Doctoral Dissertation Academic Year 2014

Analysis of Peer-to-Peer Operation in Content Delivery

Keio University

Graduate School of Media and Governance

Mohamad Dikshie Fauzie

Summary

In this research, BitTorrent as one of the most popular and successful P2P applications in the current Internet, is taken as an example in a study of uncoordinated P2P operation. The first problem to be addressed in this research is how to reveal the topology of real BitTorrent swarms, how dynamic the topology is, and how it affects overall behavior. We study BitTorrent networks, where real-world BitTorrent swarms were measured using a rigorous and simple method in order to understand the BitTorrent network topology. We propose usage of the BitTorrent Peer Exchange (PEX) messages to infer the topology of BitTorrent swarms listed on a BitTorrent tracker claiming to be the largest BitTorrent network on the Internet, instead of building small BitTorrent networks on testbeds such as PlanetLab and OneLab as other researchers have done. We also performed simulations using the same approach to show the validity of the inferred topology resulted from the PEX messages by comparing it with the topology of the simulated network. Our result, verified using the Kolmogorov-Smirnov goodness of fit test and the likelihood ratio test and confirmed via simulation, show that a power-law with exponential cutoff is a more plausible model than a pure power-law distribution. We also found that the average clustering coefficient is very low, implies the BitTorrent swarms are close to random networks. BitTorrent swarms are far more dynamic than has been recognized previously, potentially impacting attempts to optimize the performance of the system as well as the accuracy of

simulations and analyses.

In the current content delivery architecture, many CDN companies and ISPs adopt hybrid CDN-P2P because of the advantages of P2P. On the P2P side, peers are organized in a tree-based overlay on a per substream basis for live streaming. This ensures that all peers contribute some upload bandwidth. Each CDN server keeps track of clients currently assigned to it to avoid the undesirable side effects of P2P. Each client learns about other peers assigned to its designed CDN server. Since in a hybrid CDN-P2P architecture, some of the workload or data delivery is done by peers, CDN servers benefit from a potential reduction in power consumption. The second problem to be addressed in this research is the trade-off of hybrid CDN-P2P architecture compare to CDN. We solve this problem by proposing a simple model of power consumption of CDN server and router, including the cost of cooling needed generated from the power consumption of the CDN server and router. Furthermore, this power reduction can be used for capacity planning of data centers.

Finally, our proposed methodology can contribute to further characterizing P2P networks and promotion of relaxing capacity planning data center in term of energy consumption by hybrid CDN-P2P.

Keywords: P2P, BitTorrent, Peer-Assisted CDN, Power-Law, CDN, Energy.

論文要旨

本研究では、uncoordinated P2P の運用の研究のために、BitTorrent を取り上げる。BitTorrent は今日のインターネットで最も広く使われている P2P アプリケーションである。一つ目の問題は、現実の BitTorrent Swarm のトポロジが如何に動的であり、全体の動作にどう影響するのか、その実態をどうやって解明するかである。本研究では、正確で単純な手法を用いて現実の BitTorrent swarm を計測することで BitTorrent のネットワークトポロジを明らかにし、BitTorrent ネットワークについて研究した。先行研究が PlanetLb や OneLab 上に BitTorrent ネットワークの小型テストベッドを作成して研究しているのと異なり、本研究では、BitTorrent Peer Exchange (PEX) メッセージを利用して、BitTorrent tracker に記載されている BitTorrent swarm のトポロジを推測し、インターネット上で最大の BitTorrent ネットワークを求めた。この推測手法の正当性の検証にはシミュレーションを利用し、シミュレートされたネットワークと、そのネットワークの PEX メッセージから推測したネットワークを比較した。コルモゴロフ -スミルノフ検定と尤度比検定を用いて検証した結果、べき乗分布より、指数関数的カットオフの存在するべき乗分布に近いことがわかった。また、クラスタ係数の平均はとても小さく、現実の BitTorrent swarm はランダムネットワークに近いことも分かった。BitTorrent swarm は、従来考えられてきたように潜在的にパフォーマンスを最適化を試みていると言うより、動的である。

現在のコンテンツ配送アーキテクチャでは、CDN 事業者と ISP は P2P

の長所を活かしてハイブリッド CDN-P2P を採用している。P2P 側では、ライブストリーミングのために、サブストリーム毎に木構造を基本としたオーバーレイネットワークを構築している。これは、全てのピアがアップロード帯域に貢献することを保証する。各々の CDN サーバは、P2P の悪い副作用を回避するために、現在そのサーバに割り当てられているクライアントを記憶している。各々のクライアントは、自分の CDN サーバに割り当てられている他のクライアントを学習する。ハイブリッド CDN-P2P アーキテクチャなので、データ配送のワークロードのうちいくらかはピアに分担され、CDN サーバは電力消費を抑えられる。本研究で扱う二番目の問題は、CDN と比較してハイブリッド CDN-P2P アーキテクチャが持つトレードオフである。本研究では、電力消費によって必要となる冷却のコストを含め、CDN サーバとルータの電力消費量の単純なモデルを提案する。さらに、本研究による電力節約は、データセンターの容量計画にも利用できる。最後に、我々が提案する方法は、P2P ネットワークを特徴づけることにも利用でき、ハイブリッド CDN-P2P のよるエネルギー消費の意味で、データセンターの容量緩和の促進に利用できる。

キーワード: P2P, BitTorrent, Peer-Assisted CDN, Power-Law, CDN, エネルギー

Acknowledgements

I would like to thank all the people, who always encourage my research. Prof.Jun Murai, my thesis supervisor who keeps giving the direction of my research and support me with insightful comments. Prof.Kusumoto who giving me the supports. Prof.Osamu Nakamura, who giving me sharp comments of my work. I extremely grateful to Dr.Kenjiro Cho for constructive suggestions. Associate Prof.Rodney Van Meter has been kindly advised me with various research supports. Dr.Achmad Husni Thamrin who supports my research life in SFC everyday. Dr.Shigeya Suzuki who supports me in various ways including some administrative stuffs. Prof.Keiko Okawa who supports me in SFC and gave me opportunity to join School on Internet (SOI) - Asia project and WIDE project.

I am also grateful to Prof.Jin Mitsugi, Prof.Keiji Takeda, Prof.Keisuke Uehara, Dr.Hitoshi Asaeda, Dr.Kotaro Kataoka, Dr.Kenji Saito, Mr.Ucu Maksudi, and Prof. Kilnam Chon. Thanks to all members of SOI-Asia group: Dr.Achmad Basuki, Dr.Shoko Mikawa, Mr.Haruhito Watanabe, Mr.Katsuhiro Hiroba, Mr.Noriatsu Kudo, and SOI-Asia group in Keio Media Design: Mr.Marcos Sadao Maekawa and Mr.Goki Miyakita.

I would also say thank my father, my mother, and my wife for understanding of my extra student time.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Internet-based multimedia content delivery enables users to watch desired content from any location at any point of time. With the increasing capacities of end-user devices and faster Internet connections, the popularity of such services is growing steadily. Cisco VNI predicts that video streaming will significantly outweigh other types of consumer Internet traffic, such as file sharing, Web, Voice over IP (VoIP), and online gaming [11]. In contrast to file transfers, video streaming enables users to watch the video while downloading it, which imposes strict requirements on the delivery infrastructure. The users expect performance similar to traditional television, with short startup delays and without performance degradations or playback stalling during watching. This is exacerbated by the growing requirements on video quality, such as higher resolutions and additional features (high-definition and 3D videos). The higher quality typically results in increased video bitrates that require higher download bandwidth.

Today, users are increasingly able to consume videos directly from their TV screens using Internet-enabled Set-top Boxes (STBs) such as digital video

recorders, game consoles, or other entertainment devices. Questions arise as to which delivery architecture is able to provide this vast amount of video content to end-user devices and which mechanisms are required to make this architecture scalable and cost-efficient. Common solutions are centralized and decentralized delivery architectures employing various mechanisms to deliver video streams to end-users. These delivery architectures build overlay networks on top of the underlying Internet infrastructure. The simplest architecture for video streaming is based on the centralized client-server model. Here (one or many) video servers send a separate video stream to each client, which results in high bandwidth costs for popular content and potential scalability issues for large numbers of concurrent users. The peer-to-peer (P2P) paradigm offers a promising alternative to pure server-based video distribution networks. Here, the users, called peers, not only consume but also provide services to other peers. The application of the P2P paradigm to video streaming uses peers' resources, such as local storage, computational power, and bandwidth, to reduce the load and costs of content servers. In the extreme case of pure P2P streaming, there are no dedicated servers anymore and all services are provided by regular peers. If we consider a commercial streaming system, a pure P2P solution turns out to be insufficient because it lacks important properties such as service guarantees for users, security, and control by the content provider. In order to overcome these limitations, a peer-assisted architecture can combine content servers and peers intelligently.

In order to understand the relationships between entities in a P2P-CDN ecosystem and to identify the possible tensions, we must understand their roles in the architecture:

- Overlay providers contribute the initial content and host servers for content injection and indexing. In a pure commercial scenario the overlay provider also acts as a content provider, while in a scenario with user-

generated content the content is contributed by users that upload it to content servers. Typically, an overlay provider receives certain payments for the hosted content, either directly from users (usage-dependent or subscription-based) or indirectly via advertisements.

- Users consume the streamed videos but also provide their resources to the system, such as upload bandwidth, local storage space, and online time. The users typically pay flat-rate fees for the Internet access, which explains why they allow an overlay provider to use their upload bandwidth.

- Network operators provide infrastructure for Internet access and receive flat-rate payments from the users for this service. Typical delivery overlays span several network domains controlled by different network operators. Therefore, network operators must manage both the internal and external traffic flows to avoid congestion and excessive payments for traffic transit.

A peer-assisted solution shifts the main load of content delivery from the overlay providers' servers to users. However, the actual delivery costs are shifted from the overlay providers to the network operators. The reason is the widespread acceptance of flat-rate based pricing for the Internet access. These pricing schemes allow network operators to attract users and to sell high-speed Internet connections. But peer-assisted overlays can also lead to bottlenecks and link congestions, since the Internet architecture is built for the client-server traffic pattern where the traffic flows from content servers to the users. In recent years, the management of overlay traffic that crosses the boundaries of network operators' domains has gained a lot of attention in the research community. Thereby, various traffic management methods have been proposed to relieve the tension between the overlays and network operators.

However, most of them fail to satisfy the demands of both parties.

1.2 Challenges

In this section, we discuss the specific challenges of peer-assisted architectures. Most of these challenges arise from the necessity to achieve quality of service (QoS) comparable to the current client-server systems, while using the limited resources of unreliable peers.

- Limited resources of an individual peer compared to a typical server mean that the resources of many peers must be combined to serve the same streaming request. This applies in particular to the upload bandwidth, which is typically much smaller than the download bandwidth.

- Heterogeneity of peers in terms of their resources and behavior. For example, the upload capacity is a resource that differs between the users and affects the system' s performance significantly.

- Lack of service guarantees at peers makes it difficult to ensure a quality streaming experience to the users (comparable to well-dimensioned server-based systems). In a commercial scenario, in contrast to pure P2P-based systems, all users should be able to receive the quality they paid for, which makes the coupling between the peer' s contribution of resources and received streaming quality undesirable.

- Missing or insufficient incentives for users to contribute their resources are a common issue for P2P-based systems. In a commercial scenario, it can be partially solved by offering rewards or discounts for contributed resources. For example, a peer might get certain credits for each megabyte of data uploaded to other peers. However, this does not solve the issue of users that should remain online in order to provide content availability.

- Energy consumption is becoming an important challenge for content delivery. While various approaches were proposed to increase the energy efficiency of servers and routers in terms of reduced power consumption, the same issue applies for the users' devices. One interesting aspect here is whether an idle peer should stay online to serve new requests or leave the system. While the first option would maximize the peer' s contribution to the system, the second option would save energy that might be wasted if the services of this peer are not required.

- Negative impact on the network infrastructure is another issue in peer-assisted and pure P2P delivery architectures. Most P2P and peer-assisted overlays apply their own application level routing mechanisms that might have undesired effects on the underlying network such as congested links or high costs for the transit traffic.

1.3 Motivation

BitTorrent as one of the most popular P2P file sharing applications, dominated the Internet traffic and is still growing even thought recent studies suggest that its growth is slower than the growth of Internet traffic and its proportion to the Internet traffic is declining. This popularity reflects the robustness and efficiency of the BitTorrent protocol. These characteristics of BitTorrent come from its peer and piece selection strategies to distribute large files efficiently.

Many properties of BitTorrent such as upload, download performance, peer arrival and departure have been studied but only few research projects have assessed the topological properties of BitTorrent. The BitTorrent system is different from other P2P systems. The BitTorrent protocol does not offer peer traversal and the BitTorrent tracker also does not know about the relationship

between peers since peers never send information to the tracker concerning their connectivity with other peers. While a crawler can be used in other P2P networks such as Gnutella, in BitTorrent we cannot easily use a crawler to discover topology, making direct measurement of the topology very difficult and challenging. BitTorrent swarm topologies reflect peer behavior. The peer relationship behavior is very important as a basis for designing a controllable P2P system to be used together with other systems for example, peer assisted CDN or peer assisted cloud.

In the current modern content delivery, CDN providers tend to combine P2P with CDN in order to help with the scaling the services, especially related to traffic or bandwidth saving. It has been done by, for example, Akamai and Pando networks. The bigger issue is not traffic or bandwidth that can be relatively easily fulfilled by adding a network card into a router or adding more servers, but instead is the energy consumption by CDN provider itself inside the data center. Since the Internet traffic grows, demand for scaling is also grow thus demand for energy is also grow. This growing is constrained by energy supply in data center. The usage of peer assisted CDN can be seen not only for helping scaling the services, but there is potential to reduce the energy consumption furthermore this reduction can relax energy budget inside data center.

1.4 Approach

The key factors in this research are: (1) characterization of peer dynamic in P2P systems, (2) simulation of peer-assisted CDN, and (3) design of energy consumption trade-off in the using of P2P to assist CDN server.

P2P Swarm Dynamics

Real-world BitTorrent swarms were measured using a rigorous and simple

method in order to understand the BitTorrent topology. To our knowledge, our approach is the first anyone has to performed such a study on real-world BitTorrent network topologies. We used the BitTorrent peer exchange (PEX) messages to infer the topology of BitTorrent swarms listed on a BitTorrent tracker claiming the be the largest BitTorrent network on the Internet, instead of building small BitTorrent networks on testbed such as PlanetLab or OneLab as other researchers have done [14]. We also performed simulations using the same approach to show the validity of the inferred topology resulted from the PEX messages by comparing it with the topology of the simulated network.

Peer-Assisted CDN

The goal of peer-assisted CDN is to improve the delivery of content by the peer-to-peer network. In this work, we simulated how a peer-assisted CDN can deliver the content that involved 100000 peers and with a catalog that consists of 10000 videos. Our model is an improvement from previous researcher [26]. We found that our peer-assisted CDN model can increase peer contribution while maintaining optimal number of replicas. Increasing of peer contributions will impact to the energy usage in CDN side. The CDN side can reduce the workload because some workload of content delivery done by peers. The conceptual application of peer-assisted CDN in SOI-Asia/AI3 network is presented in chapter 4.

Energy Consumption of peer assisted CDN

As intermediate step in merging between P2P and CDN, this research introduce energy consumption trade-off in peer-assisted CDN. We analyze the characteristics and the requirements of peer-assisted CDN for live streaming and peer-assisted CDN for online storage. Then we proposed an energy consumption model for both peer-assisted CDN architectures. To be able to validate the result, we use the models from two architectures that are currently running

on the Internet, LiveSky [79] and FS2you [19].

1.5 Contribution

This dissertation contributes by enabling analysis into the integration of P2P services and CDN services.

- For P2P, we propose new and effective methodology to infer BitTorrent swarm topologies. We show that gathering BitTorrent swarm topologies is important as a step to understand peer behavior.
- For Peer-assisted CDN, we show by a simulation that our model can increase peer contribution while maintaining the optimum number of replicas.
- For energy trade-off, we show that both peer-assisted CDN architecture have differences energy characteristics that can be used by considering the model in the service integration between P2P and CDN. This model can be used by CDN providers as a basis for: capacity planning in data center and incentive planning to customer who will run peer-assisted mode in home gateway.

Chapter 2

P2P Content Delivery

2.1 Introduction

P2P applications such as Napster, Gnutella, FastTrack, BitTorrent, Skype and PPLive have attracted the end users. In P2P paradigm, the P2P network is formed by peers that equally share the computing resources in a cooperative manner. Each peer contributes some of its resources such as network bandwidth, storage, etc. As the number of nodes in the P2P network grows, the capacity of the network grows. This enables a P2P application to be cheap and it can be used for content delivery.

We noted that the popular P2P file sharing applications are Gnutella, eDonkey, and BitTorrent, while other type of P2P applications are storage systems, VoIP, and IPTV [9]. Gnutella is one of the earliest P2P file sharing applications in the Internet. Gnutella is a pure P2P application that does not use a centralized server. A Gnutella peer joins the network via at least one known peer. That known peer IP address is obtained via a list of pre-configured addresses. From this initial Gnutella peer, we can discover new Gnutella peers. The Gnutella peer will send the search request of a file to all connected peers. The recipient peer will answer the query if it knows anything. The recipient

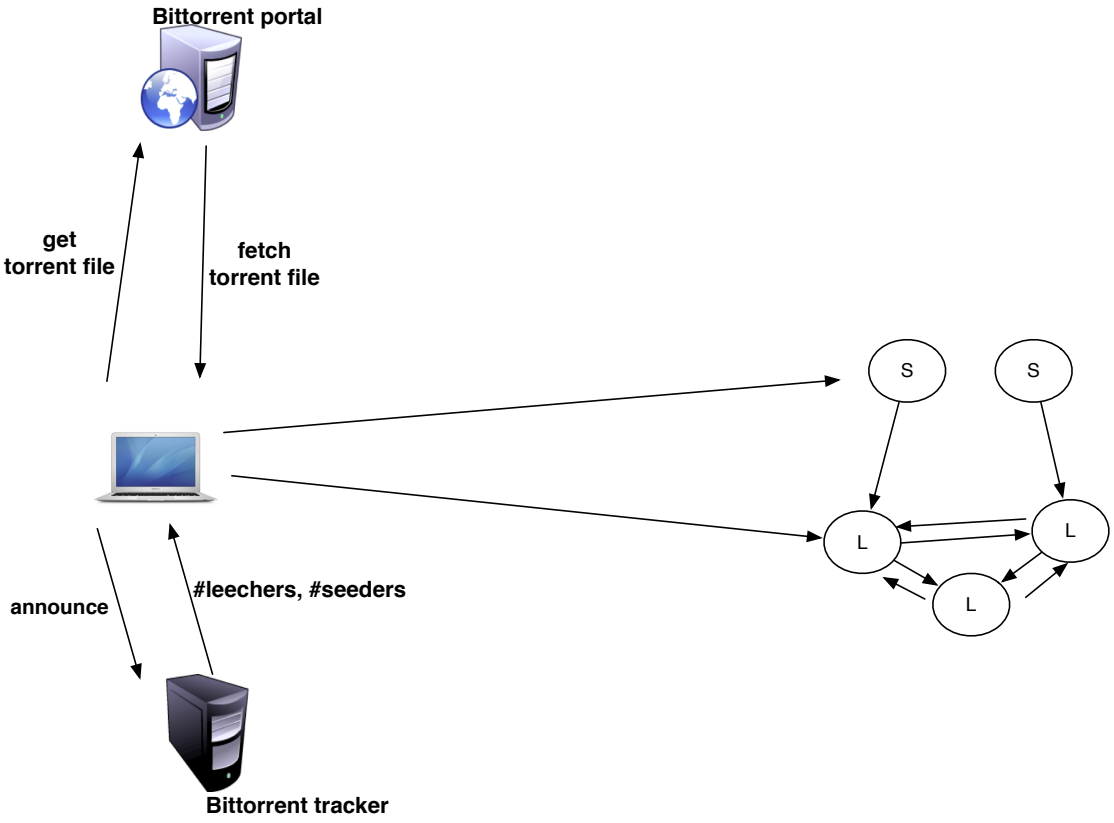

Figure 2.1: Building block of BitTorrent. L refers to Leecher and S refers to Seeder. In a very popular torrent swarm, where there may be more than 40000 seeders and more than 50000 leechers. While 'get torrent' and 'fetch torrent file' only happen in the beginning of process when a peer join a BitTorrent swarm, the announce step will be repeated for every 30 minutes as the BitTorrent specification.

peer can forward the request to other peers if it does not know the answer. Thus, the query will propagate in a flood among the Gnutella network.

Another P2P application is eDonkey. This application is very popular in Europe. The eDonkey network operates as a hybrid P2P and server. The eDonkey network consists of number of clients and a number of servers. The IP address of a server is usually pre-configured when the user installs the application for the first time. If the user wants to change the server, he or she can read on eDonkey web portal. The eDonkey server works as an index for distributing IP addresses of other eDonkey peers to the eDonkey users.

BitTorrent was created in 2002 by Bram Cohen. It runs on an open protocol specification, thus there are many BitTorrent implementations. To share a file or a set of files through BitTorrent, a torrent file must be created for the first time. The torrent file contains the information of the content, which includes the information of the tracker and the hashes of the file blocks to be distributed. The torrent file is usually distributed via BitTorent web portals. The client then contacts the trackers listed in the torrent file to obtain a list of peers that are sharing the file at the same time. Important terms in BitTorrent are: a *seeder* is a client who has complete chunks/blocks and a *leecher* is a client who download the chunks/blocks. The announce step as shown in Fig.2.1 is also a tool for a peer to get the current state of the tracker e.q list of seeders and leechers. The BitTorrent specification defines the minimum announce interval to be 30 minutes [59]. This process from initial phase to join BitTorrent swarm is shown in Fig.2.1 BitTorrent has generated great enthusiasm for P2P file sharing distribution due to simplicity. Many open source software projects use BitTorrent to distribute their applications.

Besides P2P for filesharing, we noted that P2P for streaming is also on the rise, especially in China. The target of P2P streaming is to build a scalable P2P platform for TV/music delivery. More than a dozen companies are actively working in this area; for example, UUSe, PPLive, and PPStream. Let us take PPLive as an example since it is the most popular P2P streaming service. According to [28] in 2007, the number of concurrent users for the most popular PPLive session rose to 1.5 million and the aggregate bandwidth to 100Gb/s. When the PPLive client is launched by users, it retrieves from a channel server the metadata for all channels. After the user chooses the channel, the PPLive client further talks to a tracker of that channel. Next the trackers will give a list of peers that are watching the same channel. After getting a list of peers, the PPLive client connects to a set of peers, and starts to exchange data. The

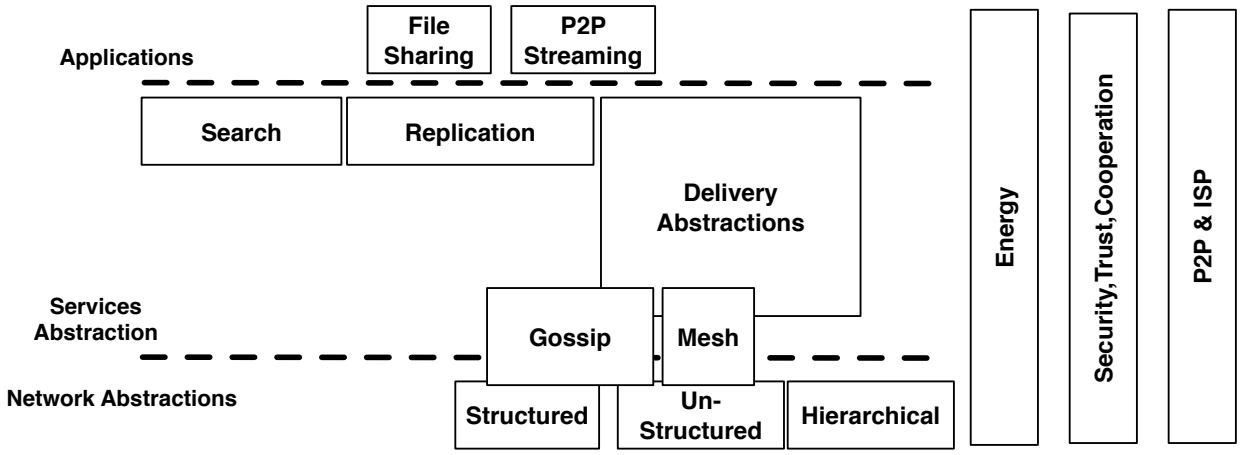

Figure 2.2: P2P Technologies Building Block. This graph is adapted from [53].

challenge in P2P streaming is to provide a sustained streaming bit rate to all peers joining the network. In P2P streaming available bandwidth is very important. Insufficient bandwidth can cause poor video quality.

Figure 2.2 shows the presentation of P2P technologies building block. The bottom level provides the basic networking abstractions, i.e., the P2P overlay networks. The middle level provides additional P2P services for delivery and management (including searching). Above them, we can see P2P applications which work on top of the previous blocks. Additional overlay abstractions can be exist in the middle to help the delivery process, for example, mesh structure and gossip service. Finally, security, trust and cooperation are seen as cross-layer issues including the mismatches between P2P application cooperation and ISPs.

All P2P networks run on top of the Internet. We often consider the P2P network as an overlay network on the top of Internet. We can classify overlays into: *structured*, *unstructured*, and *hierarchical*. In structured overlays, the network is formed in a particular structure. The function of a structured overlay is for storing and locating objects based on a defined virtual address space. The space is organized according to a given geometry which defines

neighborhood relationship between peers. For example, Pastry [57], organized peers in a logical ring and the Pastry's address position in the ring.

In unstructured systems, the unstructured overlays do not form any fixed structure in the network. Unstructured overlay networks are typically close to random graphs. This is due to the nature of every P2P applications that choose neighbors in random. In unstructured P2P networks, peers usually do have not the same role as other peers and a peer can choose any other peers as neighbors with some degree of freedom. In a hierarchical overlay network, the network is built in an unstructured architecture with superpeers. The superpeers form the top level and ordinary peers the bottom level. Peers are organized in different groups and each group can runs its own overlay. A top-level overlay can be formed by one representative per group. This architecture will flexible than a conventional unstructured network. Superpeers as a group can form a different overlay network, either structured or unstructured. Figure 2.3 shows the conceptual of a hierarchical P2P network.

Searching for content in overlay network is one of the key services in P2P. Index is a keyword in searching inside P2P network. In P2P networks, indices can be centralized, localized, or distributed. In structured overlay network centralized indices typically rely on a unique entity. In unstructured overlay network localized indices is usually use for searching. Therefore, a query looking for a particular content must be propagated among peers. However, indexing does not solve the content poisoning problem in P2P.

In a structured overlay networks, due to the fixed structured form, can provide the support for exact match queries. Let us take Distributed Hash Table (DHT) as an example for a structured overlay network [5]. DHT provides a lookup service similar to hash table; (key, value) pairs are stored in a DHT and any peers can efficiently retrieve the value associate with a given key. Therefore a search operation results is fast and efficient.

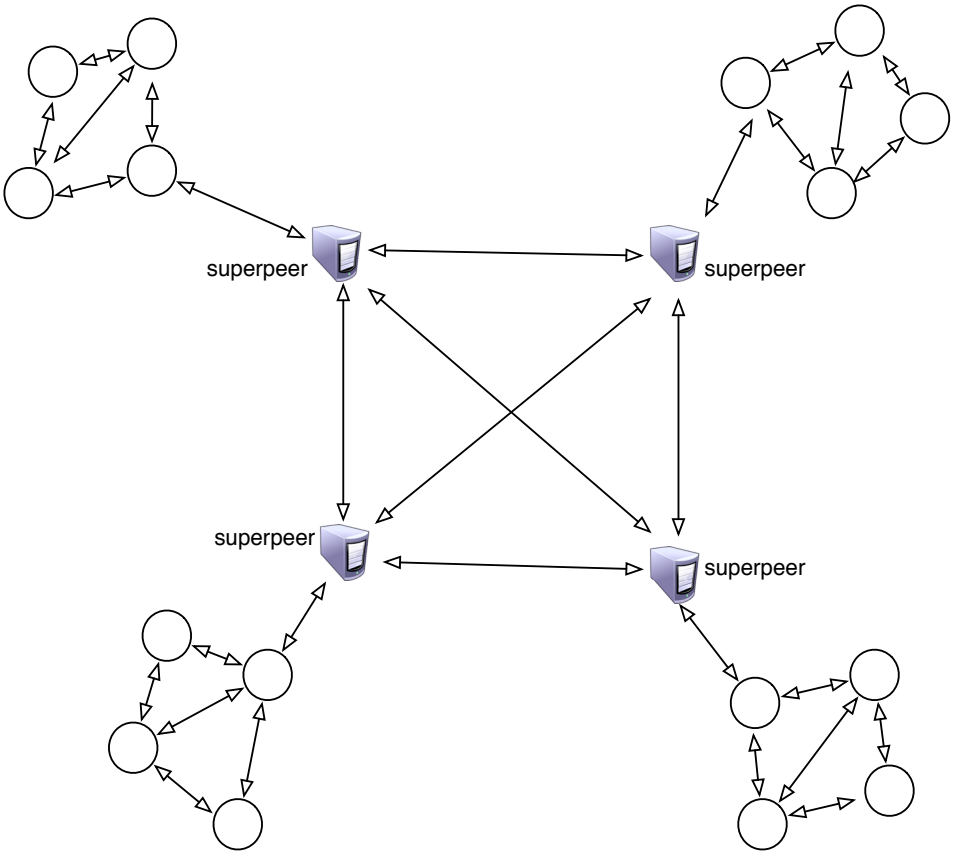

Figure 2.3: Hierarchical P2P Network.

Replication is tied to P2P networks. Take the example of BitTorrent. If a user downloads content in BitTorrent, it means that the user replicated the content. The block of content already downloaded replicated is in his/her PC and that block can be downloaded by other peers. Consistency is important in BitTorrent, that's why BitTorrent provides a hash for every block for checking for the right block and rejecting bad blocks from fake content.

Gossiping is a simple and effective mechanism to spread the content. In P2P overlay networks, each peer participating in a gossiping process contacts other peers at random and exchanges the information between them. Gossiping can be done in push or pull mode. Most P2P applications work in both modes. In BitTorrent, gossiping is done in both push and pull mode and the set of peers to contact is chosen at random.

Another important aspect in security of P2P is trust, particularly establishment of reputation. Establishing the reputation of peers requires collecting information about previous interactions. For example, each BitTorrent tracker usually has a log to record the peers' behavior. If a peer has a bad upload/download ratio, a BitTorrent tracker can blacklist that peer to join the BitTorrent swarm since the tit-for-tat policy of BitTorrent is the basis of cooperation enforcement.

We highlight an important aspect related to the impact of P2P technologies on ISP policies (cross layer issues). P2P solutions are network agnostic, in the sense that they do not take into consideration the layer 3 of the network paths. Having agnostic P2P solutions can cause problems at different levels for ISPs. It has been shown that blind selections of neighbors in P2P networks may result in unnecessary traversal of multiple links inside an ISP. Furthermore, it can significantly impact the shape and amount of traffic among different ISPs, which may be quite different from what was foreseen in ISPs peering agreements. Although, many solutions to this problem was exist, the implementations are still not deployed widely.

While algorithms in P2P systems have been analyzed by many researchers, measurement of P2P is also important. In the cross layer issue, P2P is often blind in selection of neighbors in P2P networks, affecting the underlying ISP's traffic engineering policies. This effect can only be seen if we have good real measurement instead of simulation. In the BitTorrent economic model, researchers want to understand the content publishing phenomena. The growing popularity of BitTorrent is primarily due to the availability of valuable content without any cost for users. However, publishing valuable content has legal implications for the users who publish the content. This raise the question of whether the content publisher is behaving in an altruistic manner for financial incentives. This question can only be answered by doing measurement in real

BitTorrent swarms [13].

Energy was another aspect that has been quite distant from current P2P research but now has begun to rise in importance. While popularity of P2P is declining because users now get content from the cloud, in some regions P2P filesharing applications are still popular. Even now content providers, CDN providers, and ISPs are actively exploring the use of peer-to-peer (P2P) technologies to distribute content to homes as a means of reducing both file download times and the energy consumption of data centers. This approach pushes the energy use out of the data centers and into the homes of content consumers, including P2P based applications that run on mobile devices. With millions of houses connected to the Internet and millions of mobile devices also connected to the Internet, it is very important to consider energy consumption in P2P networks.

2.2 P2P Measurement

BitTorrent is the most successful Peer-to-Peer (P2P) application and was responsible for a major portion of Internet traffic in the past. This has attracted researchers interested in evaluating the performance and the demographic aspects of BitTorrent. An example of current popular application that uses a BitTorrent-like protocol is Spotify, which is a very popular streaming music application in Europe and the US. BitTorrent has been largely studied using simulations, models and real measurements. Although simulations and modeling are easier to perform, they typically simplify the problems and they are likely to miss some of the effects which occur in real BitTorrent swarms. Since many popular applications work based on BitTorrent-like protocols, we will focus on it.

2.2.1 Measuring BitTorrent

In this subsection we describe the BitTorrent measurement techniques defined in the literature so far. We classify them into two main categories, macroscopic and microscopic, depending on the retrieved information. Table 2.1 shows the summary of different techniques.

Table 2.1: Comparison of Measurement Techniques in BitTorrent.

Property	Portal crawling	Tracker crawling	Peers crawler
Scope	macroscopic	macroscopic	microscopic
Information level	torrents level	demographics and general performance	peer level performance
Cost of crawler preparation	low	medium	high
Obtained result details	basic	medium	high
Peers population results	-	high	very high

Macroscopic Technique

The goal of a macroscopic technique is to understand the demographics of the BitTorrent ecosystem, for example the type of published content, the popularity of the content, and the distribution of BitTorrent users per country. The macroscopic measurement allows us to get the performance information such as the ratio of seeder to leechers, the session time of the BitTorrent users, the seedless state (period the torrent is without a seeder) duration, etc. We can classify the macroscopic techniques into two categories: BitTorrent portals crawling and BitTorrent trackers crawling.

- BitTorrent portals crawling: The (major) BitTorrent portals index millions of torrents in a structured way. They provide detailed information about each indexed torrent on a specific torrent web page. Once we know the how BitTorrent portals index the torrents, we can crawl the BitTorrent portals using that index. If we want to analyze the demographics of BitTorrent, we need to crawl a large number of torrents. This takes time

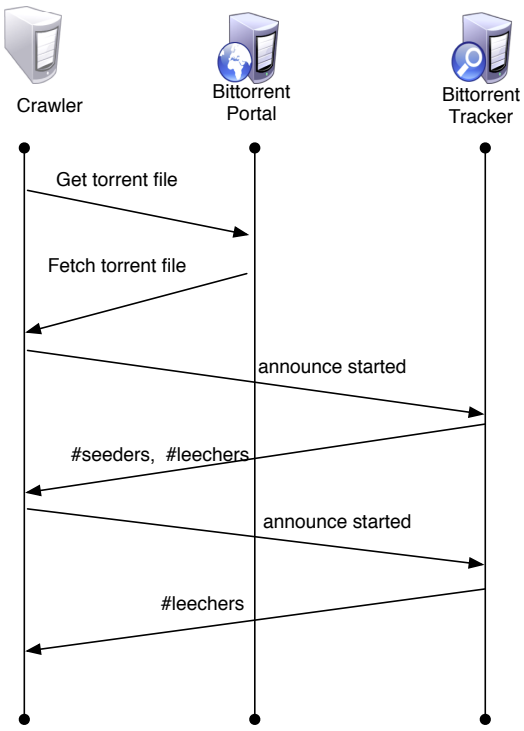

Figure 2.4: BitTorrent tracker crawling: The BitTorrent crawler fetches the torrent file from BitTorrent portals. Using information from the torrent file, the BitTorrent crawler contacts the BitTorrent tracker by sending an announce message. The BitTorrent tracker will reply by with a list of seeders and leechers. This process is repeated until all the peers in the swarm are obtained.

since millions of torrents exist in BitTorrent portals and every second a new torrent can be published to BitTorrent portals (including fake torrents). By processing the data from the BitTorrent crawling, we can get the information related to BitTorrent demographics, for example content popularity distribution, distribution of published content, and publishing rate of new torrents. It unfortunately, some BitTorrent portals do not give detail information about the torrent information.

- BitTorrent trackers crawling: While crawling BitTorrent portals can give us information about the torrent type and torrent publisher and some number of seeders and leechers, that technique is not sufficient for more detailed information such as the distribution of users per country or

performance- relevant information such as peer arrival rate and peer session time. To get that information, we need to collect the IP address of seeders and leechers in the BitTorrent swarms. This information can only be gotten from BitTorrent tracker logs by asking BitTorrent tracker owners or by crawling the BitTorrent tracker. Figure 2.4 shows the schematic of BitTorrent tracker crawling. First, a BitTorrent crawler needs to download a torrent file from a BitTorrent portal. Next, a BitTorrent crawler send an announce message to a BitTorrent tracker and the BitTorrent tracker will reply with a list of seeders' IP addresses and a list of leechers' IP addresses. The BitTorrent crawler then sends an announce message again to the BitTorrent tracker. Since the BitTorrent tracker has recorded the IP address of the BitTorrent crawler, the BitTorrent tracker will reply with a list of leechers' IP addresses only. This process is repeated as many times as possible to get a full list of leechers's IP addresses. If a BitTorrent crawler send too many announce messages, the BitTorrent tracker will block the BitTorrent crawler. Hence, this technique must be done in a friendly manner. Another challenge is that BitTorrent peers do not have permanent peer-ids. Every time a BitTorrent client is started a new random peer-id is generated thus it is very difficult to follow a peer using its peer-id. Also since most of the BitTorrent clients are home users with the IP addresses assigned dynamically by their ISP, identifying peers by their IP address can introduce inaccuracies.

Microscopic Technique

While in macroscopic techniques we can get the peers' IP level information, that information is not sufficient to infer more detailed performance metrics at the peer level such as peers' download and upload rates, and how peers

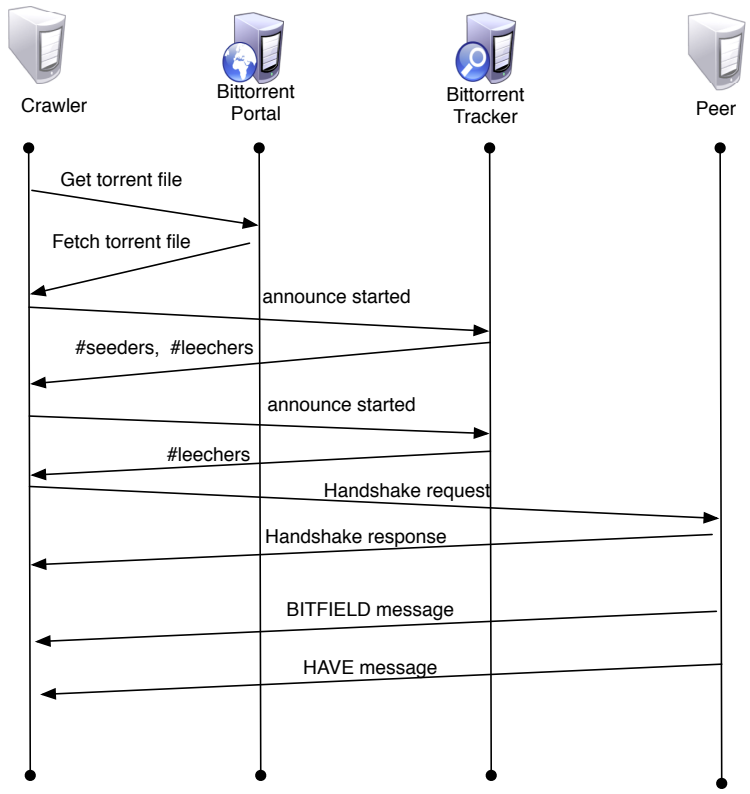

Figure 2.5: Peer crawling: The initial process is the same as the BitTorrent tracker crawling with additional steps for the BitTorrent crawler to contact all the peers in the swarm by sending handshake requests. The contacted peers will reply by sending BITFIELD messages and HAVE messages.

connect each other (the graph of the network). To get this information, we need more sophisticated techniques that implement the wire level BitTorrent protocol since the crawling techniques in this case require us to join a BitTorrent swarm directly. Figure 2.5 shows the schematic functionality of peer crawling. In this peer crawling technique, a BitTorrent crawler gets the IP address of peers participating in a swarm. The initial process is the same as tracker crawling. Afterwards, the BitTorrent crawler contact each peer and performs the handshake procedure. The handshake message must be the first message sent by a BitTorrent client to other peers. The contacted peers will reply by sending BITFIELD and HAVE messages. From the handshake response we can determine if the peer is using a public IP address or is behind

NAT. Basically if the contacted peer responds to the BitTorrent crawler, then that peer is using a public IP address space and if that peer is silent, then it is behind a NAT. From the BITFIELD message we can get information about the peers' type: seeder or leecher. By measuring the time between the reception of two consecutive HAVE messages from a peer, the BitTorrent crawler can calculate the time needed to download a chunk. Chunk size information itself is always available in torrent file. Thus, dividing the size of the chunk by the time need to download two consecutive HAVE messages, we can infer the instantaneous download rate of a peer. The BITFIELD message contains information about chunks that already are held by peer in the swarm. Thus by comparing BITFIELD message for every peer in the swarm, we can infer the chunk distribution. Challenges in this technique, include the fact that the BitTorrent crawler can be blocked by a peer because it sends too many handshake messages and the possibility of receiving many BITFIELD and HAVE messages from a peer but the BitTorrent crawler does not upload any information about the peer.

2.3 Energy Aspect of P2P Network

The work of Gupta and Singh [27] on green networking has received a lot of attention. In recent years, much effort has been expanded reducing energy expenditure. Big companies and data center operators are trying new technologies to consume less energy. For example, Google is planning to operate its data centers with a zero carbon footprint by using hydro-power, water-based chillers, and external cold air for cooling its data center. While energy is mostly a concern for companies that run hundreds of servers and hundred of networking devices inside a data center, only a little attention has been paid to home users. Recalling that P2P applications were the largest fraction of

the Internet traffic in the past, we still have many popular P2P applications. For example, Spotify is a fee based music streaming service for mobile and desktop. It is very popular in Europe and USA. Spotify works like BitTorrent though there are several differences. These figures show that making P2P applications more energy efficient is important. These days, ISPs have evolved from providing basic Internet connectivity to offering higher valued services such as Internet+TV+phone. In this value added service, the set top boxes (STB) or home gateways play an increasingly important role. Modern STBs are as powerful as desktop PCs while the form factor is quite small compared to a desktop PC. However, this power translate to higher power consumption. Since broadband home Internet users are increasing, consideration to reduce energy consumption of STB is required.

Several approaches have been considered to reduce energy consumption. For example: Dynamic Voltage Scaling (DVS) and Dynamic Frequency Scaling (DFS) can be used. With DVS, the supply voltage is reduced when the workload is low, hence DVS does not make the circuit slower unless the user do other things at the same time. DFS can reduce the number of processor instructions, thus reducing performance. Another approach is designing new network architectures. For example, placing optical amplifiers at the most important locations and the task functions near renewable sources. Performing resource consolidation, capitalizing on available energy is also considered as a way to reduce energy consumption. This can be done via traffic engineering. Another way is by migrating computation or delegate the workload, typically using virtualization to move workloads transparently.

At the network level, migrating computation tasks is another way to reduce energy consumption. For example in [20], Gianetti et al. propose the usage of a BitTorrent proxy for delegating the BitTorrent client work. Delegating some work to a BitTorrent proxy can save up to 25% of energy usage for PC

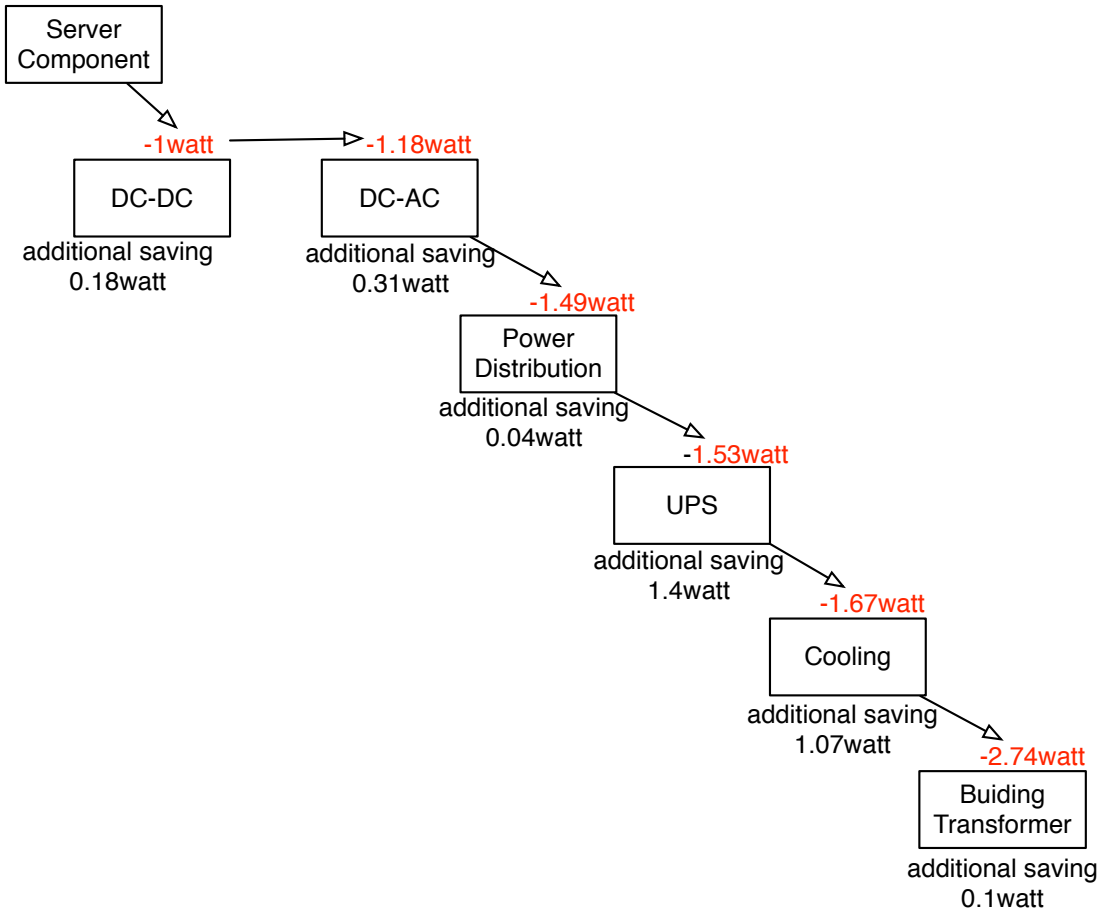

Figure 2.6: Cascade effect: 1 watt energy saving in server component can reduce 2.8 watt energy in total facility level. This figure adapted from [34]

desktop used for BitTorrent file sharing. Delegating or proxying workload is not a new idea. An example of a more primitive form of proxying is Wake on LAN technology, introduced almost two decades ago [33]. Wake on LAN technology allows a computer to be awakened by a *magic* packet.

Since energy flow to consumers is based on supply and demand, saving of small amount of energy in homes can affect the distribution side of energy. This is known as the cascade effect. It not only happens in homes and their distribution, but it also appears in data centers and their power distribution.

2.4 Related Work

BitTorrent protocol performance has been explored extensively [25] [41] [54] [67] [42] [80]. Although we know that the topology can have a large impact on performance, to date only a few papers have addressed the issue. Urvoy *et al.* [68] used a discrete event simulator to show that the time to distribute a file in a BitTorrent swarm has a strong relation to the overlay topology. Al-Hamra *et al.* [2], also using a discrete event simulator, showed that BitTorrent creates a robust overlay topology and the overlay topology formed is not random. They also showed that peer exchange (PEX) generates a chain-like overlay with a large diameter. Dale *et al.* [14], in an experimental study on PlanetLab, showed that in the initial stage of BitTorrent a peer will get a random peer list from the tracker. They found that a network of peers that unchoked each other (uploaded chunk to another peers) is scale-free and the node degree follows a power-law distribution with exponent approximately 2. Dale *et al.* [14] also showed that the path length formed in BitTorrent swarms averages four hops and BitTorrent swarms have low average clustering coefficient. However, little work has been done on confirming that such controlled experiments correspond to the system in the real world.

We emphasize that compared to Hoßfeld *et al.* [29], our work provides a completely different approach and goal. Hoßfeld *et al.* [29] discuss the AS (Autonomous System) level topology of BitTorrent swarm for optimizing overlay traffic across ASes, while our study focuses on microscopic dynamic aspects of the BitTorrent swarms topology itself (at the peer level or IP address level). The closest work to ours is Kryczka *et al.* [39] published shortly after our own work [17]. While our method is somewhat similar to theirs, they focus on clustering and locality while our focus is on node degree and clustering. They use PEX to discover peer relationship, unfortunately they do not explain in detail how to process the PEX data set. Because of differing PEX imple-

mentations between BitTorrent clients, we need to be careful with it in data processing. In our work, we describe PEX behavior and its limitation on two popular BitTorrent clients: Vuze and uTorrent, and we also explain how to treat PEX data from different BitTorrent clients. We also provide simulation result to confirm that our methods for inferring peer relationship with PEX is valid. They observed that BitTorrent swarms have slightly higher clustering coefficient compared to random graphs of the same size but neither BitTorrent swarm fulfills the properties of small world. The slight difference in clustering comes from the difference of PEX data processing. They assume that PEX is the same and complete for all BitTorrent clients.

Our results agree with previous research [14] in some areas and disagree in others, perhaps for two reasons. First, power-law claims must be handled carefully. Many steps are required to confirm the power-law behavior, including alternative model checking, and we must be prepared for disappointment since other models may give a better fit. Second, our methodology relies on real work measurement combined with simulation for validation. This real-world measurement will reflect different types of clients connected to our swarm, and each client has a different behavior. We also face difficult-to-characterize network realities such as NAT and firewalls. Our ability to reproduce key aspects of the topology dynamics suggests that these factors have only limited impact on the topology, somewhat to our surprise.

A Content delivery network (CDN) is a large distributed system of servers deployed in multiple data center across the Internet [50]. The objective of CDN is to serve content to end-users high availability and low latency manner as CDNs are distributed geographically. A CDN is owned, deployed, and maintained by a company that charges content providers or website owners for its services. CDNs with peer assist have been successfully deployed on the Internet, such as Akamai [32] and LiveSky [79]. The authors of [32] conclude

from two real world traces that hybrid CDN-P2P can significantly reduce the cost of content distribution and can scale to cope with the exponential growth of Internet video content. Yin et al. [79] described LiveSkye as commercial operation of a peer-assisted CDN in China. LiveSky solved several challenges in the system design, such as dynamic resource scaling of P2P, low startup latency, ease of P2P integration with the existing CDN infrastructure, and network friendliness and upload fairness in the P2P operation. Measurement from LiveSky showed that LiveSky can save around 40% of the bandwidth [79]. The author in [31] and [30] proposed mechanisms to minimize CDN server bandwidth to make the content distribution cheap. They designed different peer prefetching policies of video on demand system in surplus mode while ensuring user quality of experience. A similar analysis has been done in [77] for a video streaming system where the authors proposed different limited peer contribution policies to reduce CDN bandwidth requirement and eventually off load the distribution process from the CDN to the P2P system. An ISP friendly rate allocation schemes for a hybrid CDN-P2P video on demand system was proposed in [74]. These techniques try to minimize CDN server bandwidth while reducing ISP unfriendly traffic and maximizing peer prefetching. Load on the CDN servers has been shown to be reduced using this approach while reducing cross ISP traffic. The above studies were performed for video on demand or live video streaming. While video is the most popular content, the systems can be also used for other type contents. Moreover, while content based services are growing, energy consumption of a content distribution system has not been analyzed.

Related to CDN and energy usage, in a seminal work [56], the authors show that if costs are based on electricity usage and if the power prices vary in real-time, global load balancing decision can be made such that users are routed to locations with the cheapest power without significantly impacting

user performance or bandwidth cost. In [52], the author proposed utilizing batteries for CDN for reducing total supplied power and total power costs. The authors in [52] also proposed battery provisioning algorithms based on workload of CDN server. The result shows that batteries can provide up to 14% power savings.

The idea to utilize the ISP controlled home gateway to provide computing and storage services and adopt a managed peer-to-peer model is presented in [71]. Valancius et al. [71] show that distributing computing platform in NaDa (Nano Data Center) saves at least 20-30% of the energy compared to traditional data centers. The saving in NaDa comes from taking advantage of the underutilized home gateway, avoidance of cooling costs, and the reduction of network energy consumption as a result of demand and service co-localization.

The comparison between CDN architecture and peer-to-peer architecture is discussed in [6] and [18]. The authors of both papers agree that CDN architecture is more energy saving compare to peer-to-peer architecture. Another interesting study of energy efficiency in content delivery architectures is presented by Guan et al. [23]. Guan et al. [23] compare energy efficient of CDN architecture and CCN architecture. CCN is a new architecture to deliver the contents in the Internet [36]. CCN uses a data storage cache at each level of the network, e.g routers, to decrease the transmission traffic and also increase the speed of response. The authors in [23] conclude that CCN is more energy efficient in delivering popular content while the approach with optical bypass is more energy efficient in delivering infrequently accessed content.

To the best of our knowledge, the only study of energy considering peer-to-peer in CDN architecture is presented in [45]. Mandal et al. [45] mentioned that hybrid CDN-P2P systems can reduce the energy in the optical core network by around 20-40%. The authors only considered energy consumption of hardware

especially optical devices and do not include overhead inside the data center, CDN server energy consumption, and power consumed by peers. Our work is quite different , We take a number of peers with static content and add the overhead of the data center cause by temperature of hardware for different purposes, for live streaming and video on demand [16].

Chapter 3

Characteristics of BitTorrent Swarms

3.1 Introduction

BitTorrent is one of the most used application in the current Internet and is responsible for an important portion of the upstream and downstream traffic as revealed by recent reports. The significant footprint of BitTorrent in the Internet has motivated researchers and practitioners to dedicate an important amount of effort to understanding and improving BitTorrent. However, despite this effort, we still have little knowledge regarding the connectivity properties exhibited by real BitTorrent swarms at both swarm and peer level [17]. Due to the difficulty in collecting the required information from real swarms, most of the existing works that analyze connectivity properties are based on simulations or experiments in controlled environments. As a result, they are likely to miss some of the effects affecting BitTorrent swarms in the wild. The analysis of the peer level connectivity in real BitTorrent swarms can reveal important information such as: (1) efficiency of a swarm to disseminate the content; (2) the resilience of the swarm, (3) checking the locality-bias. Urvoy Keller et

al. [68] shows that number of connections from a peer maintains high entropy of the BitTorrent system thus the BitTorrent works in optimal. Resilience of the swarm is very important for content provider who use BitTorrent like protocol to disseminate their contents. The result of the information about how peer connect each other can be used for doing network partition. For example, we want to know what's happened if high degree peers are remove randomly?; what's happened with peers are removed randomly? will the swarm collapse? or will it decrease download performance? Some ISPs have started to introduce a policy to minimize the impact of cross AS traffic caused by P2P. This effect of locality will be much better observed if we have peer connectivity level properties information by measuring real BitTorrent swarms topology.

In this Chapter, we first present a methodology to collect the connectivity information at both the swarm and the peer level for the entire lifespan of a real torrent. Specifically, we discover new torrents just after their birth by using the RSS service of the most important BitTorrent portal, namely the Pirate Bay. Afterwards, we exploit the Peer Exchange (PEX) extension of the BitTorrent protocol to gather the set of neighbors for each peer. PEX is a gossiping technique which main goal is to allow peers to exchange their list of neighbors so that they can learn about other participants in the swarm without contacting the tracker. Note, that PEX has been implemented by most of the existing BitTorrent clients and in particular by the most popular ones such as uTorrent or Vuze. The information collected from PEX (i.e., a peer' s neighborhood) is the connectivity information at the peer level. Furthermore, by aggregating the neighborhood information collected from every peer in a swarm we are able to build the overlay topology of that swarm (i.e., swarm level connectivity). We retrieve the information from each active peer every 3 minutes and then study the dynamic evolution of both: the overlay topology of the swarm and the composition of each peer' s neighborhood. We have applied

the described methodology to collect the connectivity information of 50 real torrents, including more than 150 peers, since their birth during a period of 10 days.

3.2 Measurement Methodology

The difficulties in inferring topologies in BitTorrent swarms are caused by the BitTorrent mechanism itself. First, although a BitTorrent *peer* may offer some information about its peers, there is no mechanism for peer *discovery*. Second, a peer never sends information about its connections with other peers to the tracker, so we cannot infer overlay topologies by querying or hosting a tracker. Our other options inferring topologies are simulation or deploying BitTorrent nodes in a real network or in a laboratory, e.g., PlanetLab. Deploying hundreds to thousands of nodes in a real network or in the laboratory in a manner that accurately reflects the real world is a very challenging task.

We used PEX to collect information about peer neighbors (see Fig.3.1), and then we describe the network formed in terms of properties such as node degree and average clustering. Besides collecting data from real BitTorrent networks, we ran simulations similar to these of Al-Hamra *et al.* [3]. In these simulations, we assumed that peer arrivals and departures (churn) follow an exponential distribution as explained by Guo *et al.* [25]. For simplification, we assumed that nodes are not behind a NAT. Since we are only interested in the construction of the overlay topology, we argue that our simulations are thorough enough to explain the overlay properties.

Temporal graphs have recently been proposed to study real dynamic graphs, with the intuition that the behavior of dynamic networks can be more accurately captured by a sequence of snapshots of the network topology as it changes over time [22] [65]. In highly dynamic networks such as P2P, an in-

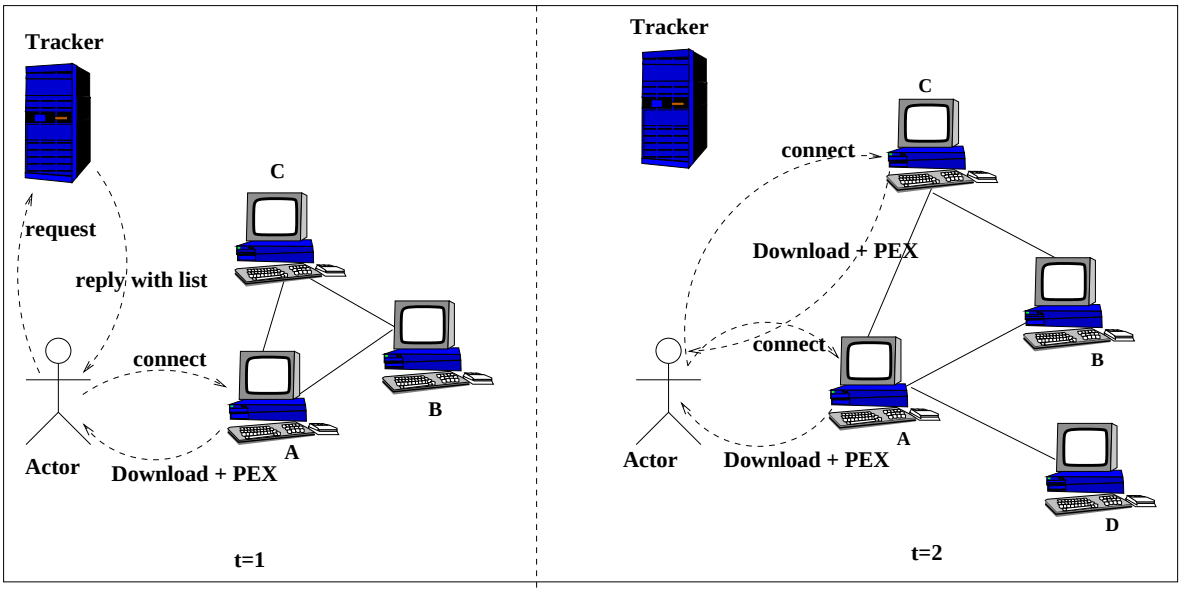

Figure 3.1: Simplified view of our approach. Left: At time $t = 1$, the actor gets a PEX message from peer A and learns that peer A is connected to peer B and C. At $t = 2$, the actor gets PEX messages from peers C and A. The actor learns that now peer A is connected to peer D. Thus the actor knows the properties of peer A at $t = 1$ and $t = 2$.

stantaneous snapshot may capture only a few nodes and links. In this chapter, we study the network dynamics by continuously taking network snapshots over a short time interval Δ, and show them as a time series. A snapshot captures all participating peers and their connections within a particular time interval, from which a graph can be generated. The snapshot duration may have minor effects on analyzing slowly changing networks. However, in a P2P network, the characteristics of the network topology vary greatly with respect to the time scale of the snapshot duration [62]. We consider $\Delta = 3$ minutes to be a reasonable estimate of minimum session length in BitTorrent [60].

3.2.1 Graph Sampling

Due to the large and dynamic nature of BitTorrent networks, it is often very difficult or impossible to capture global properties. Facing this difficulty, sampling is a natural approach. However, collecting unbiased or representative

sampling is also sometimes a challenging task.

Suppose that a BitTorrent overlay network is a graph $G(V,E)$ with the peers or nodes as vertices and connections between the peers as edges. If we observe the graph in a time series, i.e., we take samples of the graph, the time-indexed graph is $G_t = G(V_t, E_t)$. From this set of graphs, we can define a measurement window $[t_0, t_0 + \Delta]$ and select peers uniformly at random from the set: $V_{t_0,t_0+\Delta} = \bigcup_{t=t_0}^{t_0+\Delta} V_t$. In that equation, we cannot distinguish properties of the same peer at different times, thus it focuses on sampling peers instead of peer properties. Stutzbach *et al.* [61] showed that equation is only appropriate for exponentially distributed peer session lengths but we know from existing measurement that BitTorrent networks peer session lengths have very high variation [25].

For example: suppose we want to measure number of files shared by peers in BitTorrent swarm. In this BitTorrent swarm, half of the peers are up all the time and have many files, while other peers remain around for one minute and are immediately replaced by new short-lived peers who have few files. The common method is to observed the system for a long time and sample the peers randomly. This method will incorrectly conclude that most of the peers in the system have very few files. The problem with this method is that it focuses on sampling the number of peers instead of peer properties. Our objective should not be to select a vertex $v_i \in \bigcup_{t=t_0}^{t_0+\Delta} V_t$, but to sample the property of a vertex v_i at a particular instant time t. Therefore, we must distinguish the occurrences of the same peer at different times: samples $v_{i,t}$ and $v_{i,t'}$ gathered at different times $t \neq t'$ are viewed as different, even from the same peer. The key in this method is that we must be able to sample from the same peer more than once at different points in time. Thus we can formalize this into $v_{i,t} \in V_t, t \in [t_0, t_0 + \Delta]$ [61]. With that method, the sampling will not biased because we track the peer's properties overtime.

The number of peers in a swarm that is observed by our client is our population. The sampled peers set is the number of peers that exchange PEX messages with our client. Our sampled peers set through PEX messages exchange can observe about 70% of the peers in a population. This observation is consistent with [76].

3.2.2 Experimental Methodology

We joined the top 50 TV series torrents from the piratebay, which claims to be the biggest torrent tracker on the Internet. Almost all of these torrents were in steady-state phase, which is more dominant than the bootstrapping and decay phases of a torrent's lifetime. We used a modified rasterbar libtorrent [49] client that is connection greedy, where the client tries to connect to all peers it knows without a limit on the number of connections, and the client logs PEX messages received from other clients. PEX messages from old versions of Vuze BitTorrent clients contain all of peers they connected to in the past, hence these clients should be removed from the data. Removal of some peers in data processing is valid in terms of sampling with dynamics, see Sect.(3.2.1). In terms of connectivity, two popular BitTorrent clients (uTorrent and Vuze), by default try to connect to peer candidates randomly without any preference, thus we have random data sets. This implies that our data set is independent of measurement location and the number of measurement locations.

3.2.3 Data Analysis Background

Many realistic networks exhibit the scale-free property [12], though we note that *scale-free* is not a complete description of a network topology [15] [43]. It has been suggested that BitTorrent networks also might have scale-free characteristics [14]. In this chapter, we test this hypothesis. Besides testing this hypothesis, we also study the clustering property of BitTorrent swarms.

In a scale-free network, the degree distribution follows a power-law distribution. A power-law distribution is quite a natural model and can be generated from simple generative processes [46], and power-law models appear in many areas of science [12] [46].

A power-law distribution can be described as:

$$Pr[X \geq x] \propto cx^{-\alpha}. \tag{3.1}$$

where x is the quantity of distribution and α is commonly called the scaling parameter. The scaling parameter usually lies in the range $1.8 < \alpha < 3.5$. In discrete form, the above formula can be expressed as:

$$p(x) = Pr(X = x) = Cx^{-\alpha}. \tag{3.2}$$

This distribution diverges on zero, therefore there must be a lower bound of x, called $x_{min} > 0$, that holds for the sample to be fitted by a power-law. If we want to estimate a good power-law scaling parameter then we must also have a good x_{min} estimation.

Normalizing (3.2) we get

$$p(x) = \frac{x^{-\alpha}}{\zeta(\alpha, x_{min})}. \tag{3.3}$$

The most common way to fit empirical data to a power-law is to take the logarithm of Eq.(3.1) and draw a straight line on a logarithmic plot [46]. We use maximum likelihood to estimate the scaling parameter α of power-law [12]. This approach is accurate to estimate the scaling parameter in the limit of large sample size. For the detailed calculations of both x_{min} and α, see Appendix B in [12].

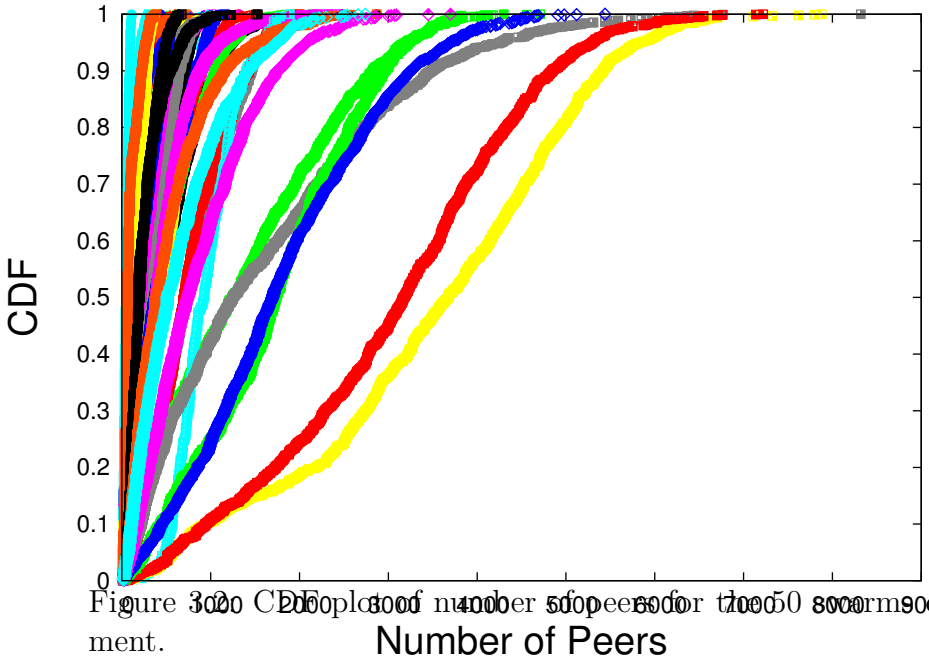

Figure 3.2: CDF plot of number of peers for the 50 swarms during measurement.

3.3 Experimental Results

Our time samples for the size of swarms are plotted as the CDF of the number of peers for every swarm during measurement with 104 to 1400 time samples for each torrent, as shown in Fig.3.2. It is clear that the number of peers has high variability due to churn in BitTorrent networks.

3.3.1 Power-law Distribution of Node Degree

The degree of a node in a network is the number of edges connected to that node. If we define p_k as the fraction of nodes in the network that have degree k, then p_k is the probability that a node chosen uniformly at random has degree k. We show node degree data in cumulative distribution function (CDF) plot, which can be expressed as

$$P_k = \sum_{k'=k}^{\infty} p_{k'}. \tag{3.4}$$

We want to know the power-law distribution of the measured BitTorrent networks, and we do not know *a priori* if our data are power-law distributed. To test the applicability of a power-law distribution, we use the goodness-of-fit test as described by Clauset *et al.* [12]. First, we fit data to the power-law model and calculate the Kolmogorov-Smirnov (KS) statistic for this fit. Second, we generate power-law synthetic data sets based on the scaling parameter α estimation and the lower bound of x_{min}. We fit the synthetic data to a power-law model and calculate the KS statistics, then count what fraction of the resulting statistics is larger than the value for the measured data set. This fraction is called the p value. If $p \geq 0.1$ then a power-law model is a good model for the data set, and if $p < 0.1$ then power-law is not a good model.

As mentioned before, a good estimation for x_{min} is important to get a overall good fit. Too small an x_{min} will cause a fit only to the body of the distribution. Too high an x_{min} will cause a fit only to the tail of the distribution. Figure 3.3 illustrates the fit for snapshots of *torrent1* and *torrent3*. For *torrent1*, setting $x_{min} = 2$ leads to $\alpha = 2.11$, while $x_{min} = 1$ gives $\alpha = 2.9$. For *torrent1*, xmin = 1 visually does not give a good fit, while for *torrent3*, setting $x_{min} = 1$ leads to a visually good fit.

Figure 3.4 shows the CDF for p values for all data sets. This figure shows that from the K-S statistics point of view, around 45% of the time a power-law distribution is not a good fit for the data. The inset figure in Fig.3.4 shows the CDF plot p value for each torrent. The dash line on p value = 0.1 is the threshold.

However, these data sets must be interpreted with care. The usage of the maximum likelihood estimators for parameter estimation in power-law is guaranteed to be unbiased only in the asymptotic limit of large sample size, and some of our data sets fall below the rule of thumb for sample size, $n = 50$ [12]. In the goodness-of-fit test, a large p value does not mean the power-law is the

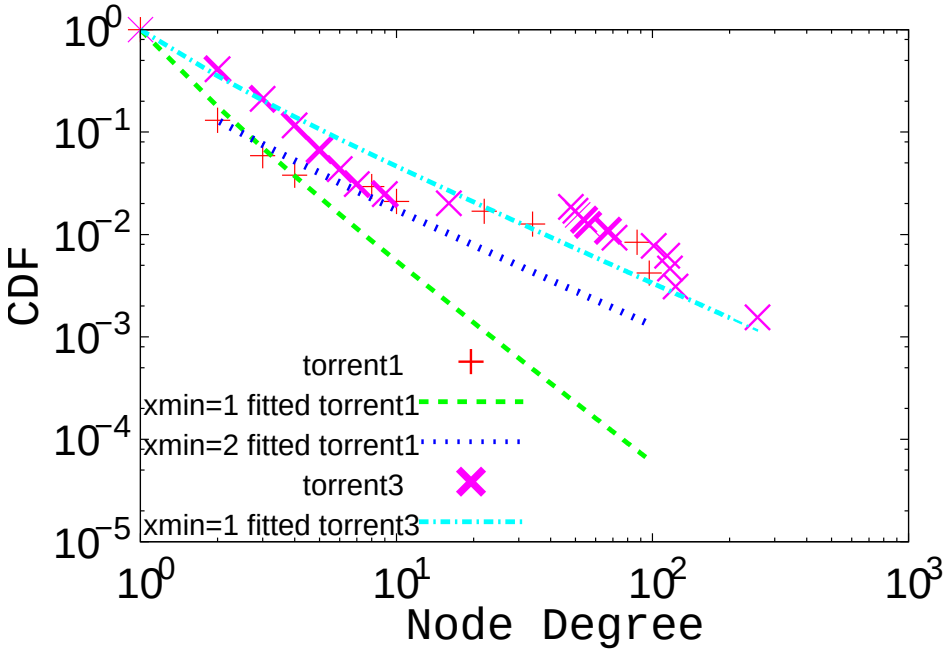

Figure 3.3: Node degree fit for snapshots of two torrents, with three fits shown in log scale.

correct distribution for data sets, because there may be other distributions that match the data sets and there is always a possibility that even with small value of p the distribution will follow a power-law [12]. We address these concerns next.

3.3.2 Alternative Distributions

Even if we have estimated the power-law parameter properly and the fit is decent, it does not mean the power-law model is good. It is always possible that non-power-law models are better than the power-law model. There are several methods for direct comparison of two distributions such as *likelihood ratio test* [73], *Bayesian test*, and *Minimum description length*. The likelihood ratio test idea is to compute the likelihood of the data sets under two distributions. The one with the higher likelihood is the better fit. We use the likelihood ratio test to see whether other distributions can give better parameter estimation.

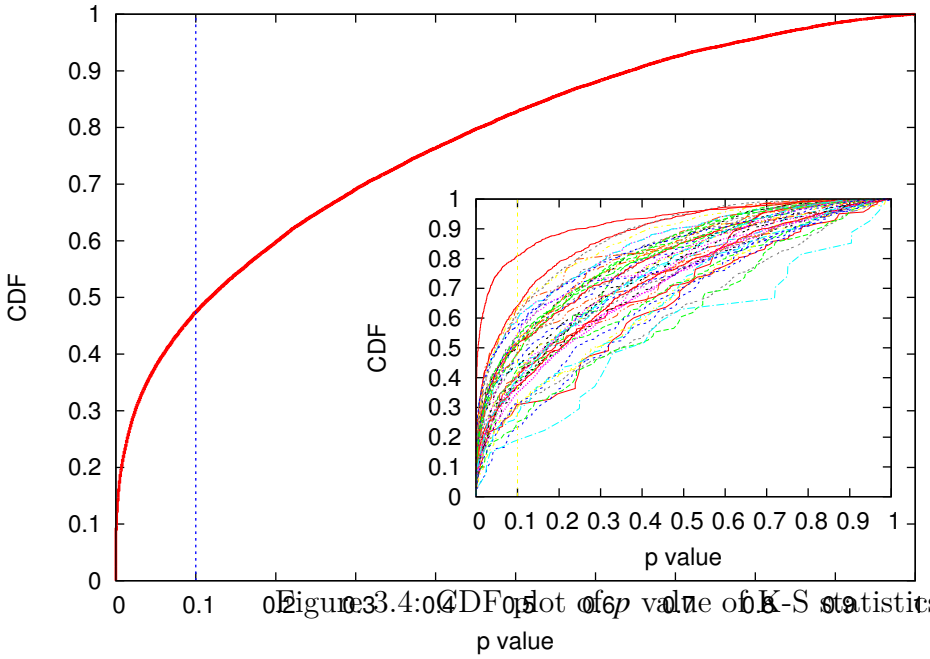

Figure 3.4: CDF plot of p value of K-S statistics.

Nested Case

We now hypothesize that the smaller family of power-law distributions may give a better fit to our data sets. We only consider a power-law model and a power-law with exponential cut-off model as examples to show model selection. Model selection for power-law model and power-law with exponential cut-off is a kind of nested model selection problem. In a nested model selection, there is always the possibility that a bigger family (power-law) can provide as good a fit as the smaller family (power-law with exponential cut-off). In a likelihood ratio test, we must provide the significance value (ρ value). Under the likelihood ratio test, we compare the pure power-law model to power-law with exponential cut-off, and the ρ value here helps us establish which of three possibilities occurs: (i) $\rho > 0.1$ means there is no significant difference between the likelihood of the data under the two hypotheses being compared and thus neither is favored over the other; if we already rejected the pure power-law model, then this does not necessarily tell us that we also can reject

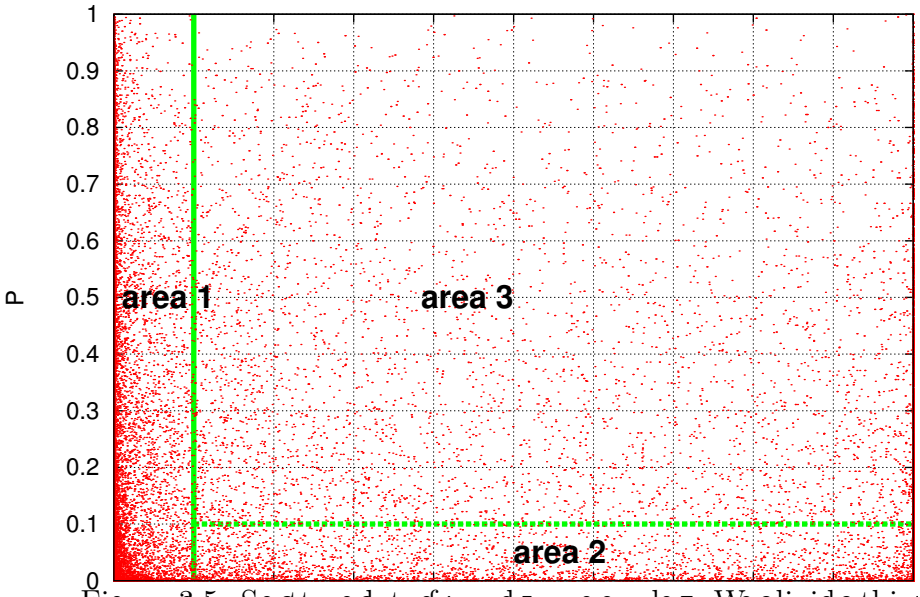

Figure 3.5: Scatter plot of p value vs ρ value. We divide this figure into three areas where the borders are vertical line $\rho = 0.1$ and horizontal line $p = 0.1$. p is goodness-fit of test for power-law model, if $p \geq 0.1$ power-law model is good model for data set. ρ is significance test for nested hypothesis. if $\rho \geq 0.1$ then there is no significant difference between the likelihood of the data under the two hypotheses being compared. if $\rho \leq 0.1$ there is significant difference between the likelihood of the data under the two hypotheses being compared. 52% of points lie in area 1, thus an alternative model may be plausible for these points.

the alternative model; (ii) $\rho < 0.1$ and the sign of likelihood ratio (LR) = negative means that there is a significant difference in the likelihoods and that the alternative model is better; if we have already rejected the pure power-law model, then this case simply tells us that the alternative model is better than the bad model we rejected; (iii) if $\rho < 0.1$ and the sign of LR = positive means that there is a significant difference and that the pure power-law model is better than the alternative; if we have already rejected the pure power-law model, then this case tells us the alternative is even worse than the bad model we already rejected.

Figure 3.5 shows a p value vs ρ value scatter plot, divided into three areas.

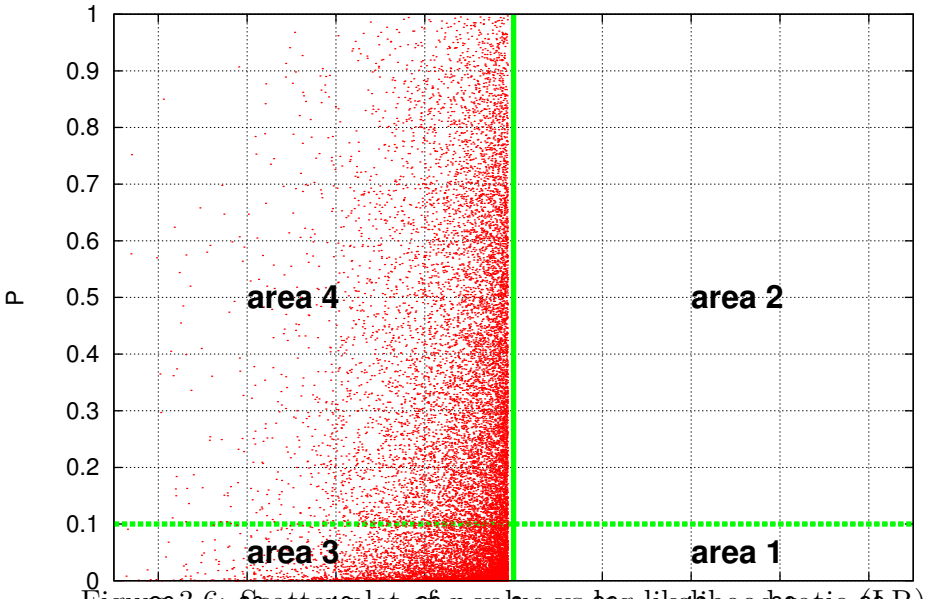

Figure 3.6: Scatter plot of p value vs log-likelihood ratio (LR) for $\rho < 0.1$. We divide this figure into four areas where the borders are vertical line $LR = 0$ and horizontal line $p = 0.1$. p is goodness-fit of test for power-law model, if $p \geq 0.1$ power-law model is good model for data set. ρ is significance test for nested hypothesis. LR is the sign of likelihood ratio value. if LR is negative, there is a significant difference in the likelihoods and the alternative model is better. 58% of points lie in area 3 and 42% lie in area 4. Because area 3 and 4 are negative value of LR, the alternative model is better.

Area 1: $\rho < 0.1$ and $p > 0$. Area 2: $\rho > 0.1$ and $p < 0.1$. Area 3: $\rho > 0.1$ and $p > 0.1$. This figure shows that 52% of the samples lie in area 1, thus an alternative model may be plausible for these samples.

Now we plot p value vs LR as shown in Fig.3.6 for $\rho < 0.1$. We divide the figure into four areas: area 1, area 2, area 3, and area 4 with green lines as borders to see how sparse the points are in each area. Area 1: LR=positive sign and $p < 0.1$. Area 2: LR=positive sign and $p > 0.1$. Area 3: LR=negative sign and $p < 0.1$. Area 4: LR=negative sign and $p > 0.1$. In this figure, 58% of the samples lie in area 3 and 42% lie in area 4, while there are no samples in areas 1 and 2, which means that the alternative model is better. Although in the case $p < 0.1$ we reject power-law as the plausible model, the alternative model

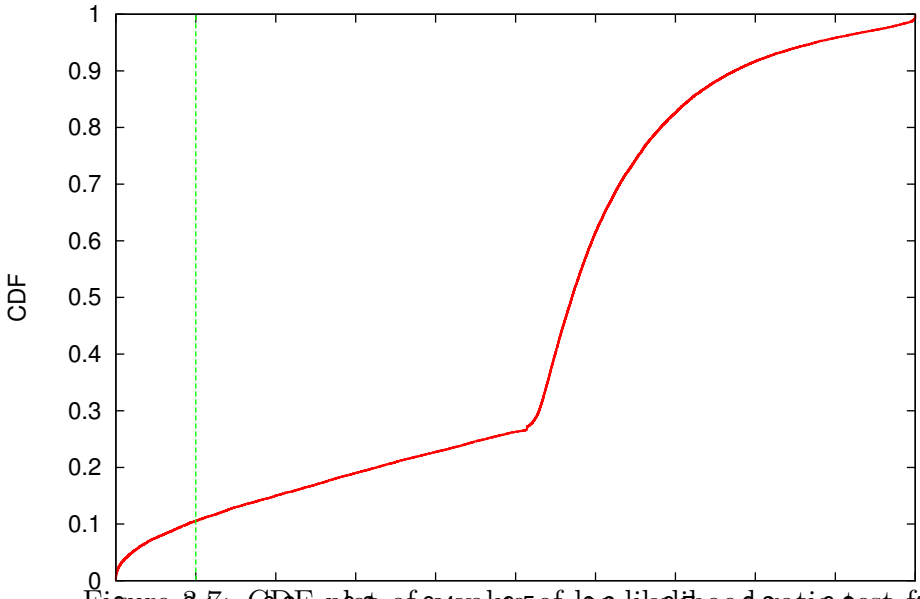

Figure 3.7: CDF plot of ρ value of log-likelihood ratio test for power-law v.s log-normal. We divide the figure into two areas with vertical line $\rho = 0.1$ as border. Under nested hypothesis, ρ value is significance value of observed sign of likelihood ratio. if $\rho \leq 0.1$ then the sign of log-likelihood ratio is a good indicator of which model is the better fit to the data. In this figure, only 13% of ρ values are less than 0.1. Since 87% of ρ values are more than 0.1, the sign of log-likelihood ratio is not good indicator and the test does not favor either model over the other.

is still better than the power-law model. We believe that these results are caused by peers that are not willing to maintain large numbers of concurrent connections (high node degree). These observations clearly demonstrate that comparing models is a very complex task in highly dynamic networks.

Non-Nested Case

In the non-nested case, we compare power-law distribution with log-normal distribution and exponential distribution. We calculate the ratio of two likelihood distributions or the logarithm of the ratio, which is positive or negative depending on which distribution is better. The positive or negative sign of log-likelihood ratio does not definitively indicate which model is the better fit.

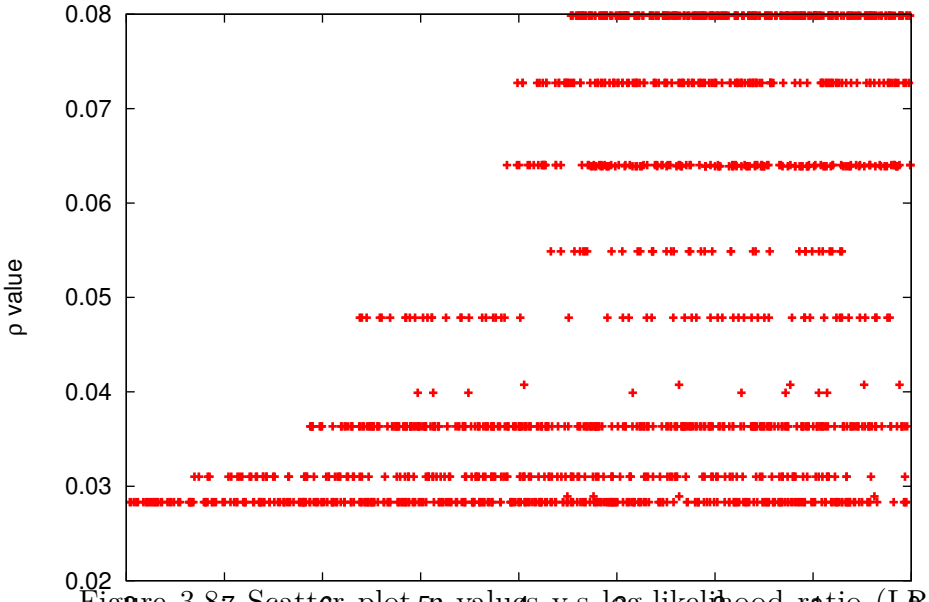

Figure 3.8: Scatter plot of p values v.s log-likelihood ratio (LR) for likelihood test for power-law vs log-normal. In this figure, we show the log-likelihood ratio (LR) values for $\rho \leq 0.1$. Since the LR values are negative, the alternative distribution (log-normal) is not better than power-law model.

Vuong's [73] method gives a ρ value that can tell us whether the observed sign of likelihood ratio is statistically significant. If this ρ value is small ($\rho < 0.1$) then the sign of log-likelihood ratio is a reliable indicator of which model is the better fit to the data. If the ρ value is large the sign of log-likelihood ratio is not reliable and the test does not favor either model over the other.

Figure 3.7 and Fig.3.9 show the CDF of the ρ value for the log-likelihood ratio between power-law and log-normal distributions and the log-likelihood ratio between power-law and exponential distributions. Both alternative distributions only show very small number of data points that have $\rho < 0.1$, each at around 13% and 5.5%. The log-likelihood ratio of these data points have negative signs as shown in Fig.3.8 and Fig.3.10, therefore the alternative distributions are not better than power-law. With the vast majority of the values for ρ being larger than 0.1, the results of the log-likelihood ratio test are ambigu-

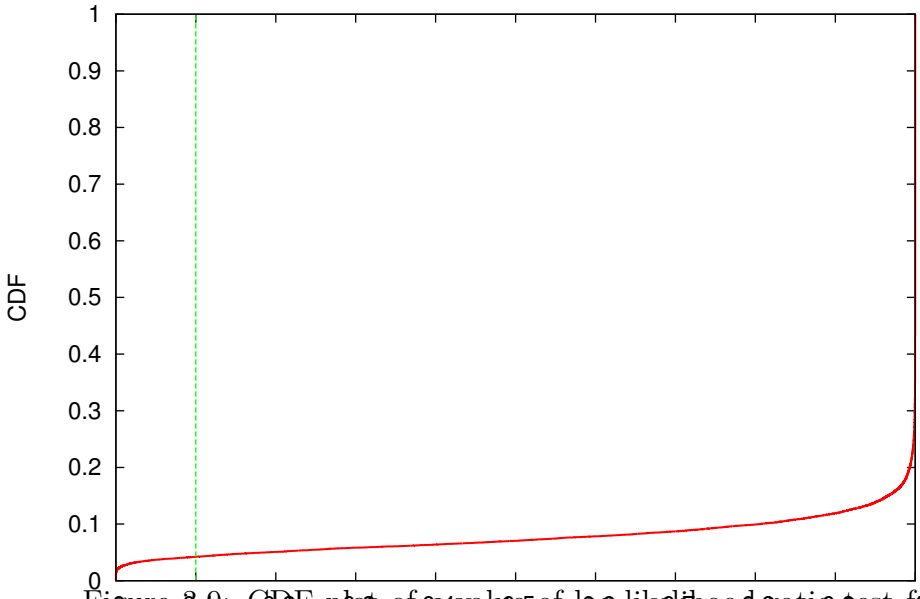

Figure 3.9: CDF plot of ρ value of log-likelihood ratio test for power-law v.s exponential. We divide the figure into two areas with vertical line $\rho = 0.1$ as border. Under nested hypothesis, ρ value is significance value of observed sign of likelihood ratio. if $\rho \leq 0.1$ then the sign of log-likelihood ratio is a good indicator of which model is the better fit to the data. In this figure, only 5.5% of ρ values are less than 0.1. Since 94.5% of ρ values are more than 0.1, the sign of log-likelihood ratio is not good indicator and the test does not favor either model over the other.

ous. Therefore, it is important to look at theoretical factors or design factors behind the systems to make a sensible judgment. For example: a leecher in a BitTorrent swarm is design to prefer the fastest seeders or leechers instead of high degree seeders or leechers. With this design factor information, we can make a sensible judgment which distributional form is more reasonable.

3.3.3 Clustering Coefficient

Networks show a tendency for link formation between neighboring vertices called *clustering* that reflects the topology robustness. The clustering around a vertex i is quantified by the clustering coefficient C_i, defined as the number of triangles in which vertex i participates normalized by the maximum possible

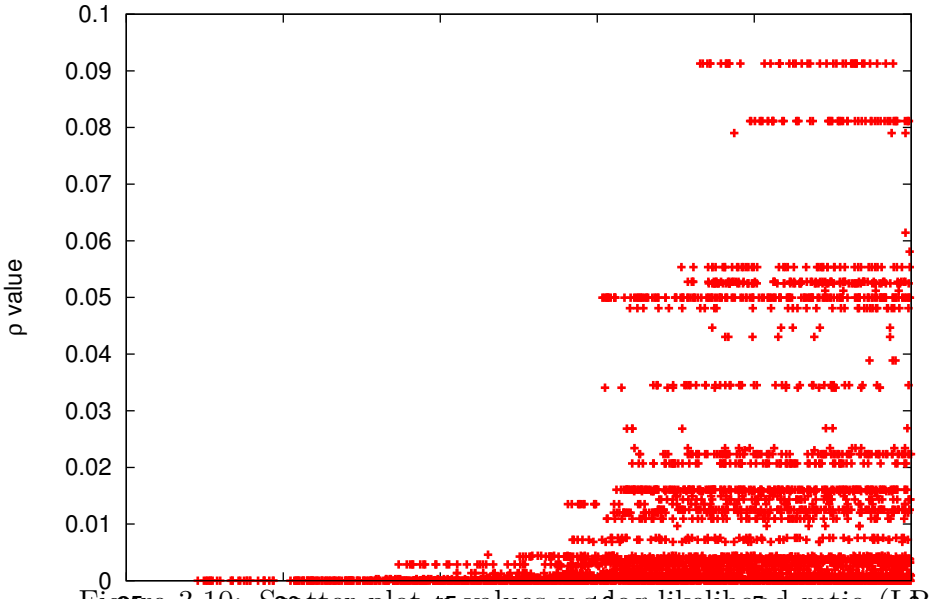

Figure 3.10: Scatter plot-ρ values v.s log-likelihood ratio (LR) for likelihood test for power-law vs exponential. In this figure, we show the log-likelihood ratio (LR) values for $\rho \leq 0.1$. Since the LR values are negative, the alternative distribution (exponential) is not better than power-law model.

number of such triangles,

$$c_i = \frac{2t_i}{k_i(k_i-1)} \tag{3.5}$$

where t_i denotes the number of triangles around i and k_i denotes vertex degree. For the whole graph, the clustering coefficient is

$$C = \frac{1}{n}\sum_{i \in G} c_i. \tag{3.6}$$

A larger clustering coefficient represents more clustering at nodes in the graph, therefore the clustering coefficient expresses the local robustness of the network. The distinction between a random and a non-random graph can be measured by clustering-coefficient metrics [75]. A network that has a high clustering coefficient and a small average path length is called a *small-world*

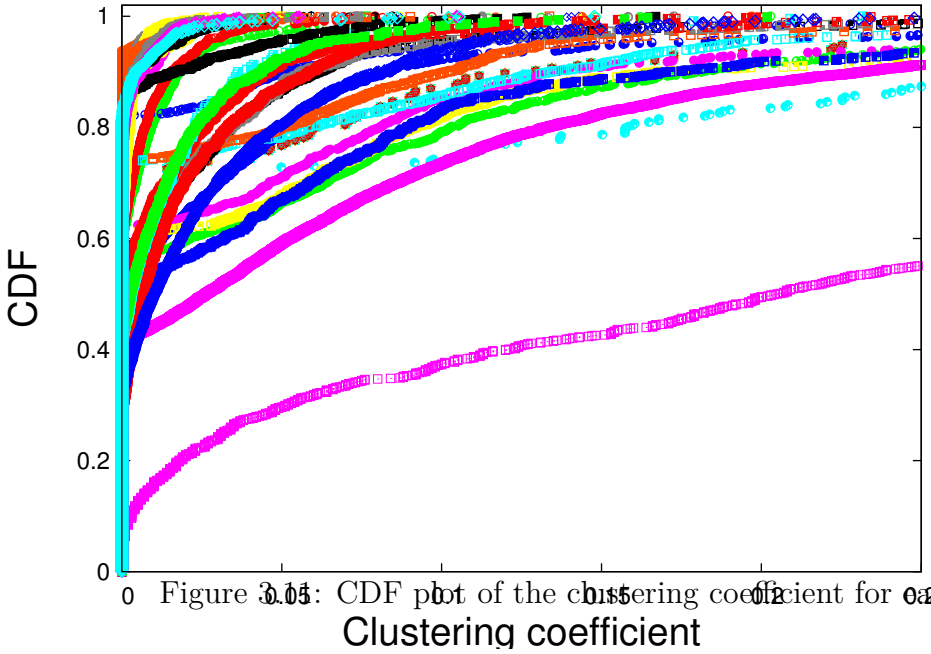

Figure 3.11: CDF plot of the clustering coefficient for each torrent.

model [75]. In BitTorrent systems, a previous study [40] mentioned the possibility that BitTorrent's efficiency partly comes from the clustering of peers. Figure 3.11 shows the CDF clustering coefficient value of our data sets. Only one torrent exhibits clustering coefficient less than 0.1 for about 40% of the snapshots, while for the other torrents, more than 70% are less than 0.1. This low clustering coefficient observation is the same as that observed by Dale *et al.* [14]. Considering only the low clustering coefficient, the BitTorrent topologies seem to be close to random graphs.

3.4 Confirmation via Simulation

We use simulations to compare the overlay topology properties based on our real-world experiments. We set the maximum peer set size to 80, the minimum number of neighbors to 20, and the maximum number of outgoing connections to 80. In our simulation, the results are quite easy to get since we

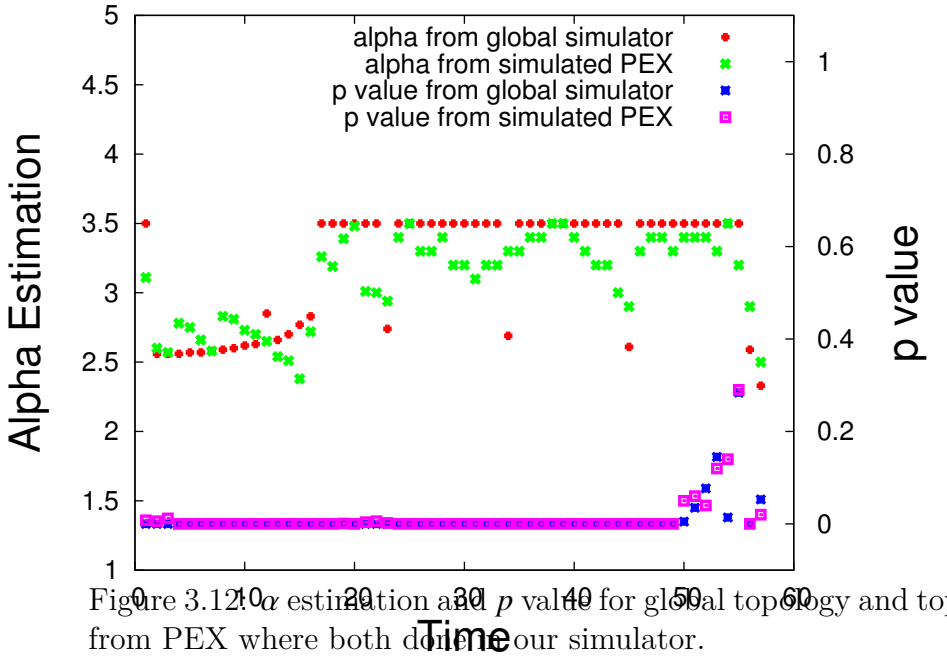

Figure 3.12: α estimation and p value for global topology and topology inferred from PEX where both done in our simulator.

are on a controlled system; we can directly read the global topology properties from our results. We also have the simulated PEX messages. We compare the global overlay topology properties as the final result from the simulator with the overlay topology that we get from PEX on the same simulator. Figure 3.12 shows the α estimate Eq.(3.1) and p Eq.(3.1) value both for the global result and the PEX result from our simulator. It clearly shows that both the global result and the PEX result from the simulator produce very low p values. We calculate the Spearman correlation for both α values from the global result and the PEX result. The Spearman rank correlation coefficient is a non-parametric correlation measure that assesses the relationship between two variables without making any assumptions of a monotonic function. The Spearman rank correlation test gives $0.38 \leq \rho \leq 0.5$, which we consider to be moderately well correlated. Therefore, the simulator confirms that the PEX method can be used to estimate α.

3.5 Summary

We have investigated the properties of BitTorrent overlay topologies from the point of view of the peer exchange protocol using real swarms from an operational BitTorrent tracker on the Internet.

We find that the node degree of the graph formed in a BitTorrent swarm can be described by a power law with exponential cut-off and the observation of a low clustering coefficient implies BitTorrent networks are close to random networks. From the BitTorrent protocol point of view, the reason that a BitTorrent swarm can be described by a power-law with exponential cut-off is: leechers in a BitTorrent swarm prefer a few good seeders or neighbors that can give high data rates to exchange the data and seeders have rich connections to leechers as seeders have complete chunks or pieces. That behavior explains why seeders have rich connections while leechers only have a few neighbors. We argue that there are two reasons for the cut-off phenomenon. First, most BitTorrent clients configure the maximum number of global connection between $200-300$, however the maximum connection per torrent (swarm) is set between $50-90$ by default [58] [70]. Some BitTorrent forums suggest decreasing the maximum connection for torrent (swarm) to between $30-40$ [69]. Second, most of the BitTorrent users are home users where their home gateway device cannot give high concurrent connections and BitTorrent is not the main online activity. We argue that the BitTorrent swarm closes to random that we infer from clustering coefficient is caused by BitTorrent mechanism itself that always choose random peers from its neighbors in the choking-unchoking algorithm, optimistic choking algorithm, and optimistic connect algorithm as we explained previously. Our approach can infer BitTorrent swarms topology and the result confirmed by simulation [17]

Chapter 4

Peer-Assisted Content Delivery

4.1 Introduction

Streaming content, especially video, represents a significant fraction of the traffic volume on the Internet, and it has become a standard practice to deliver this type of content using Content Delivery Networks (CDNs) such as Akamai and Limelight for better scaling and quality of experience for the end users. For example, YouTube uses Google cache and MTV uses Akamai in their operations.

With the spread of broadband Internet access at a reasonable flat monthly rate, users are connected to the Internet 24 hours a day and they can download and share multimedia content. P2P (peer to peer) applications are also widely deployed. In China, P2P is very popular; we see many P2P applications from China such as PPLive, PPStream, UUSe, Xunlei, etc. [72]. Some news broadcasters also rely on P2P technology to deliver popular live events. For example, CNN uses the Octoshape [51] solution that enables their broadcast to scale and offer good video quality as the number of users increases.

From the Internet provider point of view, the presence of so many always-on users suggests that it is possible to delegate a portion of computing, storage

and networking tasks to the users, thus creating P2P networks where users can share files and multimedia content. Starting from file sharing protocols, P2P architectures have evolved toward video on demand and support for live events.

Alternatively, video contents can be efficiently distributed on services offered by managed network architectures and CDN companies. The major issues of CDN are high deployment cost and good but not unlimited scalability in the long term. Given the complementary features of P2P and CDN, in recent years some hybrid solutions have been proposed and applied to the operational of CDN [32, 37, 78] to take the best of both approaches. In Peer assisted CDN, users can download content from CDN nodes from or other users or peers. A user may cache the content after download to serve requests from other users. Due to the complexity of the behavior of peers, the process should be done in the home gateway user where the ISP can control it.

In this work, we will revisit Guo et al.'s PROP, [26] as a basis to evaluate peer-assisted CDN and propose an improvement to the model for the PROP called CPPro (this system is called CPPro abbreviated from our technical term "CDN-P2P Project"). In PROP's utility function, the difference between very popular videos and unpopular video is very difficult to differentiate. For an unpopular video, $f(p)$ will be very close to $f(p_{min})$ thus $f(p) - f(p_{min})$ will be very close to 0 then the utility function becomes very small. For a very popular video, $f(p)$ will be very close to $f(p_{max})$, thus $f(p_{max}) - f(p)$ will be very close to 0 and the utility function becomes very small. We will take Youtube as an example of an Internet VoD service model. In the Youtube service model, we can get data such as (1) the time when a video is uploaded and (2) the number of accesses or number of views. We can get such data using Youtube's API. Borghol et al., [8] used the above information to estimate when a video will become very popular. They divided a video's popularity into three

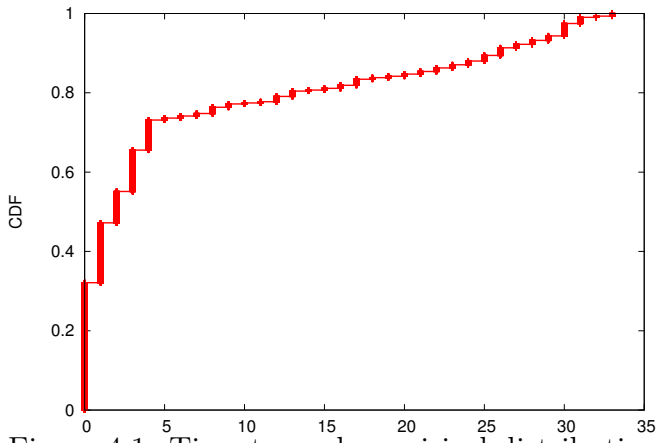

Figure 4.1: Time to peak empirical distribution data from [8].

phases: before-peak phase, at-peak phase, and after-peak phase. We will use an estimate of a video's popularity phases for helping PROP. We will explain video popularity in Sec.4.2. Our contribution is as follows: (1) We use the idea of VoD view popularity model to aid the PROP. To the best of our knowledge, the combination of the PROP and the VoD view popularity model is new. (2) From simulation-based experiments, we find that peer contributions in CPPro are almost as good as PROP while the numbers of replicas are lower, achieving the same performance with fewer resources.

4.2 Determining Internet VoD Popularity Phase

The objective of determining the Internet VoD popularity phase is to determine whether a video is at before-peak, at-peak, or after-peak phase, to be used by peers in their caching strategy. For this purpose we use the Youtube content popularity dataset from Borghol et. al., [8] which contains data about 29791 videos, including the view count and upload time, during 36 weeks of measurements. Figure 4.1 is the cumulative distribution function (CDF) of the time-to-peak distribution from Borghol et. al., [8] which shows that around three-quarters of the videos peak within the first six weeks after upload. The

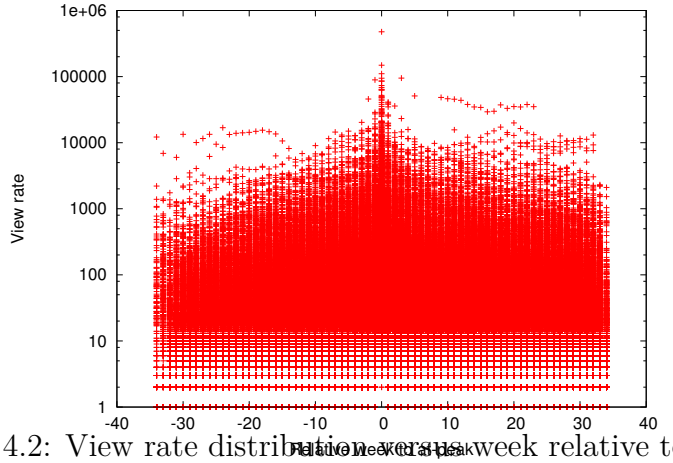

Figure 4.2: View rate distribution versus week relative to at-peak phase week for every video in Youtube datasets, where y-axis in log scale. We shift the week relative to to at-peak phase week. Every point lies in negative x-axis mean view rate of every video in before-peak phase. Every point lies in x-axis= 0 mean view rate of every video at-peak phase. Every point lies in positive x-axis mean view rate of every video in after-peak phase. Data from [8].

time-to-peak is exponentially distributed up to the sixth week, and it is uniform beyond the sixth week. Borghol et al., [8] define time-to-peak for a video as its age (time since upload) at which its weekly viewing rate is the highest during measurement (from the first week until end of measurement). Because we know the peak time (at-peak phase) of every video, we can also find the before-peak phase and after-phase of every video. Borghol et al., [8] mentioned that view count evolution follows beta distribution. For detail we refer the readers to [8].

Our algorithm for determining a requested video is quite simple. As we know the age of a requested video in week number and its view rate, the algorithms looks for Borghol dataset with the same week, one week before, and one week after for the same view rate. We are averaging the relative week to at-peak values in the same week, one week before, and one week after.

Suppose a video v in the Borghol dataset has a viewing rate $r_v(t)$, $0 <= t < t_f$, and $r_v(t)$ peaks at t_{vp}. The data is transformed by including the relative time-

to-peak, such that each data point is a 2-tuple: video rate and relative time to peak:

$$r_p^v(t) = \langle r_v(t), t_p(t) \rangle \tag{4.1}$$

$$t_p(t) = t - t_{vp}. \tag{4.2}$$

Figure 4.2 shows the Borghol's dataset with time axis for each video is translated by t_{vp} to the left, such that each video peaks at time 0.

In determining the phase of a requested video e with known age t_e and view rate r_e at t_e, we find the three r_v data points whose rates are closest to r_e at t_e, $(t_e - 1)$, and $(t_e + 1)$ for all videos from Borghol dataset, and then average the t_p of the three data points. The phase of the requested video e is estimated to be before-peak, at-peak, or after-peak based on whether the average is negative, 0, or positive. The view rate r_e of a video is calculated by subtracting the view counts at the time of the current and the previous video requests. The pseudo code for averaging the t_p is shown in algorithm 1.

But when a video is being requested for the first time, the phase can only be estimated using the age of the video. In this case, we draw 36 weighted random integer numbers s_i, $0 <= i <= 35$ where the weighted factor is from time-to-peak phase CDF as shown in fig. 4.1 then calculate the count of each integers between 0 and t_e from the drawn numbers then divide the result by 36, i.e., $estphase = \sum_0^{t_e} \frac{count(i,s)}{36}$. The number 36 come from the duration of measurement and each week has its own probability as we shown in fig. 4.1. This result represents the estimated phase. From time to peak distribution, 50% of video reach peak within four weeks. At that level, we expect that half of videos may reach at-peak and half of videos are not yet reach at-peak. Therefore we put 0.5 as low threshold. Still from the same time to peak distribution 75% of video reach peak within six weeks, and beyond six week

Algorithm 1 Averaging relative weeks from the nearest neighbor points

Require: dataset that consist of week number, view rate, and relative week to at peak.

$t, r_v, t_p \leftarrow$ read Borghol's dataset {where t is week number (1st column), r_v is view rate (2nd column), and t_p is relative week to at-peak values (3rd column).}
$len \leftarrow 29791 * 36$ {we have 29791 videos * 36 weeks thus we have 1072476 rows and 3 columns of Borghol's dataset.}
$t_e \leftarrow time_{cur} - time_{upload}$ {calculate week number of a requested video which is current time a requested video subtract by upload time of a requested video.}
$r_e \leftarrow viewcount_{cur} - viewcount_{lastweek}$ {calculate view rate of a requested video which is current view count of a requested video subtract by last week view count of a requested video.}
$t_e^{before} \leftarrow (t_e - 1)$ {at one week before.}
for $i = 0$ to len **do**
  if $t == t_e$ and $r_v == r_e$ **then**
    $t_p^{before} \leftarrow t_p$ {put t_p value into t_p^{before} list.}
  end if
end for
$t_e^{at} \leftarrow (t_e)$ {at same week.}
for $i = 0$ to len **do**
  if $t == t_e$ and $r_v == r_e$ **then**
    $t_p^{at} \leftarrow t_p$ {put t_p value into t_p^{at} list.}
  end if
end for
$t_e^{after} \leftarrow (t_e + 1)$ {at one week after.}
for $i = 0$ to len **do**
  if $t == t_e$ and $r_v == r_e$ **then**
    $t_p^{after} \leftarrow t_p$ {put t_p value into t_p^{after} list.}
  end if
end for
$t_p^{final} \leftarrow average(t_p^{before}, t_p^{at}, t_p^{after})$
if $t_p^{final} < 0$ **then** {if relative week to at-peak value is less than 0, estimate a requested video as before-peak.}
  $phase \leftarrow before$
else if $t_p^{final} == 0$ **then** {if relative week to at-peak value is equal to 0, estimate a requested video as at-peak.}
  $phase \leftarrow at$ {if relative week to at-peak value is more than 0, estimate a requested video as after-peak.}
else
  $phase \leftarrow after$
end if

Algorithm 2 Determine phase for the first time access of a requested video

Require: t_e and time-to-peak distribution
1: $len \leftarrow 35$
2: **for** $i = 0$ to len **do**
3: draw integer random number between 0 and 35 respect to time-to-peak distribution: $d \leftarrow draw\_integer\_random\_number()$
4: **end for**
5: $total \leftarrow 0$
6: **for** $i = 0$ to t_e **do**
7: $total \leftarrow total + count(d,i)$ {counting how many integer random number and sum those number}
8: **end for**
9: $estphase \leftarrow total/36$
10: **if** $estphase > 0.75$ **then**
11: phase ← after-peak
12: **else if** $estphase \leq 0.75$ and $estphase > 0.5$ **then**
13: phase ← at-peak
14: **else**
15: phase ← before-peak
16: **end if**

the distribution is considered, it means there are not much additional view count. In other words, beyond six weeks we consider videos reach after-peak phase. Therefore we put 0.75 as high threshold. The pseudo code for this purpose is shown in algorithm 2.

4.3 System Description

4.3.1 System Overview

The main components of the system are: (1) a CDN and (2) a set of peers which are self organized into a P2P overlay network. Each peer in the system has two functions. First, a peer is a client that requests a video. Second, a peer is a contributor that shares the cached video with other peers in the system. Peers control the number and utilization of their connection based on current resources availability. In fig.4.3, we describe the process of a peer that requests a video which derived from PROP. When a video is requested for the

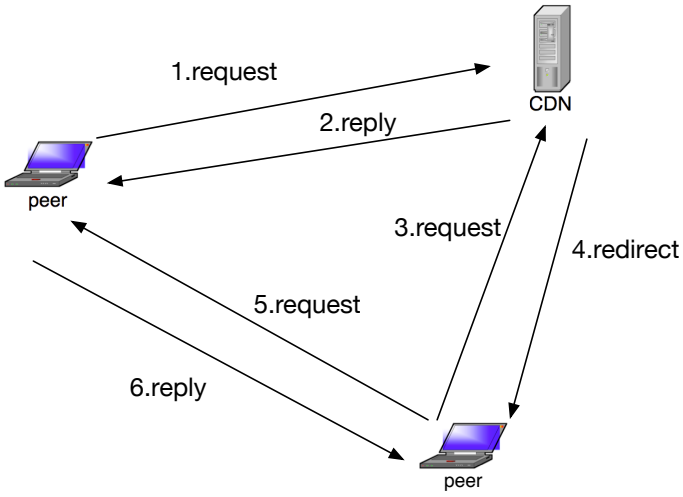

Figure 4.3: Peer assisted CDN works as follows: when a peer requests a video, it always goes to a CDN server (step 1). The CDN provides the videos to the peer (step 2). If there is another peer request same video, that request will go to CDN (step 3). A CDN will check its record to see if there is some peers cache that requested video. If there is some peers cache that requested video, a CDN will reply with redirect message that asking a peer to download requested video from other peer (step 4). If there s no peers have requested video, a CDN will serve the video. A peer then can request the video to other peer and get the video (step 5 and step 6).

first time, the CDN is responsible for delivering the requested video. When a CDN receives a query for the same video, a CDN will find suitable peers that currently have a copy of the requested video. The CDN then returns information about these peers to the querying peer.

4.3.2 Peer caching strategy

A peer stores the requested videos into its cache based on the value of a utility function, i.e., in a peer with a full cache a newly requested video will replace one or more videos in the cache whose utility values are smaller than that of the new video. The utility function reflects the popularity of a video in the system that considering number of copy of its video or replica. PROP's

u value itself lies in interval $[0,2]$ Guo et al., [26]. Our utility function u is:

$$u = \frac{(f(p) - f(p_{min}))(f(p_{max}) - f(p))}{r^{\alpha+\beta}} + z(t) \tag{4.3}$$

where the first term is the utility function of PROP, and the second term, $z(t)$, is the phase factor that we introduced in CPPro, where $z(t)$ is large if the video phase is at-peak. p, p_{min}, and p_{max} are the popularity of the video, the minimum popularity, and the maximum popularity in the system. We follow [26] for $f(p) = log(p)$. The popularity of a video is the determined by the view rate of the video. r is the number of replicas. α and β are adjustment factor [26].

Following [26], we can calculate p as follows:

$$p = min\left(\frac{n_i^r}{t_i^r - t_a^i}, \frac{1}{t_{cur} - t_i^r}\right) \tag{4.4}$$

Where n_i^r is number of requested video, t_i^r is last time the video is requested, t_a^i is the time when the video is accessed for the first time, and t_{cur} is the current time. To able to track the simulation, we use default value from PROP thus we refer the readers to [26] for the details.

The $z(t)$ values are set as follows:

$$z(t) = \begin{cases} 0.15 & \text{if phase estimation is before-peak} \\ 0.47 & \text{if phase estimation is at-peak} \\ 0.38 & \text{if phase estimation is after-peak} \end{cases} \tag{4.5}$$

where the numbers are based on the proportion of video counts in the Youtube dataset before the peaks, around the peaks, and after the peaks. $z(t)$ for at-peak is determined by the total proportion of video counts from 1-week-

before until 1-week-after the peak, which accounts to almost half of the total. The value of $z(t)$ is assigned after we finish to determine a video popularity phase. For example: if we determine a video popularity phase is at-peak, then we assign $z(t) = 0.47$.

4.4 Evaluation

We performed simulations to compare the performance of CPPro against PROP in two main metrics: (1) peer contributions in video delivery, and (2) number of video replicas in peers. The peer contribution is the number of videos served by a peer instead of by the CDN server. The number of video replicas shows how much storage is used in peers to contribute in video delivery. The peer contribution metric is related to the byte-hit-ratio. The byte-hit-ratio is defined as the total bytes of content served by peers normalized by the total bytes of video all peers and the CDN consume. With more peer contributions, we will have higher byte-hit-ratio because peer can get content from other peers. However, because we only interested in peer performance, we compare peer contribution between PROP and CPPro. Contribution ratio of peer to total contribution (comparing to CDN contribution) becomes irrelevant in this case.

4.4.1 Simulation Design

This peer-assisted CDN is simulated using an event driven simulation implemented in Python. Peers request videos from a video catalog where the peer request as well as the videos in the catalog are generated using certain distributions.

Video Catalog

Each video in the catalog has the following properties: video-id,size,upload time, final view count, view count function parameters. View count parameters are the distribution parameters. The final view count is the total number of views of a video at the end of simulation and it is generated using uniform distribution sampling from Borghol dataset ranging from 27 until 1069385. The upload time interval is a Poisson process with $\lambda = 1$. The video size is generated using a uniform distribution. Because of the very weak relationship between video size and popularity [1] and because our work focuses on the impact of the popularity aspect on the utility function rather than storage optimization we believe that the choice to assign a random uniform video size from the Youtube dataset does not effect our results. The view rate progression from the upload time until the end of simulation time is modeled using a Beta distribution [8]. As Borghol et al., [8] showed that view rate of a video can be modeled using beta distribution we can calculate α and β parameters. We can use beta distribution mode formula $m = \frac{\alpha-1}{\alpha+\beta-2}$ to calculate α or β. In this case, we choose α random uniform between 1 and 3 and m is view rate at-peak, thus we can get $\beta = \frac{\alpha-1}{m} - \alpha - 2$.

Peer Request Generator

Peers request videos from the catalog using a Poisson process with $\lambda = 1$ [81] for the inter-arrival time. For the requested videos there are three scenarios A,B,and C. In Scenario A, the video popularity in the peer-assisted CDN system follows the global popularity of the video. In Scenario B, the video popularity in the peer-assisted CDN system lags four weeks behind the global popularity of the video. We choose four weeks lags based on the probability from time-to-peak distribution that half of videos are already reach peak within four weeks. In Scenario C, the video popularity in the peer-assisted CDN

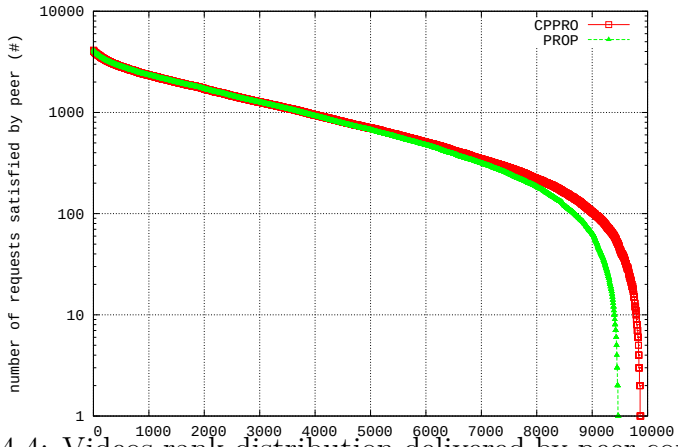

Figure 4.4: Videos rank distribution delivered by peer compared between CP-Pro and PROP for scenario A (y-axis in log-scale).

system does not follow the global popularity of the video. We use a Zipf distribution with rate= 0.9 for this purpose [24].

Simulation Parameters

The simulation parameters are follows:

- Length: 360 days.
- Video size: uniform random between 1MB and 200MB.
- Peer storage capacity: 500MB.
- CDN storage capacity: 10000MB.
- Number of peers: 10000.
- Number of videos: 10000.
- Peer's caching strategy: CPPro, PROP.

Finally, we compare our results to PROP [26].

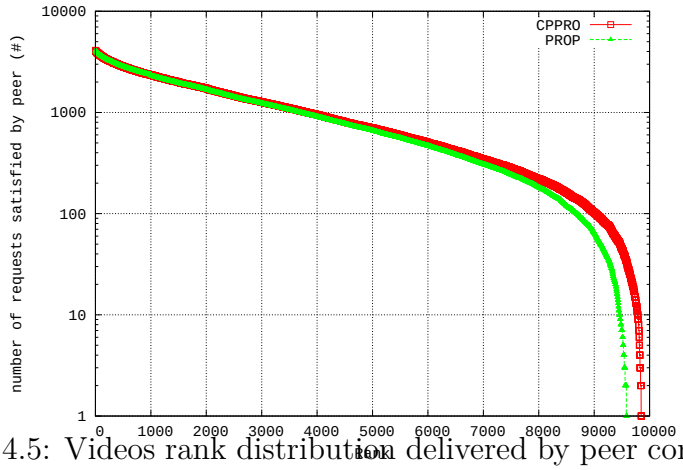

Figure 4.5: Videos rank distribution delivered by peer compared between CP-Pro and PROP scenario B (y-axis in log-scale).

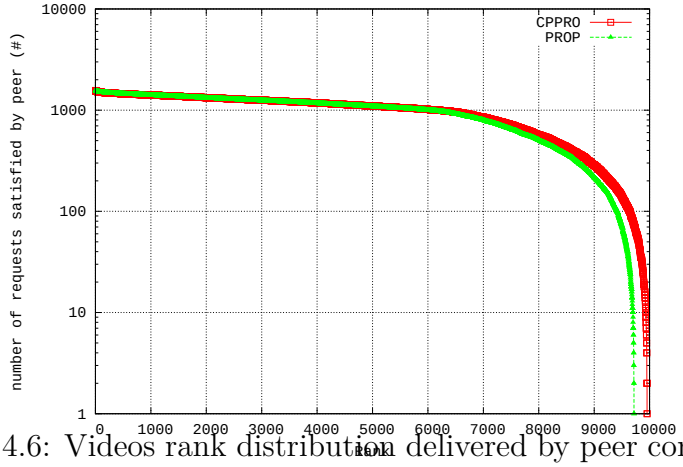

Figure 4.6: Videos rank distribution delivered by peer compared between CP-Pro and PROP scenario C (y-axis in log-scale).

4.4.2 Result and Discussion

Figure 4.4, 4.5, and 4.6 shows number of requested satisfied by peer. They exhibit a similar pattern and only differ in the tails for all scenarios, where CPPro gives higher, albeit insignificant, results. The difference between CPPro and PROP in scenario A is 1.1%, 1.7% in scenario B, and 2% for scenario C. These results show that the introduction of $z(t)$ component to the utility function does not affect the performance in terms of peer contributions. The

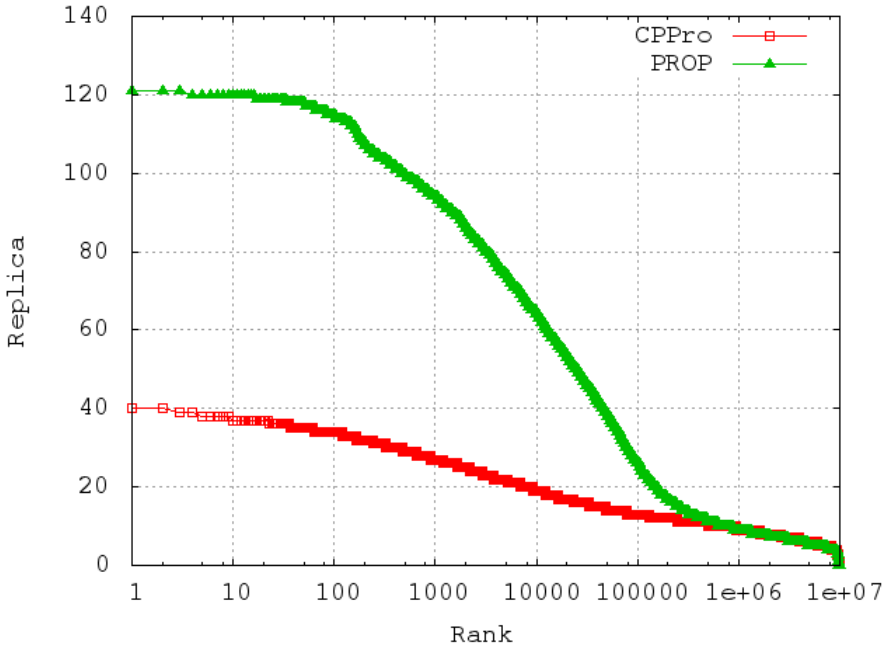

Figure 4.7: Comparison of available replicas between CPPro and PROP when a peer requests a video for scenario A (x axis in log-scale).

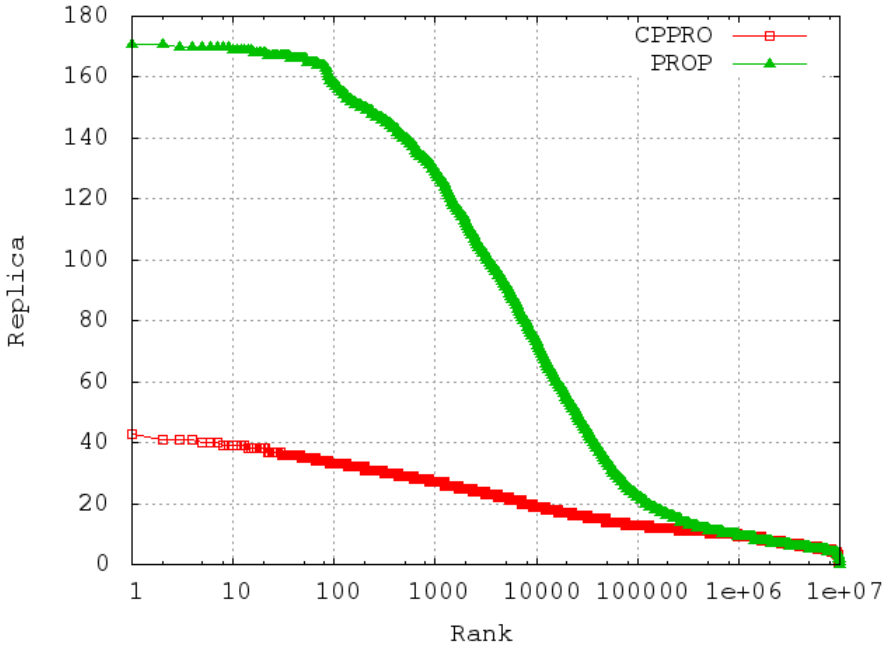

Figure 4.8: Comparison of available replicas between CPPro and PROP when a peer requests a video for scenario B (x axis in log-scale).

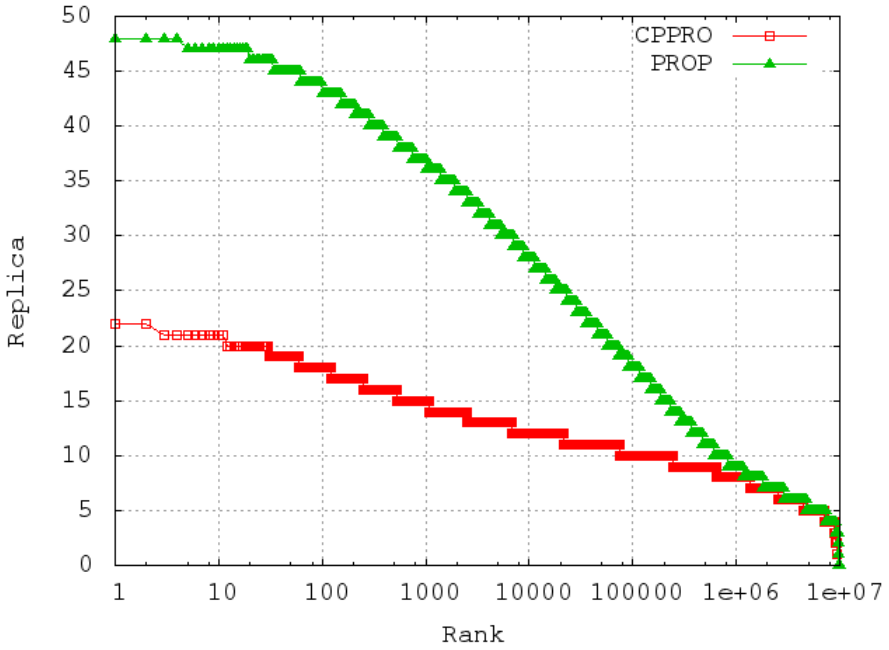

Figure 4.9: Comparison of available replicas between CPPro and PROP when a peer requests a video for scenario C (y axis in log-scale).

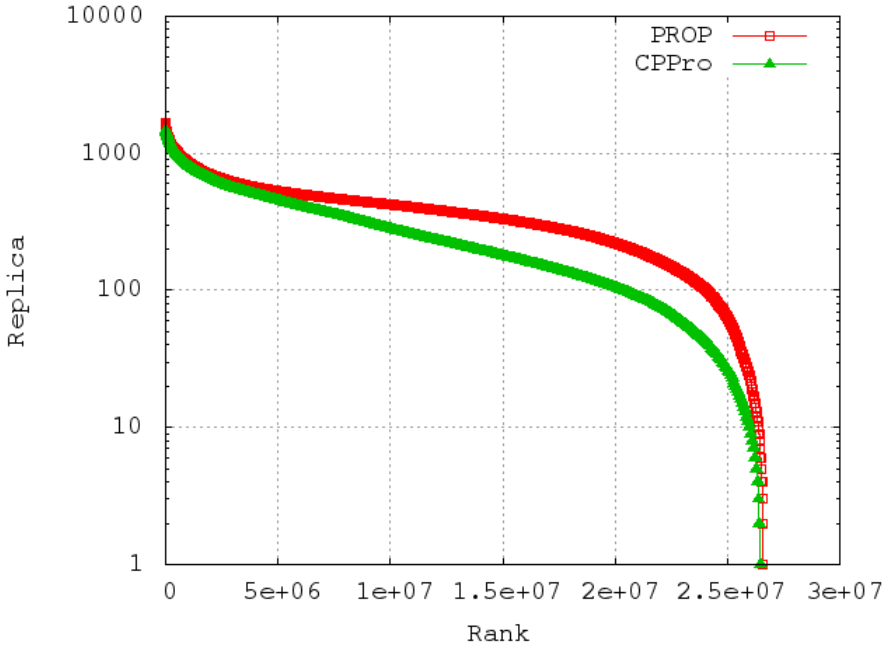

Figure 4.10: Comparison of available replicas between CPPro and PROP when a peer requests a video for scenario A with 100000 peers (x axis in log-scale).

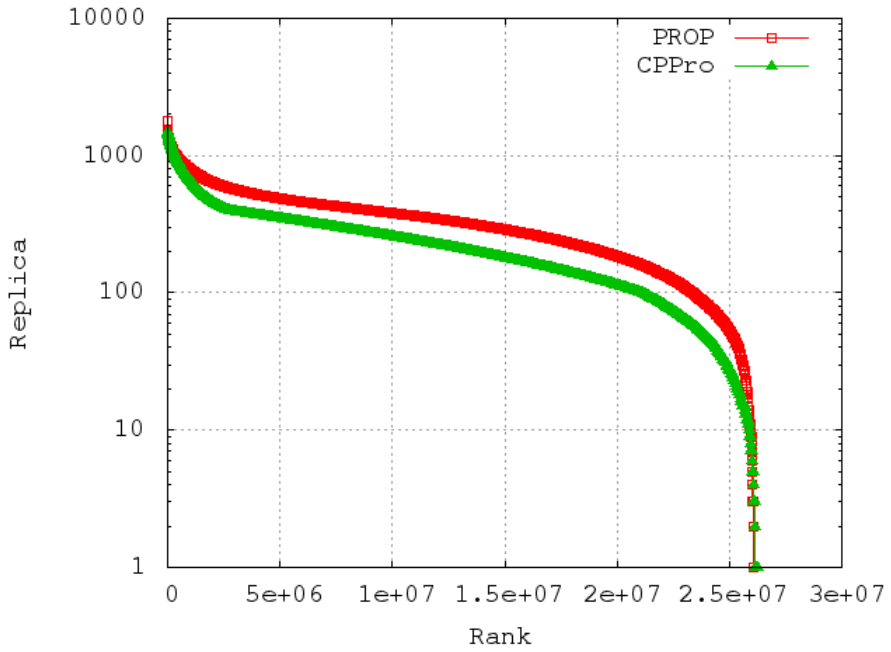

Figure 4.11: Comparison of available replicas between CPPro and PROP when a peer requests a video for scenario B with 100000 peers (x axis in log-scale).

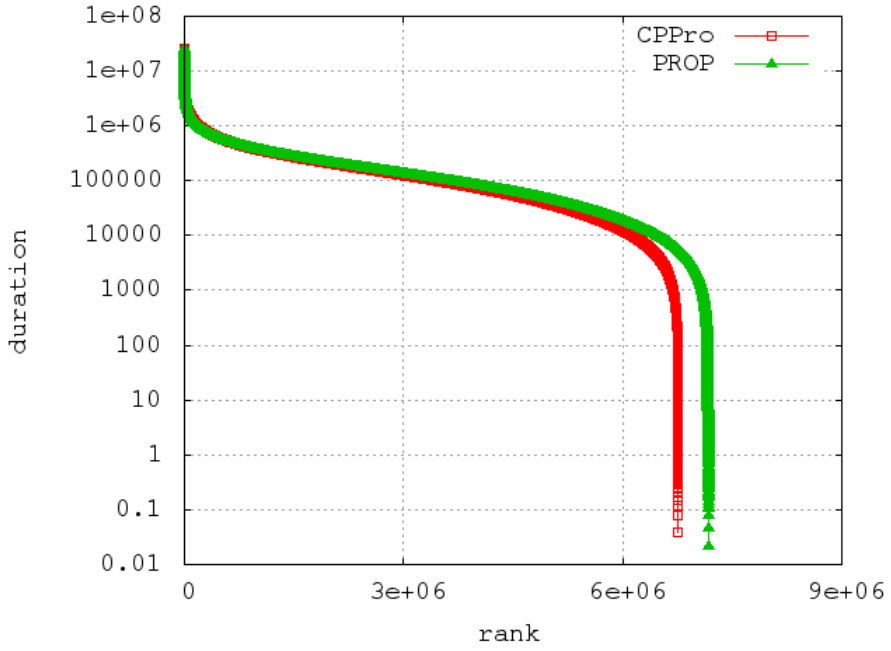

Figure 4.12: Cache duration of a video in peers for scenario A.

advantage of CPPro to PROP is evident in fig. 4.7, 4.8, and 4.9, which show the number of replicas of the requested videos at each request event.

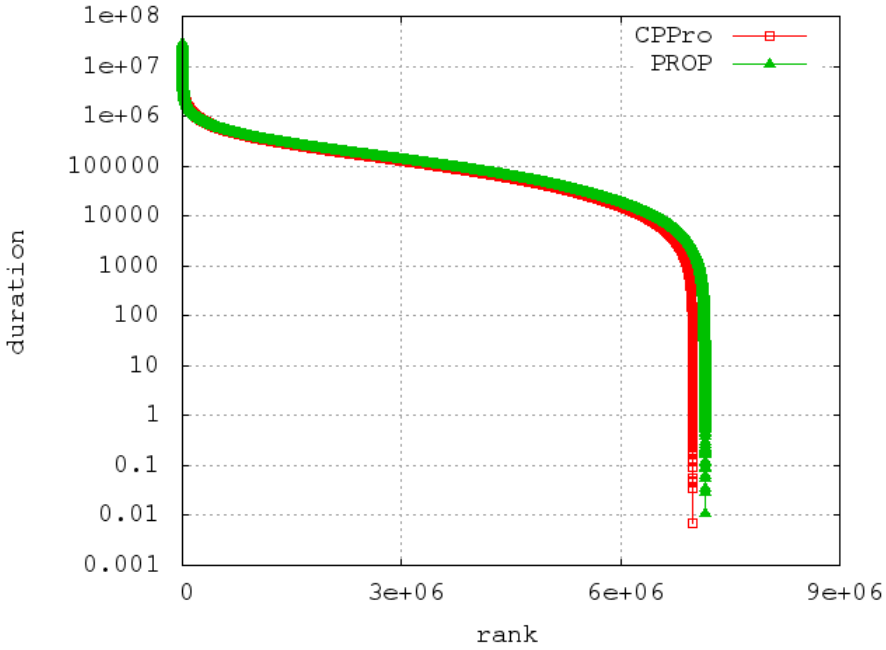

Figure 4.13: Cache duration of a video in peers for scenario B.

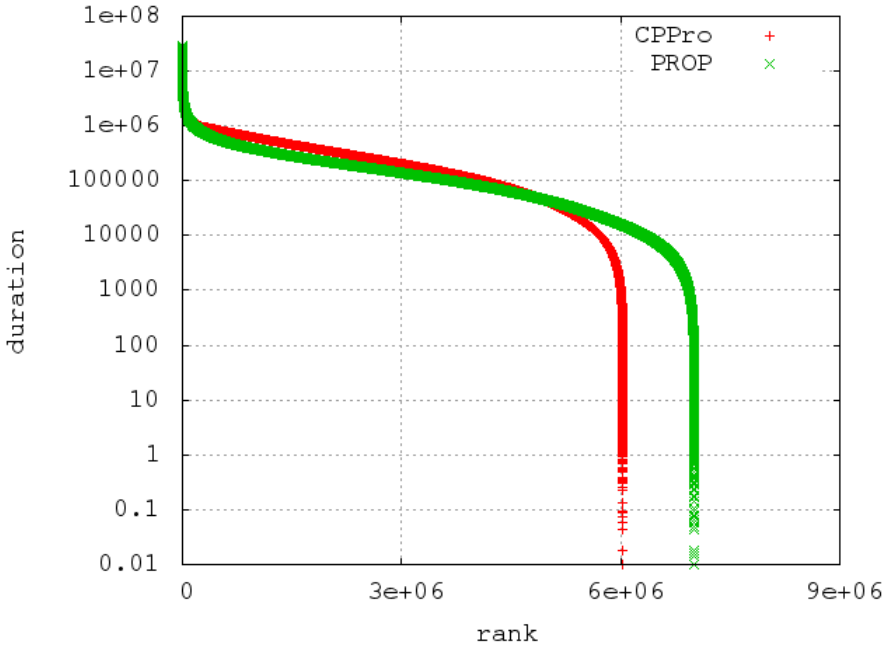

Figure 4.14: Cache duration of a video in peers for scenario C.

When a peer requests a video from another peer, a peer has to decide whether a peer wants to cache the video or not to cache the video based on the calculated utility function value. If a peer decided to cache a video, we called it

cached event. If a peer did not decide to cache a video, we called it not-cached event. We breakdown how many cache events occur in our simulation as shown in table.4.1. Furthermore, we breakdown again by video popularity phase for CPROP's cached and not-cached events as shown in table. 4.2. From table.4.1, we can see that CPPro cached less videos compared to PROP for all scenarios. CPPro cached events are fewer than PROP because in CPPro the probability of cached events are fewer than not-cached events. We want to know what makes cached events in CPPro are fewer than PROP. Assume a peer requests a video with utility function u_r and the minimum utility function inside the peer's cache is u_c-min.Intuitively, when a requested video is in the after-peak phase and other videos in peer's storage are in the at-peak phase, then in the CPPro case there is a high probability that a requested video will not be cached. In the PROP case, since PROP does not use the video popularity phase, the probability of a requested video will not be cached is lower than CPPro. We can take the same intuition for before-peak and at-peak phase.

The cached events happen more frequently in scenario A comparing to scenario B, because in scenario B when we shift requests four weeks later, estimated of requested videos will probably fall at-peak phase or after-peak phase and the videos in peer's cache are also probably estimated at-peak phase or after-peak thus probability of not-cached events are higher in scenario B than scenario A. In table.4.2, the at-peak phase has low proportion comparing to before-peak and after-peak phase because when we estimate popularity phase of a requested video, the estimate of video to be at-peak phase occurs when average of relative week is exactly 0. In table.4.2, the after-peak phase has high proportion comparing to before-peak and after peak phase in scenario A and B because video popularity evolution model that we use to estimate a new requested video assume that after four weeks most of video will enter after peak phase. In scenario B, its higher than scenario A since we shifted

four weeks. In scenario C, while after-peak phase has higher proportion than before-peak and at-peak phase, its proportion is lower than scenario A an B because different popularity video choice. In table.4.2, the before-peak phase in scenario A has higher proportion comparing to scenario B because in scenario B we shift the video request time four weeks thus the probability of a requested video to be before-peak is lower than scenario A. In scenario C, the proportion before-peak phase is higher than scenario A and B. This probably due to different popularity of video choice that a requested video is estimated to be before-peak phase but the video inside peer's cache are having lower utility function.

In fig. 4.7, 4.8 , and 4.9 show number of replicas available for each scenario when a peer request a video. Maximum replica for PROP is around three times from CPPro in scenario A and B, while in scenario C the maximum value for PROP is around two times from CPPro. Numbers of replicas for each scenario are different between CPPro and PROP in the head of distribution and similar in the body and the tail of distribution. Although, we can see two and three times fold difference in the head of distribution, in fact total difference is only 1% for scenario A and B and 5% for scenario C. This is small difference because of small number of peers in the simulation, which is same with number of available video. The probability of different peers chooses a same video is high here. We also performed simulation with 100000 peers and number of available replicas for scenario A and B are shown in fig. 4.10 and 4.10. In 1000000 peers case all videos are uploaded (in Poisson process) in the beginning of simulation. In case of 100000 peers, we can see the difference between CPPro and PROP in number of available replicas. In this case, the probability of different peers chooses a same video is low here. We calculate the Kolmogorov-Smirnov statistic on two samples and we find that for scenario A and scenario B the p-values are less than 0.1 thus we consider the result are

Table 4.1: Percentage of Cached events and Not-Cached events in CPPro and PROP.

Scenario	Type	Cached (times)	Not-Cached (times)
A	CPPro	32%	68%
	PROP	71%	29%
B	CPPro	30%	70%
	PROP	72%	28%
C	CPPro	48%	52%
	PROP	70%	30%

Table 4.2: Percentage cached and not-cached events for each video popularity phase in CPPro.

Scenario	Type/Events	Before-Peak	At-Peak	After-Peak
A	CPPro Cached	2.1%	0.5%	29.5%
	CPPro Not-Cached	2.8%	0.7%	64.4%
B	CPPro Cached	1.6%	0.5%	27.8%
	CPPro Not-Cached	2.2%	0.8%	67.1%
C	CPPro Cached	13.7%	1.7%	33.8%
	CPPro Not-Cached	8.2%	1.2%	41.4%

significant.

Figure 4.12, 4.13, 4.14 show cache duration from all scenarios. In scenario A and B, cache duration between PROP and CPPro are similar except in tail of distribution. In scenario C, CPPRO has longer cache duration compared to PROP in the body of distribution and lower cache duration in the tail of distribution. Long cache duration means video stay longer than another video inside peer's cache. If some videos has long cache duration in peer's cache, then big probability that peer's do not cache a requested video because the videos inside the peer's cache have higher utility function. The implication is the peer has low cached events. That's what wee see in table 4.1.

4.5 Application

Conceptual application of peer-assisted CDN in AI3 network is shown in fig 4.15. AI3 network [55] is a satellite based testbed network that operating mainly in south east Asia. As shown in fig 4.15, the satellite feeder is located

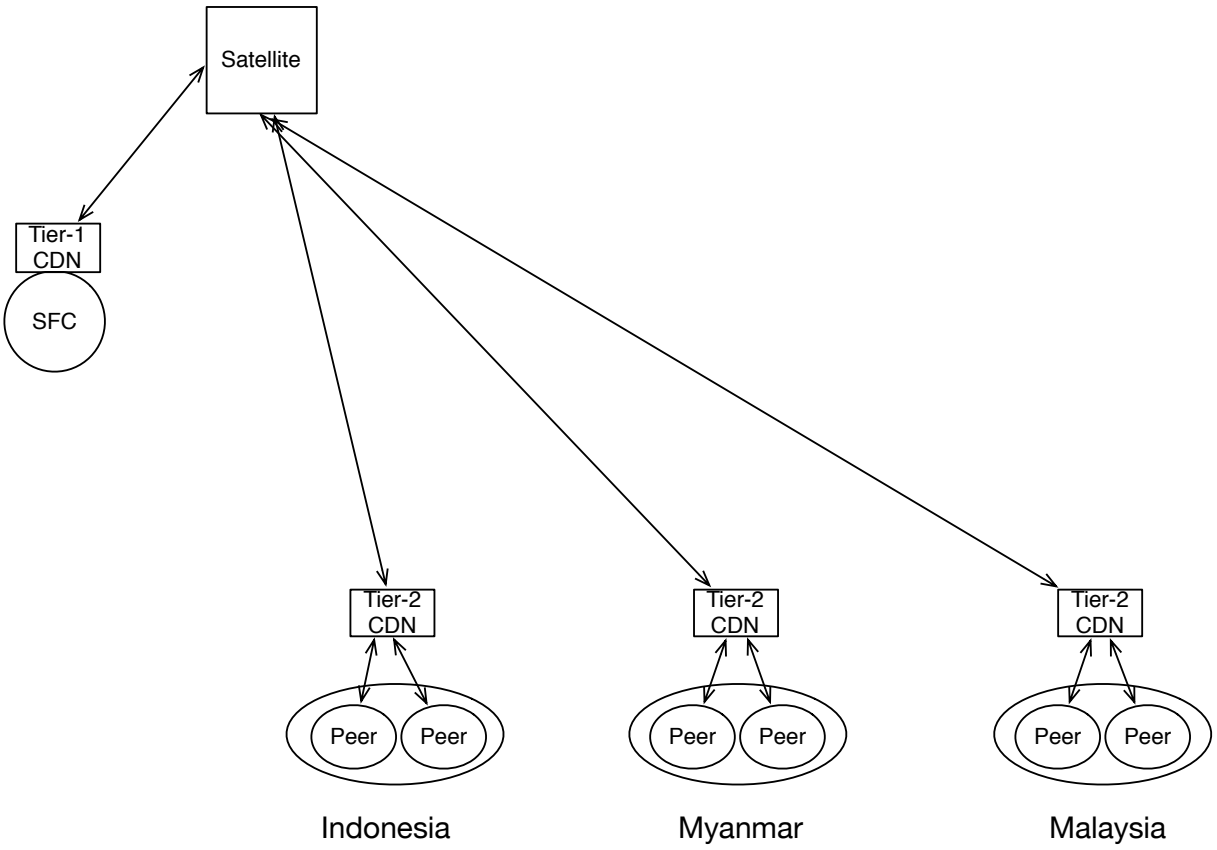

Figure 4.15: Conceptual Application of Peer-Assisted CDN in AI3 network.

in Keio university Shonan Fujisawa Campus (SFC), Japan. The receivers are mainly located in several South-East Asia countries such as Indonesia, Malaysia, and Myanmar.

The main CDN (tier-1 CDN) is located in SFC and tier-2 CDNs are located in receiver sites. Every receiver sites have several peers. Every time a peer ask a content, it will goes to the tier-2 CDN. If the content is not available, the tier-2 CDN will forward the request to the main CDN, the main CDN will delivery the content to the tier-2 CDN, and the tier-2 CDN will delivery the content to a peer. When another peer ask the same content it will goes to the tier-2 CDN. If other peer inside the same site has the content, the tier-2 CDN will redirect the request to other peer. If there are no peers have the content, the tier-2 CDN will deliver the content.

4.6 Summary

This chapter presented a scheme for a peer-to-peer network that can help a CDN to deliver the content over the Internet. We show that by introducing the z factor to the cache utility function CPPro can maintain the same peer contribution while reducing number of replicas. We found that there is no much difference for all scenarios in terms of peer contribution to deliver a video. However, we found that in all the scenarios, CPPro gives lower replicas than PROP. This is because in CPPro, we found intuitively that probability that not-cached events occur is higher than cached events. Furthermore compared to PROP, the probability that a peer does not cache a requested video is higher in CPPro compared to PROP. Therefore, in CPPro the numbers of available replicas are lower than PROP. We also did the significance test to the number of replicas using the Kolmogorov-Smirnov statistic on two samples and we find that for all scenarios the p-values are less than 1% thus the results are significant.

Some areas of improvement that we have identified for future work are: (1) since we know CPPro offers a little bit of peer contribution with fewer replicas, we are interested in knowing the energy trade-off of this peer-assisted CDN architecture in order to know how much energy saving by ISP and how much increase of energy at users home gateway side in this architecture since we have higher peer contribution. (2) Involving the different popularity model from different geographically VoD service. This is very important to see the system behavior when receiving requests from different locations with different popularity while still in the same global CDN service. Furthermore, we can exploit this behavior for commercial ads in VoD service.

Chapter 5

Energy Savings in Peer-Assisted CDN

5.1 Motivation

We discussed peer-assisted CDN in previous chapter. One of the implications is the possibility to delegate workload from CDN servers to peers. As we mentioned earlier, CDN servers are placed in data centers globally in order to serve clients as fast as possible and as low latency as possible. On the other hand, the data center where the CDN server is placed faces costs for powering the data center. The Uptime Institute, a global data center authority, surveyed 1100 data center owners and operators in 2012 and reported that 55% of organizations will increase their financial budget 10% over 2011 [35]. 30% of organizations were expected run out of data center capacity (power, cooling, space, and network) by the end of 2012 [35]. More than 50% of the organizations surveyed reported that saving energy [1] is a major priority. Even in the data centers using the state of art cooling technologies heat dissipation accounts for at least 20% and as much as 50% of the total power consump-

[1] As we are discussing steady-state operation, energy and power are in direct correspondence so we use the terms interchangeably.

tion [21]. The increases in energy cost and the demand due to growth of traffic urges the data center operators and owners to look for ways to reduce energy usage in the years to come. Although reducing energy consumption can effectively reduce overall cost, this will limit the capacity for growth and scalability of the service provisioning. For example: routers and servers spend most of their energy on the baseline activities such as running the fans, spinning disk, powering the backplane, and powering the memory. Even in an idle state, modern systems can be consuming anything from 50% to 80% of the power consumed under maximum load [7, 10].

Alternatively, the data center can be revamped by relocating some services to end-host computers or peers. Peers contribute their communication, storage, and computation resources to exchange data and provide services while the data center performs central administration and authentication as well as backend processing. A P2P network formed by peers offers flexibility and scalability in service delivery.

We study the energy consumption of hybrid CDN-P2P in two use cases: live streaming and online storage services. It has been shown that CDN energy consumption is better than P2P architecture [6, 18]. The questions are: with the opportunity to offload the CDN' s workload to the peers, how much energy saving can the CDN server get and how large is the difference compared to a pure CDN architecture. If we can estimate the difference between a CDN architecture and a peer-assisted CDN combined with an estimate of peer power consumption, we can use this difference as a basis calculation for giving an incentive to users since peer assisted relies heavily on the user' s uptime and upload rate. Furthermore, since the power consumption is reduced, the power requirement inside the data center can be reduced thus relaxing capacity planning.

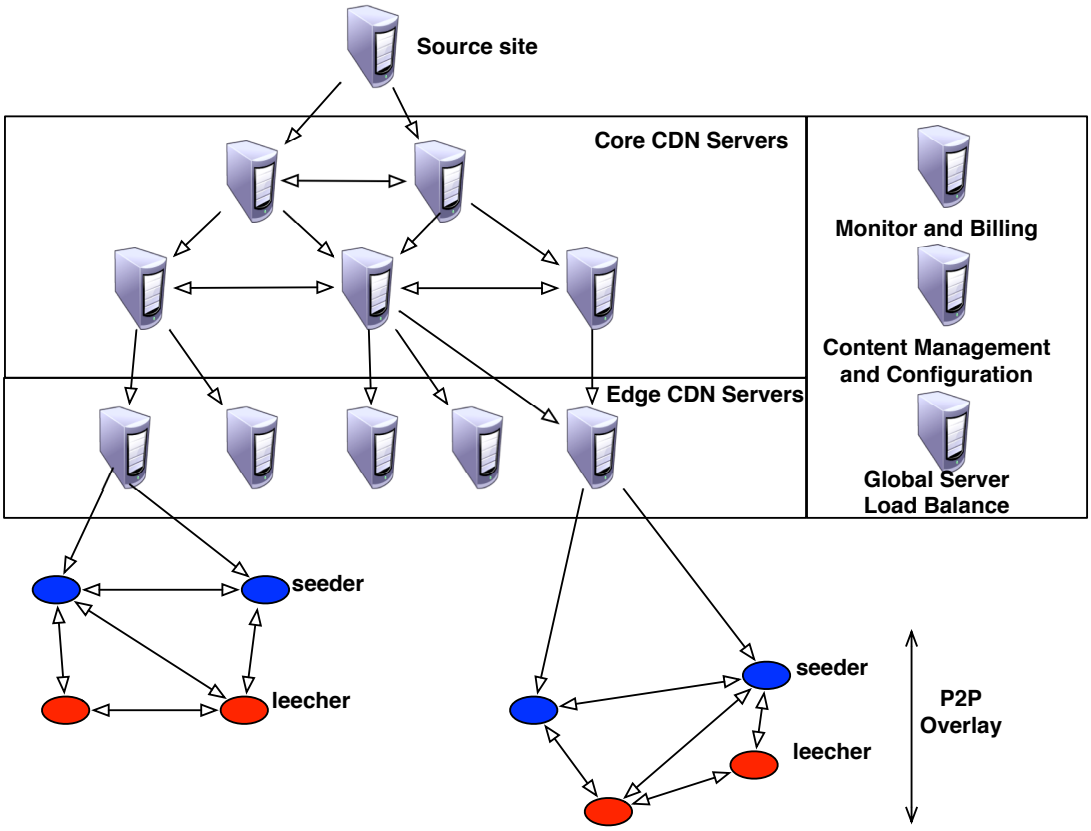

Figure 5.1: Example model of peer-assisted online storage architecture.

5.2 System Description

5.2.1 Live Streaming

Figure 5.1 shows an example model of a peer-assisted CDN for live streaming adapted from [79]. CDN servers deliver video contents from the content provider to end-users. The CDN usually is organized into several tiers usually to cope with the scale demand. Edge CDN servers are directly responsible for serving end users. The goal of the server side peer is for efficient data distribution with some measures to guard against some node failures and network delay.

The CDN overlay is largely tree-based. To provide greater reliability, a CDN node may allow retrieving the content either from other nodes. Edge

CDN servers are responsible for serving end users

For this system, we introduce the concepts of seeder and leecher. A peer that is served by an edge CDN server is called a seeder while a peer that is served by seeders is called a leecher.

A peer obtains the URL from a content source. The global server load balancer finds a suitable edge CDN node for this peer. The peer is then redirected to the nearest edge CDN. The edge CDNs has decision logic that decides if a new peer should be served directly by the edge CDN or if it should be redirected to the P2P overlay.

In the P2P overlay, the stream is separated into several substreams according the stream id and peers are organized in a tree-based overlay. A working peer-assisted CDN live streaming system is defined by parameters such as: (1) video bitrate, (2) the total number of peers, (3) the edge CDN servers bandwidth, and (4) peer upload bandwidth capacity and churn rates.

The maximum number of seeders is bounded by the CDN's capacity, while the maximum number of leechers is bounded by the number of seeders with a certain upload rate. Let us denote the number of the seeders by n_s, the number of leechers by n_l, the maximum bitrate supplied by seeders to leechers by ρ, and the video bitrate by r. The number of leechers that can be supported by seeders is:

$$\lfloor n_l \rfloor = n_s.\rho \tag{5.1}$$

The number of seeders that support or upload content to leechers is:

$$n_s^u = n_l.\frac{r}{\rho} \tag{5.2}$$

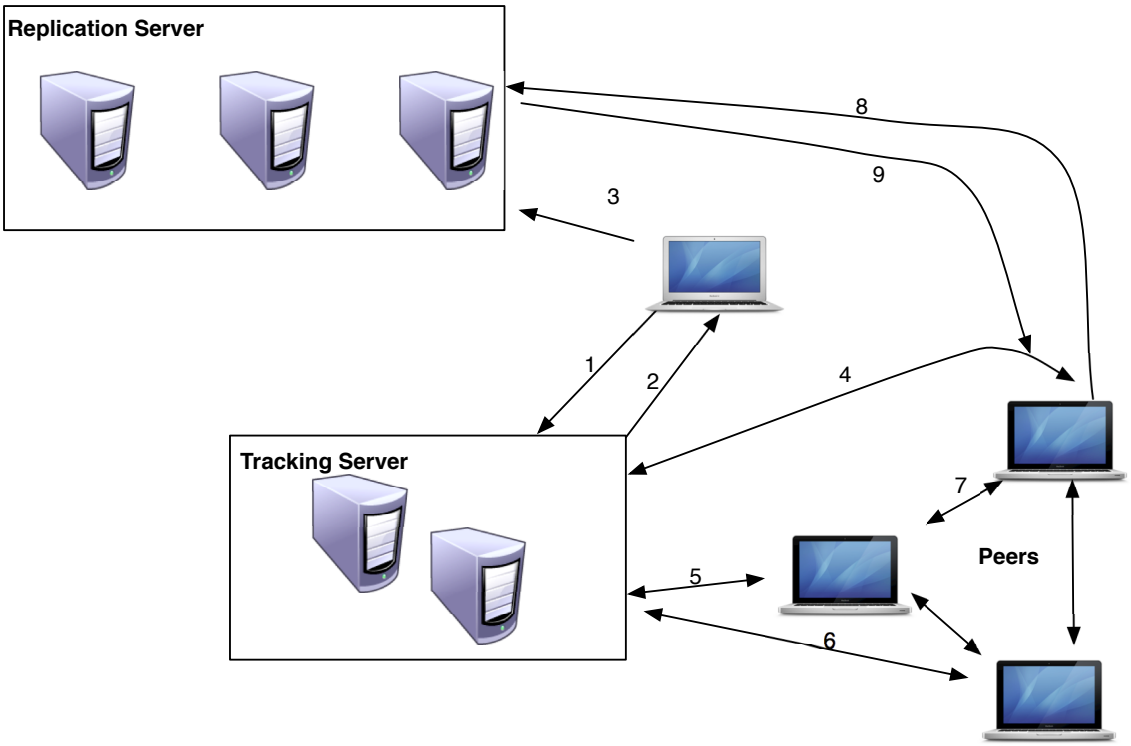

Figure 5.2: Example model of peer-assisted online storage architecture.

In peer-assisted live streaming, we introduce the utilization policy where the CDN server admits peers as seeders as long as the CDN utilization does not exceed 50%, which we defined as 50% of the capacity of a Gigabit Ethernet. When the utilization hits 50%, incoming peers are admitted as leechers, hence they receive the contents from seeders. When more peers join the system and the upload capacity of the seeders is exceeded, the policy raises the utilization cap and the server admits the newly joined peers as seeders. We consider this policy to be better than adding a new server from the point of view of energy consumption.

5.2.2 Peer-Assisted Online Storage

Figure 5.2 illustrates the architecture of peer-assisted online storage for a file hosting system (one-click hosting service with peer-assistance) and inter-

actions among the main components [64]. In this system, each file provided by users is treated as a swarm. Each end user is treated as a peer.

In Fig.5.2, arrows 1, 2, and 3 denote the interaction between a participating peer and tracking server and replication servers for uploading a new file. Arrows 4, 5, and 6 denote the interaction between peers and the tracking server to maintain the peer topology. Arrow 7 denotes the sharing of the file and exchange of availability data among peers. Arrows 8 and 9 represent peer requests and server response.

The tracking server' s function is to maintain swarm information and bootstraps peers. Replication servers working as dedicated content servers have a function for maintaining the availability of swarms when peers do not actively serve them alone.

We choose this peer-assisted online storage model because this model has been implemented widely in China, e.g. FS2You [19], and because one-click file hosting services are very popular right now [44]. Such services rely heavily on server farms inside the data center, thus energy cost becomes important [4]. In this model, since the server holds an important role in this system, we present a simple mathematical model of server bandwidth allocation strategies as a basis for energy calculations [63, 66], as follows:

- Type-1 represents less popular files and type-2 represents popular files.
- S_{t1} represents server bandwidth allocated to a type-1 file and S_{t2} represents server bandwidth allocated to a type-2 file.
- S is the total server bandwidth.
- S_{max1} is the maximum amount of server bandwidth that can be assigned to a file of type-1 and S_{max2} is the maximum amount of server bandwidth that can be assigned to a file of type-2.

- M_{t1} is the number of type-1 files and M_{t2} is the number of type-2 files.
- μ is upload rate of a peer.
- α_{t1} is the arrival rate of new peers in type-1 file and α_{t2} is the arrival rate of new peers in type-2 file.
- $\alpha = M_{t1}\alpha_{t1} + M_{t2}\alpha_{t2}$
- $M = M_{t1} + M_{t2}$
- η_{t1} is the file sharing effectiveness. It is the fraction of the upload capacity of peers that is being utilized for type-1 file.
- η_{t2} is the file sharing effectiveness. It is the fraction of the upload capacity of peers that is being utilized for type-2 file.
- T_d is the average downloading time.
- x_i is the average number of peers.

There are three server bandwidth allocation strategies: (1) lower bound of the average downloading time; (2) request driven strategy; (3) water leveling strategy. In the lower bound strategy, the server uses the bandwidth for type-1 files until S_{t1} reaches its maximum value, then the residual server bandwidth can be assigned to type-2 files. In the request driven strategy, the server serves every request from peers. The server bandwidth is equally divided among all the peers. Let' s assume that the number of requests for a file to the server is proportional to the peer arrival rate of the file. Let' s also assume that when the amount of server bandwidth assigned to one of the two types of files has reached its maximum value, the residual server bandwidth will be assigned to the other type of file. In the water leveling strategy, the server bandwidth is equalized across all the files by taking file popularity into consideration. The server serves the requests from peers according to a certain probability, which

is inversely proportional to the peer arrival rate of the file. Let' s assume that the number of requests for a file to the server is proportional to the peer arrival rate of the file, the server will serve the same number of requests for different files and therefore the server bandwidth is equally allocated across all the files. In order to be able to calculate our power consumption, we need to get the number of peers in the system that can be expressed as [63]:

$$\sum x_i = T_d . \sum \lambda_i \tag{5.3}$$

Furthermore, we can calculate T_d:

$$T_d = \frac{1}{M_{t1}.\lambda_{t1} + M_{t2}.\lambda_{t2}} \left(\frac{M_{t1}.f_{t1}.\lambda_{t1}.\eta_{t2} + M_{t2}.f_{t2}.\lambda_{t2}.\eta_{t1}}{\mu.\eta_{t1}.\eta_{t2}} - \frac{S_{t1}(M_{t1}.\eta_{t2} - M_{t2}.\eta_{t1}) + S.M_{t2}.\eta_{t1}}{\mu.\eta_{t1}.\eta_{t2}} \right) \tag{5.4}$$

5.2.3 Energy Model

Our goal is to provide a general view and fair comparison of the energy consumed by a pure CDN and a hybrid CDN-P2P architecture. To do so, we designed a series of models and performed an analysis. Our energy model is similar to the models used in [48]. The differences with [48] are, firstly, our baseline energy is not a function of bitrate flow. Our baseline energy is based on the minimum energy required to turn on the device without any traffic flowing through the device. Secondly, our overhead ratio is based on the Coefficient of Performance (COP) of the cooling cycle in data center, which we will explain at the end of this section.

Let E_s, E_r, and E_p denote the energy consumption of a single request at each a CDN server, router, and peer respectively. Next, we define baseline

energy consumption as the energy consumed to keep the device on, even when there is no traffic. Let E_{s-base}, E_{r-base}, and E_{p-base} denote the baseline energy consumption for CDN server, router, and peer respectively; and E_{s-max}, E_{r-max}, E_{p-max} denote the power consumption of server, router, and peer when operating at the maximum capacity.

Next, we introduce work-induced energy as the energy consume per bit transferred. Let δ_s,δ_r, and δ_p denote the work-induced energy consumed per additional bit transferred by each CDN server, router, and peer,

$$\delta_s = \frac{(E_{s-max} - E_{s-base})}{M_s} \tag{5.5}$$

$$\delta_r = \frac{(E_{r-max} - E_{r-base})}{M_r} \tag{5.6}$$

$$\delta_p = \frac{(E_{p-max} - E_{p-base})}{M_p} \tag{5.7}$$

Furthermore, we can get:

$$E_s = \delta_s B + E_{s-base} \tag{5.8}$$

$$E_r = d\delta_r B + E_{r-base} \tag{5.9}$$

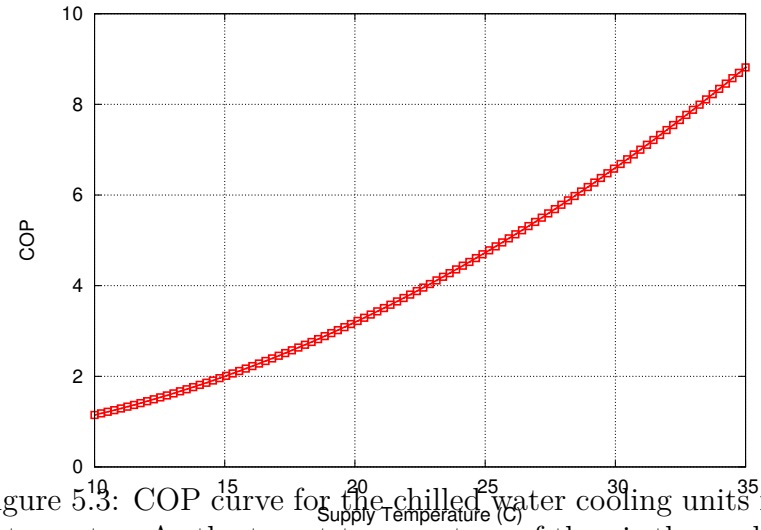

Figure 5.3: COP curve for the chilled water cooling units from HP Lab utility data center. As the target temperature of the air the cooling unit pumps into the floor plenum increases, the COP increases.

$$E_p = \delta_p B + E_{p-base} \tag{5.10}$$

where d is the number of hops and B is the size of file to be transferred in bits.

We now introduce the overhead for the server and routers. The only overhead that we will consider here is cooling power. Since servers and routers are placed in the data center, the data center needs to be provisioned with adequate cooling. This cooling overhead in the data center is quantified by the coefficient of performance (COP). The COP value itself has been empirically determined to be [47]:

$$COP(T) = 0.0068.T^2 + 0.0008.T + 0.458 \tag{5.11}$$

Where T is the temperature supplied by the cooling unit in Celsius. Figure

5.3 shows the $COP(T)$ value for every T. Finally, the cooling cost can be calculated as [47]:

$$C = \frac{Q}{COP(T)} \tag{5.12}$$

Where Q is the amount of power consumed by the servers and hardware. We assume a uniform T at each cooling unit. Taking into account the cooling energy overhead, the total energy consumption is as follows:

$$E_t = E_s\left(1 + \frac{1}{COP(T)}\right) + E_r\left(1 + \frac{1}{COP(T)}\right) \tag{5.13}$$

We do not include the cooling overhead in the peer energy consumption because most of the peers in homes do not need a separate cooling supply.

5.3 Result and Analysis

5.3.1 Numerical Parameters

The parameters used in this analysis were adapted from [6, 48, 63, 71]. The parameters values are shown in Table 5.1. We choose the numerical parameters from [6, 48, 63, 71] because these parameters were gathered from empirical measurements.

5.3.2 Live Streaming Service

In the live streaming case, each video stream flows from the CDN node to the network, in this case a router, and then arrives at the seeders. If seeders need to upload data to leechers it will flow through the router then arrive at

Table 5.1: Numerical Simulation Parameters.

Symbol	Description	Values
δ_s	Work induced at server per bit transferred	$5.2.10^{-8}$ (J/b)
δ_r	Work induced at router per bit transferred	$8.0.10^{-9}$ (J/b)
δ_p	Work induced at peer per bit transferred	$16.10^{-}9$ (J.b)
E_{r-base}	Router baseline power consumption	750 watt
E_{s-base}	Server baseline power consumption	290 watt
E_{p-base}	Peer baseline power consumption	13.5 watt
r	Video bitrate in live streaming	1Mbps
d	Number of hops	1
N_s^u	Upload rate of peers in live streaming	$[0.25, 0.5, 0.75, 1]$ Mbps
N	Number of peers in live streaming	$[100, .., 1000]$
δ_{t1}	Type-1 peer arrival rate to less popular files in online storage (Poisson process)	0.1
δ_{t2}	Type-2 peer arrival rate to less popular files in online storage (Poisson process)	1
η_{t1}	File type-1 sharing effectiveness. The fraction of the upload capacity of peers that is being utilized in online storage	0.5
η_{t2}	File type-2 sharing effectiveness. The fraction of the upload capacity of peers that is being utilized in online storage	1
M_{t1}	Number of files in type-1 files or less popular files	10
M_{t2}	Number of files in type-2 files or less popular files	1
$f_{t1} = f_{t2}$	File size in online storage	100 MB
μ	Upload rate of peers in online storage	0.5Mbps
c	Downloading rate of peers in online storage	1Mbps
T	Air temperature supplied form cooling unit in data center	$[20, 25]$ correspond to COP value $[3.194, 4.728]$

the leechers. We show the logical network architecture of peer-assisted CDN for live streaming in Fig.5.1. Fig.5.4 shows a simplified physical representation of the network. While in a logical network the peer can communicate directly with another peer, in a real physical world the communication between peers always passes through a router inside the data center.

Figure 5.5 shows energy usage for CDN server, router, and the total energy consumption for the CDN scenario (without peer assist). We plot the energy consumption for CDN server, router, clients, and total energy for two COP coefficient values (T). All energy consumption components increase in value

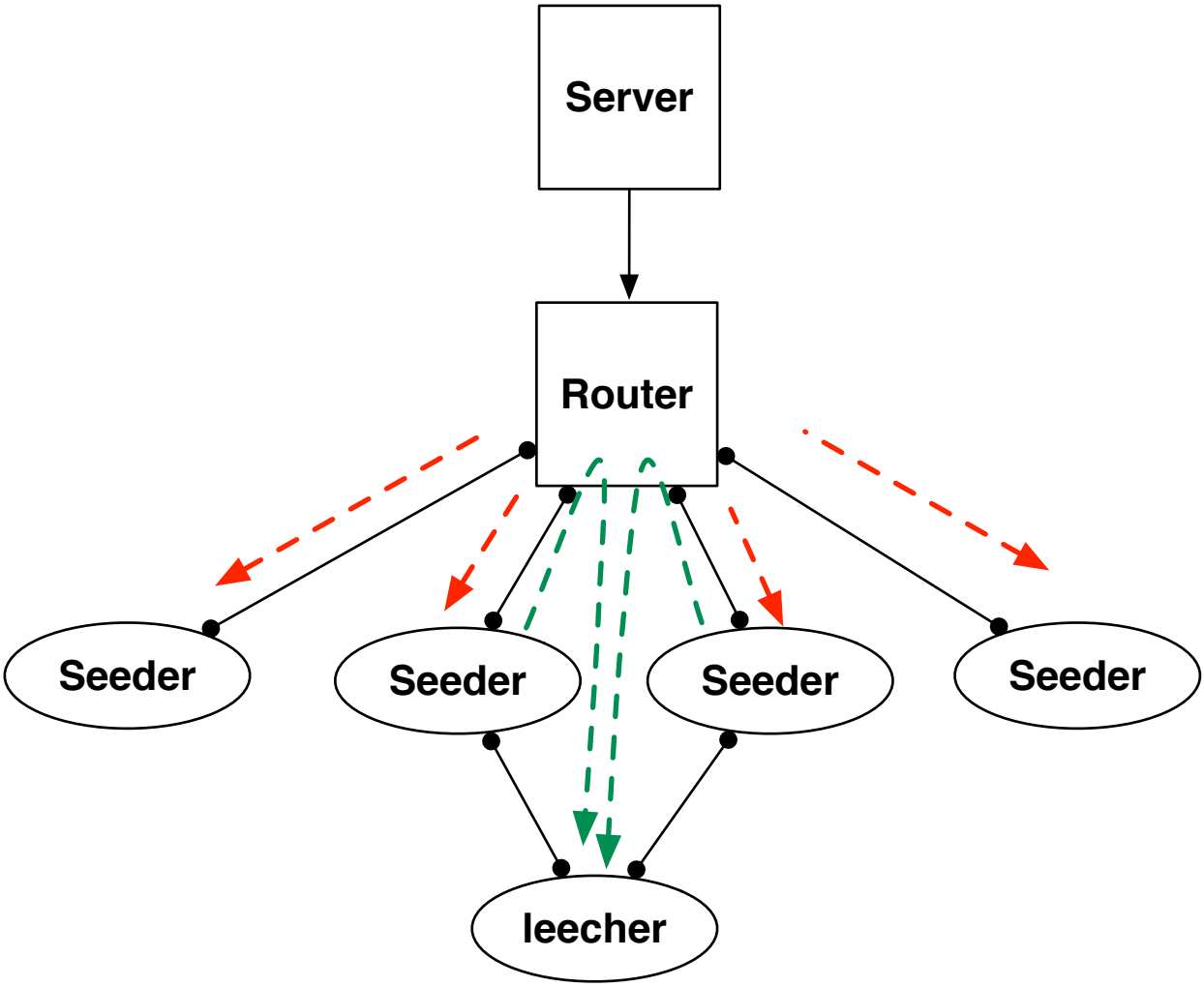

Figure 5.4: Simplified physical representation flow of data.

as the number of peers increases; and peers consume most of the energy. The effect of variations in T on total energy is small

Figure 5.6 shows the energy consumption of all components for the CDN-P2P scenario. We use the peer upload rate $N_s^u = 0.75$ in Fig.5.6. We observe that there is almost no change in peers' power consumption compared to the pure CDN. The router consumes more power with a higher rate of increase because in CDN-P2P peers the traffic originated from the seeders passes trough the router twice. The CDN server power consumption has small increases between $N = 100$ and $N = 500$, while there are sections with no power increase

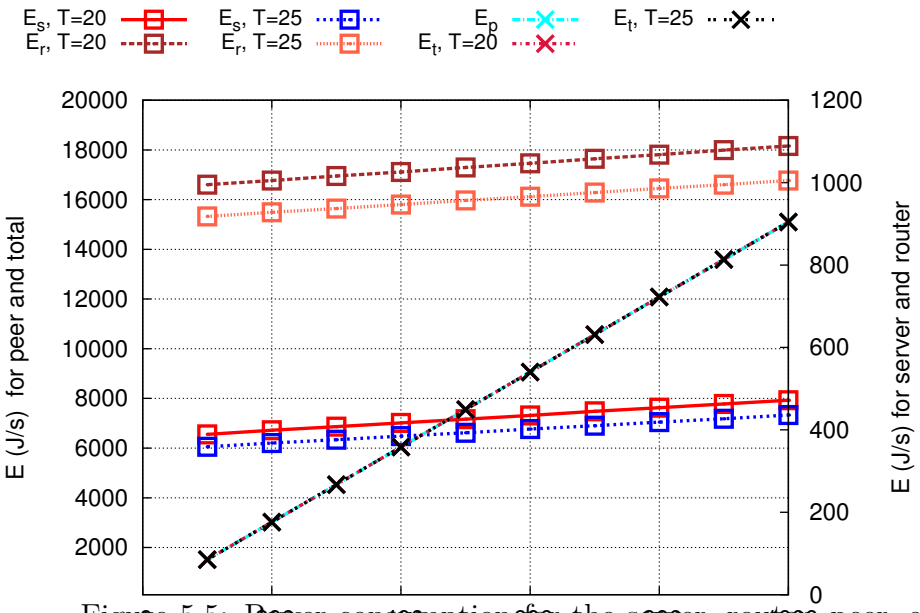

Figure 5.5: Power consumption for the server, router, peer, and total system for CDN architecture. Note that the server and router energy are plotted using the right hand scale, and the peer and total energy are plotted using the left hand scale.

at $N > 500$, which is where the seeders are uploading contents to the leechers. The server power consumption remains flat as long as the upload rate does not exceed the defined peer upload rate.

Figure 5.7 shows the energy savings of CDN-P2P compared to CDN architecture for CDN server with the utilization policy as explained in Sec.5.2 for $N_s^u = [0.25, 0.5, 0.75, 1]$.

Let's take $N_s^u = 0.75$ as an example. The first 500 nodes are served directly by the CDN server since the utilization of the CDN server is 50% or less. We consider 500 nodes to be 50% utilization because a video-rate of 1 Mbps will result in total network traffic of 0.5 Gbps, which is half of a Gigabit Ethernet interface. When more peers join the system, these peers will be treated as leechers as long as the upload ratio condition is fulfilled. The number of leechers that can be supported by seeders is 375. Therefore from $N = 500$ to $N = 875$, the CDN server does not need to increase utilization because the

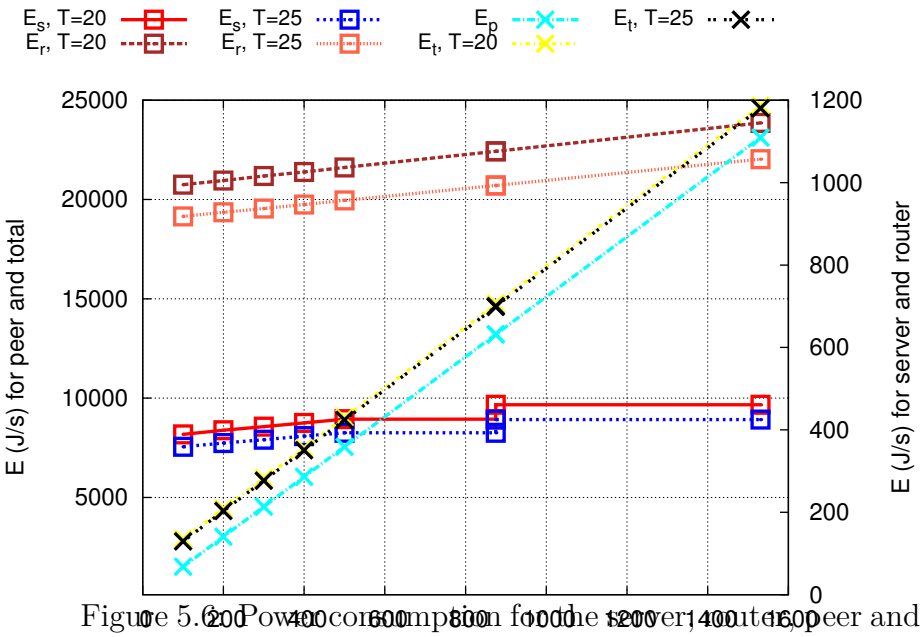

Figure 5.6: Power consumption for the server, router, peer and total system for peer-assisted CDN with $N_s^u = 0.75$. Note that the server and router energy are plotted using the right hand scale, and the peer and total energy are plotted using the left hand scale.

leechers can be supported by seeders, thus we see that the CDN server saves energy. In this phase, compared to CDN architecture, CDN-P2P energy saving is around 7.8%. Next, we have 875 total peers in the system, is apportioned into 500 peers as seeders and 375 as leechers. Since more peers joining the system, the CDN increases the utilization from 50% to 87.5% so all current 875 peers become seeders. In this phase, 875 seeders can support an additional 656 leechers. Therefore, from $N = 875$ to $N = 1531$ the CDN utilization is flat at 87.5% because 875 seeders can support 656 leechers thus we have energy savings around 11% compared to CDN architecture. Other values of N_s^u have same pattern as shown in Fig.5.7.

Figure 5.8 shows the total energy savings of CDN-P2P compared to CDN architecture for $N_s^u = [0.25, 0.5, 0.75, 1]$. As the savings only occurs in the CDN server, we see the same patterns as in Fig 5.7 but with a much lower percentage of energy savings, which is 1%.

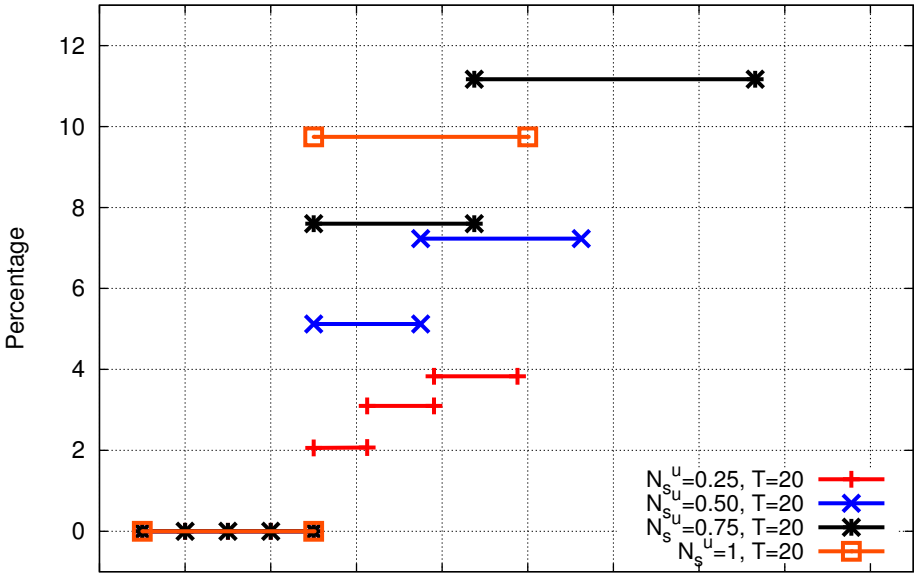

Figure 5.7: Savings in power consumption between CDN architecture and peer-assisted CDN for server with $N_s^u = [0.25, 0.5, 0.75, 1]$.

5.3.3 Online Storage

To calculate the energy consumption in peer-assisted online storage, we must be able to determine the number of peers in the system. We get average downloading time (T_d) values by varying server bandwidth values (S) from 0 to 150 MBps using Eq 5.4. After getting the average downloading time, we can get the average number of peers using Eq 5.3. We found that the number of peers is inversely related to the server bandwidth. The number of peers is the horizontal axis in Fig 5.9 and Fig 5.10. The figures cover more number of peers compared to the live streaming service and we can look at the comparable number of peers in both cases when we want to do comparisons.

Figure 5.9 shows the power consumption for the lower-bound strategy for the server, router, peers, and the total system. We found that increasing the number of peers decreases CDN server power consumption because the bandwidth usage of the CDN server decreases. The router power consumption

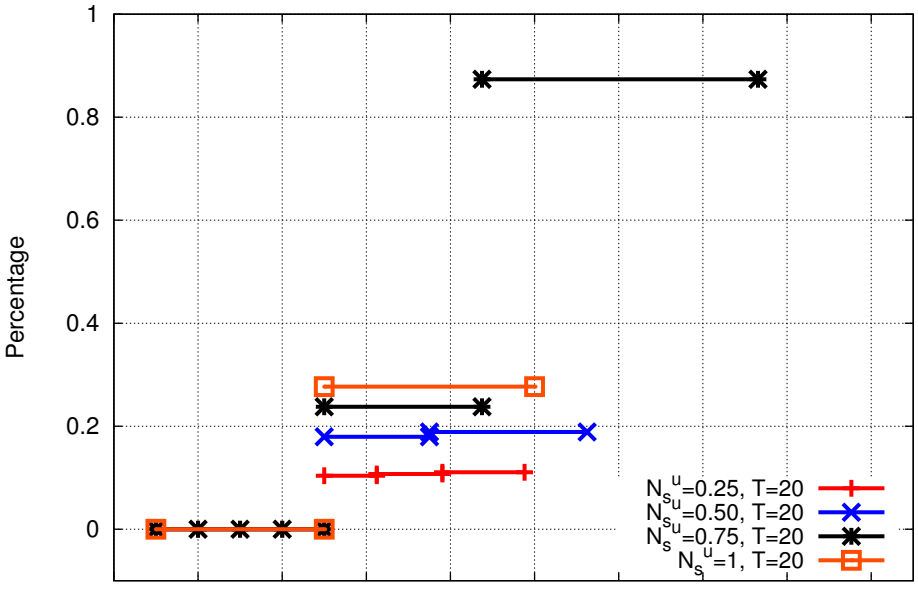

Figure 5.8: Savings in power consumption between CDN architecture and peer-assisted CDN for total system with $N_s^u = [0.25, 0.5, 0.75, 1]$.

is flat at around 1000J/s because the server bandwidth reduction is offset by the increasing number of peers. We also found that the other strategies, request driven and water leveling, have the same pattern as the power consumption of the lower-bound strategy.

Figure 5.10 shows a comparison of the energy consumption between the request driven and the lower bound strategy, and between the water-leveling strategy and the lower bound strategy for different numbers of peers. Compared to the water-leveling strategy, the request driven strategy required more energy because the request driven strategy equalizes the server bandwidth across all the peers. The water-leveling strategy equalizes server bandwidth across all the files by taking file popularity into consideration, thus minimizing downloading time. We mentioned before that the number of peers is inversely related to the server bandwidth, therefore for the same server bandwidth, we get different numbers of peers for each strategy. This implies that for the same number of peers, we get different server bandwidth. That is the reason

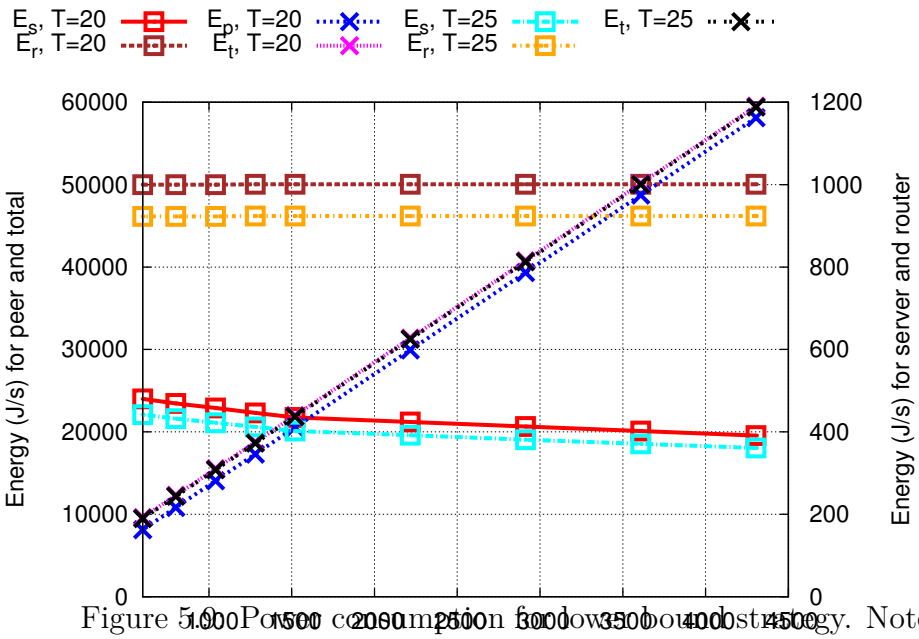

Figure 5.9: Power consumption for low bound strategy. Note that the server and router energy are plotted using the right hand scale, and the peer and total energy are plotted using the left hand scale.

for $1000 < N < 2500$ the power consumption diverges. In very limited server bandwidth (less than 45MBps) and sufficient server bandwidth (more than 120MBps) each strategy has the same downloading performance. That is the reason for $N < 1000$ and $N > 2500$ we have the same the number of peers for same bandwidth. As shown in Fig 5.10, for $N < 1000$ and $N > 2500$ the savings for each strategy is relatively the same.

File popularity has a strong correlation with downloading performance. We examine popularity by varying the peer arrival rate of less popular files while the server has fixed bandwidth. Specifically, we increase the type-1 files popularity from 0.1 to 1 and we choose a fixed server bandwidth $S = 50$ MBps which is similar to FS2You. Figure 5.11 shows the difference in total power consumption (left axis) and router power consumption (right axis) of that case. Since the server bandwidth is fixed, we only show the power consumption changes in the routers and the total. When the arrival rate is less than 0.5,

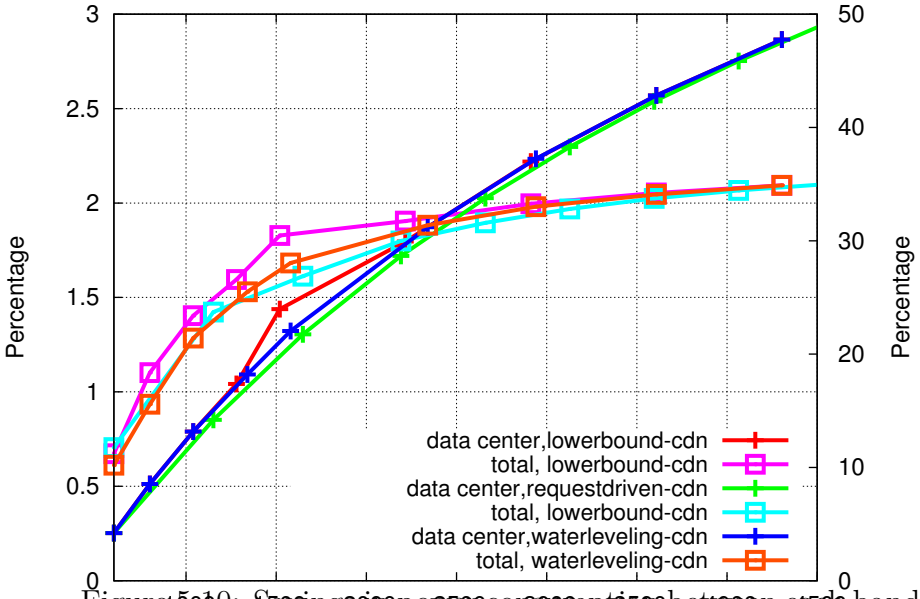

Figure 5.10: Saving in power consumption between end bandwidth allocation and CDN architecture for total and server. Note that the server and router energy (data center) are plotted using the right hand scale, and the peer and total energy are plotted using the left hand scale.

the request driven strategy has worse downloading performance compared to the water leveling strategy. This implies more peers exist in the request driven strategy than the water leveling strategy. Therefore, the energy consumption of the request driven strategy is higher than the water leveling strategy. Generally for each strategy, increasing the peer arrival rates to less popular file makes the total energy consumption and router energy consumption increase because more peers are present in the system. Increasing the peer arrival rates to the less popular file makes both the request driven and the water leveling strategy energy consumption converge to a lower bound. This is because more peers in the system improve P2P content availability, thus improving downloading performance that converges to the lower bound strategy.

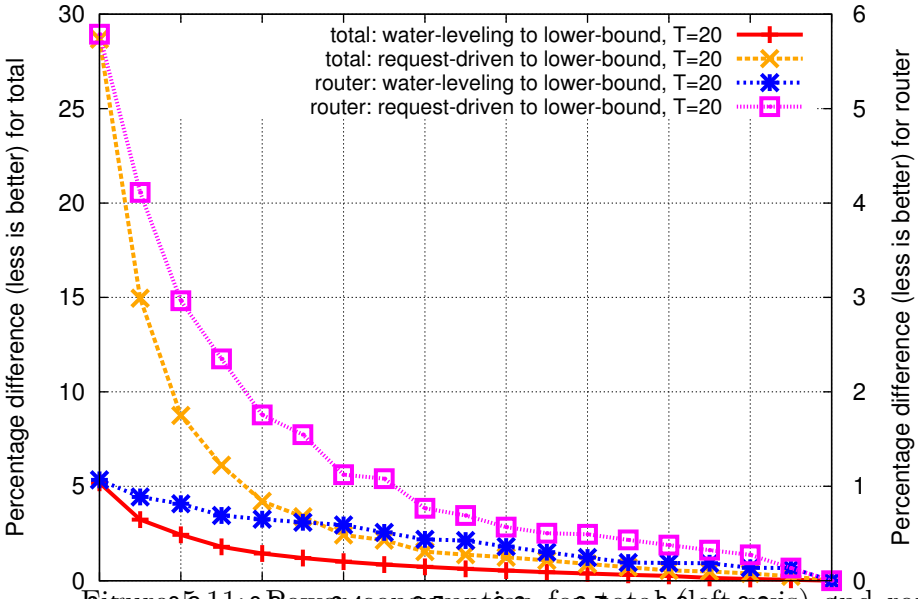

Figure 5.11: Power consumption for total (left axis) and router (right axis) under different server bandwidth allocation strategies when peer arrival rate of less popular varies. We use fixed server bandwidth $S = 50$ MBps.

5.4 Summary

We compared the energy consumption between peer-assisted CDN and pure CDN for live streaming and online storage services both at the data center as well as in total. Employing peer-to-peer capability to assist a CDN is thought to lower the energy requirements at data centers, and we found that the maximum savings at the data centers are 11% and 21%, respectively for the live streaming and online storage services. These savings may change depending on the COP values used and should be better if a new generation of power proportional server were used. One thing to note is that as the number of peers increases, the server' s energy consumption increases for the live streaming and decreases for the online storage service due to the differences in the ways both services handle peers. However, the server's energy consumption is swamped by the peers' energy consumption. Despite this difference in behavior in the two cases, when comparing Peer Assisted CDN to pure CDN,

we found the total energy savings of less than 1%. Nevertheless, the total energy consumption is large, so that even a small percentage improvement results in valuable net reduction. Several areas that we identified as the future work are: 1. The effect of peer' s uptime variations; 2. More realistic file popularity models for the online storage service; and 3. How CDN providers or ISPs give incentives to the peers based on the understanding of the energy consumptions.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

We have investigated the properties of BitTorrent overlay topologies from the point of view of the peer exchange protocol using real swarms from an operational BitTorrent tracker on the Internet.

Our results agree in some particulars and disagree in others with prior published work on isolated testbed experiments, suggesting that more work is required to fully model the behavior of real-world BitTorrent networks.

We find that the node degree of the graph formed in a BitTorrent swarm can be described by a power law with exponential cut-off and the observation of a low clustering coefficient implies BitTorrent networks are close to random networks. From the BitTorrent protocol point of view, the reason that a BitTorrent swarm can be described by a power-law with exponential cut-off is: leechers in a BitTorrent swarm prefer a few good seeders or neighbors that can give high data rates to exchange the data and seeders have rich connections to leechers as seeders have complete chunks or pieces. That behavior explains why seeders have rich connections while leechers only have a few neighbors. We argue that there are two reasons for the cut-off phenomenon. First, most Bit-

Torrent clients configure the maximum number of global connection between $200-300$, however the maximum number of connections per torrent (swarm) is set between $50-90$ by default [58] [70]. Some BitTorrent forums suggest decreasing the maximum connection for torrent (swarm) to between $30-40$ [69]. Second, most of the BitTorrent users are home users where their home gateway device cannot support a high number of concurrent connections and BitTorrent is not the main online activity. We argue that the BitTorrent swarm closes to random graph that we infer from clustering coefficient is caused by BitTorrent mechanism itself that always choose random peers from its neighbors in the choking-unchoking algorithm, optimistic choking algorithm, and optimistic connect algorithm as we explained previously.

In peer-assisted CDN, we show that by introducing the z factor, which represents the activity phase into utility function of the PROP algorithm, we can increase the peer contribution to deliver the content while decreasing the required number of replicas. Implication of higher peer contribution means CDN can reduce workload because some workload is delegated to the peer-to-peer network.

Integrating peer-to-peer capability to assist the existing CDN has the potential to save energy. In this study, we show that event without explicitly considering energy consumption while assigning content, the peer assisted CDN can save energy. Although the energy savings depend on the number of request (number of clients), number of routers and their configuration, for total system energy saving is around 0.5% to 1.2%. If we break it down per component, the CDN server is the part that can be pushed to save energy, up to 11% and could save if the new generation of power proportional server is used [38]. We agree with Chabarek et al., [10], router component in the other side is quite difficult for energy saving, because different chassis size, different network interface type slot, and different configuration has different energy

consumption. Several areas that we have been identified for future work are: more correlation analysis of time period to peer energy usage pattern in live streaming, continued characterization of different peer energy usage based on flash memory storage, and comparing energy model with different file popularity models.

6.2 Future Work

Some areas of improvement that we have identified for future work are: more correlation analysis of the number of peers with α and p value, continued characterization with NATed peers, wider likelihood ratio test with other models and comparing the results with simulation for global graph properties such as distance distribution. We hope to incorporate these properties into a complete dK series for the evolution of a real-world BitTorrent overlay as it evolves over time [43]. We conclude that further work throughout the community is necessary to continue to improve the agreement of simulation and controlled experiment with the real world, and that such work will impact our understanding of BitTorrent performance and its effects on the Internet. Several areas that we have been identified for future work in energy consumption of peer-assisted CDN: more correlation analysis of time period to peer energy usage pattern in live streaming, continued characterization of different peer energy usage based on flash memory storage, and comparing energy model with different file popularity models.

Appendices

Appendix A

Some Detail on BitTorrent Terminology

In this appendix, we will give more detail explanation some BitTorrent terminology based on BitTorrent specification.

A.1 Torrent File

Torrent file or metainfo file has always ending in ".torrent". This file is a bencoded dictionary, containing structure as follows:

- info: a dictionary described the file of the torrent.
- announce: the announce URL of the tracker.
- creation date: the creation time of the torrent in standard UNIX epoch format.
- comment: free text contain comments from the author.
- created by: name and version of the application used to create this torrent file or metainfo file.

- encoding: the string encoding format used to generate the pieces/chunks/blocks of the info dictionary in torrent file.

info itself has structure as follows:

- piece length: the length of a piece/chunk/block in bytes. it is usually in power of 2. Common size for piece length is 256KB.
- pieces: concatenation of all 20 byte SHA1 hash value each pieces/chunks/blocks.
- private: this optional to denote private torrent.

A.2 Handshake

Handshake is the first message that must sent by BitTorrent client when contacting other peers. Handshake message format as follows:

`handshake: <pstrlen><pstr><reserved><info_hash><peer_id>`

The details as follows:

- pstrlen is string length of <pstr>.
- pstr is string identifier of the protocol.
- reserved is reverse bit for changing the behavior of protocol.
- info_hash is 20 byte SHA1 hash of the info key in torrent file.
- peer_id is 20 byte string used as a unique ID for the client.
- name: the file name of content.
- length: length of the file in bytes.
- md5sum: hexadecimal string corresponding to the MD5 sum of the file.

A.3 HAVE

HAVE message is the message that sent by a peer to other peers. The purpose of this message to let other peers know the pieces/chunks that has just downloaded. The format of HAVE message as follows:

```
have: <len=0005><id=4><piece index>
```

piece index is the index of a piece that has just successfully downloaded and verified by the hash.

A.4 BITFIELD

BITFIELD message is the message that sent by peer to other peers. The purpose of this message to let other peers know total pieces/chunks that a peer have. The format of BITFIELD message as follows:

```
bitfield: <len=0001+X><id=5><bitfield>
```

The BITFIELD message has variable length because its depend on number of information of pieces/chunks that have been downloaded.

A.5 PEX

There are two PEX implementation which are Azareus and Libtorrent (UT_PEX). We will use the UT_PEX as reference. Since PEX is extension, to be able to use that extension, the reserved field in handshake message must be filled. In this case, the 5th bit of the 6th byte of the reserved. Next, the handshake message informs other peers which extended messages are supported and what's the extended id will be used.

The payload of a peer exchange message is a bencoded dictionary with the following keys:

- added: contains list of peers in the compact tracker format since the last peer exchange message.
- added.f: one byte of flags for each peer in the above added string. This contains information about other supported things for UT_PEX. for example in UT_PEX, byte 1 is for encryption thus the peer support encryption.
- dropped: contains list of peers that dropped since the last peer exchange message.

Appendix B

Likelihood Ratio Test

Assume we have two different distributions with PDFs $p_1(x)$ and $p_2(s)$. The likelihood of a given data set within two distributions are

$$L_1 = \prod_{i=1}^{n} p_1(x_i) \qquad L_2 = \prod_{i=1}^{n} p_2(x_i) \tag{B.1}$$

and the likelihood is:

$$R = \frac{L1}{L2} = \prod_{i=1}^{n} \frac{p_1(x_i)}{p_2(x_i)} \tag{B.2}$$

taking log, the log likelihood ratio is:

$$\mathcal{R} = \sum_{i=1}^{n} [ln\, p_1(x_i) - ln\, p_2(x_i)] = \sum_{i=1}^{n} [\ell_i^{(1)} - \ell_i^{(2)}] \tag{B.3}$$

where $\ell_i^{(j)}$ is the log-likelihood in distribution j. As we assume that x_i are independent, thus $\ell_i^{(1)} - \ell_i^{(2)}$ is also independent. By the central limit theorem, their sum $\mathcal{R}$ will be in normal distribution as n grows. Expected variance can be estimate as:

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}[(\ell_i^{(1)} - \ell_i^{(2)}) - (\bar{\ell}_i^{(1)} - \bar{\ell}_i^{(2)})]^2 \tag{B.4}$$

where

$$\bar{\ell}^1 = \frac{1}{n}\sum_{i=1}^{n}\ell_i^{(1)} \qquad \bar{\ell}^2 = \frac{1}{n}\sum_{i=1}^{n}\ell_i^{(2)} \tag{B.5}$$

The probability that the measured of log-likelihood ratio has magnitude as large as or larger than observed value $|\mathcal{R}|$ is:

$$p = erfc(\frac{|\mathcal{R}|}{\sqrt{2n\sigma}}) \tag{B.6}$$

and

$$erfc(z) = 1 - erf(z) \tag{B.7}$$

is the complementary Gaussian error function which is available widely in scientific computing library. This p value is also call significance value for log-likelihood test. In nested hypothesis where we compare two distributions under same family, if p-value is small, the smaller family is better than larger family. if not then we can not say no evidence that larger family is needed, while the smaller family gives better fit.

Appendix C

Research History

C.1 Journal

1. Mohamad Dikshie FAUZIE, Achmad Husni THAMRIN, Rodney VAN METER, Jun MURAI, "Assessing the Dynamics of Bittorrent Swarms Topologies Using the Peer Exchange Protocol" IEICE Transactions on Communications, VOL.E95-B, NO.5, pp.1566-1574, May 2012.

2. Mohamad Dikshie Fauzie, Achmad Husni Thamrin, Jun Murai, "Energy Consumption in Peer-Assisted CDN", International Journal of Advanced Research in Computer Science (IJARCS). Volume 4.No.9. July-August 2013. ISSN 0976-5697.

C.2 International Conference

1. Mohamad Dikshie Fauzie, "A Temporal View of The Topology of Dynamic Bittorrent Swarms", Third International Workshop on Network Science for Communication Networks at IEEE INFOCOM 2011, April 2011, Shanghai, CHINA.

C.3 Miscellaneous Publication

1. Mohamad Dikshie Fauzie, Achmad Husni Thamrin, Rodney Van Meter, and Jun Murai, "Capturing The Dynamic Properties of Bittorrent Overlays", Asia Future Internet August 2010 Summer School.
   `http://www.asiafi.net/meeting/2010/summerschool`

Bibliography

[1] Abhari, A., and Soraya, M. Workload generation for youtube. *Multimedia Tools and Applications 46*, 1 (2010), 91–118.

[2] Al Hamra, A., Legout, A., and Barakat, C. Understanding the properties of the bittorrent overlay. *CoRR abs/0707.1820* (2007).

[3] Al-Hamra, A., Liogkas, N., Legout, A., and Barakat, C. Swarming overlay construction strategies. In *Computer Communications and Networks, 2009. ICCCN 2009. Proceedings of 18th Internatonal Conference on* (2009), IEEE, pp. 1–6.

[4] Arstechnica. Megaupload data cost. `http://arstechnica.com/tech-policy/2012/03/isp-storing-25-petabytes-of-megaupload-data-costs-us-9000-a-day/`.

[5] Balakrishnan, H., Kaashoek, M. F., Karger, D., Morris, R., and Stoica, I. Looking up data in p2p systems. *Commun. ACM 46*, 2 (Feb. 2003), 43–48.

[6] Baliga, J., Hinton, K., and Tucker, R. S. Energy consumption of the internet. In *Optical Internet, 2007 and the 2007 32nd Australian Conference on Optical Fibre Technology. COIN-ACOFT 2007. Joint International Conference on* (2007), IEEE, pp. 1–3.

[7] Barroso, L., and Holzle, U. The case for energy-proportional computing. *Computer 40*, 12 (2007), 33–37.

[8] Borghol, Y., Mitra, S., Ardon, S., Carlsson, N., Eager, D., and Mahanti, A. Characterizing and modelling popularity of user-generated videos. *Perform. Eval. 68*, 11 (Nov. 2011), 1037–1055.

[9] Brands, E. H. T. B., and Karagiannis, G. Taxonomy of p2p applications. In *GLOBECOM Workshops, 2009 IEEE* (Nov 2009), pp. 1–8.

[10] Chabarek, J., Sommers, J., Barford, P., Estan, C., Tsiang, D., and Wright, S. Power awareness in network design and routing. In *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE* (2008), pp. 457–465.

[11] Cisco. Cisco visual networking index. `http://www.cisco.com/en/US/netsol/ns827/networking_solutions_sub_solution.html`.

[12] Clauset, A., Shalizi, C., and Newman, M. Power-law distributions in empirical data. *SIAM review 51*, 4 (2009), 661–703.

[13] Cuevas, R., Kryczka, M., Cuevas, A., Kaune, S., Guerrero, C., and Rejaie, R. Unveiling the incentives for content publishing in popular bittorrent portals. *Networking, IEEE/ACM Transactions on 21*, 5 (Oct 2013), 1421–1435.

[14] Dale, C., Liu, J., Peters, J., and Li, B. Evolution and enhancement of bittorrent network topologies. In *Quality of Service, 2008. IWQoS 2008. 16th International Workshop on* (2008), IEEE, pp. 1–10.

[15] Doyle, J., Alderson, D., Li, L., Low, S., Roughan, M., Shalunov, S., Tanaka, R., and Willinger, W. The robust yet

fragile nature of the internet. *Proceedings of the National Academy of Sciences of the United States of America 102*, 41 (2005), 14497.

[16] Fauzie, M. D., Thamrin, A. H., and Murai, J. Energy consumption in peer-assisted cdn. *International Journal of Advanced Research in Computer Science (IJARCS)* (2012).

[17] Fauzie, M. D., Thamrin, A. H., Van Meter, R., and Murai, J. A temporal view of the topology of dynamic bittorrent swarms. In *Proc. IEEE Conf. Computer Communications Workshops (INFOCOM WKSHPS)* (2011), pp. 894–899.

[18] Feldmann, A., Gladisch, A., Kind, M., Lange, C., Smaragdakis, G., and Westphal, F.-J. Energy trade-offs among content delivery architectures. In *Telecommunications Internet and Media Techno Economics (CTTE), 2010 9th Conference on* (2010), IEEE, pp. 1–6.

[19] FS2You. Fs2you. `http://www.rayfile.com/en/`.

[20] Giannetti, I., Anastasi, G., and Conti, M. Energy-efficient p2p file sharing for residential bittorrent users. In *Computers and Communications (ISCC), 2012 IEEE Symposium on* (2012), pp. 000524–000529.

[21] Google. Google data center power usage. Available on `http://www.google.com/corporate/datacenters/measuring.html`.

[22] Grindrod, P., and Higham, D. J. Evolving graphs: dynamical models, inverse problems and propagation. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Science 466*, 2115 (Mar. 2010), 753–770.

[23] Guan, K., Atkinson, G., Kilper, D., and Gulsen, E. On the energy efficiency of content delivery architectures. In *Communications*

Workshops (ICC), 2011 IEEE International Conference on (2011), pp. 1–6.

[24] Guillemin, F., Houdoin, T., and Moteau, S. Volatility of youtube content in orange networks and consequences. In *Communications (ICC), 2013 IEEE International Conference on* (June 2013), pp. 2381–2385.

[25] Guo, L., Chen, S., Xiao, Z., Tan, E., Ding, X., and Zhang, X. Measurements, analysis, and modeling of bittorrent-like systems. In *Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement* (2005), USENIX Association, p. 4.

[26] Guo, L., Chen, S., and Zhang, X. Design and evaluation of a scalable and reliable p2p assisted proxy for on-demand streaming media delivery. *Knowledge and Data Engineering, IEEE Transactions on 18*, 5 (May 2006), 669–682.

[27] Gupta, M., and Singh, S. Greening of the internet. In *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications* (New York, NY, USA, 2003), SIGCOMM '03, ACM, pp. 19–26.

[28] Hei, X., Liang, C., Liang, J., Liu, Y., and Ross, K. A measurement study of a large-scale p2p iptv system. *Multimedia, IEEE Transactions on 9*, 8 (2007), 1672–1687.

[29] Hossfeld, T., Hock, D., Oechsner, S., Lehrieder, F., Despotovic, Z., Kellerer, W., and Michel, M. Measurement of bittorrent swarms and their as topologies. Technical Report 464, University of Wurzburg, November 2009.

[30] Huang, C., Li, J., and Ross, K. Peer-assisted vod: Making internet video distribution cheap. In *Proc. of IPTPS* (2007), vol. 7.

[31] Huang, C., Li, J., and Ross, K. W. Can internet video-on-demand be profitable? *SIGCOMM Comput. Commun. Rev. 37*, 4 (Aug. 2007), 133–144.

[32] Huang, C., Wang, A., Li, J., and Ross, K. W. Understanding hybrid cdn-p2p: why limelight needs its own red swoosh. In *Proceedings of the 18th International Workshop on Network and Operating Systems Support for Digital Audio and Video* (New York, NY, USA, 2008), NOSSDAV '08, ACM, pp. 75–80.

[33] Inc., A. Available on `http://support.amd.com/us/Embedded_TechDocs/20213.pdf`.

[34] Inc., C. Available on `http://www.cisco.com/web/partners/downloads/765/other/Energy_Logic_Reducing_Data_Center_Energy_Consumption.pdf`.

[35] Institute, U. the 2012 uptime institute data center industry survey. Available on `http://uptimeinstitute.com/2012-survey-results`.

[36] Jacobson, V., Smetters, D. K., Thornton, J. D., Plass, M. F., Briggs, N. H., and Braynard, R. L. Networking named content. In *Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies* (New York, NY, USA, 2009), CoNEXT '09, ACM, pp. 1–12.

[37] Jiang, H., Li, J., Li, Z., and Liu, J. Efficient hierarchical content distribution using p2p technology. In *Networks, 2008. ICON 2008. 16th IEEE International Conference on* (dec. 2008), pp. 1 –6.

[38] Krioukov, A., Mohan, P., Alspaugh, S., Keys, L., Culler, D., and Katz, R. Napsac: design and implementation of a power-

proportional web cluster. *SIGCOMM Comput. Commun. Rev. 41*, 1 (Jan. 2011), 102–108.

[39] Kryczka, M., Cuevas, R., Guerrero, C., and Azcorra, A. Unrevealing the structure of live bittorrent swarms: Methodology and analysis. In *Proc. IEEE Int Peer-to-Peer Computing (P2P) Conf* (2011), pp. 230–239.

[40] Legout, A., Liogkas, N., Kohler, E., and Zhang, L. Clustering and sharing incentives in bittorrent systems. In *Proceedings of the 2007 ACM SIGMETRICS international conference on Measurement and modeling of computer systems* (2007), ACM, pp. 301–312.

[41] Legout, A., Urvoy-Keller, G., and Michiardi, P. Rarest first and choke algorithms are enough. In *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement* (2006), ACM, pp. 203–216.

[42] Li, Z., Goyal, A., Chen, Y., and Kuzmanovic, A. Measurement and diagnosis of address misconfigured p2p traffic. In *INFOCOM, 2010 Proceedings IEEE* (2010), IEEE, pp. 1–9.

[43] Mahadevan, P., Krioukov, D., Fall, K., and Vahdat, A. Systematic topology analysis and generation using degree correlations. In *Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications* (2006), ACM, pp. 135–146.

[44] Mahanti, A. Measurement and analysis of cyberlocker services. In *Proceedings of the 20th international conference companion on World wide web* (New York, NY, USA, 2011), WWW '11, ACM, pp. 373–378.

[45] Mandal, U., Lange, C., Gladisch, A., and Mukherjee, B. Should isps adopt hybrid cdn-p2p in ip-over-wdm networks: An energy-efficiency

perspective? In *Advanced Networks and Telecommuncations Systems (ANTS), 2012 IEEE International Conference on* (2012), pp. 6–11.

[46] Mitzenmacher, M. A brief history of generative models for power law and lognormal distributions. *Internet mathematics 1*, 2 (2004), 226–251.

[47] Moore, J., Chase, J., Ranganathan, P., and Sharma, R. Making scheduling" cool": temperature-aware workload placement in data centers. In *Proceedings of the annual conference on USENIX Annual Technical Conference* (2005), pp. 5–5.

[48] Nedevschi, S., Ratnasamy, S., and Padhye, J. Hot data centers vs. cool peers. In *Proceedings of the 2008 conference on Power aware computing and systems* (Berkeley, CA, USA, 2008), HotPower'08, USENIX Association, pp. 8–8.

[49] Nordberg, A. Rasterbar libtorrent. `http://www.rasterbar.com/products/libttorrent`.

[50] Norton, W. B. *The Internet Peering Playbook: Connecting to the Core of the Internet*, 2nd ed. DrPeering Press, December 2012.

[51] Octoshape. Cnn uses octoshape. Available on `http://octoshape.com/cnn-com-using-octoshapes-p2p-for-live-feed/`.

[52] Palasamudram, D. S., Sitaraman, R. K., Urgaonkar, B., and Urgaonkar, R. Using batteries to reduce the power costs of internet-scale distributed networks. In *Proceedings of the Third ACM Symposium on Cloud Computing* (New York, NY, USA, 2012), SoCC '12, ACM, pp. 11:1–11:14.

[53] Passarella, A. A survey on content-centric technologies for the current internet: Cdn and p2p solutions. *Computer Communications 35*, 1 (2012), 1 – 32.

[54] Pouwelse, J., Garbacki, P., Epema, D., and Sips, H. A measurement study of the bittorrent peer-to-peer file-sharing system. *Delft University of Technology Parallel and Distributed Systems Report Series, Tech. Rep. Technical Report PDS-2004-007* (2004).

[55] Project, A. Ai3 project description. Available on `http://www.ai3.net`.

[56] Qureshi, A., Weber, R., Balakrishnan, H., Guttag, J., and Maggs, B. Cutting the electric bill for internet-scale systems. *ACM SIGCOMM Computer Communication Review 39*, 4 (2009), 123–134.

[57] Rowstron, A. I. T., and Druschel, P. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In *Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms Heidelberg* (London, UK, UK, 2001), Middleware '01, Springer-Verlag, pp. 329–350.

[58] Setting, V. B. C. G. Available on `http://wiki.vuze.com/w/Good_settings`.

[59] Specification, B. Available on `https://wiki.theory.org/BitTorrentSpecification`.

[60] Stutzbach, D., and Rejaie, R. Understanding churn in peer-to-peer networks. In *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement* (2006), ACM, pp. 189–202.

[61] Stutzbach, D., Rejaie, R., Duffield, N., Sen, S., and Willinger, W. Sampling techniques for large, dynamic graphs. In *INFO-*

COM 2006. 25th IEEE International Conference on Computer Communications. Proceedings (2007), IEEE, pp. 1–6.

[62] Stutzbach, D., Rejaie, R., and Sen, S. Characterizing unstructured overlay topologies in modern p2p file-sharing systems. *Networking, IEEE/ACM Transactions on 16*, 2 (2008), 267–280.

[63] Sun, Y., Liu, F., Li, B., and Li, B. Peer-assisted online storage and distribution: modeling and server strategies. In *Proceedings of the 18th international workshop on Network and operating systems support for digital audio and video* (New York, NY, USA, 2009), NOSSDAV '09, ACM, pp. 13–18.

[64] Sun, Y., Liu, F., Li, B., Li, B., and Zhang, X. Fs2you: Peer-assisted semi-persistent online storage at a large scale. In *INFOCOM 2009, IEEE* (2009), pp. 873–881.

[65] Tang, J., Musolesi, M., Mascolo, C., and Latora, V. Temporal distance metrics for social network analysis. In *Proceedings of the 2nd ACM workshop on Online social networks* (New York, NY, USA, 2009), WOSN '09, ACM, pp. 31–36.

[66] Tewari, S., and Kleinrock, L. Analytical model for bittorrent-based live video streaming. In *Consumer Communications and Networking Conference, 2007. CCNC 2007. 4th IEEE* (jan. 2007), pp. 976 –980.

[67] Tian, Y., Wu, D., and Ng, K. Modeling, analysis and improvement for bittorrent-like file sharing networks. In *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings* (2007), IEEE, pp. 1–11.

[68] Urvoy-Keller, G., and Michiardi, P. Impact of inner parameters and overlay structure on the performance of bittorrent. In *INFOCOM*

2006. 25th IEEE International Conference on Computer Communications. Proceedings (2007), IEEE, pp. 1–6.

[69] uTorrent Bittorrent Client's Forum. Available on `http://forum.utorrent.com/viewtopic.php?id=47523`.

[70] uTorrent Bittorrent Client's Good Setting. Available on `http://forum.utorrent.com/viewtopic.php?id=58404`.

[71] Valancius, V., Laoutaris, N., Massoulié, L., Diot, C., and Rodriguez, P. Greening the internet with nano data centers. In *Proceedings of the 5th international conference on Emerging networking experiments and technologies* (2009), ACM, pp. 37–48.

[72] Vu, L., Gupta, I., Nahrstedt, K., and Liang, J. Understanding overlay characteristics of a large-scale peer-to-peer iptv system. *ACM Trans. Multimedia Comput. Commun. Appl. 6*, 4 (Nov. 2010), 31:1–31:24.

[73] Vuong, Q. Likelihood ratio tests for model selection and non-nested hypotheses. *Econometrica: Journal of the Econometric Society 57*, 2 (1989), 307–333.

[74] Wang, J., Huang, C., and Li, J. On isp-friendly rate allocation for peer-assisted vod. In *Proceedings of the 16th ACM international conference on Multimedia* (New York, NY, USA, 2008), MM '08, ACM, pp. 279–288.

[75] Watts, D., and Strogatz, S. Collective dynamics of small-world networks. *Nature 393*, 6684 (1998), 440–442.

[76] Wu, D., Dhungel, P., Hei, X., Zhang, C., and Ross, K. Understanding peer exchange in bittorrent systems. In *Peer-to-Peer Comput-*

ing (P2P), 2010 IEEE Tenth International Conference on (2010), IEEE, pp. 1–8.

[77] Xu, D., Kulkarni, S. S., Rosenberg, C., and Chai, H.-K. Analysis of a cdn–p2p hybrid architecture for cost-effective streaming media distribution. *Multimedia Systems 11*, 4 (2006), 383–399.

[78] Yin, H., Liu, X., Zhan, T., Sekar, V., Qiu, F., Lin, C., Zhang, H., and Li, B. Design and deployment of a hybrid cdn-p2p system for live video streaming: experiences with livesky. In *Proceedings of the 17th ACM international conference on Multimedia* (New York, NY, USA, 2009), MM '09, ACM, pp. 25–34.

[79] Yin, H., Liu, X., Zhan, T., Sekar, V., Qiu, F., Lin, C., Zhang, H., and Li, B. Livesky: Enhancing cdn with p2p. *ACM Trans. Multimedia Comput. Commun. Appl. 6*, 3 (Aug. 2010), 16:1–16:19.

[80] Zhang, C., Dhungel, P., Liu, Z., and Ross, K. Bittorrent darknets. In *INFOCOM, 2010 Proceedings IEEE* (2010), IEEE, pp. 1–9.

[81] Zink, M., Suh, K., Gu, Y., and Kurose, J. Characteristics of youtube network traffic at a campus network - measurements, models, and implications. *Comput. Netw. 53*, 4 (Mar. 2009), 501–514.