) query resolve time
- add shortcuts to reduce time to minimally O(logn)
- each peer periodically pings its connected peers to check that they are alive, if one dies then choose the next to replace it.

4 P2P file sharing: BitTorrent

- register with a tracker to get a list of peers, and then connect to neighbors
- while downloading use upload to distribute chunks to other peers
- peer may change those it serves
- bittorrent has churn
- once the peer has recieved an entire file it can choose to leave (selfish), or remain in torrent (altruistic).

4.1 How it works

- make a GET request to announce that you're joining and to get peers
- start a tcp connection to uploading peer, exchange messages to download pieces
- can send or request different pieces to/from different peers at the same time
- peer to send to first is chosen randomly out of those who have requested
- peers develop preference based on how fast file transfers are completed via different peers

5 Video Streaming and CDNs

- most of internet bandwidth is video streaming
- it is a challenge to scale for billions of users with different capabilities
- we solve this with distributed application-level infrastructure
- we can use redundancy between frames to send less information per frame for the same video