<u>Communication Networks</u>
<u>Project 1</u>

Shivani Chepuri 2018122004
Gowri Lekshmy 20171053

Modeling and Performance Analysis of BitTorrent-Like Peer-to-Peer Networks

Objectives:

- Studying and Understanding (the research paper) “**Modeling and Performance Analysis of BitTorrent-Like Peer-to-Peer Networks**”.
- Literature Survey (on P2P networks and BitTorrents) within the scope of the main paper.

Main paper link: http://conferences.sigcomm.org/sigcomm/2004/papers/p444-qiu1.pdf
Main Paper Title: **Modeling and Performance Analysis of BitTorrent-Like Peer-to-Peer Networks**

Literature Survey Outline and Scope:

In regard to the topics and insights mentioned in the main research paper, we want to explore more papers on BitTorrent and P2P networks, keeping the scope limited to the ideas in the main research paper.
(survey papers/links are mentioned in the References section)

Abstract of the paper:

- Develop simple models to study the performance of BitTorrent
- Present a simple fluid model and study the scalability, performance, and efficiency of such a file-sharing mechanism and Steady-State network performance.
- Consider the built-in incentive mechanism of BitTorrent and study its effect on network performance.

Approach and Workflow:

- Understand P2P networks, their architecture, and their mechanism.
- Study and understand the terminology, architecture, and working of a BitTorrent.
- Understand a simple Fluid Model and Steady-state performance
- Explore various file-sharing mechanisms and algorithms in BitTorrent or P2P networks.
- Explore Free-Riding and Optimal Unchoking in BitTorrent or P2P networks.

Important Terms:

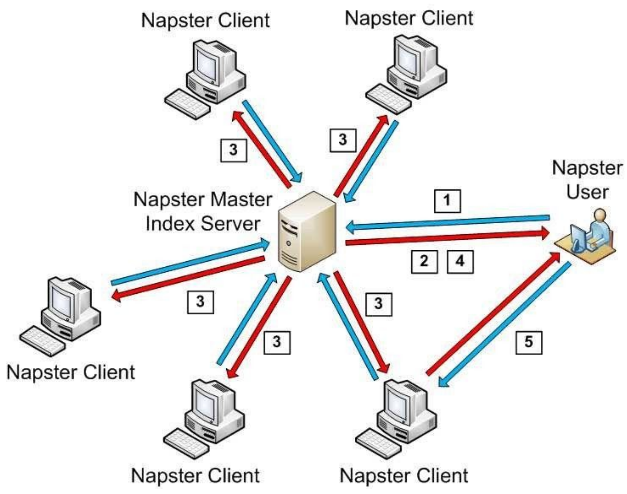
Peers: Equally privileged, equipotent participants in the application
Client Peer: Peer that is sending a request for a file/object
Server Peer: Peer that is accepting a request for a file/object, from a Client Peer
Torrent: In BitTorrent, a file is referred to as a torrent.

Swarm: The set of all peers that takes part in a torrent
Seed: A peer in a swarm that has the complete content file
Downloader/Leech: A peer that has only part of the file and wants to download the rest
Swarm = Seeds + Leeches
Tracker: A central node that tracks the operation of the swarm

<u>Overview of Peer-to-Peer Networks</u>

P2P Networks:

- P2P Computing or Networking is a distributed application architecture that partitions tasks or workloads between peers.
- It is a file-sharing technology, allowing users to access mainly multimedia files like videos, music, e-books, games, etc. The peers (individual users) request for the files from other peers by establishing TCP or UDP connections.
- **Requirements to join a P2P network:** Internet Connection and P2P software.
- **Examples of P2P Software:** BitTorrent, Gnutella, Freenet, BearShare, Soulseek.
- **Evolution of P2P networks:** Napster's P2P file-sharing - Oldest P2P file-sharing protocol, used for downloading .mp3 files. It is based on a centralized architecture. Later, Decentralized or pure P2P structures came up.



P2P File Sharing:

- Peers in the network make their files to share, available to the rest of their peers.
- An interested peer can connect to the peer with the file and download it and make it available for other peers. As more peers join and download that file, more copies of the file become available to the group.
- Since lists of peers may grow and shrink, the paradigm needs to keep track of loyal peers and the location of the files.
- The underlying architecture supported by the P2P system can get the IP addresses of all those peers, who have a copy of the requested file.
- Three such architectures exist, viz, Pure (full decentralization), Hybrid(Partial Decentralization) and Centralized.

Advantages of P2P Networks:

- Easy to set up and maintain, as each computer on the network is equipped to maintain itself.
- Eliminates the extra cost to set up and maintain a server
- Each device is a master of its own and independent of other computer operations.

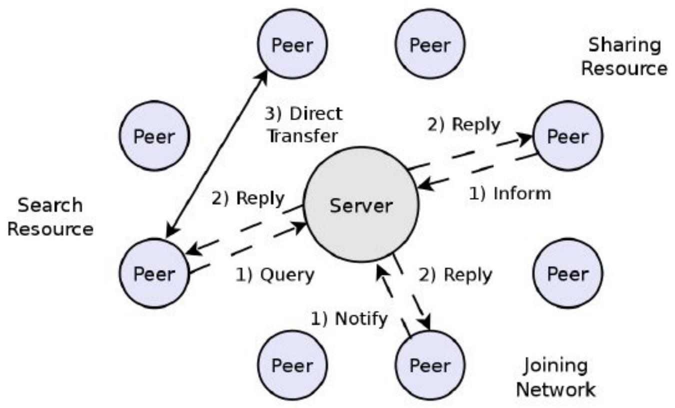
Disadvantages of P2P Networks:

- The absence of a centralized server makes it difficult to backup data as data is located on different workstations.
- There is no central point of data storage for file archiving.
- Security is weak as each system manages itself only.

P2P Architecture:

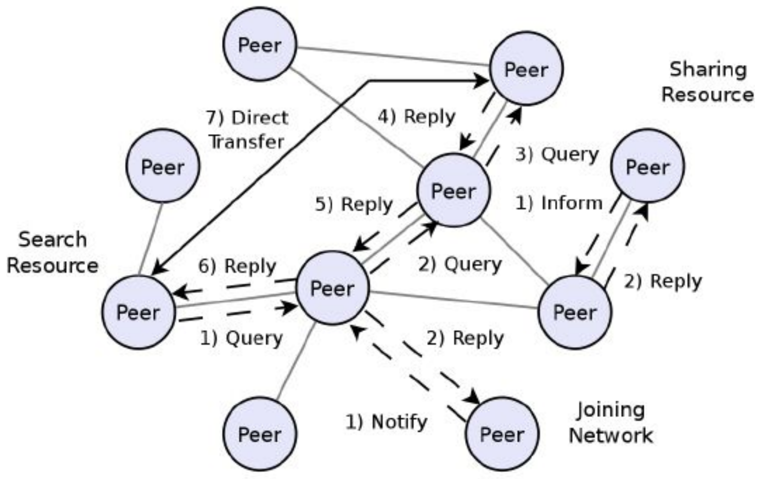
- Peers(can act as both client and server simultaneously) share both hardware and software resources, through a distributed architecture, without going through a separate server computer (unlike the server-client network) for processing requests.

1. **Centralized P2P (Hybrid P2P) Network:**
   The directory system (listing of the peers and what they offer) used a client-server paradigm. Storing and downloading of the files are done using the peer-to-peer paradigm. Eg: Napster(shown above). Diagram of a Fully Centralized P2P architecture:



2. **Decentralized Network:**
   Peers are arranged into an overlay network (A logical network above the physical network).
   Depending on how the nodes in the overlay network are linked, a decentralized P2P network is classified as either **unstructured** (nodes linked randomly. Eg. Gnutella, Freenet, etc) or **structured** (predefined set of rules to link nodes, Uses Distributed Hash Table. Eg. BitTorrent).

   Diagram of a Fully Centralized P2P architecture:



BitTorrent:

- A P2P protocol for sharing large files among a set of peers called a swarm.
- File sharing is done in a collaborating process called a **torrent**.
- A BitTorrent can be **implemented with a tracker being centralized or in a decentralized** fashion where a tracker's functions are distributed among the nodes.
- A downloader can upload data to other peers even though it may only have parts of a file (η parameter, the effectiveness of file sharing)

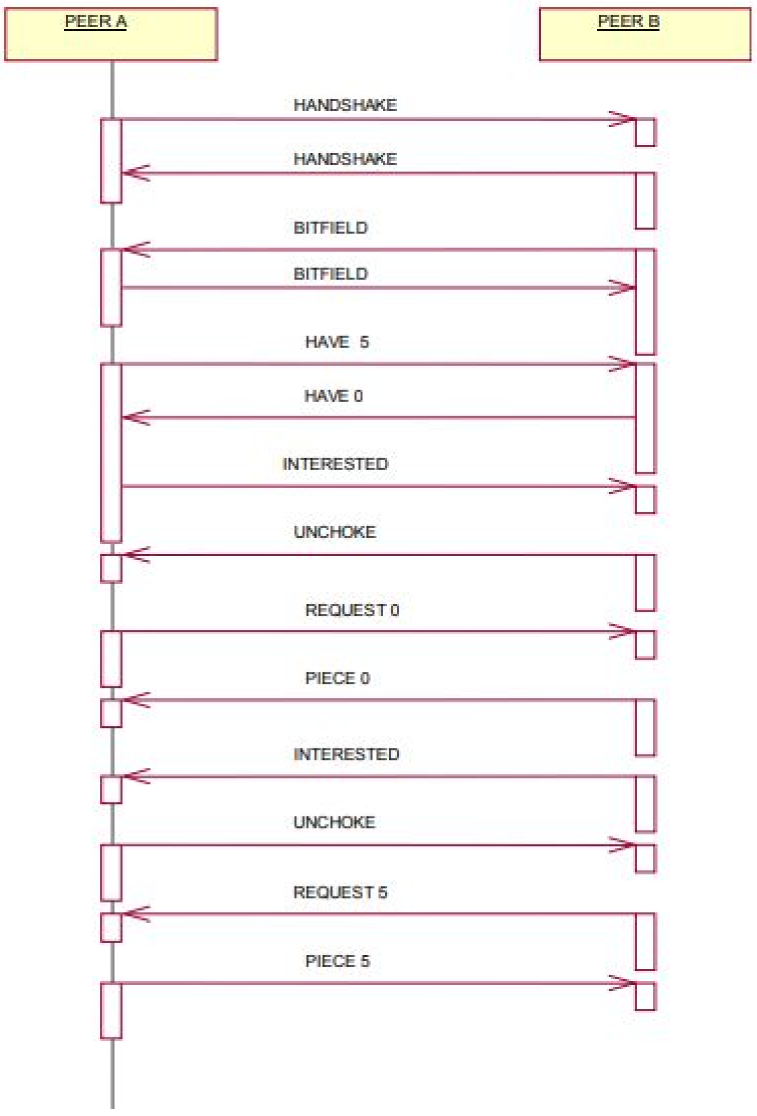
- **BitTorrent protocol** applies a set of policies
  - To provide fairness
  - To encourage the peers to exchange pieces with other peers
  - To prevent overloading a peer with requests from other peers
  - To allow a peer to find peers that provide better service.

- The typical value of the number of concurrent neighbors is four.
- Two types of peers :
  - **Downloaders:** Peers who only have a part (or none) of the file
  - **Seeds:** Peers who have all the pieces of the file but stay in the system to allow other peers to download from them.
- The **unchoked group** is the list of peers that the current peer has concurrently connected to. It continuously uploads and downloads pieces from this group.
- The **choked group** is the list of neighbors that the peer is not currently connected to but may connect to in the future.

- Consider a peer that wants to download a torrent. We will refer to it as the **local peer** and others as **remote peers**.
- Using a **tracker server**, the local peer discovers a list of remote peers in the swarm, selects *N* remote peers, and establishes TCP connections with each. Then the peers use the **Peer Exchange Protocol** to exchange pieces (i.e. upload and download the file).
- The **strategy for selecting peers** to exchange pieces is **important for maximizing the download rate**, as well as uploading content for other peers to access.

- Of the ***N*** remotes peers that the local peer is connected to, it will choose ***U*** unchoked peers to exchange pieces with.
- The local peer chooses these unchoked peers from those that it has the best download rates from, and from those that are interested in exchanging pieces with the local peer.
- Regular updates of choked/unchoked peers are performed, as well as occasional optimistic unchoking in the hope of maximizing the download rate.

BitTorrent Message Flow:



Messages:

1. BITFIELD: it is a required message and must be the first message transmitted by the client, representing the pieces that have been successfully downloaded.
2. HAVE: it is used to inform other peers about the pieces that have been downloaded by that peer.
3. INTERESTED: it is used to inform other peers that it is ready to download a piece from it and will start requesting from it.

4. NOT INTERESTED: it is used when a peer is not ready to download a piece from another.
5. REQUEST: it is used to ask for a particular piece from another peer.
6. PIECE: it indicates the transfer of a requested piece to the peer.
7. CHOKE: it is a notification that no requests will be answered until the client is “unchoked”.
8. UNCHOKE: it is a notification that the requests for pieces will now be answered

Peer Selection Mechanisms:

- These mechanisms form the base for the choking algorithm utilized by a peer.
- Aim is to improve the downloading performance of peers that help induce the file exchange by uploading pieces to other peers, and punish free-riders.

Tit-for-Tat and Fairness in BitTorrent:

- The tit-for-tat strategy is taken up by a peer to preferentially upload to its neighbor peers who provide it the optimum downloading rates.
- This mechanism is very important to encourage downloaders and punish free-riders, thus preventing leechers from downloading without uploading anything.
- An important step towards fairness (“receive as much as you give”).

Optimistic Unchoking Strategy:

- A peer randomly promotes a single interested peer, regardless of its uploading rate, from the choked group and flags it as unchoked.
- To show that rational users would set their uploading rate to be equal to the maximum possible limit.
- To show that the maximum downloading rate of a free-rider is limited to $1/(n_u + 1)$.
- To allow a newly joined peer (does not yet have a piece to share)
- To receive pieces from other peers every 30 seconds(typical OU limit).

Need: If a peer uses only the TFT mechanism, there will be no opportunity for discovering other peers that can provide higher uploading rates.

An Example BitTorrent File Sharing Scenario with free riding and OU strategy:

- Consider a network with a group of peers (g_1) that have the same uploading bandwidth μ.
- The number of peers in the group is N.
- Assume that each peer has nu uploads and one optimistic unchoking upload.
- A new peer j with zero uploading bandwidth joins the network. Each peer $i \in g_1$ will randomly choose a peer that it is not currently uploading to as the target of its optimistic unchoking.

- So, for peer i, on average, 1/ ($N-n_u$) of the time, it will optimistically upload to peer j.
- Since there is a total of N peers in g_1, the total average downloading rate of peer j will be (for large N),

$$N\frac{1}{N-n_u}\frac{\mu}{n_u+1} \approx \frac{\mu}{n_u+1},$$

- In current BitTorrent, n_u = 4 and thus a free-rider gets 20% of the possible maximum downloading rate.

Tradeoff:
It would seem that n_u can be increased to reduce the amount that a free-rider can get. However, choosing n_u to be large means that multiple TCP connections have to share the same bandwidth and thus may lead to more time-outs and result in poor performance.

How does it induce Free Riding?

- Analysis and the experimental results show that the original incentive mechanism of BitTorrent can induce free riding because it is **not effective in rewarding and punishing downloaders properly.** Eg. TFT builds a robust mechanism against free riders
- We address this issue in upcoming sections.

Upload Only:

- A peer becomes a seed, once it finishes downloading the entire file.
- Seeds cannot select peers based on downloading rates, as seeds have nothing to download.
- The seeds prefer to upload to peers with better uploading rates.
- They need to be rewarded.

Anti-snubbing:

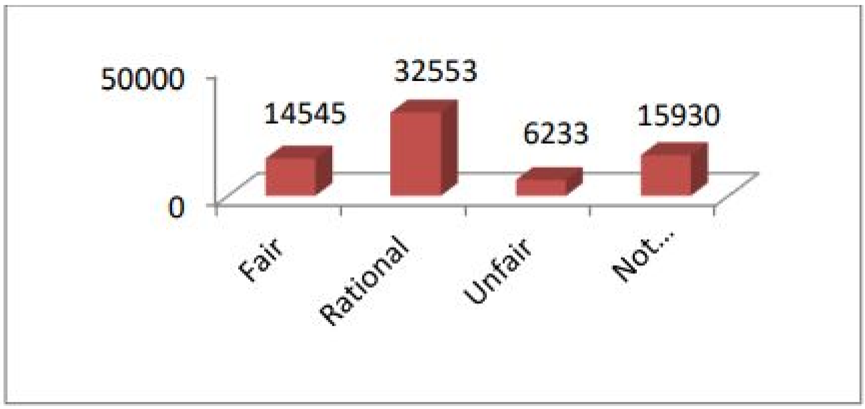
- A peer may be choked by the peers it was earlier downloading from, thereby getting poor downloading rates.
- To address this problem, when a peer notices that some time has elapsed without getting a single piece from a neighboring peer, the leecher assumes it is **snubbed** by that peer and does not upload to it any further through regular unchoke.

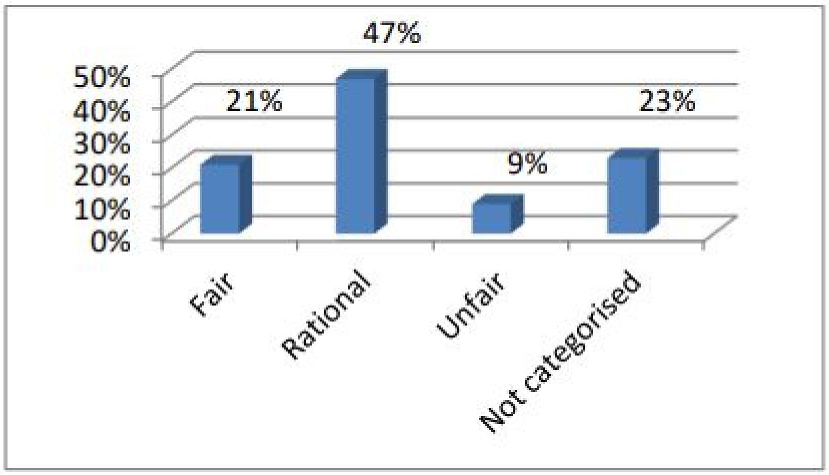
Reciprocal Behaviour:

- Behavioral performance of peers in BitTorrent in real scenarios
- The trackers were robust enough to provide download volume, upload volume, connection time, etc for each peer in the swarm.

- Free Riders are classified according to the **volume uploaded** by them in a content having **multiple torrents**.
    - **Fair** - A peer appearing free rider in one torrent and having altruistic behavior in other torrents
    - **Unfair** - A free rider who never uploads
    - **Rational** - A peer free rider rarely uploading to fellow peers in the swarm
    - **Not Recognized**

- **An experimental setup by a survey:**
    - 47 four part contents
    - 53 five part contents
    - 40 six part contents
    - Number of Torrents = 4*47+5*53+6*40 = 693.
    - Average part size 850 MBs
    - Part size varies from 750 MB to 1100 MBs
    - On an average, 520 peers per torrent among 360746 peers participated. The percentage of free riders is around 20% of the participating peers





Piece Selection Strategies:

Strict Priority:
Peers concentrate on downloading a whole piece of a file, before requesting for another piece.

Endgame Mode:
- An applied algorithm for downloading the **last few pieces** of a torrent.
- A protocol that is used to prevent users who have all but a few pieces from **waiting too long** to finish their download.
- In typical client operation, the last download pieces arrive more slowly than the others. This is because the faster and more easily accessible pieces should have already been obtained.
- In order to prevent the last pieces from becoming unobtainable, BitTorrent clients attempt to get the last missing pieces from all of their peers. Upon receiving the last pieces a cancel request command is sent to other peers.

Rarest first Strategy:
- A strategy to provide a balance between the number of pieces each peer may have at each moment.
- A peer tries to **first download** the pieces with the **fewest repeated copies** among the neighbors so that the pieces are circulated faster.

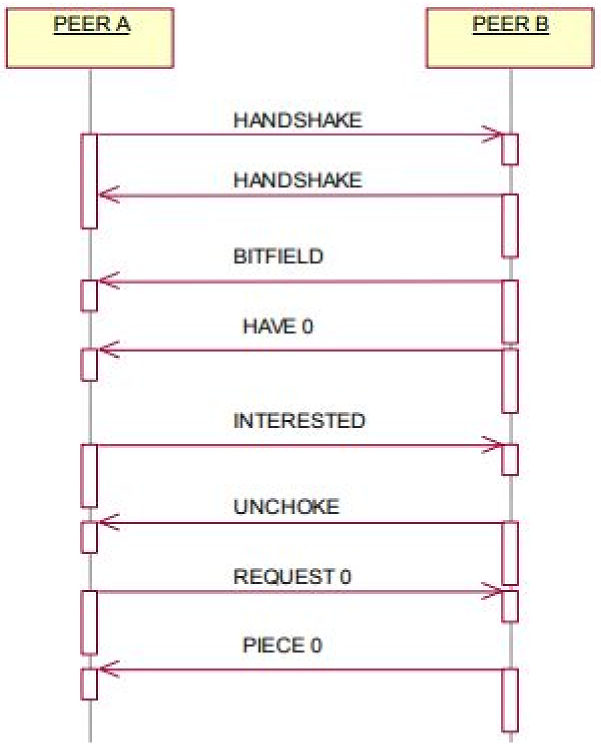
Random First Piece:
- An exception exists to the rarest first policy when a peer first joins a torrent.
- Since the peer does not have any pieces, it would like to download a whole piece quickly so that it can get ready to reciprocate with the TFT algorithm.
- In this case, the new peer will not take the rarest piece. Instead, it will download the first piece randomly in order to make sure it can get a whole piece as soon as possible.

Free-Riding:
- Free-riding means that a **peer does not contribute** anything to the system, while it attempts to obtain service (or downloading) from other peers.
- If peers have global information, the free-riding problem can be solved by not uploading to peers with zero uploading bandwidth.
- Very often a peer is found only downloading large files and not allowing uploading of files so as to minimize bandwidth utilization at its own end. Therefore, a detection-cum-punishment mechanism is needed.
- **In reality, peers use optimistic unchoking to explore the network and this gives an opportunity to free-ride.**

Flow of messages between a free-rider (A) and non-free-rider(B):

The peer A does not periodically broadcast a HAVE message and also refrain from sending an initial BITFIELD message. Peer A, a random peer can therefore be called a Free-Rider.

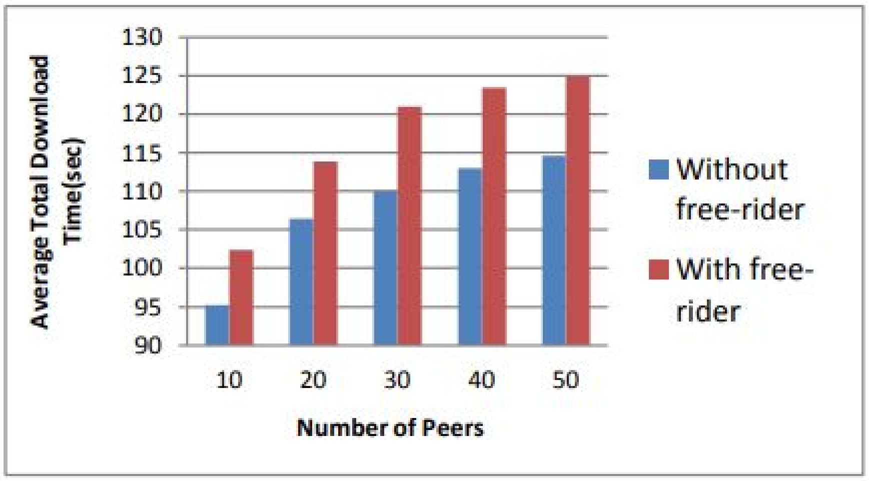
If peers have global information, the free-riding problem can be solved by not uploading to peers with zero uploading bandwidth.



Results and Observations of the analysis:
The average total download time (TDT) of peers was opted as output parameter in both the analysis.

Swarm-level:

- Number of peers - 10 to 50.
- The peers of the system with free-riders have more download finish time compared to the system without any free-rider.
  Conclusion: This shows that free-riders in a P2P network increase the average total download time for all peers.

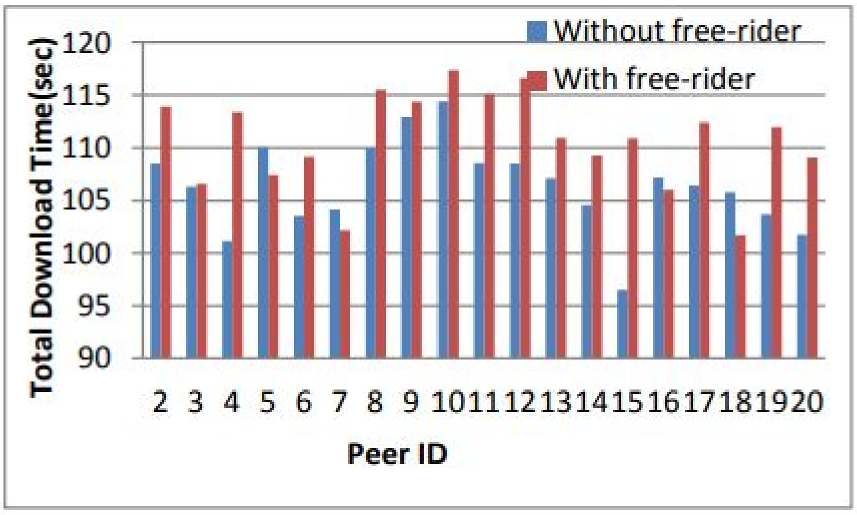


Peer-level:

- The total download time (TDT) for each peer was measured for a swarm of 20 peers.
- IDT is increasing for all peers except the freerider peer #7
- It suggests that the free-riders are not effectively punished by BitTorrent protocol's standard mechanisms.
  Conclusion: Free-riders are not effectively punished by standard mechanisms in the P2P network. Also, the presence of free-riders degrades the download performance of contributing peers i.e. non-free-riders.

In both(swarm and peer level) analysis,

- The download rate for all the peers was configured to 2000 Kbps
- The size of the file to be downloaded was specified as 20 MB.
- 10% out of the total peers were randomly chosen to behave as free-riders.



How does free riding reduce Network's performance?

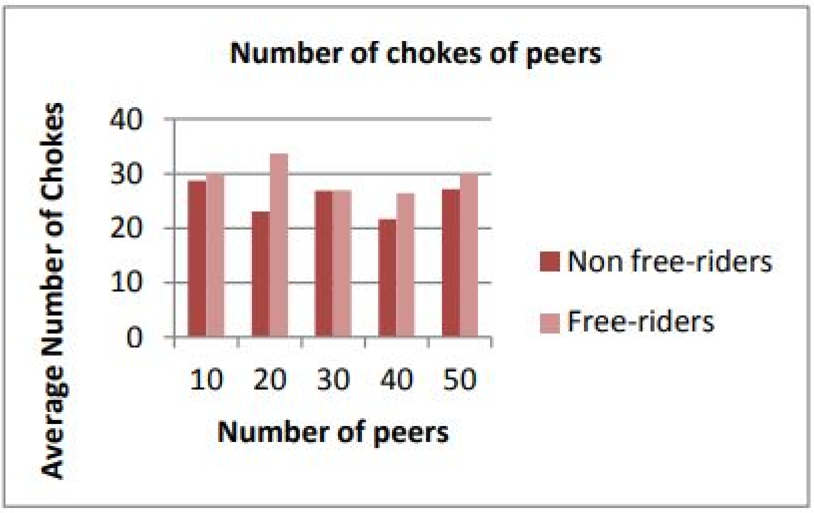
- Increases the download time of peers in P2P networks.
- BitTorrent's Tit-For-Tat (TFT) policy fails to prevent unfairness across nodes in terms of volume of content served.

How can we detect Free-Riding?
A peer is declared as a free-rider under the following conditions:

- It must download the minimum threshold, of the file size (here 20 MB)
- It must not upload any file piece.

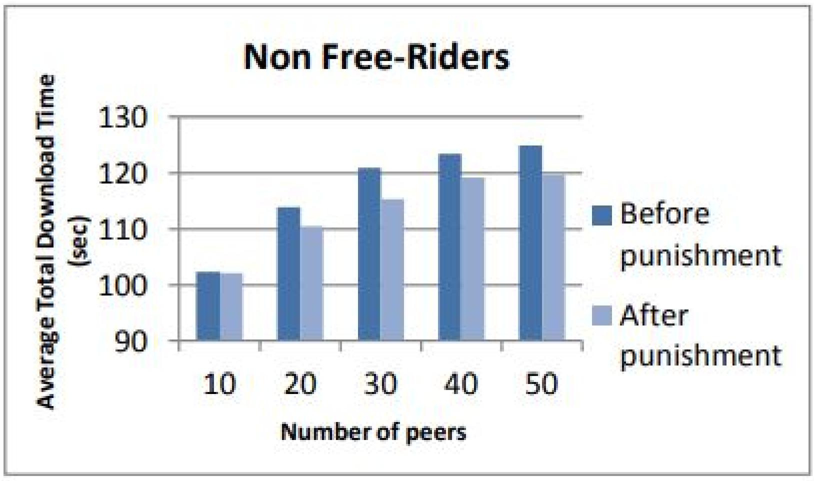
How to Punish the free riders?
By decreasing the download times of non-free-rider peers of the network and punishing free-riders by increasing their download times.
Principle: "The more a peer gets choked, the more will be its download time".
Therefore, free-riders were continuously choked every time the decision of unchoking is made by an optimistic unchoking mechanism.

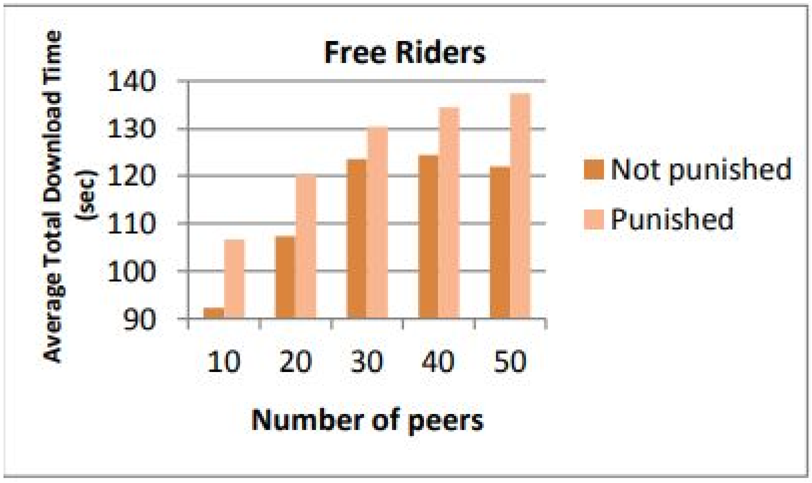
The following **result** shows free-riders were choked more number of times than non-free-riders:



Results of applying the above Detection and Punishment Mechanisms:

- There is a 3 % decrease in the total download time of non-free-riders.
- An increase of an average of 10% in total download time is observed on the free-riders.





Simple Fluid Model:

A mathematical model used to describe the fluid level in a reservoir subject to randomly determined periods of filling and emptying. They are used to predict queue lengths and waiting times.

Purpose:

It obtains the average file-transfer time(an important factor in network performance analysis) and characterizes the variability of the number of peers around the equilibrium values predicted by the deterministic fluid model.

Model Parameters:

$x(t)$ number of downloaders (also known as leechers) in the system at time t
$y(t)$ number of seeds in the system at time t
λ the arrival rate of new requests
Assumption: Poisson Process
μ the uploading bandwidth of a given peer
Assumption: All peers have the same uploading bandwidth
c the downloading bandwidth of a given peer
Assumption: All peers have the same downloading bandwidth, **$c>=\mu$**

θ the rate at which downloaders abort the download.

Assumption: Each downloader independently aborts its download after a certain time, which is exponentially distributed with mean 1/θ.

γ the rate at which seeds leave the system

η indicates the effectiveness of the file sharing

η takes values in [0, 1]

A deterministic fluid model for the evolution of the number of peers is as follows, they are calculated on the basis of Markov Chain Modeling.

$$\begin{aligned}\frac{\mathrm{d}x}{\mathrm{d}t} &= \lambda - \theta x(t) - \min\{cx(t), \mu(\eta x(t) + y(t))\},\\ \frac{\mathrm{d}y}{\mathrm{d}t} &= \min\{cx(t), \mu(\eta x(t) + y(t))\} - \gamma y(t),\end{aligned}$$

Markov chain modeling:

- A Markov chain is a stochastic model describing a sequence of possible events in which the probability of each event depends only on the state attained in the previous event.
- A Stationary distribution of a Markov chain is used to find the mean, the number of jobs, servers, and delay..
- We assume that the probability that some downloader becomes a seed in a small interval δ is given by min(cx, μ(ηx + y))δ and use this probability to model the Markov Chain.

The system in Steady-State:

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = \frac{\mathrm{d}y(t)}{\mathrm{d}t} = 0$$

Little's Law:

- L = λW
- L – the average number of items in a queuing system
- λ – the average number of items arriving at the system per unit of time
- W – the average waiting time an item spends in a queuing system

Using Little's Law, we have

$$\frac{\lambda - \theta\bar{x}}{\lambda}\bar{x} = (\lambda - \theta\bar{x})T,$$

$$T = \frac{1}{\theta + \beta}.$$

Where, $\frac{1}{\beta} = \max\{\frac{1}{c}, \frac{1}{\eta}(\frac{1}{\mu} - \frac{1}{\gamma})\}$

Inferences from the Results:

- T is independent of λ (the request arrival rate). Hence, the BitTorrent P2P system scales very well.
- When η increases, T decreases. This is because the peers share the file more efficiently.
- When γ increases, T increases because a larger γ means that there are fewer seeds in the system.
- Bottlenecks in uploading and downloading bandwidths.
- comment on the case η = 0 (downloaders do not upload data to each other and only download from seeds) :
- **γ<μ** - T = 1/c.
- **γ>μ -** from deterministic fluid model equations,
    - $$\frac{dy(t)}{dt} \leq (\mu - \gamma)y(t).$$
- This tells us that y(t) decreases at least exponentially.
- The number of seeds will exponentially decrease to zero and the system dies.
- So, it is **very important for the downloaders to upload data to each other**.

A Study on η, the parameter that captures the effectiveness of file sharing:
Obtained Expression:
For a given downloader i, we assume that it is connected to k = min{x − 1, K} other downloaders, where x is the number of downloaders in the system and K is the maximum number of downloaders to which a peer can connect.

$$\eta \approx 1 - \left(\frac{\log N}{N}\right)^k .$$

Assumptions made to obtain this expression:

- Each downloader has the information about which pieces the connected peers have.
- Piece distributions between different peers are independent and identical (i.i.d).
- For each downloader, the number of pieces it has is uniformly distributed in {0, ··· , N − 1}, where N is the number of pieces of the served file.
- Given n_i, these pieces are chosen randomly from the set of all pieces of the file, where n_i is the random variable describing the number of pieces at downloader i.
- This is a reasonable assumption because BitTorrent takes a **rarest first piece selection policy** when downloading.

Inferences:

- N is of the order of several hundreds (In BitTorrent, each piece is typically 256KB). Hence, even if k = 1, η is very close to one. Typically K = 40 in BitTorrents. This tells us that BitTorrent is very efficient in sharing files.

- **When k increases**, η also increases but very slowly and the **network performance increases slowly.** Note that, since k depends on the number of other peers in the system, it may be related to the arrival rate λ (previous observation)

Characterizing Variability:

- When the request arrival rate is large (which also means a large number of downloaders and seeds), the fluid model is a good approximation of the real system.
- To understand how the number of seeds and downloaders vary around the numbers predicted by the deterministic model.
- Characterization of the variance of x(t) and y(t) around their equilibrium values, using a Gaussian approximation.

Experimental Results and Analysis on The Proposed Fluid Model:

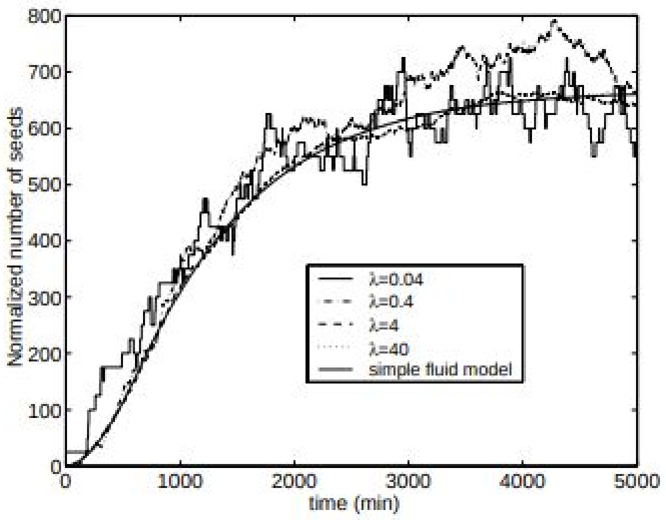
Experiment 1

γ = 0.001

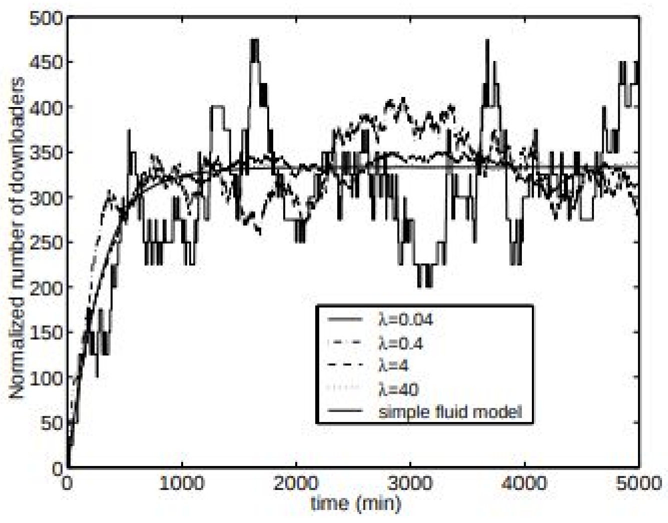
Comparison of the proposed simple deterministic fluid model with the results from a discrete-event simulation of a BitTorrent-like network.

For a file which had a total of less than 100 completed downloads, the match between the fluid model and the observed data is quite close (with relevant parameters and initial conditions)

The evolution of the number of seeds as a function of time



The evolution of the number of downloaders as a function of time

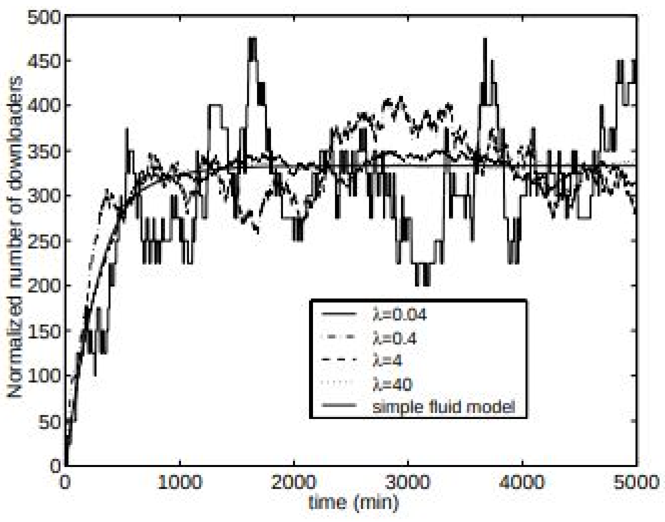


<u>Experiment 2:</u>

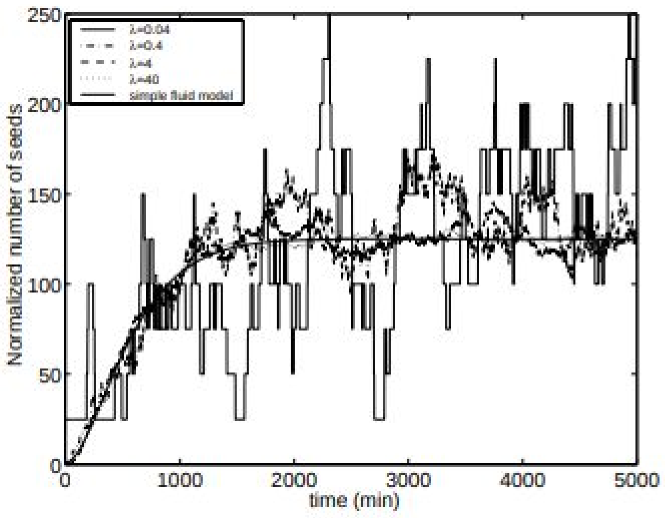
Experiment 1 with γ = 0.005

The uploading bandwidth becomes the bottleneck.
Simple fluid model is accurate when λ is large, but performs well even for smaller λ.

The evolution of the number of downloader:



The evolution of the number of seeds:

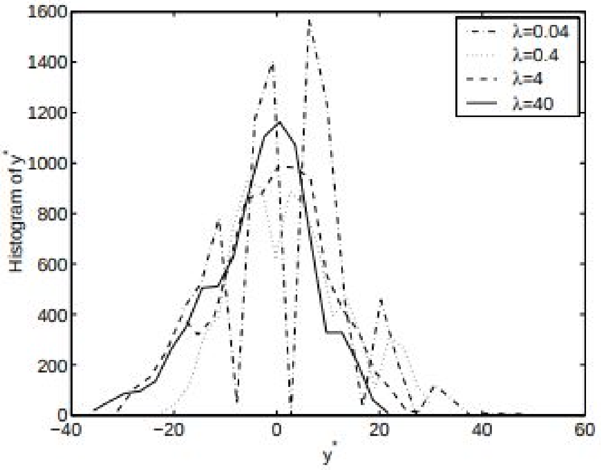


Histogram of the variation of the number of seeds around the fluid model:

- Histograms look roughly Gaussian (figures for sufficiently large λ).
- The variances of ˆx and ˆy do not change much when λ changes from 0.04 to 40.
- **Actual simulation values/parameters:**
  - x_sim(t) is the #downloaders
  - y_sim(t) is the #seeds
- **Deterministic fluid model values/parameters:**
  - x(t) is the #downloaders
  - y(t) is the #seeds

$$\hat{x}(t) = \frac{x_{sim}(t) - x(t)}{\sqrt{\lambda}}$$

$$\hat{y}(t) = \frac{y_{sim}(t) - y(t)}{\sqrt{\lambda}},$$

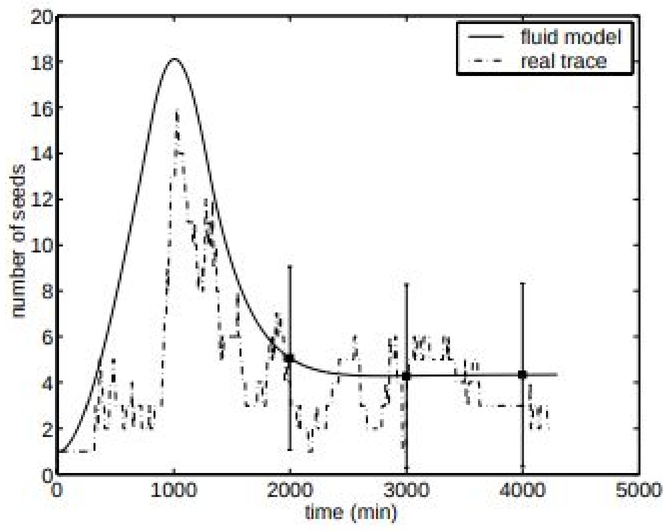


Experiment 3:

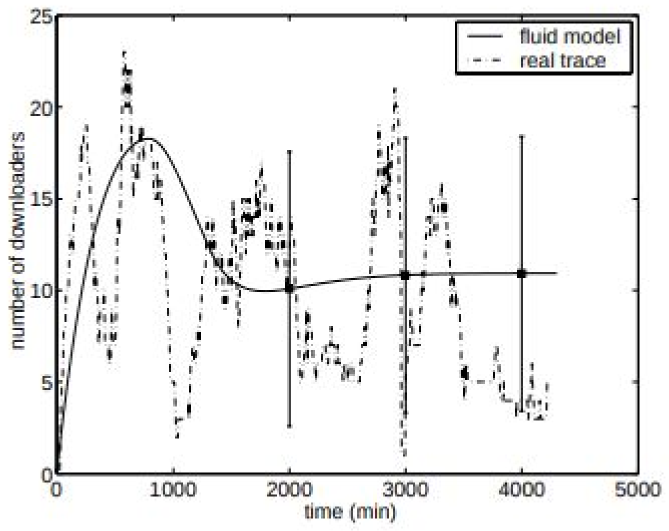
Setup:

- A file is introduced into the BitTorrent network and the log files of the BitTorrent tracker are collected for a time period of around three days.
- Not possible to determine whether the uploading bandwidth or the downloading bandwidth is the bottleneck from the tracker log files (assume the uploading bandwidth is the bottleneck).
- Relevant parameters and initial conditions are given/assumed.

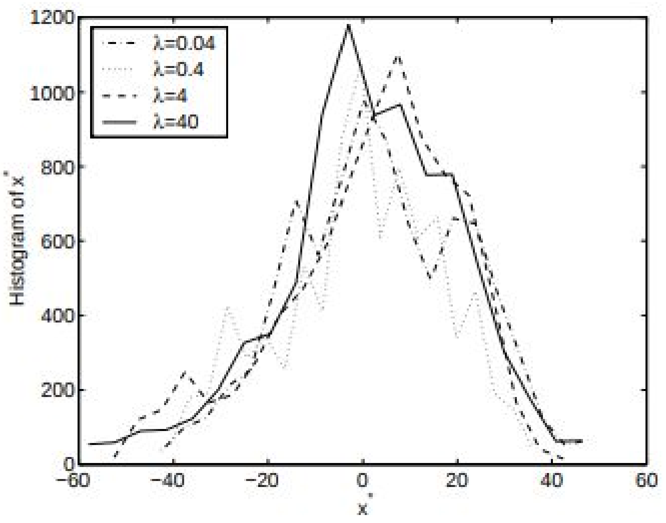
Evolution of the number of seeds:



Evolution of the number of downloaders:



Histogram of the variation of the number of downloaders around the fluid model:



Performance Metrics:

We obtained the expressions for the average number of seeds, the average number of downloaders, and the average downloading time as functions of the peer arrival rate, downloader leaving rate, seed leaving rate, uploading bandwidth, etc, which explicitly give us insight on how the network performance is affected by different parameters.

Peer Evolution:

- In BitTorrent (P2P file sharing), the number of peers in the system is an important factor in determining network performance.
- Therefore, it is useful to study how the number of peers evolves as a function of the request arrival rate, the peer departure rate, the uploading/downloading bandwidth of each peer, etc.

Scalability:

- The network performance to not deteriorate, and preferably to actually improve, as the size of the network increases.
- Network performance can be measured by the average file downloading time and the size of the network can be characterized by the number of peers, the arrival rate of peers, etc.

File-Sharing Efficiency:

- Peers have different uploading/downloading bandwidths.
- A file may be broken into smaller pieces and the pieces may be distributed at random among the peers in the network.
- To efficiently download the file, it is important to design the file-sharing protocol such that each peer is matched with others who have the pieces of

the file that it needs and further, to ensure that the downloading bandwidth of each peer is fully utilized.

<u>Iterative Prisoner’s Dilemma:</u>

- BitTorrent can be modelled as “**iterated prisoner’s dilemma**” using game theoretic framework using which we can understand the conflict and cooperation between the peers
- We relate the download process in BitTorrent, by building a payoff matrix for the fragment exchange in BitTorrent.
- Let d (>0) denote the utility of downloading a fragment and u (>0) denote the negative utility that represents the cost incurred in uploading a fragment.
- The components of the payoff matrix are assigned as follows:
    - R = d u; T = d; S = u; P = 0:
- Conditions of the payoff matrix
    - T >R>P >S,
    - 2R>S+T
- The payoff matrix we present satisfies the two conditions above, provided d>u, which should be true of those who join the BitTorrent network.
- After all, the peer-to-peer computing, including BT, has emerged as users can share their resources in return for some benefit (i.e., d>u)

<u>Incentive Mechanism based on the Fluid Model:</u>

- The objective of the incentive mechanism is to encourage users to contribute. There are three algorithms in BitTorrent which are intended to **discourage free-riding.**
- Incentive mechanisms in P2P systems can’t be overemphasized and they deserve more attention.
- The lack of reward and punishment can induce free riding to such a degree that the system becomes inefficient.
- To address this issue, we propose a new mechanism that is robust against free riders and therefore, with this mechanism, the system should be sustainable in the long run.
- **Original Incentive Mechanism**
    - Find **best k neighbors**
    - Unchoke those k neighbors and choke all other neighbors
    - Choking algorithm (every 30 seconds)
        - Filter out uninterested neighbors
        - Sort neighbors in decreasing order of download rates
        - Pick first k neighbors
    - Optimistic unchoking: 30 seconds of unconditional uploading
- **Peer Selection Algorithm**
    - We assume there is no optimistic unchoking
    - We sort the peers in descending order of their uploading bandwidth
    - **Algorithm:**

Let Total number of peers is N
Uploading bandwidth of peer i is $\boldsymbol{\mu}_i$
At step i, peer i selects peers to upload according to the following rules:

- If the downloading rate from peer j to peer i is greater than or equal to the uploading rate of i, peer i should upload to peer j to try to keep the downloading rate high.
    - (a) If $\mu_{k1} > \mu_{k2}$, select k1.
    - (b) If $\mu_{k1} = \mu_{k2}$ and $n^i_{k1} < n^i_{k2}$, select k1.
    - (c) If $\mu_{k1} = \mu_{k2}$, $n^i_{k1} = n^i_{k2}$, and k1 < k2, select k1.
- Takes care of the last several peers and makes sure that all peers have n_u (mostly 4) uploads.

- **Peer Strategy**
    - Peer strategy describes how the users set their bandwidth.
    - Let the uploading bandwidth of peer i be μ_i
    - Let p_i be the physical uploading bandwidth of peer i
    - Here, we assume that maximizing the gain has priority over minimizing the cost.
    - When $\mu_i = p_i$, peer i gets the maximum downloading rate.
    - But setting $\mu_i = p_i$ is not necessarily the best strategy for peer i.
    - A peer will want to maximize the gain, but at the same time, also minimize the cost.

Applications:

Performance Analysis of BitTorrent and Its Impact on Real-Time Video Applications

Uses a detailed model of the BitTorrent protocol to analyze its performance and impact on real-time video traffic.

The Model shows the following:

- Increasing the number of BitTorrent clients and/or upload connections can cause a decrease in download rate due to delayed TCP acknowledgments.
- Effect of access router buffer size on performance:
    - Too small **reduces** BitTorrent’s upload rate
    - Too large **increases** video jitter and delay.
- **Relationship between BitTorrent clients**
- **Upload connections**
- **Access router buffer size on delay**
- Upload/download rates
- Packet drops for BitTorrent and video applications.

Bottleneck

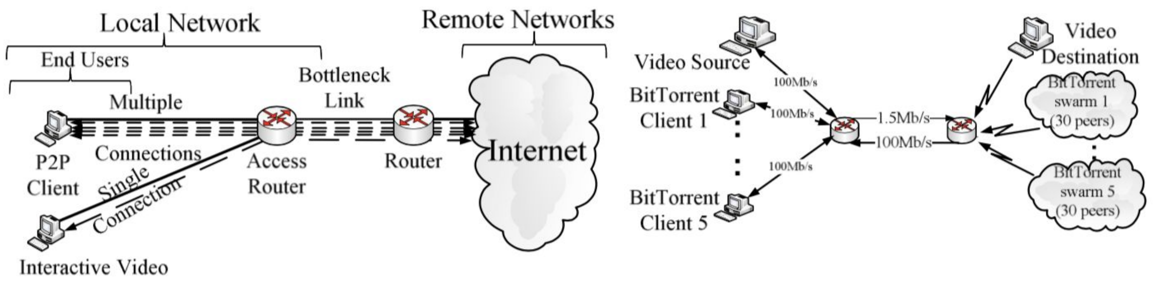
- The popularity and efficiency of P2P file-sharing have resulted in performance problems for end-users and Internet Service Providers (ISPs).
- Increasing the delay experienced when web browsing etc, happens due to the protocol's interaction with other applications.
- Particularly **detrimental** to **real-time voice** and **video applications**.

Performance Issues

- Using multiple TCP connections can increase reliability and distribute traffic load to multiple peers.
- It can also impact **on how link bandwidth is shared** among applications and users.
- Considering an end-user running **multiple applications**, the user may experience **unfairness in the link bandwidth allocation** for each of the applications because of the different number of TCP connections established.

- 2 types of Bottlenecks
    - Multiple applications at end user side - **Controllable**
    - Multiple end users to ISP access router - **Uncontrollable by user**
        - Access router uplink to next router - bottleneck
        - Large delay and high packet drops when queue is full
        - Arises when there are a large number of local hosts, each with reasonable uplink speeds, but the uplink speed from an access router is insufficient for all hosts uploading at the same time.

A diagrammatic representation of sharing of access link among end-users running different applications with different number of connections



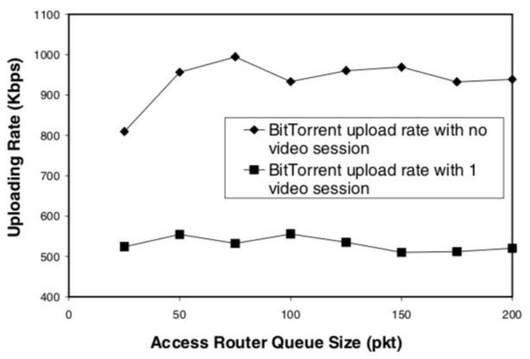
Drop-tail queue discipline

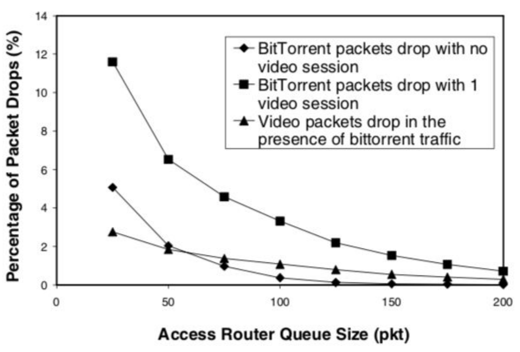
- Drop tail is a queue management algorithm used by network schedulers in network equipment to decide when to drop packets.
- When the queue is filled to its maximum capacity, the newly arriving packets are dropped until the queue has enough room to accept incoming traffic.

Analysis

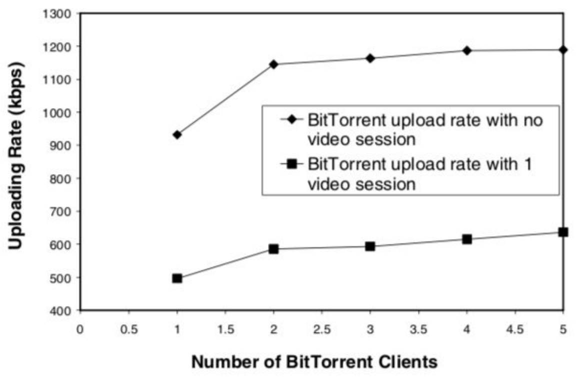
- Two different simulation configurations were carried out:
  - BitTorrent traffic with no video session
  - BitTorrent traffic with 1 video session

- **Access Router Queue size**
  - Uploading Rate
  - Percentage of Packet Drops

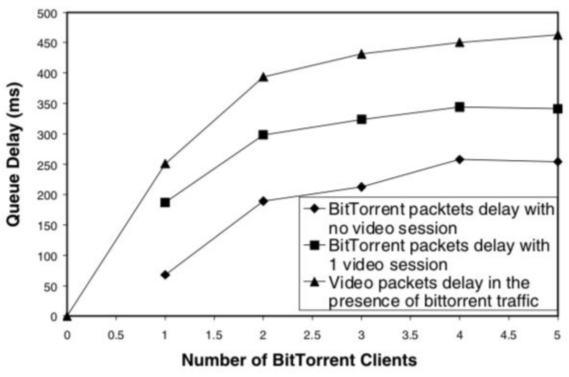




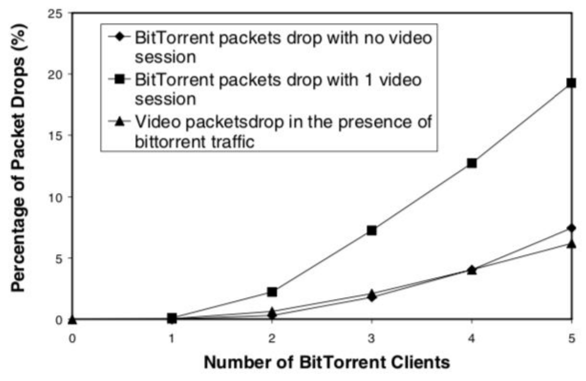
- **Impact of varying the number of local BitTorrent clients from 0 up to 5**
  - Uploading Rate
  - Queue Delay
  - Percentage of Packet drops
  - Downloading Rate
  - Inter arrival rate



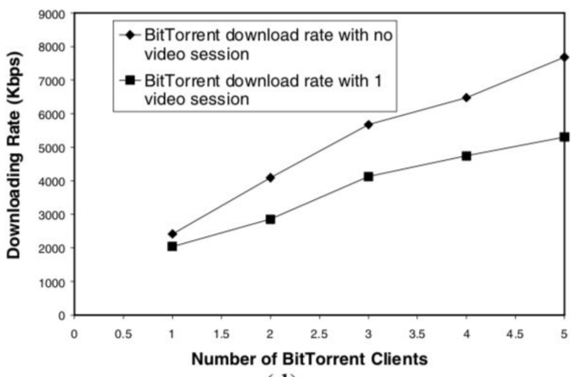
(a)



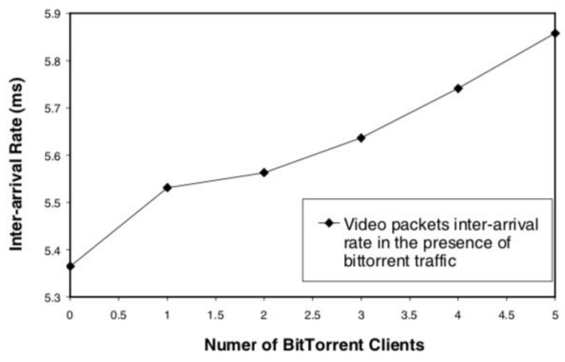
(b)



(c)



(d)



(e)

References:

Research papers (and References) for Survey (within the scope of the main paper and the project objective):

“Fairness, incentives and performance in peer-to-peer network”
http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=C49D7A1A99BDFB6C1C3B5D1C2640F07E?doi=10.1.1.121.9069&rep=rep1&type=pdf

“Performance analysis of BitTorrent (P2P) protocol”
(Real-Time Video Applications)

- https://www.researchgate.net/publication/220717176_Performance_Analysis_of_BitTorrent_and_Its_Impact_on_Real-Time_Video_Applications

- https://www.researchgate.net/publication/261269751_Performance_analysis_of_BitTorrent_protocol
- https://web.njit.edu/~dingxn/papers/BT-JSAC.pdf

“On Free Riding”

- https://www.ijact.org/ijactold/volume5issue5/IJ0550007.pdf
- http://netecon.seas.harvard.edu/P2PEcon05.html/Papers/Jun_05.pdf
- https://www.usenix.org/legacy/event/nsdi07/tech/piatek/piatek_html/bittyrant.html
- https://www.researchgate.net/publication/271547138_Free-rider_detection_and_punishment_in_BitTorrent_based_P2P_networks