P2P file distribution and BitTorrent

Group - 18

- Aditya Pote (2020IMT-069)
- Sameer (2020IMT-086)
- Suyash Vikram Singh (2020IMT-104)

Scenario

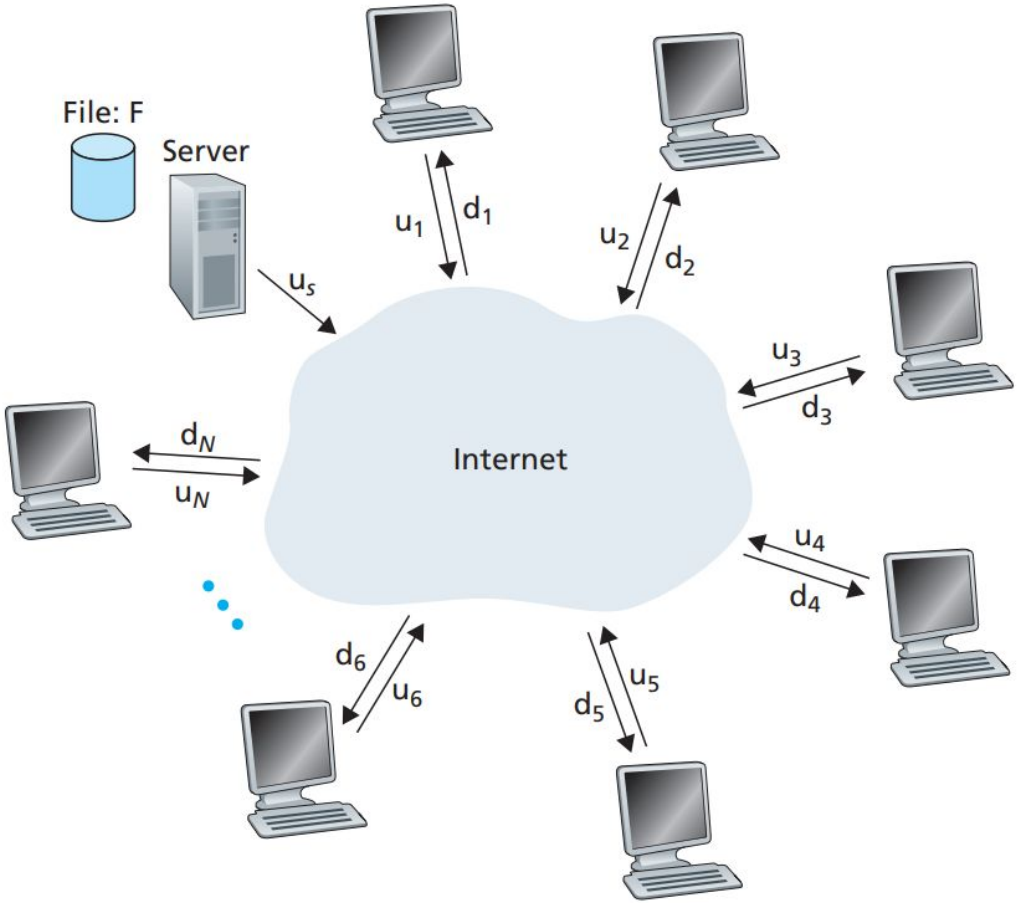
u_s = server upload speed

u_i = i^{th} host upload speed

d_i = i^{th} host download speed

F = file size

N = # of peers



File sharing using Client-Server architecture

- Time taken to upload N copies of file by the server = $\frac{NF}{u_s}$
- Bottleneck at client's end = $\frac{F}{d_{min}}$ $\quad d_{min} = min(d_1, d_2, ..., d_N)$
- Combing above 2 equations = $\left(D_{cs} \geq \max\left\{ \frac{NF}{u_s}, \frac{F}{d_{min}} \right\} \right)$

File sharing using P2P architecture

- Time taken to upload 1 copy of file by the server = $\frac{F}{u_s}$
- Bottleneck at client's end = $\frac{F}{d_{min}}$ $\quad d_{min} = min(d_1, d_2, ..., d_N)$
- The system must upload NF bits so that all peers can download the file. Assuming simultaneous upload, minimum time taken is at least = $\frac{NF}{u_s + \sum_1^N u_i}$

File sharing using P2P architecture

Putting together the previous 3 observations, we get as

$$D_{\text{P2P}} \geq \max\left\{\frac{F}{u_s}, \frac{F}{d_{min}}, \frac{NF}{u_s + \sum_{i=1}^{N} u_i}\right\}$$

Comparison between CS and P2P architectures

Assumptions: $d_{min} >= u_s \mid u_i = u \ \forall \ i \mid u_s = 15u$

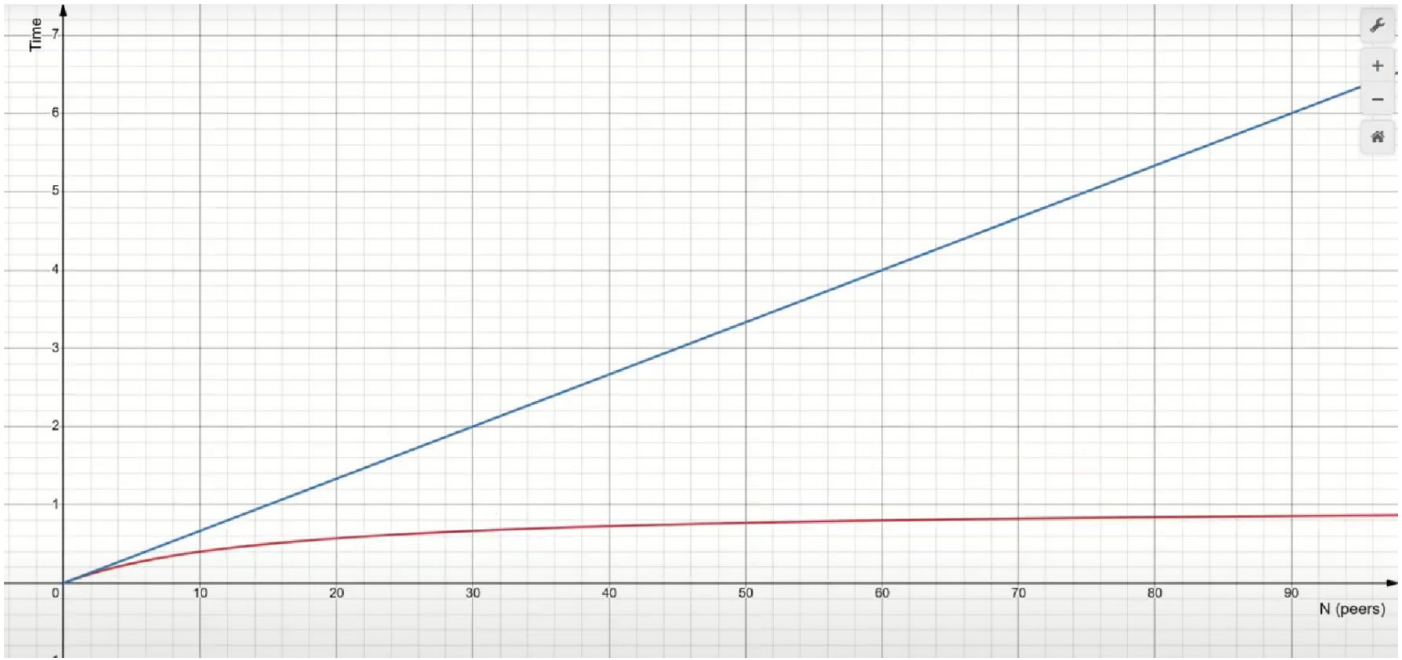
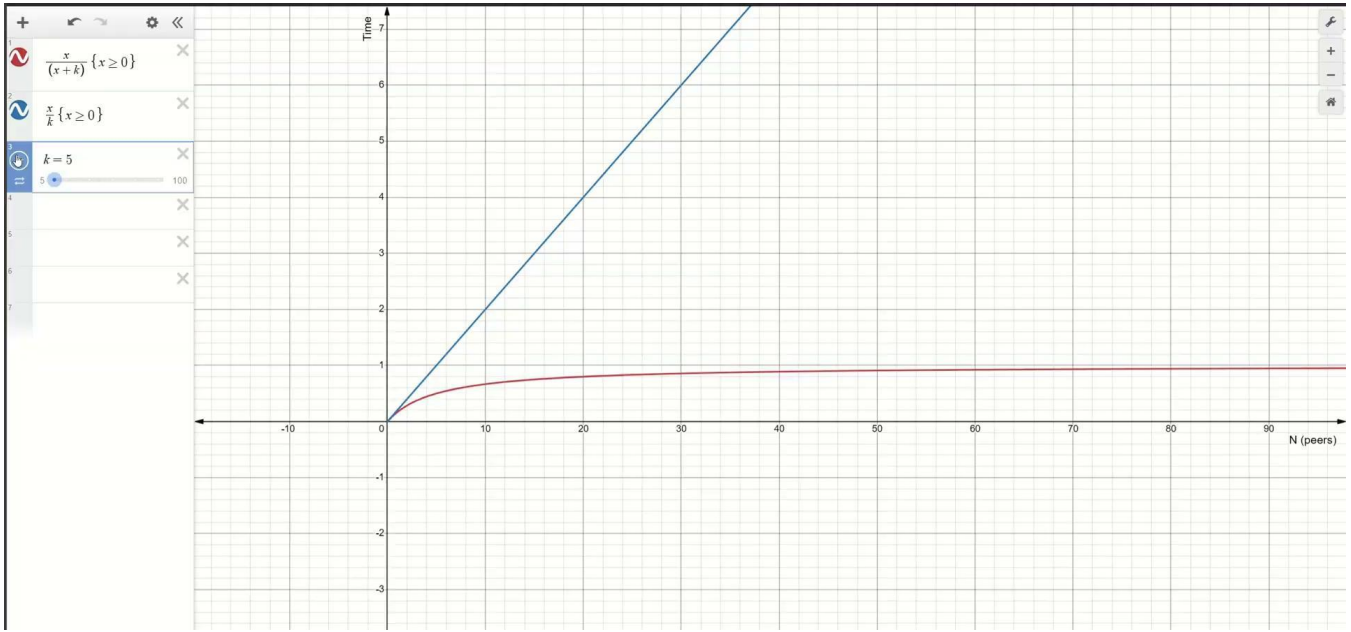


Fig. Distribution time for P2P and client-server architectures

Comparison between CS and P2P architectures

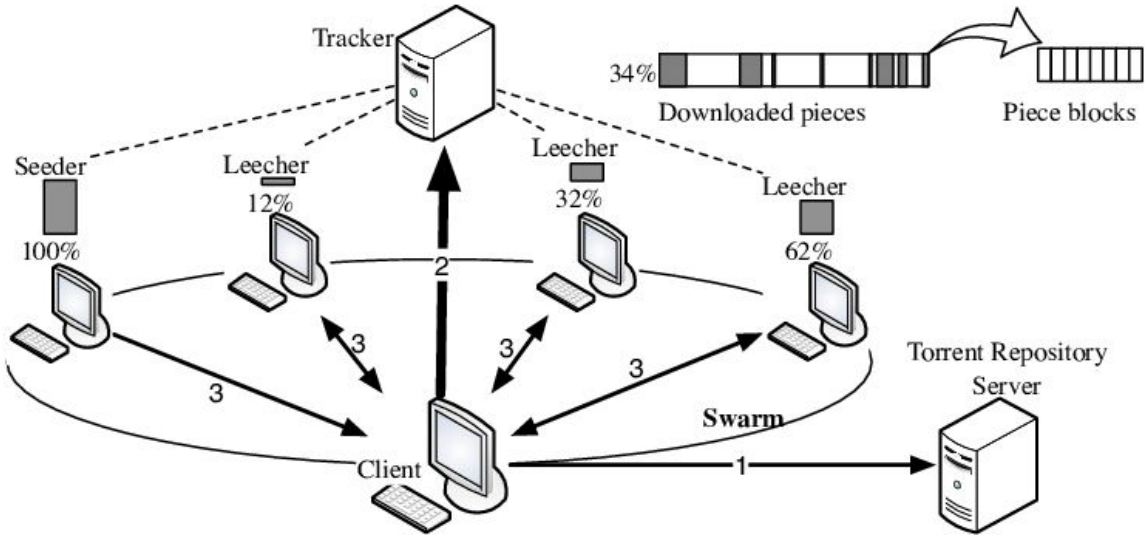
Assumptions: $d_{min} >= u_s$ | $u_i = u \ \forall \ i$ | $u_s = ku$



Vid. Distribution time for P2P and client-server architectures

BitTorrent

BitTorrent is a communication protocol for peer-to-peer file sharing (P2P)



BitTorrent features

- The file being distributed is divided into segments called **pieces**. Each piece is protected by a cryptographic hash
- Pieces are typically downloaded non-sequentially, and are rearranged into the correct order by the BitTorrent client.
- **BitTorrent trackers** provide a list of files available for transfer and allow the client to find peer users, known as "seeds", who may transfer the files.

uTorrent Transport Protocol

Traditional TCP connections used by BitTorrent distribute bandwidth evenly across all connections, giving applications with multiple connections, like BitTorrent, an unfair bandwidth advantage.

uTP solves these problems by adjusting its send rate based on the router's queue size. When the queue gets too large, indicating network congestion, uTP reduces its send rate.

uTP operates over UDP and implements its own congestion control mechanisms, differing from TCP primarily in its use of delay-based congestion control rather than packet loss as a signal for congestion.

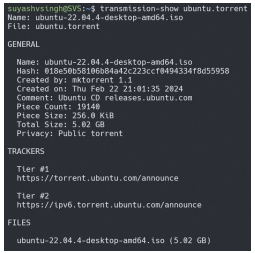
Working of BitTorrent

Repository

Download .torrent file

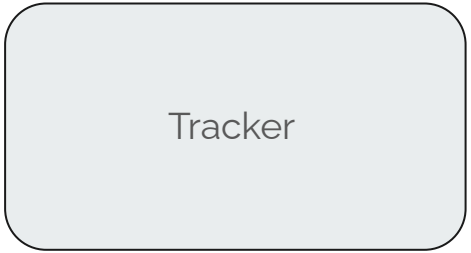
```
suyashvsingh@SVS:~$ transmission-show ubuntu.torrent
Name: ubuntu-22.04.4-desktop-amd64.iso
File: ubuntu.torrent

GENERAL

  Name: ubuntu-22.04.4-desktop-amd64.iso
  Hash: 018e50b58106b84a42c223ccf0494334f8d55958
  Created by: mktorrent 1.1
  Created on: Thu Feb 22 21:01:35 2024
  Comment: Ubuntu CD releases.ubuntu.com
  Piece Count: 19140
  Piece Size: 256.0 KiB
  Total Size: 5.02 GB
  Privacy: Public torrent

TRACKERS

  Tier #1
  https://torrent.ubuntu.com/announce

  Tier #2
  https://ipv6.torrent.ubuntu.com/announce

FILES

  ubuntu-22.04.4-desktop-amd64.iso (5.02 GB)
```

Working of BitTorrent

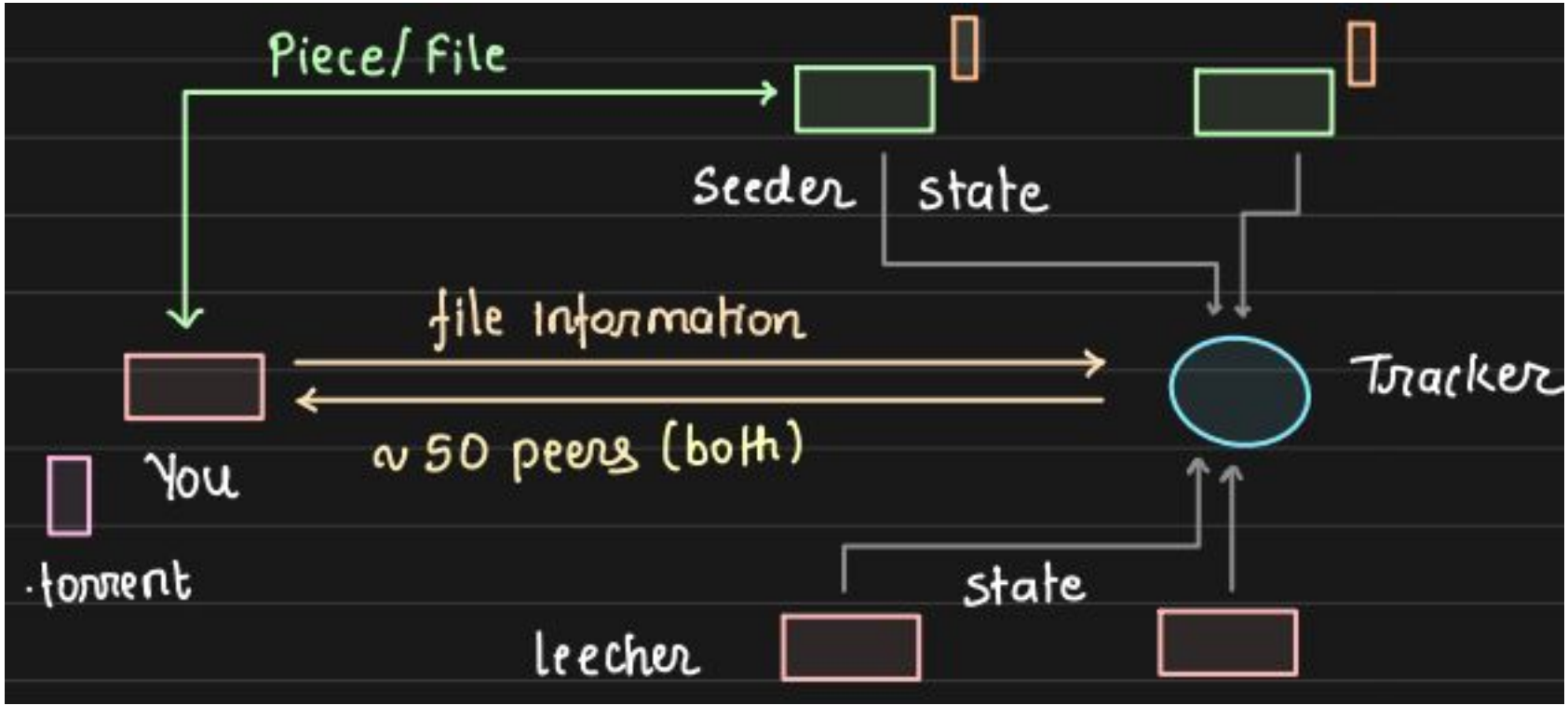


.torrent file

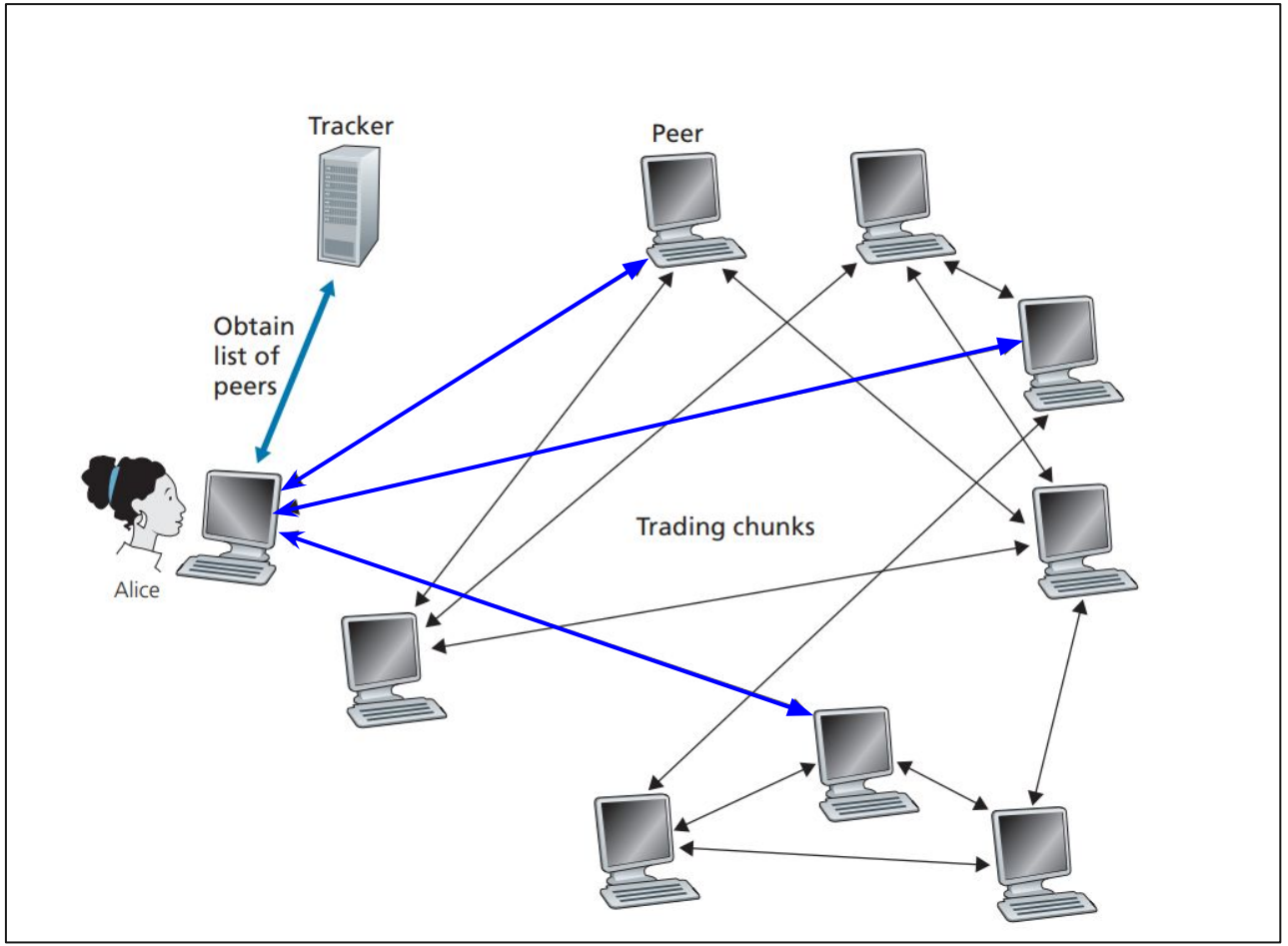
BitTorrent client, like µTorrent



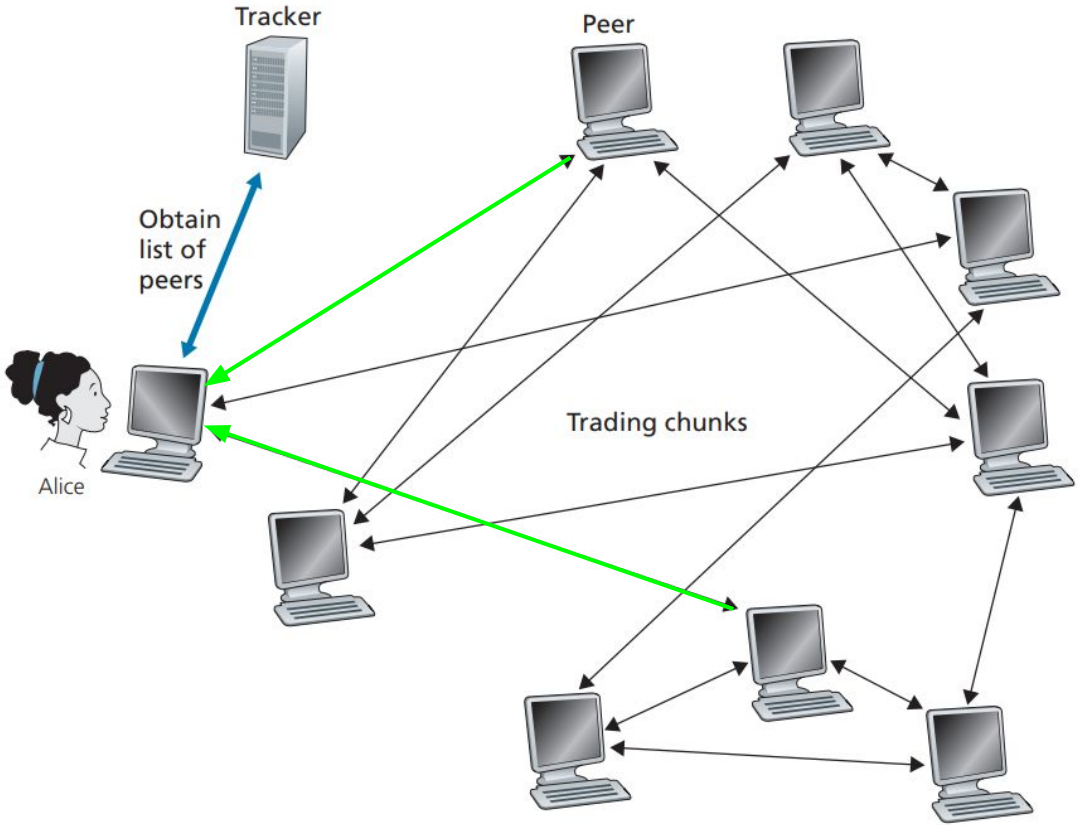
Working of BitTorrent



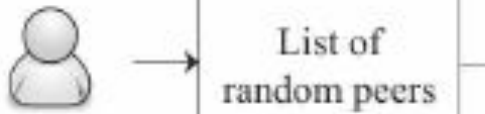
Attempts to establish concurrent TCP connections with all the peers on this list

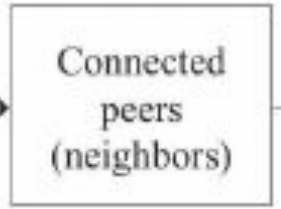


Succeeds in establishing a TCP connection with some of them



Each Connected peer will have a subset of chunks from the file, with different peers having different subsets. Periodically, it will ask each of her neighboring peers for the list of the chunks they have.





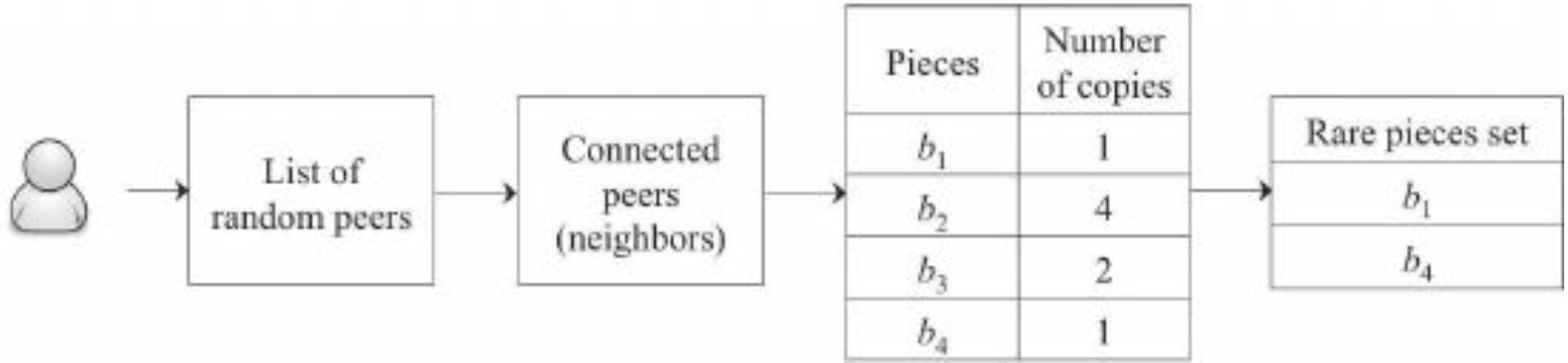
Pieces	Number of copies
b_1	1
b_2	4
b_3	2
b_4	1

Two questions arises

First, which chunks should be requested first from connected peers? And

Second, to which of her neighbors should it send requested chunks?

Rarest first Algorithm

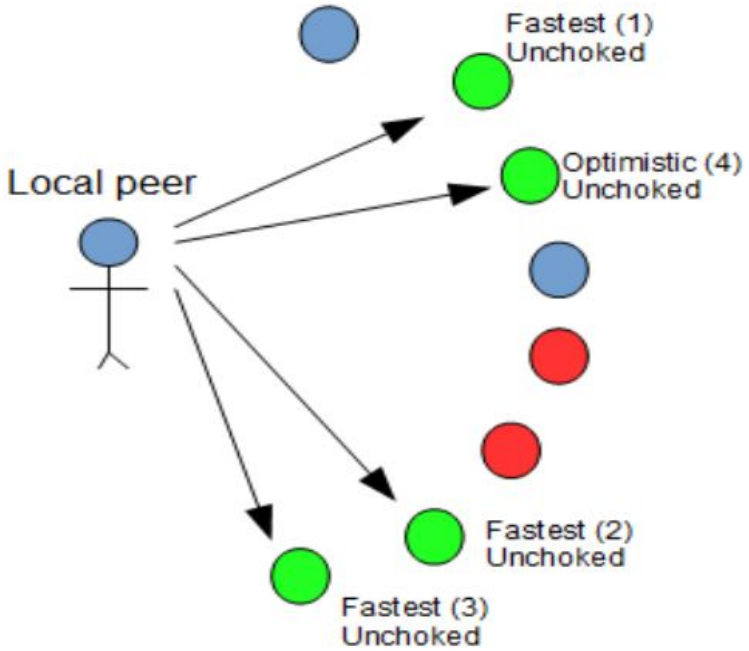


In this manner, the rarest chunks get more quickly redistributed, aiming to (roughly) equalize the numbers of copies of each chunk in the torrent.

Unchoked Peers

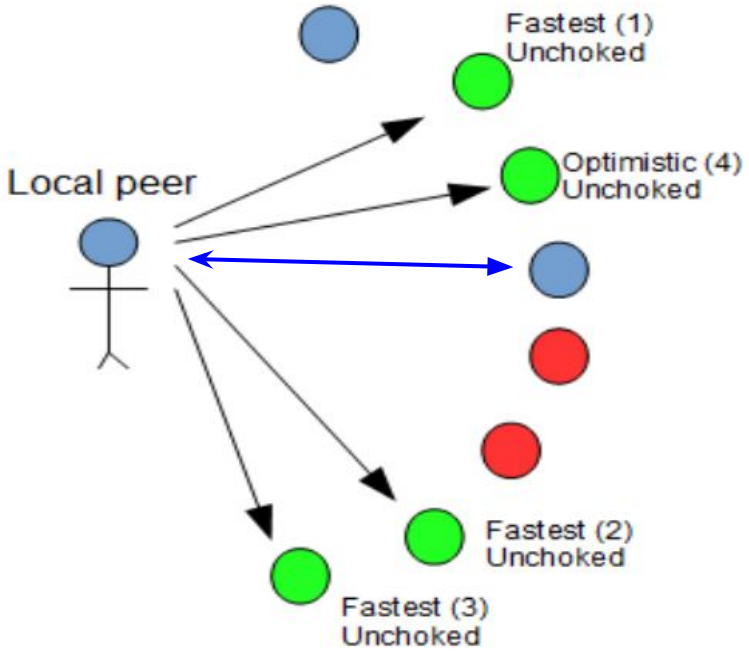
It will priorities to the peers that are currently supplying it data at the highest rate. Specifically, for each of her peer, Alice continually measures the rate at which she receives bits and determines the four peers that are feeding her bits at the highest rate

Every 10 seconds, she recalculates the rates and possibly modifies the set of four peers.



Optimistically unchoked

Every 30 seconds, she also picks one additional neighbor at random and sends it chunks, a practice known as Optimistically unchoking. This enables both peers, if compatible, to become top uploaders for each other, enhancing the exchange efficiency and integrating new peers into the trading ecosystem.



Disadvantages torrent

Copyright Risks: Torrents facilitate the unauthorized distribution of copyrighted content, posing legal risks to users and financial harm to creators.

Lack of Anonymity: Users' activities on P2P networks are easily traceable, exposing them to surveillance and potential legal repercussions without the protection of anonymity.

Security Vulnerabilities: Torrents can be a vector for malware, risking device security, while efforts to remain anonymous, like using VPNs, may face limitations or reduce network performance.

Thank You!!