Performance Evaluation of BitTorrent-like Peer-to-Peer Systems in the Presence of Network Address Translation Devices

by

Yangyang Liu

B.Eng., University of Electronic Science and Technology of China, 2006

A Thesis Submitted in Partial Fulfillment of the

Requirements for the Degree of

Master of Science

in the Department of Computer Science

Performance Evaluation of BitTorrent-like Peer-to-Peer Systems in the Presence of Network Address Translation Devices

by

Yangyang Liu

B.Eng., University of Electronic Science and Technology of China, 2006

Supervisory Committee

Dr. Jianping Pan, Supervisor

(Department of Computer Science)

Dr. Kui Wu, Departmental Member

(Department of Computer Science)

Dr. Yvonne Coady, Departmental Member

(Department of Computer Science)

Supervisory Committee

Dr. Jianping Pan, Supervisor
(Department of Computer Science)

Dr. Kui Wu, Departmental Member
(Department of Computer Science)

Dr. Yvonne Coady, Departmental Member
(Department of Computer Science)

ABSTRACT

There is no doubt that BitTorrent nowadays is one of the most popular peer-to-peer (P2P) applications on the Internet, contributing to a significant portion of the total Internet traffic and being a basis for many other emerging services such as P2P Internet Protocol Television and Video on Demand. On the other hand, Network Address Translation (NAT) devices have become pervasive in almost all networking scenarios. Despite of the effort of NAT traversal, it is still very likely that applications, especially P2P ones, cannot receive incoming connection requests properly if they are behind NAT. Although this phenomenon has been widely observed in measurement work, so far there is no quantitative study in the literature examining the impact of NAT on P2P applications. In this work, we build analytical models to capture the performance of BitTorrent-like P2P systems in a steady state, in the presence of homogeneous and heterogeneous NAT peers. We also propose biased optimistic unchoke strategies, in order to improve the overall system performance and fairness

metrics considerably. The analytical models have been validated by simulation results, which also reveal some interesting facts about the coexistence of NAT and public peers in P2P systems.

Contents

List of Tables

List of Figures

ACKNOWLEDGEMENTS

I would like to thank all people who have helped, encouraged, and inspired me during my studies. I am particularly grateful to my supervisor Dr. Jianping Pan for his endless support, constructive guidance, and understanding throughout my program. With huge patience, Dr. Pan led me to the research community for a newcomer. It has always been an enjoyable and fruitful experience to work with him in the last two years.

I also want to express my gratitude to my thesis committee members, Dr. Kui Wu and Dr. Yvonne Coady, for their valuable advice on my thesis research.

Last but not least, I want to thank my parents and my wife for their enduring support. Without them, I would never go so far.

DEDICATION

To my beloved parents

and dear wife Echo

Chapter 1

Introduction

In recent years, the peer-to-peer (P2P) paradigm has emerged as a promising architecture for bulk data transfer. The typical P2P applications include file sharing and media streaming. By taking the bandwidth contribution of participating users, the P2P paradigm dramatically shifts the workload from servers to users. As a result, the P2P paradigm is intrinsically scalable and can sustain a large number of concurrent users.

As the most popular P2P file sharing protocol, it was reported that BitTorrent-like applications contributed 53% of all P2P traffic on the Internet in June 2004 [4]. With an out-of-band mechanism to distribute file metadata and one or a group of tracker servers to keep track of participating peers, peers exchange file pieces between themselves directly, which greatly reduces the workload associated with the single-point data distribution model in the client/server (C/S) paradigm. In addition, BitTorrent-like protocols have been a basis for other emerging services such as P2P Internet Protocol Television (IPTV) [5, 6, 7] and Video on Demand (VoD) [8].

In BitTorrent user communities, there are othen complaints about Network Address Translation (NAT) devices [23]. Being a crucial technique to temporarily alle-

viate the IPv4 address shortage problem, NAT enables multiple machines to access the Internet with one public IP address. On the other side, NAT limits the direction of connectivity, i.e., incoming connection requests will be dropped if no prearrangement has been made on the NAT devices. Even with manual configuration and all kinds of NAT traversal techniques, many NAT users still complain about lower P2P performance than their counterparts not behind NAT [24, 25, 26, 27]. Furthermore, there have been observations on a large percentage of NAT users in P2P IPTV and VoD systems and the negative impact on service capacity [5, 6, 7, 8].

NAT may affect BitTorrent-like P2P applications in at least two ways. First, P2P applications running behind NAT, if NAT traversal is not successful, cannot receive incoming connection requests from other peers. Therefore, NAT peers will have fewer neighbors when compared with public ones, which slows down their progress due to the optimistic unchoke strategy employed by BitTorrent. Second, NAT peers (often behind home routers through DSL or cable modem Internet access) have lower uplink capacities than public peers (often residing in campus or enterprise networks with high-speed, dedicated Internet access). Due to the tit-for-tat strategy in BitTorrent, NAT peers are more likely to be choked by public peers since they cannot upload as fast as public peers. As a result, public peers mainly serve each other themselves, generating clusters as identified in [16].

In this thesis, we build analytical models to quantify the performance of BitTorrent-like P2P systems in the presence of homogeneous and heterogeneous NAT peers. Following the connection and rate conservation principles, we derive the ratio between public and NAT neighbors for a given peer, as well as the ratio between them chosen for upload. Then we obtain the download time, one of the most important user-perceived performance metrics, for both public and NAT peers with homogeneous and heterogeneous uplink capacities, respectively. The analytical models are vali-

dated by extensive simulations, which also reveal some counter-intuitive but indeed reasonable facts about the coexistence of NAT and public peers in BitTorrent-like systems. Driven by the theoritical results, we then employ the BitTorrent simulator to examine the actual upload and download rate of public and NAT peers throughout the download process. After that, we study the fairness metrics of the system, including the share ratio and Jain's Fairness Index [9]. According to the analytical and simulation resutls, we further propose a biased optimistic unchoke strategy: by slightly degrading the performance of public peers with a controllable parameter, we can significantly improve the performance and fairness metrics of NAT peers, and therefore improve the overall system performance and fairness considerably.

1.1 Main Contributions

The contributions of our work are fourfold. We list them as follows:

1. The negative impact of NAT on P2P systems has been speculated for a while, however, there is no quantitative study on the actual scope and severity of such impact yet. Our work first quantifies such impact with tractable mathematical models, which are also applicable to other BitTorrent-like P2P systems;

2. BitTorrent employs the tit-for-tat strategy to encourage contributions from peers, creating a fair system for rational participants. In the absence of NAT peers, some research efforts have proven that the tit-for-tat strategy is enough to construct a near-optimal system with respect to fairness metrics. However, to the best of our knowledge, there has been no research work on studying the impact of NAT on fairness metrics particularly in a steady state. In this thesis, a group of simulations is carried out to examine the fairness metrics in a system with the existence of NAT peers;

3. Based on the insights extracted from analytical models and simulations, we propose new mechanisms that can remarkably improve the system performance and fairness metrics without modifying the existing protocol too much. The new mechanisms can be applied to any P2P systems that use BitTorrent as the underlying protocol;

4. We also extend a BitTorrent simulator with the capabilities of handling NAT peers and their interaction with other entities in a typical P2P system. This simulator can be used by other researchers investigating the impact of NAT on similar systems.

1.2 Thesis Organization

The remainder of this thesis is organized as follows. In Chapter 2, we give an overview on BitTorrent, with a focus on the tit-for-tat, optimistic unchoke, and rarest-first strategy and their effectiveness in the large-scale file swapping. We also briefly review NAT and NAT traversal techniques and explain why not all NAT devices can be traversed. In addition, we describe the existing research work on BitTorrent-like systems from the viewpoint of measurement, modeling, analysis, simulation, and experimentation. Chapter 3 presents our analytical models on the performance of BitTorrent systems in the presence of homogeneous and heterogeneous NAT peers. We evaluate our analytical models and explore fairness metrics through simulations in Chapter 4. We further discuss possible approaches to improving the overall system performance and fairness metrics in Chapter 5. Finally, we conclude the thesis and point out future directions in Chapter 6.

Chapter 2

Background and Related Work

The P2P paradigm and NAT devices have gained great success since their emergence, thanks to their effectiveness in solving problems in their own domain. In this chapter, we first give an overview of a general P2P paradigm and explain why it outperforms the traditional client/server (C/S) model especially for high-bandwidth applications. We then describe the BitTorrent protocol with a focus on the tit-for-tat, optimistic unchoke, and rarest-first strategy. After that we introduce how NAT devices operate, which NAT traversal techniques are widely used, and most importantly why NAT devices cause problems for P2P applications particularly for BitTorrent-like systems. There has been a variety of research work on measurement, modeling, analysis, simulation, and experimentation of BitTorrent-like systems. As a result, we conclude this chapter by discussing the related work.

2.1 Peer-to-peer Applications

In recent years, many high bandwidth applications, such as file sharing and media streaming, have emerged and become very popular on the Internet. With the traditional C/S model, servers can easily become a bottleneck of such applications

especially when there is a large number of users attempting to download a popular file or watch a TV program at the same time. A basic architecture of a C/S model is illustrated in Figure 2.1. All users download the file or stream the media content only from the server. If we assume the uplink capacity of the server is C and the user population is N. Theoretically, the per-user throughput T is given by $T = C/N$. When N is large, T is very small. As a result, a C/S model usually has poor scalability and robustness.

On the contrary, P2P systems offer a promising approach to alleviating the cost at the server side and thus provide high scalability and robustness. Clients in P2P applications connect not only with servers but also with other clients, which forms an application layer overlay. P2P designs exploit the idea that once a client has received a data block, it also serves as an additional server by redistributing that data block. This means that the original servers are no longer the only suppliers in a system. All participating clients act as suppliers as well. Figure 2.2 describes a typical scenario of P2P systems. The most significant difference between Fig. 2.1 and Fig. 2.2 is that in a P2P model, in order to support data exchange among clients, clients usually have direct connections between themselves. If we assume the uplink capacity of the server and each client is C and c, respectively. With N users, the optimal per-user throughput T is given by

$$T = (C + N * c)/N. \tag{2.1}$$

When N is large, T can be approximated as c. Therefore, because of the contributed capacity from clients, a P2P paradigm avoids the bottleneck associated with the single-point data distribution model in a C/S paradigm.

The P2P paradigm has identified its value in several kinds of popular applications with its first successful deployment in file sharing domain. Well-known file sharing systems include Napster [1], Gnutella [2], and BitTorrent [3]. Napster relies on a

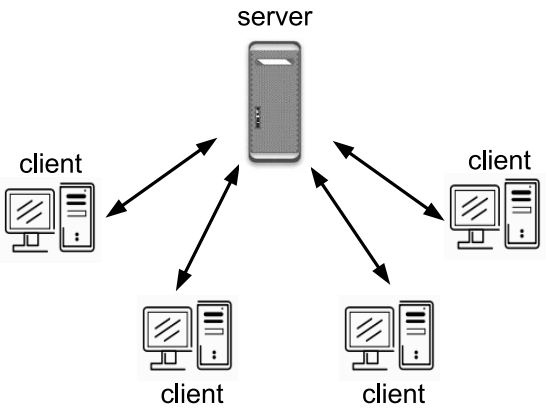

Figure 2.1: A typical client/server application model.

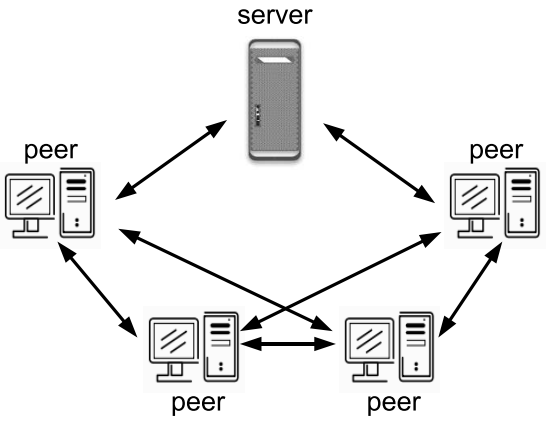

Figure 2.2: A typical peer-to-peer application model.

central server to help one peer find other peers who own the file of its interest. A peer in Gnutella network floods file queries to all its neighbours who may further flood this query to their neighbours until the query is satisfied or the specified query scope is reached. Using a central server based approach, Napster may experience the bottleneck problem at the server when a large amount of peers want to download files simultaneously. Flooding based approach in Gnutella makes it notoriously poor in scalability [29]. Supported theoretically by the epidemic algorithm [30], BitTorrent has achieved great success since its first emergence. A file in BitTorrent is chopped into serveral pieces and peers exchange file pieces among themselves. To facilitate the download process, a group of strategies has been used in BitTorrent, including the tit-for-tat, optimistic unchoke, and rarest-first strategy.

Another "killer" application, Voice over IP (VoIP), also achieves great success by using the P2P paradigm. Skype [31] is a good example. In Skype, peers form an overlay network and messages among peers are routed through this overlay network. In addition, P2P paradigms have been successfully applied to Internet Protocol Television (IPTV) and Video-on-Demand (VoD) systems which are predicted as the next killer Internet applications. CoolStreaming [5] is regarded as the first successfully deployed P2P IPTV system. Since its first release in March 2004, the number of concurrent users has reached 80,000 at peak with an average bit rate of 400 Kbps [6]. PPLive [32] is another popular P2P IPTV system originated in China. Due to the real-time feature of IPTV, peers usually contribute a small-sized buffer and share the content in this buffer among their neighbours. In addition to IPTV services, PPLive also provides VoD service, which is also constructed on the P2P paradigm [8]. Unlike P2P IPTV, P2P VoD systems usually require their clients contribute a fixed amount of hard disk storage (e.g., 1 GB in PPLive) in order to support the VCR functionality. This contributed storage makes the entire system form a distributed P2P storage

system. Combined with the unique features of VoD service, P2P VoD systems are a hot research topic in recent years and the interested readers are recommended to refer to the discussion in [8].

In our work, we focus our attention on P2P file sharing applications, particularly BitTorrent-like P2P systems due to their popularity in both industry and academia. We describe the BitTorrent protocol in the next section.

2.2 BitTorrent Overview

In BitTorrent, in order to expedite the file distribution and make good use of uplink capacities of all participating peers, a file is split into many equal-sized pieces (from 32 to 512 KB each). Peers download pieces from different neighbors simultaneously and also upload the downloaded pieces to their neighbors in return. According to the completeness of the file they are having, there are two kinds of peers: *seeds* are the peers that are having the complete file, and *leechers* are the peers that are still missing part or all of the file. In this thesis, we use *peers* to refer to both seeds and leechers.

In BitTorrent, *trackers* are the only centralized component in the system. They keep track of which peers are having which files either completely or partially. Peers sharing the same file form a *torrent*. Figure 2.3 describes a typical process for a peer to join a torrent. As we can see, a file download process starts by obtaining a .torrent file from a Web server. This .torrent file contains the meta-information of a file to be downloaded, such as the piece size, the SHA-1 hash value of each piece, and most importantly the IP address of trackers. To join a torrent, a new peer then contacts a tracker for a list of peers that are already in the torrent. The tracker returns a random list containing the information of K peers. Usually K is around 50. With this random

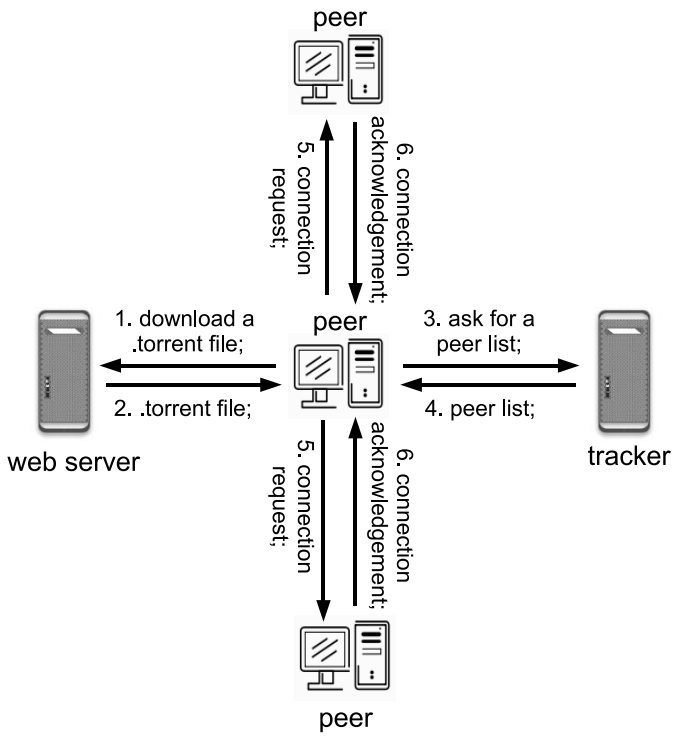

Figure 2.3: The sequence of events when a peer joins a torrent.

list, the new peer begins to build an initial peer set. Specifically, it tries to initiate connections with all peers on the list and connection attempts to peers behind NAT or firewall will fail. Once a connection is established, peers exchange the so-called file bitmap, which shows the information about which pieces they have downloaded. The bitmap helps each other better arrange from which peers to download which pieces. During the download process, once a peer has completely obtained a file piece, it will send a message to notify its neighbors about the availability of that file piece. In the BitTorrent terminology, uploading is *unchoke* and the refusal of it is *choke*.

To coordinate the download process of hundreds or even thousands of peers, BitTorrent employs a group of strategies. They include the tit-for-tat, optimistic unchoke, and rarest-first strategy.

2.2.1 Choking Algorithms

Every peer in BitTorrent can upload to M neighbors ($M = 5$ by default) at the same time. Usually, a peer might receive more than M download requests from its neighbors and hence it has to determine which neighbors it unchokes. There are two stategies employed by peers to make choking decision. One is the *tit-for-tat* stategy and the other is the *optimistic unchoke* strategy.

Tit-for-tat

The tit-for-tat strategy is regarded as a main incentive to reward rational uploaders with a better download performace and delay free-riders [33] which download files without contributing anything as reciprocation. In order to encourage contribution by uploading, once every 10 seconds by default, each leecher calculates the current download rates from its neighbors and ranks peers according to these rates. Then the peer unchokes the top $(M-1)$ of these neighbhors, leaving the remaining one picked using optimistic unchoke (introduced later). It is a non-trival task to obtain a strict instant download rate, and in BitTorrent this download rate is computed based on the last 20 seconds, i.e., the average download rate in the previous 20-second period. The way to calculate the download rate intrinsically means that the tit-for-tat strategy considers not only the absolute uplink capacities of neighbors but also their consistent upload capability [15]. It is apparent that the tit-for-tat strategy can be carried out without a centralized control in an ideal case.

Seeds do not download from others and thus the tit-for-tat strategy is not applicable for them. As the source of a file in a system, seeds usually want to distribute the first complete copy of a file in the fastest way, after which they can leave the system safely if they have to. To accelerate the distribution, each seed thus unchokes $(M-1)$ leechers that have the highest download rates from it. The other one is picked using

optimistic unchoke as well.

Optimisic Unchoke

Newly joining leechers have nothing to contribute and hence their download process will be delayed or even blocked if tit-for-tat is the only unchoke strategy. In BitTorrent, in order to bootstrap brand new leechers into the tit-for-tat process, both older leechers and seeds utilize the optimistic unchoke strategy: once every 30 seconds by default, each peer randomly picks one leecher from its neighbors to unchoke. In some BitTorrent clients, to further help new leechers, peers optimistically unchoke new leechers with a higher probability than those who have already downloaded some pieces. In addition to helping new peers, the optimistic unchoke strategy together with tit-for-tat provides peers with an ability to discover neighbors with higher upload rates (for leechers) or higher download rates (for seeds).

2.2.2 Piece Selection

In BitTorrent a file is chopped into many pieces for efficent sharing among peers. As a result, when being unchoked, a leecher usually has to decide which piece to download from that neighbor. To maximize the utilization of leechers' uplink capacities, BitTorrent specifies that a leecher first downloads the piece that is currently least owned by its neighbors. The purpose of this local *rarest-first* strategy is to reduce the overlap of pieces owned by peers, which accordingly increases the probability of a leecher always having some pieces that are of interest to its neighbors. This increased diversity of pieces gives peers a constant ability to contribute, which correspondingly makes peers have more chances to obtain pieces from its neighbors due to the tit-for-tat strategy. The theoretical analysis in [10] proved the rarest-first strategy leads to the high effectiveness of file sharing in BitTorrent. The measurement work in [11]

confirmed such efficacy in real systems.

2.3 Network Address Translation

Network Address Translation (NAT) was initially proposed as a temporary solution to alleviate the IPv4 address exhaustion problem, and now becomes a standard, indispensable feature in almost all networking scenarios, even in some cases where there are enough IP addresses [23]. By using private IP addresses, an entire network only needs very few public IP addresses for external communication, which is advantageous if the assigned public IP addresses are not sufficient or when the IP address allocation is changed (e.g., renumbering). The basic principle behind this is that NAT devices (usually NAT-enabled routers) change the source private IP address and port number to public ones (usually the public IP address of routers) for outgoing connection requests and packets. This re-writing process thus creates a mapping between the pair of (private IP address : port1) and (external IP address : port2). This mapping allows the return packets to reach the original initiator, and in this case, NAT is "transparent" to both endpoints. Depending on how a mapping is created by outgoing packets and applied to incoming packets for filtering, there are different types of NAT devices: full cone NAT, IP-restricted cone NAT, port-restricted cone NAT, and symmetric NAT [22].

Figure 2.4 describes a typical operation of NAT-enabled routers. Consider that machine A at private address (10.0.0.1) wants to send a packet through port 4000 to another machine (128.119.40.188) at port 8000. When receiving this outgoing packet, the router replaces the source IP address and port number with its public IP address (138.76.29.8) and an arbitrarily generated port number 3500, respectively. This creates a mapping entry between (10.0.0.1 : 4000) and (138.76.29.8 : 3500) in

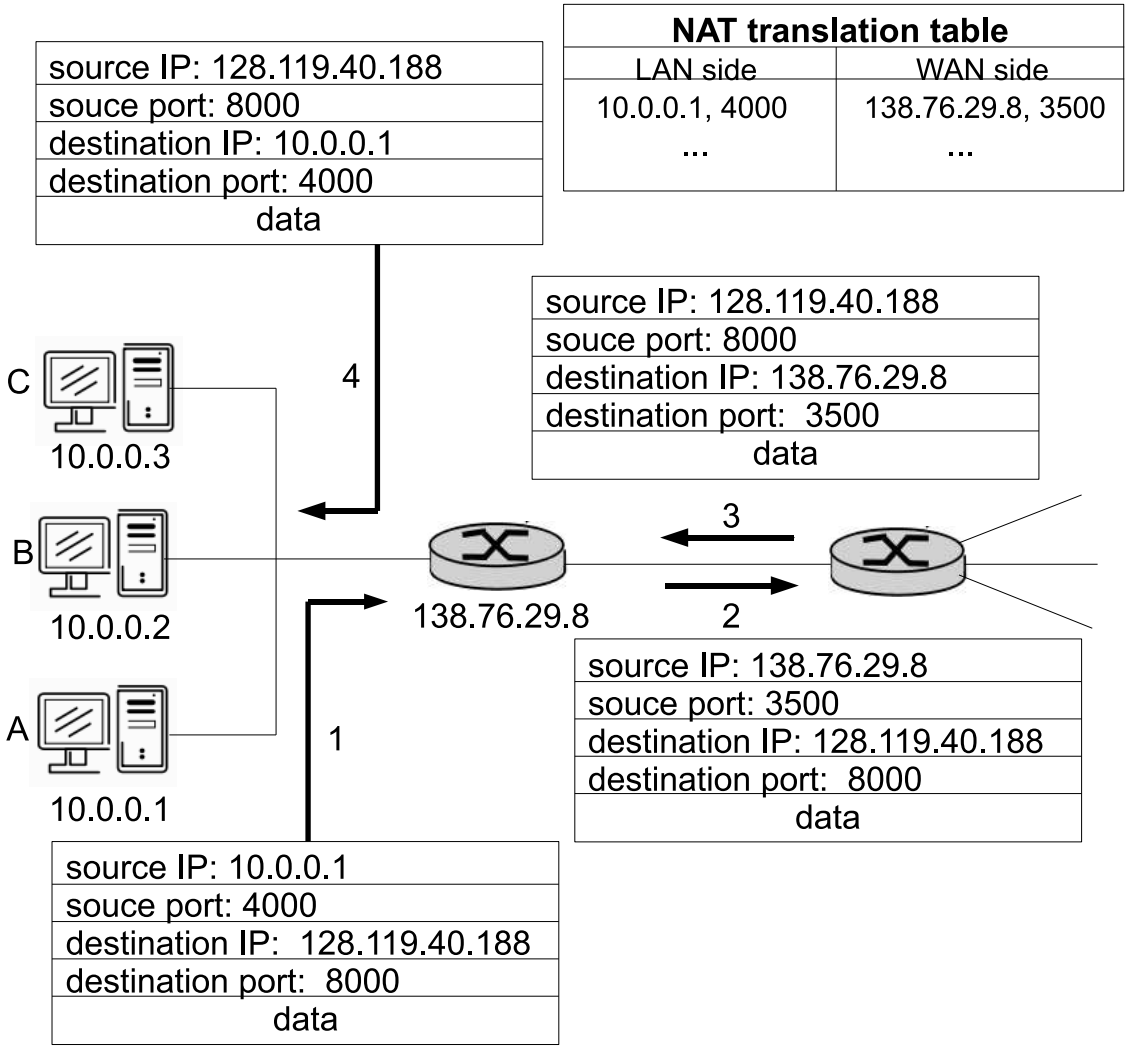

Figure 2.4: An illustration of Network Address Translation.

the router's NAT translation table. As a result, any incoming packet with destination IP address 138.76.29.8 and port number 3500 will be translated back and forwarded to port 4000 at machine A (10.0.0.1).

The existence of NAT devices increases the connectivity for end hosts with private IP addresses, but they also break the originally envisioned end-to-end IP connectivity model across the Internet [34], i.e., a connection has to be initiated by an endpoint behind NAT, and the one initiated by an external machine will be blocked by NAT, since there is no suitable mapping created by outgoing packets yet. This is a serious problem especially when both endpoints are behind different NAT, because no matter which endpoint initiates the connection, it will be blocked by the other NAT. Therefore, some prearrangement has to be made in order for incoming connection requests to traverse NAT devices.

There have been many techniques proposed for NAT traversal, such as Universal Plug-and-Play (UPnP), Simple Traversal of UDP through NAT (STUN), Traversal using Relay NAT (TURN) and Interactive Connectivity Establishment (ICE) [24, 25, 26], in addition to manual configuration and application-layer gateway running on NAT. UPnP allows endpoints to punch "holes" on NAT for incoming requests and has difficulty with cascaded NAT or NAT in another domain. STUN relies on STUN servers to discover the existence and type of the NAT closest to the server, and depending on the type of NAT involved, traverses the NAT when possible. If not, TURN can be used to relay the connectivity between endpoints, i.e., the direct connectivity between endpoints is not achieved. ICE is an integrated framework of these approaches and attempts to traverse NAT from the most optimistic to pessimistic way.

Even with these NAT traversal techniques, letting incoming connection requests traverse NAT and reach endpoints behind NAT is still a nontrivial job, especially for

symmetric NAT that allocate different port numbers for different internal and external endpoint pairs [27]. Some guesswork on the port allocation behavior is possible, but it is very unreliable due to the fact that NAT had not been standardized before it dominated the marketplace, and different vendors tend to choose different port allocation strategies.

2.4 NAT and P2P Applications

NAT is a particularly hard problem for P2P applications, including P2P file sharing, P2P VoIP, and P2P media streaming. Recall from Section 2.1 that P2P applications rely on direct connections between peers, such as that in BitTorrent to exchange piece availability information and pieces themselves. The Internet measurement work has shown many BitTorrent users, particularly residential ones, are behind NAT [28]. Usually these peers have to undergo inferior download performance when compared with the peers in public domain. In order to get around the connectivity problem due to NAT, Skype employs the so-called super nodes, which have globally routable IP addresses, to serve as relays between two peers befind different NAT [31]. As NAT becomes more pervasive on the Internet and symmetric in port allocation, NAT traversal becomes harder and in many cases impossible. Therefore in the rest of this thesis, we use NAT peers to refer to the peers not only behind NAT, but also not NAT traversalable.

Depending on the size of the mapping table and vendor-specific timeout value, peers behind NAT may be limited by how many connections they can initiate in a time window. Also since NAT peers cannot take incoming connection requests, it means that NAT peers have fewer neighbors. In BitTorrent, a peer relies on the optimistic unchoke strategy to pair with other peers with higher uplink capability,

which accordingly helps itself to optimize the benefit (download rate) in the tit-for-tat process. However, fewer neighbors mean that NAT peers have a lower opportunity to be unchoked through the optimistic unchoke strategy, and also NAT peers have more difficulty to find the pieces in need and locate peers with better performance in its smaller neighbor set, i.e., the progress of NAT peers will be slowed down.

In addition, since NAT peers are usually residential users with lower uplink capacities, the situation of NAT peers thus becomes even worse. Even if a NAT peer is unchoked by a public peer, which in most cases has a higher uplink capacity, bacause the NAT peer cannot upload to the public peer as fast as other public peers, the NAT peer will be choked very soon by the public peer due to the tit-for-tat strategy. Therefore, public peers not only have more neighbors (including both public and NAT peers), but also are more likely to only serve other public peers most of the time, i.e., further slowing down the progress of NAT peers [16].

In BitTorrent user communities, there are many complaints about NAT devices and there are many websites instructing users on how to deal with NAT (even by manual configuration). However, due to the sophistication of manual tuning, the trend of more symmetric NAT, and the variety of P2P applications, most users still cannot get around the problem, and thus suffer much lower performance than their counterparts not behind NAT. This is reflected by the much prolonged download time for BitTorrent file sharing applications, and the undesirable start-up latency and inferior playback continuity for media streaming applications.

2.5 Related Work

BitTorrent has been a hot topic in recent years. People have intensively studied BitTorrent through measurement, modeling, analysis, simulation, and experimentation.

In this section, we give an overview of the existing research efforts with a focus on the work that is the most relevant to, supports, or inspires ours.

In [12], the analysis on real BitTorrent log files showed the effect of flash crowd for a new file, the number of active peers in a torrent, and the correlation between upload and download rates. [13] measured the content availability and integrity, the number of users, the flash crowd effect, and the download rate of users. They noticed that only a small portion of peers tend to stay in the system after they finish downloading and the peer arrival and departure processes are not Poisson, which had been assumed in modeling work. The flash crowd effect captured in [12] and [13] motivates us to believe that it is worthwhile to examine the flash crowd period at which our work thus targets. [14] demonstrated that the performance of peers in small torrents fluctuates widely and to improve the performance, inter-torrent collaboration was proposed.

To identify the factors that determine the download performance of peers, two BitTorrent clients were started side-by-side on a campus network in [15]. The results showed that with the similar system and network conditions, the download performance gap of peers is mainly contributed by the peer's frequency of changing its "active" peers (the peers that are uploading to it). Peers having a stable "active" peer set usually achieve better download performance than those who frequently change their "active" peer set. By tracking more than 40 nodes in instrumented torrents, [16] demonstrated the choke strategy used by BitTorrent leads to the formation of clusters of similar bandwidth, i.e., a peer usually uploads to and downloads from the peers that have similar bandwidth. This is an expected behavior due to the tit-for-tat strategy. As a result, our theoretical models are partially inspired by the clustering phenomenon. In [11], to evaluate the efficacy of the rarest-first and optimistic unchoke strategy, an instrumented client participated in a large number of torrents. Results showed that both strategies play significant roles in the system's performance. A

simulation-based approach was used in [17] to verify that peers in BitTorrent perform near-optimally regarding their uplink utilization and download time. A block-based tit-for-tat strategy was also proposed in the same paper to improve the fairness.

Several analytical models have been proposed as well. [10] developed a fluid model to study BitTorrent's scalability and peer's download time in the equilibrium state. Also in the same paper, the authors presented a simple probabilistic model to demonstrate the high efficiency of file sharing among peers, i.e., peers can always own useful pieces to contribute and accordingly high uplink utilization can be achieved. The tradeoff between the performance and the fairness metric was studied in [18] by using a simple mathematical model for heterogeneous bandwidth settings. It revealed that the number of peers selected by tit-for-tat strategy and optimistic unchoke strategy is very important in determining the performance and fairness. The clustering of peers in terms of uplink bandwidth was also mathematically verified in [18]. Inspired by the clustering of peers observed in [16], [19] used a rate balance model to study the download time of high bandwidth and low bandwidth peers in a flash crowd scenario. In addition, a token-based tit-for-tat strategy was proposed to improve the fairness for high bandwidth peers while maintaining that for the entire system at a reasonable level. The model in [20] is similar to that in [19] in the sense of assuming the clustering of peers. The main difference lies in that [20] targeted at the steady state and also considered the effect of free-riders. In [35], the authors derived the upper bound of the ratio of firewalled peers above which it is impossible to get a fair share ratio among all peers.

To the best of our knowledge, we are the first to study the impact of NAT peers on BitTorrent-like systems. In particular, we create mathematically tractable models to quantify such impact in the presence of homogeneous and heterogeneous NAT peers. Similar to [19] and [20], we assume the fair sharing of uplink capacity. However, [19]

and [20] only considered the scenario with heterogeneous uplink capacities and without NAT peers. In contrast, our models include NAT peers in both homogeneous and heterogeneous uplink capacity scenarios. [35] targeted at a scenario similar to ours in which peers are classified into firewalled peers (NAT peers in our case) and connectable peers (public peers in our case). The connectivity scenario is thus the same as ours. However, we conduct the work from different perspectives. In particular, [35] mainly explored the fairness embedded in an entire session, while our concentration is on the quality of service in the form of download time and the fairness metrics at a steady state in which all peers are busy with uploading and downloading. Therefore, the work in [35] and ours naturally complement to each other.

2.6 Summary

In this chapter, we described a general P2P paradigm, the BitTorrent protocol, NAT and NAT traveral techniques, and the related work on BitTorrent-like systems. In the next chapter, we are going to introduce our analytical models for the BitTorrent-like P2P systems in the presence of homogeneous and heterogeneous NAT peers.

Chapter 3

Analytical Models

In this chapter, we design models for analyzing the performance of BitTorrent-like systems in the presence of NAT peers. The system scenario is illustrated in Fig. 3.1. As mentioned in Chapter 2, in our scenario there are two kinds of peers: (i) NAT peers that can only have public neighbors; (ii) public peers that can have both public and NAT neighbors. We concentrate on the steady state performance, i.e., the number of active peers remains stable. We create models for the original BitTorrent protocol with the main focus on the average download time of public and NAT peers, respectively. We further consider two cases: (i) homogeneous scenarios in which NAT and public peers have the same uplink capacity; (ii) heterogeneous scenarios in which public peers have a higher uplink capacity than NAT peers, which is quite pervasive in real systems. The peers in each class have the same uplink capacity.

For easy understanding, our notations are quite similar to those in [19], and we summarize them in Table 3.1.

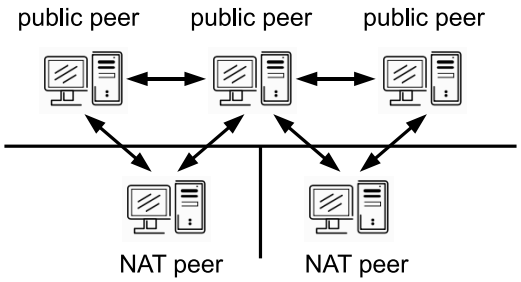

Figure 3.1: A generic P2P system with NAT peers.

Table 3.1: Notations and Definitions

Notation	Definition
X, Y	Type of peers: S (seed), P (public), N (NAT leechers)
F	Size of the file to be downloaded
K	Number of peers on the list returned by the tracker
K_X	Number of Xs on the list returned by the tracker
M	Maximal number of concurrent uploads by a peer
N	Number of peers in the system (seeds included)
N_X	Number of Xs in the system
α	Percentage of NAT peers out of all leechers
C_X	Uplink capacity of X
U_X	Upload data rate of X
U_{XY}	Upload data rate per connection from X to Y
D_X	Download data rate of X
n^u_{XY}	Number of Ys to whom X is uploading
n^d_{XY}	Number of Ys from whom X is downloading
L_X	Average number of neighbors of X
L_{XY}	Average number of Y neighbors of X
T_X	Average download time of X

3.1 Assumptions for Modeling

Due to the complexity of the BitTorrent protocol and the dynamics of the Internet, it is not an easy task, if not impossible, to capture all the features in a BitTorrent system using mathematical modeling. In our analysis, we pay attention to the main mechanisms employed in BitTorrent and thus make the following assumptions:

- There is no downlink bottleneck at each peer. The measurement results in [13] showed that 90% of peers in BitTorrent systems have download rate below 520 Kbps, which is slower than most broadband Internet access links. This assumption is also made in [10] and [18].

- The concurrent upload connections of a peer fairly share the uplink capacity of the peer. This is a common assumption usually made when people model BitTorrent systems, such as those in [19] and [20].

- There are no limits on the maximal and minimal number of neighbors a peer can sustain. These two constraints were neglected in [19] and [20] as well to avoid unnecessarily complicating the analytical models.

- The participating peers can fully utilize their uplink capacity, which has been proved theoretically in [10]. The simulation results in [17] also demonstrated the high uplink utilization of peers in BitTorrent systems.

- Leechers leave the torrent right after they have collected the entire file. Since we are more interested in exploring the influence of NAT peers on BitTorrent-like systems, this assumption is thus quite reasonable. Due to this assumption, we point out that our theoretical models derive the worst-case bound performance. In addition, the measurement work in [13] observed the similar peer

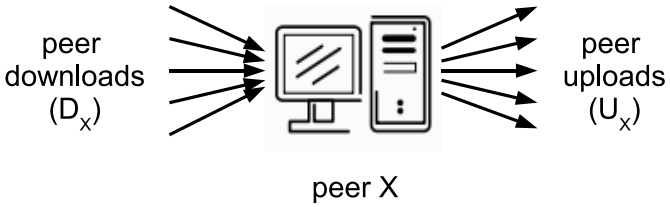

Figure 3.2: A basic analytical model.

behavior, i.e., peers do not tend to serve as seeds for a long time after finishing downloading.

- The number of seeds is much smaller than the number of leechers in the system. Our main focus on public and NAT peers makes this assumption quite acceptable and it is also made in [18] and [19].

With these assumptions, we plot Fig. 3.2 to illustrate the basic ideas about our models. In this thesis, the main issue we want to address is to capture the average file download time T_X of public and NAT peers in the steady state. In order to calculate T_X, we need to derive the average download rate D_X of peer X. It is obvious that D_X is directly determined by peer downloads n^d_{XY} (the average number of neighbors who are uploading to peer X) and the download rate per connection. We have assumed that the upload connections of a peer fairly share its upload capacity, therefore it is an easy task to get the average download rate of a connection. From the view of the whole system, peer downloads n^d_{XY} depends on peer uploads n^u_{XY} (the average number of neighbors to who peer X is uploading). While, peer X's number of peer uploads relies on the average number of neigbhors L_X, the average number of Y neighbors L_{XY}, the tit-for-tat and optimistic unchoke strategy. Therefore, we start by deriving the expressions for L_X and L_{XY}.

3.2 Peer Neighbors

When a new peer contacts a tracker to join a particular torrent, the tracker randomly selects K peers and returns their information to that peer. To alleviate the workload of a tracker, which is the sole central point in BitTorrent, a tracker usually does not utilize NAT travesal techniques to determine the accessibility of peers. As a result, the returned list may contain seeds, public leechers and NAT leechers. We assume that all peers join the system at the same time for the saturated system performance. Also, with the assumption that $N_S << (N_P + N_N)$, we have $K_P \approx (1-\alpha)K$ and $K_N \approx \alpha K$.

3.2.1 Seeds

Seeds are functioning as servers in a torrent and do not need to initiate any connections with other peers, i.e., all the connections between seeds and leechers are initiated by leechers. As a result, the total number of connection attempts sent from public leechers and NAT leechers to seeds is given by $\frac{N_S}{N}KN_P$ and $\frac{N_S}{N}KN_N$, respectively. All the connection attempts are uniformly distributed among seeds. The average number of public and NAT neighbors of a seed is thus given by:

$$L_{SP} = \frac{N_P}{N}K \approx (1-\alpha)K, \tag{3.1}$$

$$L_{SN} = \frac{N_N}{N}K \approx \alpha K, \tag{3.2}$$

i.e., the average number of neighbors for a seed is:

$$L_S = L_{SP} + L_{SN} = K. \tag{3.3}$$

3.2.2 Public Peers

Due to the accessibility of peers, the neighbor relationship between public and NAT peers can only be set up by NAT peers. Consequently, L_{PN} is calculated by:

$$L_{PN} = K_P N_N / N_P \approx (1-\alpha) K N_N / N_P = \alpha K. \tag{3.4}$$

Public peers are able to initiate connections with other public peers while attempts to initiate connections with NAT peers will fail. The total number of connection requests sent among all public peers is $N_P K_P$. However, the connection request sent from peer A to B and that sent from B to A only establish one connection between A and B. Therefore, the total number of connections Z among all public peers is given by:

$$Z = N_P K_P - \binom{N_P}{2} (\frac{K_P}{N_P - 1})^2 = N_P K_P - \frac{N_P K_P^2}{2(N_P - 1)}. \tag{3.5}$$

The second term in (3.5) is to count for the case when any two public peers happen to choose each other as a neighbor. Since a connection involves two peers, L_{PP} is obtained as:

$$L_{PP} = \frac{2Z}{N_P} \approx (2 - \frac{K}{N_P + N_N}) K_P. \tag{3.6}$$

In a large-scale BitTorrent system especially during a flash crowd period, it is not uncommon to have $K << (N_P + N_N)$. L_{PP} can thus be approximated as:

$$L_{PP} \approx 2K_P \approx 2(1-\alpha)K. \tag{3.7}$$

Then, the total number of neighbors for a public peer is:

$$L_P = L_{PP} + L_{PN} = (2 - \alpha)K. \tag{3.8}$$

3.2.3 NAT Peers

NAT peers can only have public peers as their neighbors and the neighboring relationship can only be established by NAT peers. Therefore we can obtain L_N as follows:

$$L_N = K_P \approx (1 - \alpha)K. \tag{3.9}$$

3.3 Peer Uploads

As we mentioned before, we assume the fair share of uplink capacity among concurrent connections and no bottlenecks due to downlink capacity, therefore the download time of a peer is associated with the average number of peers it is downloading from, i.e., n^d_{XY} ($X \in \{P, N\}$). From the view of the whole system, n^d_{XY} is directly related to n^u_{XY} ($Y \in \{P, N\}$). As a result, we first need to obtain n^u_{XY}, i.e., peer uploads.

In a flash crowd period, a peer usually has more than M ($M = 5$ by default) neighbors, and the rarest-first strategy contributes to high-efficient file sharing. Therefore, it is reasonable to assume that a peer can receive download requests from at least M neighbors in the steady state. NAT peers can only upload to public peers, therefore, we obtain n^u_{NP} as follows:

$$n^u_{NP} = M. \tag{3.10}$$

Each seed unchokes $(M - 1)$ leechers who have the highest download rates from it and optimistically unchokes another one. With the assumption of no downlink bottlenecks, seeds tend to randomly choose NAT and public leechers to upload. As a

consequence, n^u_{SN} and n^u_{SP} can be expressed as follows:

$$n^u_{SN} = \frac{L_{SN}}{L_S} M = \alpha M, \tag{3.11}$$

$$n^u_{SP} = \frac{L_{SP}}{L_S} M = (1 - \alpha) M. \tag{3.12}$$

We will obtain n^u_{PN} in the homogeneous and heterogeneous case, respectively. Once we have n^u_{PN}, n^u_{PP} is derived as:

$$n^u_{PP} = M - n^u_{PN}. \tag{3.13}$$

3.3.1 Homogeneous Uplink Capacity

The fair share of uplink capacity means that for a particular peer the download rates of concurrent connections are the same in the homogeneous case. The rarest-first strategy keeps a peer consistently uploading to unchoked neighbors. Therefore, in an ideal situation, a public peer randomly unchokes neighbors through tit-for-tat unchoke and optimistic unchoke. We assume in the steady state a public peer is uploading to G NAT peers and H public peers through tit-for-tat unchoke, respectively, and uploading to g NAT peers and h public peers through optimistic unchoke, respectively. Thus, we can express n^u_{PN} as:

$$n^u_{PN} = G + g, \tag{3.14}$$

where G and g can be derived from the following equations:

$$G + H = M - 1, \tag{3.15}$$

$$g + h = 1, \tag{3.16}$$

$$\frac{H+h}{M\alpha/(1-\alpha)} = \frac{H}{G}, \tag{3.17}$$

$$\frac{L_{PP}-H}{L_{PN}-G} = \frac{h}{g}. \tag{3.18}$$

Equations (3.15) and (3.16) are due to the number of neighbors a peer is uploading to through tit-for-tat unchoke and optimisitc unchoke, respectively. Equation (3.17) comes from the fact that in the homogeneous case, the number of neighbors unchoked by a public peer through tit-for-tat should be proportional to the number of neighbors unchoking that peer. Particularly, the numerator in the left side of Eq. (3.17) is the average number of public peers uploading to a public peer, and the denominator reflects the average number of NAT peers uploading to a public peer. Similarly, Eq. (3.18) means that the number of neighbors unchoked by a public peer through optimistic unchoke is proportional to the number of remaining neighbhors that are not unchoked by that public peer through tif-for-tat.

3.3.2 Heterogeneous Uplink Capacity

With the heterogeneous uplink capacity, tit-for-tat would match high-capacity peers with mostly high-capacity peers and low-capacity peers with mostly low-capacity peers, making peers having similar uplink capacities form clusters. This clustering effect has been observed experimentally in [16] and proved theoretically in [18]. Applying this observation to our models, a public peer will serve $(M-1)$ other public peers through tit-for-tat unchoke and might serve a NAT peer only through optimistic unchoke. As a result, n^u_{PN} is calculated by:

$$n^u_{PN} = \frac{L_{PN}}{L_P - M + 1}. \tag{3.19}$$

3.4 Download Time

In the previous section, we have derived n^u_{XY}. As a consequence, we can obtain n^d_{XY} as follows:

$$n^d_{PP} = n^u_{PP} = M - n^u_{PN}. \tag{3.20}$$

$$n^d_{PN} = \alpha n^u_{NP}/(1-\alpha). \tag{3.21}$$

$$n^d_{PS} = n^u_{SP} N_S/N_P. \tag{3.22}$$

$$n^d_{NP} = (1-\alpha) n^u_{PN}/\alpha. \tag{3.23}$$

$$n^d_{NS} = n^u_{SN} N_S/N_N. \tag{3.24}$$

Since peers can fully utilize their uplink capacity and usually they have more than M neighbors, we can easily get the average upload rate of each connection, i.e., U_{XY} as follows:

$$U_{SP} = U_{SN} = \frac{U_S}{M},\ U_{PP} = U_{PN} = \frac{U_P}{M},\ U_{NP} = \frac{U_N}{M}. \tag{3.25}$$

As a result, we get the average download rate of public and NAT peers as follows:

$$D_P = n^d_{PP} U_{PP} + n^d_{PN} U_{NP} + n^d_{PS} U_{SP}, \tag{3.26}$$

$$D_N = n^d_{NP} U_{PN} + n^d_{NS} U_{SN}. \tag{3.27}$$

Public peers usually have much higher average download rate than NAT peers. Therefore, we derive T_P as follows:

$$T_P = F/D_P. \tag{3.28}$$

When all public peers finish downloading and leave, NAT peers have downloaded

only $F_d = T_P D_N$. After that, the system degenerates to follow a C/S model. Specifically, each seed serves M NAT peers at the same time until these M peers download the file completely, after which the uplink resource of the seed is contributed to another group of M NAT peers. For the first group of M NAT peers, they need to spend $T'_R = \frac{F-F_d}{U_S/M}$ on obtaining the remaining part from the seed. For the following second group of selected M NAT peers, they have already waited T'_R and will spend another T'_R on downloading the missing part. We assume N_N is an integer multiple of M, i.e., we can divide all NAT peers into $\frac{N_N}{M}$ groups. For the last group of M unlucky peers, they thus have to spend $(\frac{N_N}{M} - 1)T'_R$ and T'_R on waiting for the seed and downloading the file, respectively. Since we have N_S seeds, the average download time of each group is given by:

$$T_R = (N_N + M)(F - F_d)/(2N_S U_S). \tag{3.29}$$

Equation (3.29) indicates that in order to minimize the average time NAT peers spend on the C/S model, it is preferable that when the system operates in the C/S model, each seed uploads with full rate to only one NAT peer until it finishes. As a result, in our analysis NAT peers need to wait $T'_N = (N_N + 1)(F - F_d)/(2N_S U_S)$ on average to obtain the remaining part of the file from seeds, i.e., T_N is expressed as follows:

$$T_N = T_P + T'_N. \tag{3.30}$$

For all peers including public and NAT ones, the average download time is given by:

$$T = (1 - \alpha)T_P + \alpha T_N. \tag{3.31}$$

3.5 Summary

In this chapter, we built analytical models to capture the file download time of peers in BitTorrent-like systems in the presence of homogeneous and heterogeneous NAT peers. The merit of the proposed models, compared with some existing analytical models, is that they are the first ones to quantify the impact of NAT on BitTorrent-like systems particularly in a flash crowd period. In the next chapter, we are going to validate our models through simulations and study the fairness of the system.

Chapter 4

Performance Evaluation

In this chapter, we use an event-driven BitTorrent simulator created by [17] to validate the models proposed in Chapter 3. This simulator has been used in several papers such as [19] and [20]. It implements the main BitTorrent mechanisms that we have discussed in Chapter 2, including the tit-for-tat, optimistic unchoke, and rarest-first strategy. Similar to most existing work on P2P systems, this simulator treats the Internet as a reliable black box, considering only the transmission delay and assuming no packet error or loss will happen. Besides, the concurrent connections of a peer fairly share its uplink capacity, which is consistent with our model assumption. We modified this simulator to accommodate NAT peers. In particular, peers are divided into public peers and NAT peers. A public peer can initiate connections with other public peers but cannot do so with NAT peers. A NAT peer can initiate connections with public peers but cannot connect with other NAT peers. Once established, a connection is bidirectional for data transfer, which is the case for TCP connections.

In the following simulations, unless otherwise specified, we have 500 leechers join the torrent in 5 seconds to simulate a flash crowd scenario which has been observed in [12] and [13]. A new leecher is behind a NAT device and untraversable with

Table 4.1: Uplink Capacities of NAT and Public Peers

Homogeneous case		Heterogeneous case	
NAT peer	public peer	NAT peer	public peer
384 Kbps	384 Kbps	384 Kbps	1,024 Kbps

probability α. There is only 1 seed in the simulation serving as an original source of a 100 MB file that peers want to download, but the case can be extended to multiple seeds. The uplink capacity of the seed is $1,024$ Kbps. Leechers will leave the system once they collect all file pieces each of which is 64 KB. We note that the piece size is carefully chosen according to the experimental results in [21] and the mechanisms used in the simulator. To help the system move quickly from the ramp-up stage to the steady state, each new leecher comes into the torrent with 5% of the file. This method was also used in [19]. All the peers can upload to $M = 5$ leechers simultaneously and the tracker returns the information of $K = 50$ peers. We investigate the homogeneous case and heterogeneous case, respectively. The uplink capacities of NAT and public peers in both cases are summarized in Table 4.1, which is based on the charaterizations of end-host capacities in [36]. All peers have downlink capacity of $3,000$ Kbps, which is reasonable in real systems and large enough to avoid any downlink bottlenecks.

In the following sections, we first evaluate the models for peer neighbors, and then validate the models for the original BitTorrent with homogeneous and heterogeneous uplink capacities, respectively. In the end, we study the fairness of such systems.

4.1 Peer Neighbors

Figure 4.1 illustrates the average number of neighbors for the seed and leechers obtained from the simulation and by our model. We notice that our model can accurately

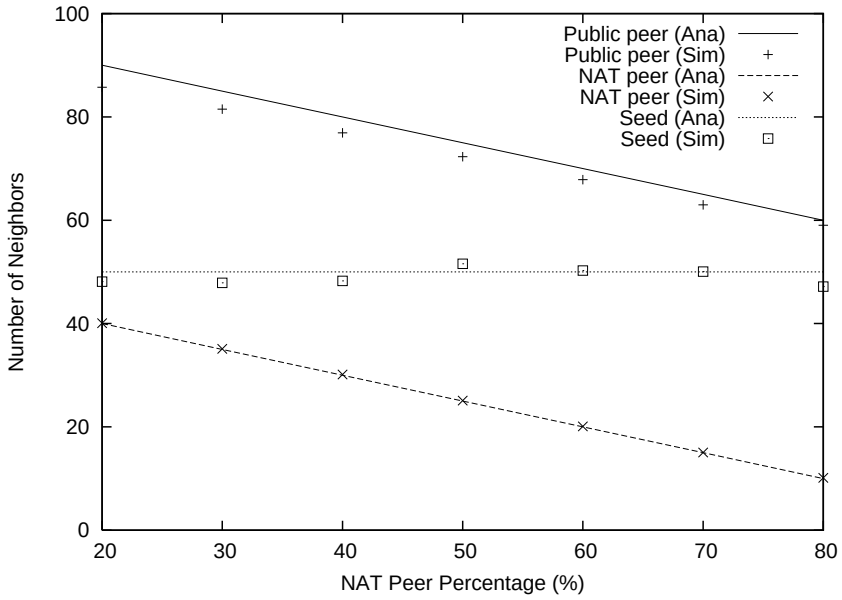

Figure 4.1: Average number of neighbors for a peer.

predict the number of neighbors for public and NAT leechers and the seed. Besides, Fig. 4.1 clearly shows that a public leecher has many more neighbors than a NAT leecher, which results in a huge difference in the quality of service experienced by public and NAT leechers. We will explore this further in the following sections. Our model can also accurately calculate L_{PN}, L_{PP}, L_{SN}, and L_{SP} as illusrated in Fig. 4.2 and Fig. 4.3. As a result, no matter working as a seed or a leecher, a BitTorrent client can employ our model to flexibly adjust their connection limits especially during flash crowd periods.

4.2 Download Time

This section applies our models to calculating the average download time of peers with different NAT peer percentage (α) and compares it with simulation results. We present the homogeneous case and heterogeneous case, respectively.

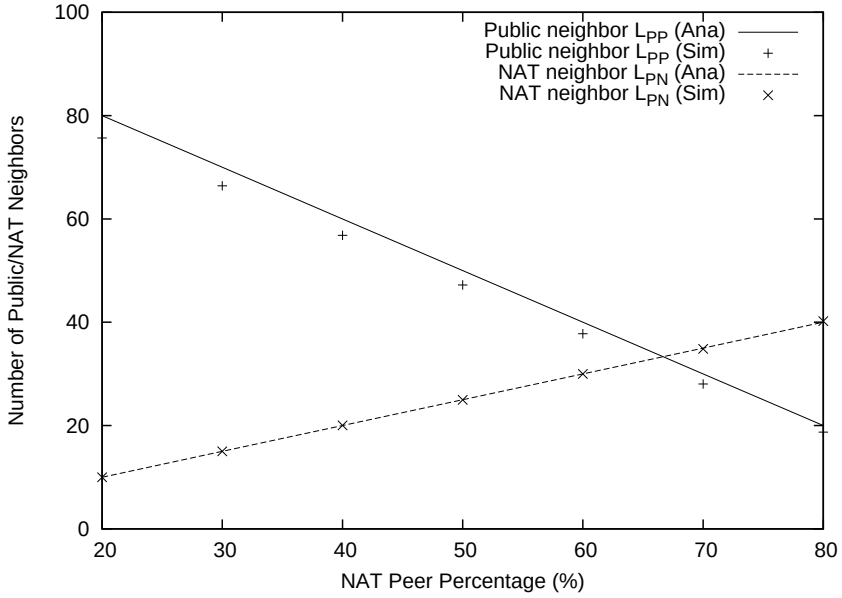

Figure 4.2: Average number of public/NAT neighbors for a public leecher.

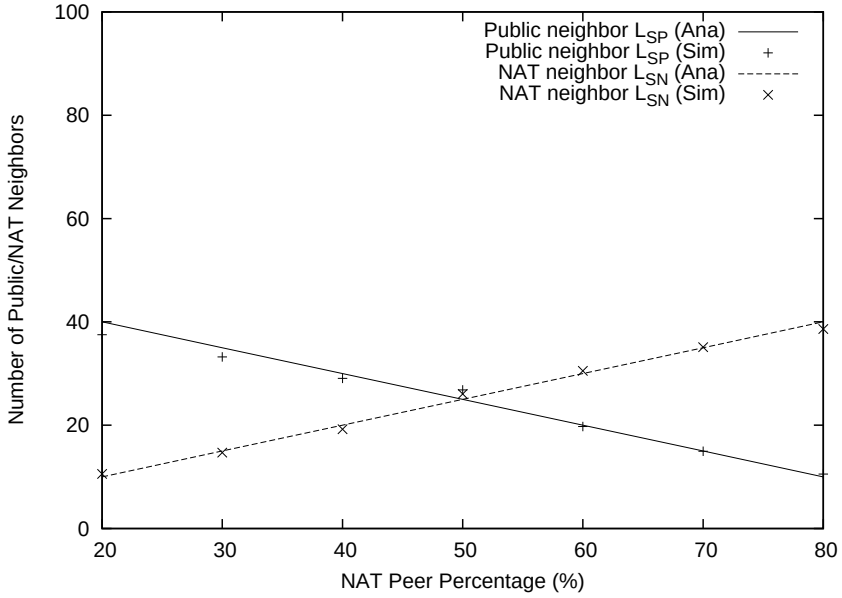

Figure 4.3: Average number of public/NAT neighbors for a seed.

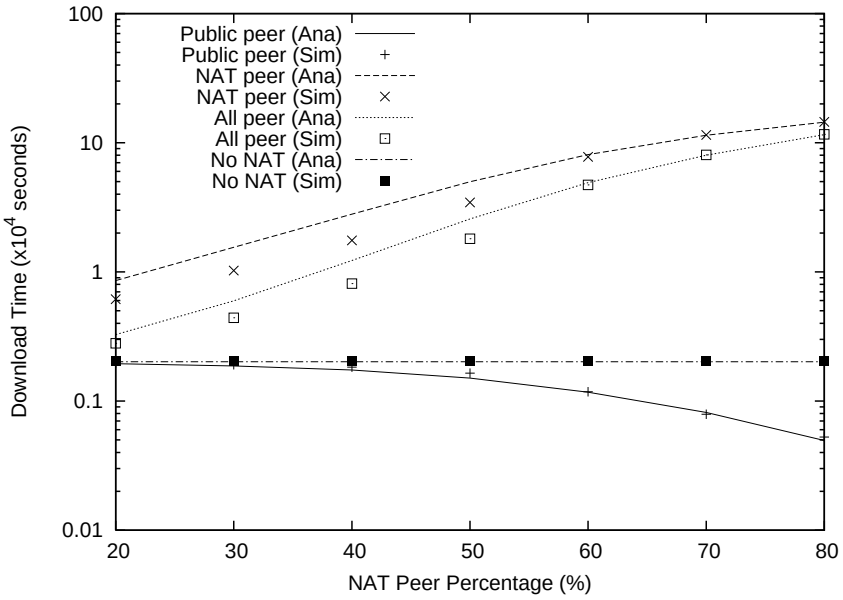

Figure 4.4: Average download time of peers in homogeneous scenarios.

4.2.1 Homogeneous Uplink Capacity

Figure 4.4 demonstrates the performance of BitTorrent systems with peers having the same uplink capacity. We note that the Y-axis is in log scale in Fig. 4.4 and also in the following figures in which Y-axis represents the download time. To better illustrate the impact of NAT peers and provide a fair comparison, we add one more case, which is a "No NAT" system. Our model can be easily tuned to obtain the average download time of peers in this case. Particularly, this case corresponds to $\alpha = 0$, where the seed only uploads to public leechers and also public leechers upload between themselves.

Figure 4.4 clearly shows that the average performance of all peers will deteriorate with the increase of the NAT peer percentage. Also NAT peers will have to spend much more time in downloading the file when the majority of peers are behind NAT. For example, when there are only 20% NAT peers, their download time is tripled when compared with public peers. When there are about 50% NAT peers, that gap is

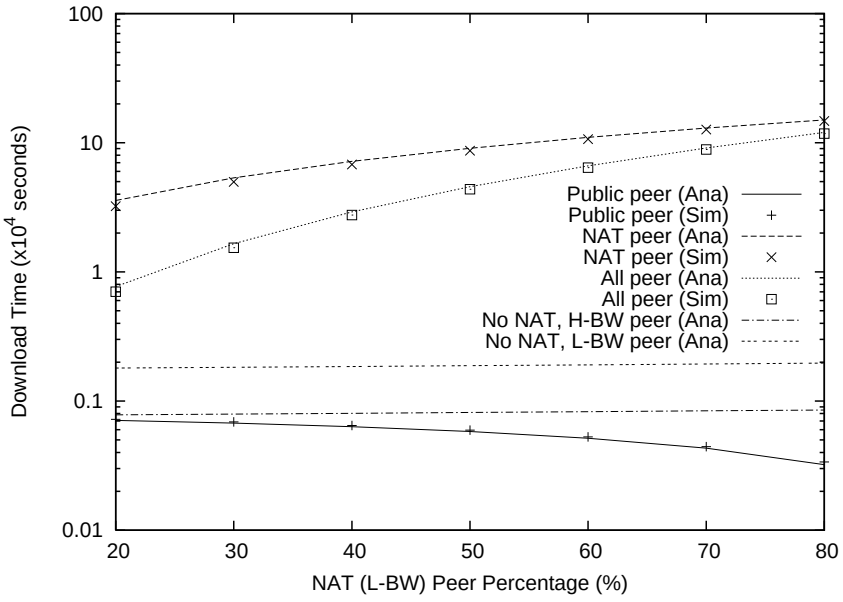

Figure 4.5: Average download time of peers in heterogeneous scenarios.

increased to 20 times. These observations are expected considering the inaccessibility of NAT peers from public or other NAT peers and the fewer neighbors of NAT peers.

Figure 4.4 also shows one counter-intuitive but indeed reasonable fact about the coexistence of NAT and public peers in P2P systems. We notice that when the system has more NAT peers, public peers on the contrary spend less time on downloading the file, even compared with the case in which all peers are publicly accessible. For example, where there are 50% NAT peers, the download time of public peers is 20% less than that in the "No NAT" case. The reason is as follows. NAT peers can only contribute their uplink capacities to public peers, while public peers have to share their uplink capacities between public and NAT peers. Therefore, when there are more NAT peers, the total uplink capacities contributed by NAT peers to the fewer public peers will be increased greatly per public peer and thus the download time of public peers is reduced.

4.2.2 Heterogeneous Uplink Capacity

We now explore the heterogeneous scenario which is more likely in reality. To better understand the impact of NAT peers, we also add one more case in Fig. 4.5. In this new case, all peers are public but are classified into high bandwidth (H-BW) peers and low bandwidth (L-BW) peers. The uplink and downlink capacities for H-BW peers and L-BW peers are the same as those of public and NAT peers in the heterogeneous case, respectively. The α value in this case accordingly corresponds to the percentage of L-BW peers in the system. We obtain the theoretical download time of H-BW peers and L-BW peers using the models proposed in [19].

Figure 4.5 demonstrates a similar trend when compared with Fig. 4.4. In particular, we notice that the overall system performance and the performance of NAT peers become even worse with the increased percentage of NAT peers. Public peers, on the other hand, will benefit a lot when more peers are behind NAT. The reason here is similar to the homogeneous case. We want to emphasize that in [19] and [20], no NAT peers have been considered. However, we find in Fig. 4.5 that if we take into account the pervasive existence of NAT peers, the overall system performance and also the performance of L-BW peers (usually NAT peers) will be much worse than the case when the existence of NAT peers is ignored. Further, H-BW peers (usually public peers) will achieve a better quality of service when NAT peers are involved.

According to Fig. 4.4 and Fig. 4.5, we conclude that the existence of NAT peers will deteriorate the overall system performance and also that of NAT peers significantly. However, since peers can fully utilize their uplink capacities and a NAT peer cannot contribute directly to other NAT peers, the existence of more NAT peers in fact can improve the performance of public peers in both homogeneous and heterogeneous scenarios. We also want to highlight that the existence of NAT peers will greatly decrease the performance of L-BW peers (usually NAT peers) and improve

the performance of H-BW peers (usually public ones).

4.3 Upload and Download Process

In the last section, we discussed a peer's download time, i.e., its entire lifetime in the system, in both the homogeneous and heterogeneous scenarios. We identified the download performance gap between public peers and NAT peers. This performance gap is caused by the inaccessibility of NAT peers. In this section, we employ our simulator to capture the average download and upload rate of public and NAT peers during the download process. When all the public peers have collected the whole file and left, the system degrades to a C/S model. Therefore, in this section, we only focus on the P2P portion of the download process. We explore the system in the presence of homogeneous and heterogeneous NAT peers, respectively, with $\alpha = 0.4$.

4.3.1 Homogeneous Uplink Capacity

Figure 4.6 depicts the average upload and download rate of public and NAT peers in homogeneous scenarios with $\alpha = 0.4$. We first notice that the simulation for the flash crowd scenario can quickly converge into the steady state from the ramp-up period. In our simulation, we have new leechers join the torrent with 5% of the file, which has accelerated this transition process. Besides, we see the download rate of peers in the steady state closely match with our analytical results obtained using (3.26) and (3.27). Figure 4.6 also shows the near-optimal utilization of uplink capacity of both public peers and NAT peers. However, we note that along the download process, public peers are always rewarded with much faster download rate than that of NAT peers, even though public and NAT peers have very similar upload rate. This is still due to the inaccessibility of NAT peers, which benefits public peers. This benefit is

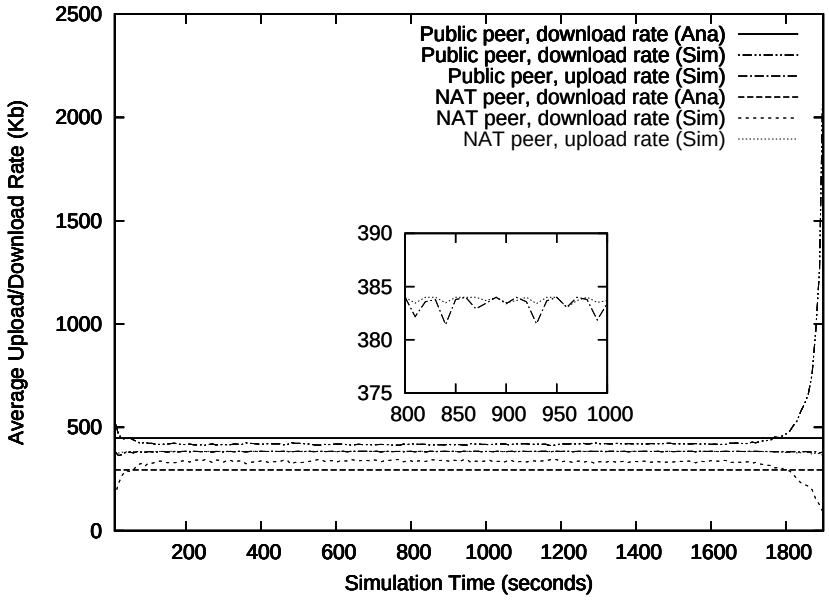

Figure 4.6: Average upload and download rate of peers in homogeneous scenarios ($\alpha = 0.4$).

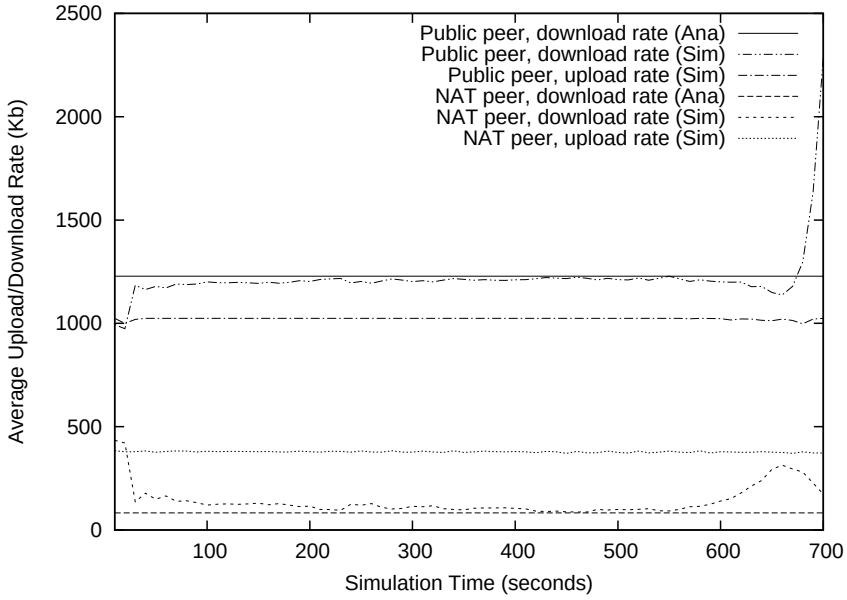

Figure 4.7: Average upload and download rate of peers in heterogeneous scenarios ($\alpha = 0.4$).

dramatically increased when there are fewer public peers in the system as presented in Fig. 4.6. Particularly, we observe that when most of the public peers finish and leave (after around 1800-th second), the remaining public peers obtain a significantly increased download rate. This is because all the possible uplink capacity of NAT peers is now shared by fewer and fewer public peers. On the contrary, NAT peers suffer from the departure of public peers and their download rate starts to decrease gradually.

4.3.2 Heterogeneous Uplink Capacity

Our analytical models can accurately predict the download rate of peers in the steady state in heterogeneous scenarios as shown in Fig. 4.7. The quick transition to the steady state and the near-optimal utilization of uplink capacities of peers are observed as well in heterogeneous scenarios. Different from the homogeneous scenario, the steady state in the heterogeneous case indicates that public peers form clusters to constantly serve each other, which is a result of the tit-for-tat strategy. In addition, public peers still benefit from the inaccessibility of NAT peers and thus as illustrated in Fig. 4.7 they are getting much higher download rate than their upload rate. On the other hand, NAT peers might be unchoked by public peers only through optimistic unchoke. Therefore, we notice that NAT peers download data at a rate much slower than that at which they contribute to the system. It is obvious that the altruism discussed in [37] appears in our scenario, i.e., NAT peers make regular contributions to the system, which do not help improve their download performance.

In addition, we note in Fig. 4.7 the up-and-down trend for the download rate of NAT peers when public peers are close to finishing. When public peers collect most of the file, it becomes hard for them to serve their public neighbors only due to the data availability among themselves. As a consequence, the clusters are partially

broken, which allows public peers to unchoke NAT peers though tit-for-tat as well. This increases the download rate of NAT peers. When a large number of public peers finish and leave (after around 650-th second), NAT peers have to experience a reduced download rate. On the contrary, with fewer public peers sharing the uplink resource of NAT peers, the download rate of public peers increases remarkably (after around 650-th second). There is no clustering effect in homogeneous scenarios, so we do not observe a similar up-and-down trend for the download rate of NAT peers in Fig. 4.6.

Both Fig. 4.6 and Fig. 4.7 have reflected unfairness in the system. In specific, we notice that due to the inaccessiblity of NAT peers, public peers usually download faster than they upload and NAT peers, on the contrary, contribute data faster than they download. This unfairness motivates us to study fairness issues embedded in such scenarios. We discuss them in the next section.

4.4 Fairness Metrics

A well designed P2P file sharing application should provide fair service to participating peers. Specifically, the amount of data contributed by a peer in an ideal case should be comparable to the amount of data it receives in the same time unit. In order to achieve the fairness, BitTorrent exploits the tit-for-tat strategy, encouraging each peer to contribute data as quickly as they receive. When designing BitTorrent, Cohen [3] did not take into consideration the existence of NAT peers, leaving the fairness issue an interesting topic in the scenarios with NAT. In this section, we apply our simulator to study two fairness metrics: 1) *share ratio*, which is a direct metric to measure a peer's contribution (upload) over its benefit (download); 2) *Jain's Fairness Index* [9], which indicates the fairness among competing and cooperating entities. The steady state spans a wide range of time as depicted in Fig. 4.6 and Fig. 4.7. As

a result, in this section, we focus on the steady state, i.e., a snapshot of the system.

The issue of fairness in BitTorrent has been studied intensively in the research community, such as [18] and [35]. However, [18] only explored Jain's Fairness Index in a scenario where all peers are accessible. Although similar to our scenario, [35] only focused on the share ratio embedded in a whole session, i.e., when all the peers complete their downloading. Instead, in our work we concentrate on the steady state in which all the peers are busy with downloading and uploading the data.

4.4.1 Share Ratio

Share ratio R_X, defined as $R_X = U_X/D_X$, is intuitively a good and direct indicator to measure the contribution of a particular peer with respect to the obtained service. From a peer's point of view, BitTorrent systems usually require U_X be comparable to D_X, i.e., the value of R_X is close to 1. In our scenario, R_X is based on the average upload and download rate of peers in the steady state. Similar to the previous section, to better understand the unique features of a system with NAT peers, we also plot the curves corresponding to a "No NAT" system. We summarize the results in Fig. 4.8 and Fig. 4.9, respectively.

Homogeneous Uplink Capacity

Figure 4.8 illustrates the share ratio of peers in homogeneous settings. Public peers and NAT peers have the same upload rate, but they behave differently. In specific, Fig. 4.8 shows that NAT peers usually contribute much faster than they obtain from the system, with a share ratio greater than 1; public peers, on the contrary, download faster than they contribute, with a share ratio below 1. This is due to the reasons we discussed in Section 4.2. We also notice that the tit-for-tat strategy does perform ideally in a system when all peers are accessible, allowing peers to download data as

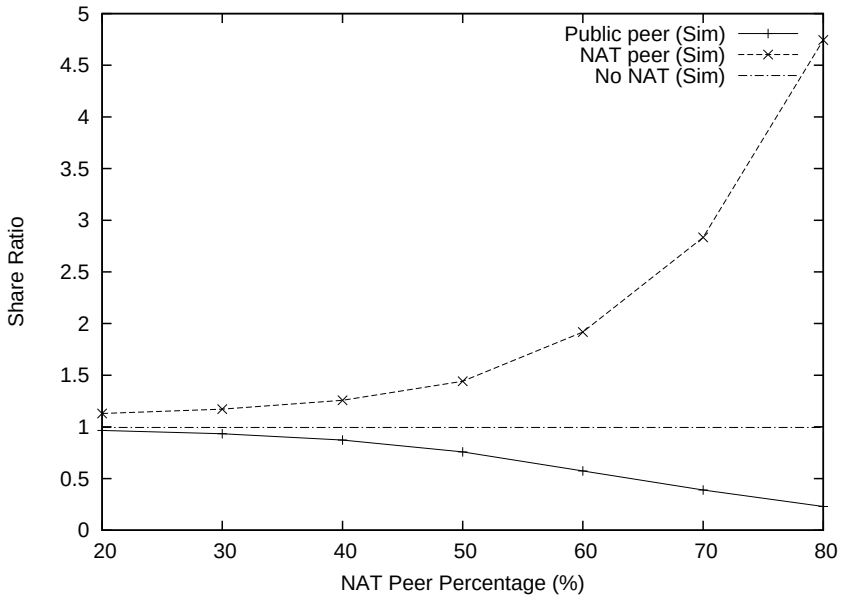

Figure 4.8: Average share ratio of peers in homogeneous scenarios.

fast as they send, with a share ratio close to 1.

Heterogeneous Uplink Capacity

Similar results are also shown in heterogeneous scenarios as depicted in Fig. 4.9. Taking into account the near-optimal utilization of uplink capacity of peers revealed in Fig. 4.7, we see from Fig. 4.9 that public peers always have download rate larger than 1,024 Kbps or even as high as twice of their maximal uplink capacity. In contrast, NAT peers contribute their uplink capacity completely; however, the download rate rewarded to NAT peers is far below their upload rate. In a system without NAT peers, the clustering effect leads to the observation that both H-BW peers and L-BW peers have share ratios close to 1. We notice that when there are more L-BW peers in the system, the share ratio of H-BW peers is slightly higher than 1. This is due to the increased probability that H-BW peers unchoke L-BW peers optimistically, i.e, the temporary breakage of formed clusters. This temporary breakage also gives rise

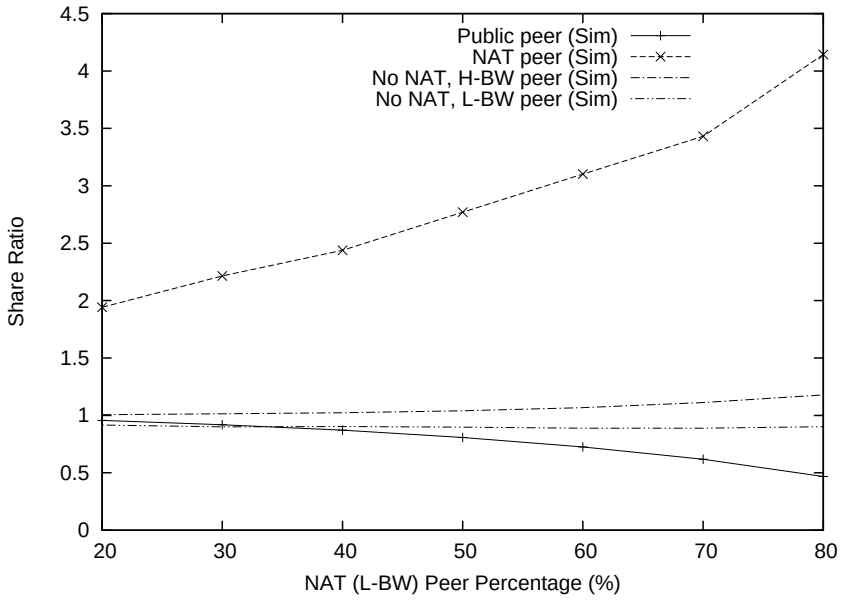

Figure 4.9: Average share ratio of peers in heterogeneous scenarios.

to the share ratio of L-BW peers slightly lower than 1.

From Fig. 4.8 and Fig. 4.9 we conclude that when there are more NAT peers in the system, because of the inaccessibility among them, NAT peers contribute much more than they download, while public peers benefit in the sense that their download rate is higher than their upload rate.

4.4.2 Jain's Fairness Index

Share ratio is applied to measure a peer's contribution over the obtained service. Jain's Fairness Index [9], on the other hand, quantifies the fairness among competing and cooperating entities, i.e., from the whole system's perspective. Jain's Fairness Index is expressed as follows:

$$F = \frac{(\sum_{i=1}^{n} x_i)^2}{n(\sum_{i=1}^{n} x_i^2)}. \tag{4.1}$$

In our case, x_i represents the share ratio of peer i in the steady state. Equation (4.1) indicates that when peers have the same share ratio, Jain's Fairness Index equals 1;

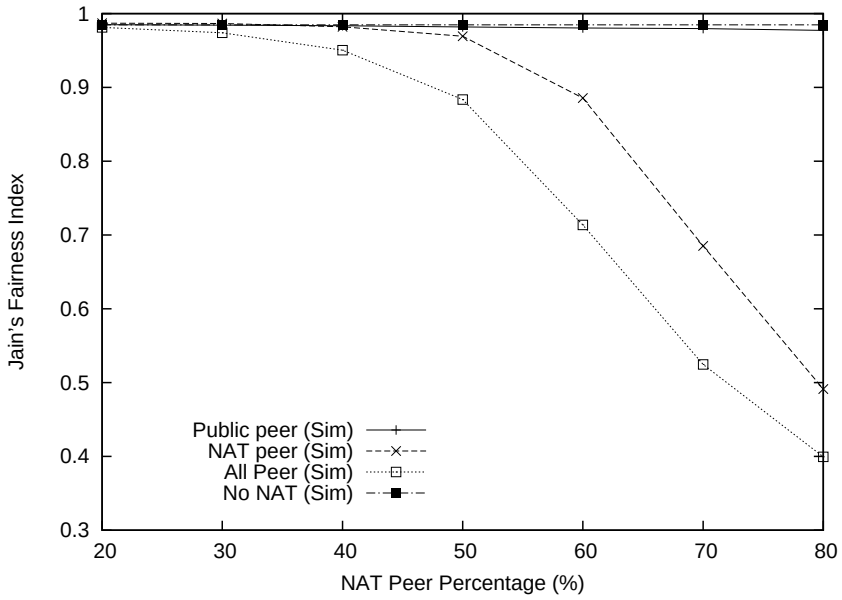

Figure 4.10: Jain's Fairness Index of peers in homogeneous scenarios.

otherwise, it is less than 1. We employ Jain's Fairness Index to explore the fairness among public peers, NAT peers, and the entire system in homegeneous and heterogeneous scenarios, respectively.

Homogeneous Uplink Capacity

Figure 4.10 demonstrates Jain's Fairness Index of peers in homogeneouse scenarios. Jain's Fairness Index reflects how equally a system is providing service to peers. We can see in Fig. 4.10 that no matter how many NAT peers are in the system, BitTorrent systems provide almost the same service to public peers. When there is a larger population of NAT peers (over 50%), however, the system is providing remarkably unfair service among NAT peers. Besides, the service unfairness between public peers and NAT peers becomes noticeable when more peers are behind NAT. Figure 4.10 also shows BitTorrent's incentive mechanisms function desirably in the "No NAT" system, i.e., Jain's Fairness Index is very close to 1.

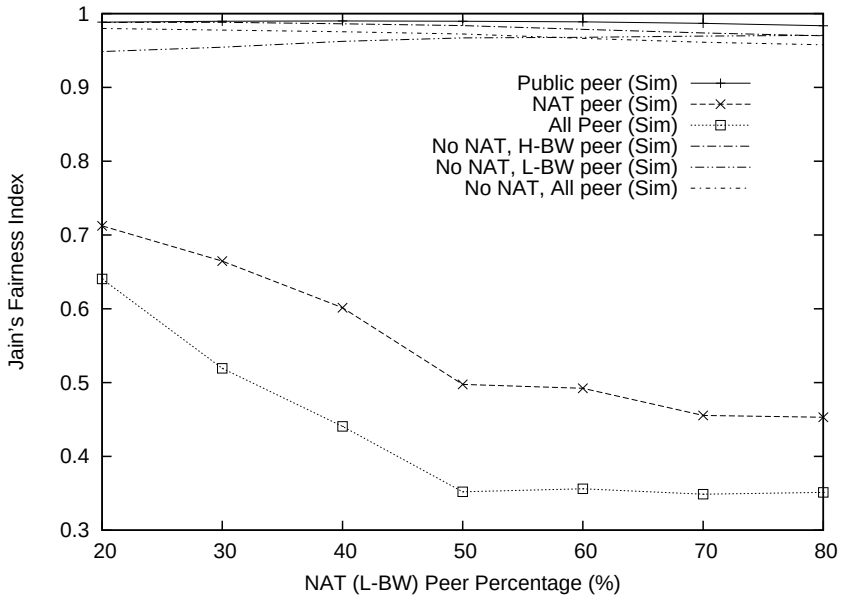

Figure 4.11: Jain's Fairness Index of peers in heterogeneous scenarios.

Heterogeneous Uplink Capacity

Figure 4.11 shows that in heterogeneous settings, BitTorrent systems also provide the same service to public peers, which is independent of the percentage of NAT peers. On the contrary, we notice that the share ratio of NAT peers is increasingly different when more NAT peers exist in the system. Additionally, we observe that the quality of service gap between public and NAT peers is becoming remarkably large when there are more NAT peers in the system. This unfairness is not noticeably shown in the "No NAT" system.

We conclude from Fig. 4.10 and Fig. 4.11 that the existence of NAT peers destroys the fairness embedded in the original BitTorrent protocol, which behaves almost ideally when all peers are accessible. Particularly, we see that NAT peers themselves have different share ratio and the share ratio gap between public peers and NAT peers becomes larger when there are more NAT peers.

4.5 Summary

In this chapter, we first validated the analytical models through simulations. We noticed that our models can accurately capture the average number of neighbors of a peer and the average download time of a peer. The models also revealed some interesting facts about the coexistence of NAT and public peers in P2P systems. In specific, we found that the overall system performance degrades rapidly when there is a large number of NAT peers in the system. On the contrary, public peers benefit a lot from the existence of NAT peers due to the fact that NAT peers cannot serve each other directly.

In addition, we also captured the average upload and download rate of public and NAT peers throughout the download process. We discovered that public peers usually download faster than they contribute, while NAT peers behave the other way around. This observed unfairness motivated us to quantify the fairness metrics of such a system by means of calculating the share ratio and Jain's Fairness Index. We found that public peers and NAT peers receive different level of quality of service with regard to their contribution. The unfairness is noticeably enlarged when there are more NAT peers in the system.

In the next chapter, we are going to propose a simple but effective strategy to improve the download performance of NAT peers and the fairness metrics without sacrificing the performance of public peers noticeably.

Chapter 5

Performance Improvement

Chapter 4 has identified and quantified that NAT peers have to spend much more time in downloading a file when compared with public peers. In the steady state, we also discovered that NAT peers contribute data at a rate much higher than that at which they obtain the data, while public peers perform the other way around. This unfairness is amplified when there is a large number of NAT peers. In addition, NAT peers receive remarkably different levels of service among themselves, resulting in their Jain's Fairness Index usually far below 1.

In this chapter, we explore possible approaches to improving the performance of NAT peers without obviously sacrificing that of public peers in both homogeneous and heterogeneous scenarios. [18] has mathematically proved that the tit-for-tat strategy plays a critical role in determining the tradeoff between the system performance and the fairness in BitTorrent-like systems. We thus do not want to change the tit-for-tat strategy for this purpose. On the contrary, we investigate whether we can adjust the optimistic unchoke strategy slightly to achieve our goal. In particular, we let public peers optimistically unchoke NAT peers with probability P_o. Intuitively, higher values of P_o mean public peers favor NAT peers more. We emphasize that it is not difficult

for public peers to detect whether the connected neighbors are behind a NAT device or not. For example, in terms of symmetric NAT devices, when a public peer receives a connection request, it sends messages via another port to the requesting neighbor. Whether or not the neighbor replies with acknowledgement can help the public peer infer the accessibility of that neighbor. The detailed techniques go beyond the scope of our work and the interested readers can refer to [24, 25, 26, 27] and the references therein.

We use the simulation-based approach to understanding the impact of P_o on the performance of peers in terms of download time and fairness metrics. All other simulation parameters are identical to those in Chapter 4. We refer to the regular BitTorrent system as R-BT, and the system favoring NAT peers as B-BT.

We choose two typical systems to examine: 1) $\alpha = 0.2$ when fewer NAT peers are in the system; 2) $\alpha = 0.4$ when there are relatively more NAT peers in the system. According to the measurement results in [5] and [6] and considering the possibly successful employment of NAT traversal techniques, the fewer- and more-NAT systems are good representations of real-world scenarios. Again, we discuss the homogeneous case and heterogeneous case, respectively, in terms of the download time, share ratio, and Jain's Fairness Index.

5.1 Download Time

5.1.1 Homogeneous Uplink Capacity

Figure 5.1 and Fig. 5.2 indicate that in both of the fewer- and more-NAT system with homogeneous peers , when P_o is higher, i.e., when public peers favor NAT peers more during optimistic unchoke, the download performance of NAT peers and that of the entire system can be improved tremendously, while public peers only need to spend

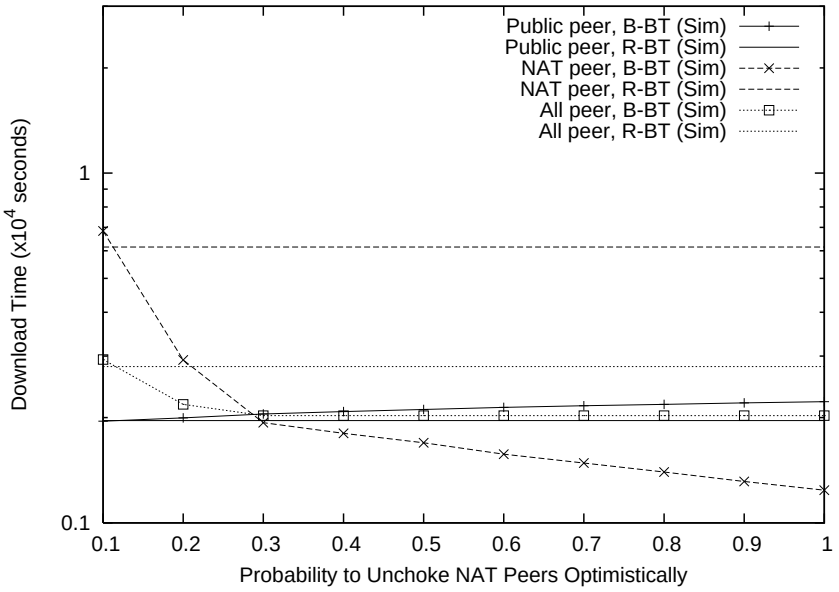

Figure 5.1: Average download time in homogeneous scenarios ($\alpha = 0.2$).

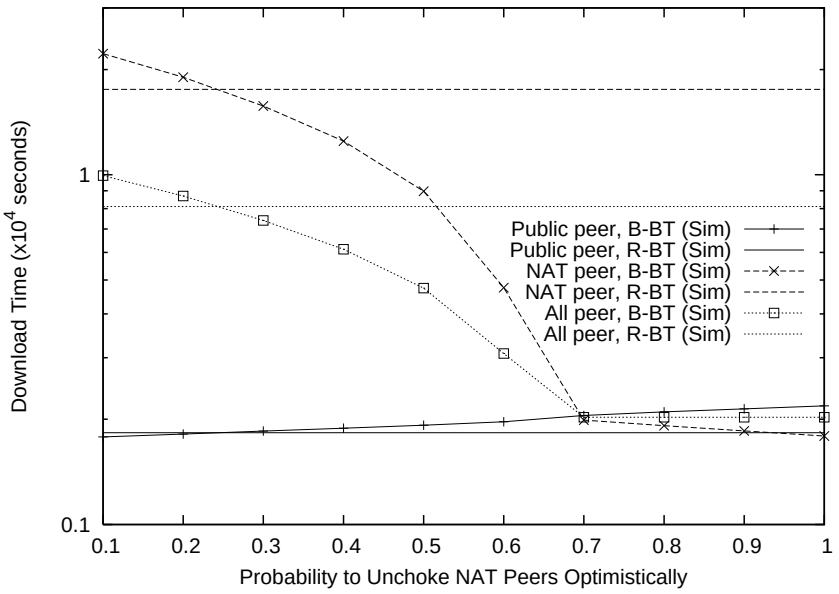

Figure 5.2: Average download time in homogeneous scenarios ($\alpha = 0.4$).

a little bit more time on downloading than that in the ordinary BitTorrent (R-BT) system. Particularly, in Fig. 5.1 when $\alpha = 0.2$, i.e., with fewer NAT peers, we notice that when P_o is larger than 0.3, NAT peers can experience better quality of service than public peers. In Fig. 5.2 when $\alpha = 0.4$, i.e., with more NAT peers, we see that when P_o is larger than 0.7, NAT peers have better performance than public peers as well.

5.1.2 Heterogeneous Uplink Capacity

Higher values of P_o can also significantly improve the download performance of NAT peers in heterogeneous cases as depicted in Fig. 5.3 and Fig. 5.4. Especially, we find that when fewer peers are behind NAT ($\alpha = 0.2$), if public peers optimistically unchoke NAT peers with a higher probability ($P_o \geq 0.9$), NAT peers can obtain the same quality of service as public ones, despite of the difference in uplink capacity. Considering the clustering effect of peers in the heterogeneous scenario, the biased optimistic unchoke strategy shows its potentiality of reducing the download time of NAT peers. We also observe that the performance gain, as demonstrated in Fig. 5.3 and Fig. 5.4, does not come at the cost of sacrificing public peers remarkably.

5.2 Fairness Metrics

In the last section, the proposed biased optimistic unchoke strategy has demonstrated its feasibility in helping NAT peers download a file faster. In this section, we explore how the biased optimistic unchoke strategy impacts the share ratio of peers and Jain's Fairness Index.

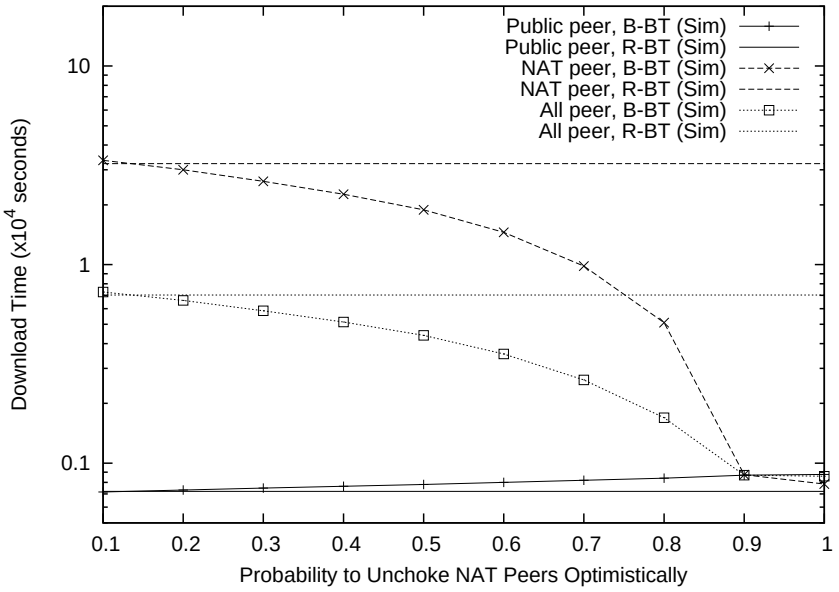

Figure 5.3: Average download time in heterogeneous scenarios ($\alpha = 0.2$).

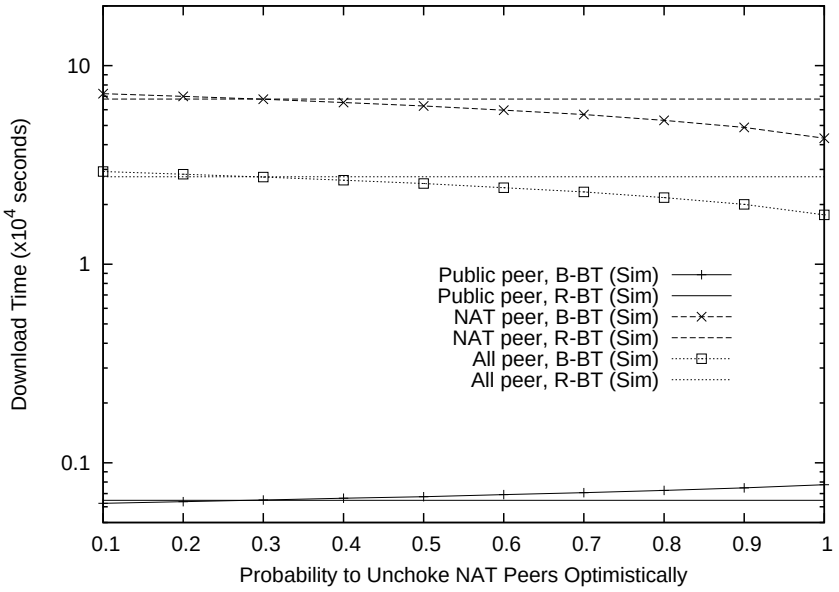

Figure 5.4: Average download time in heterogeneous scenarios ($\alpha = 0.4$).

5.2.1 Share Ratio

Homogeneous Uplink Capacity

Figure 5.5 and Fig. 5.6 indicate that in the homogeneous scenario, the proposed strategy can considerably reduce the share ratio of NAT peers and increase that of public peers to nearly 1, which is the expected value in the spirit of P2P applicaitons. We also notice in Fig. 5.5 that P_o cannot be arbitrarily large. Large values of P_o might result in excessively lowered share ratio of NAT peers (as low as 0.7) and increased share ratio of public peers (up to 1.1), which further causes a poor Jain's Fairness Index that will be discussed in the following sections.

Heterogeneous Uplink Capacity

The biased optimistic unchoke strategy is also able to tremendously reduce the share ratio of NAT peers to nearly 1 as shown in Fig. 5.7 and Fig. 5.8. On the other hand, the share ratio of public peers will not be affected hugely even when $P_o = 1.0$, i.e., public peers always choose NAT peers for optimistic unchoke. Figure 5.7 also suggests that P_o cannot be arbitrarily large. The share ratio of NAT peers can go quickly from 0.8 down to 0.4 when P_o is larger than 0.5.

5.2.2 Jain's Fairness Index

Homogeneous Uplink Capacity

Figure 5.9 and Fig. 5.10 demonstrate that the proposed strategy has the ability to make the system provide fairer service to competing and cooperating peers in the homogenous scenario. For example, in the fewer-NAT peers system, when public peers optimistically unchoke NAT peers with probability 0.2 to 0.3, the Jain's Fairness Index of the whole system is very close to 1, i.e., the system is offering almost the

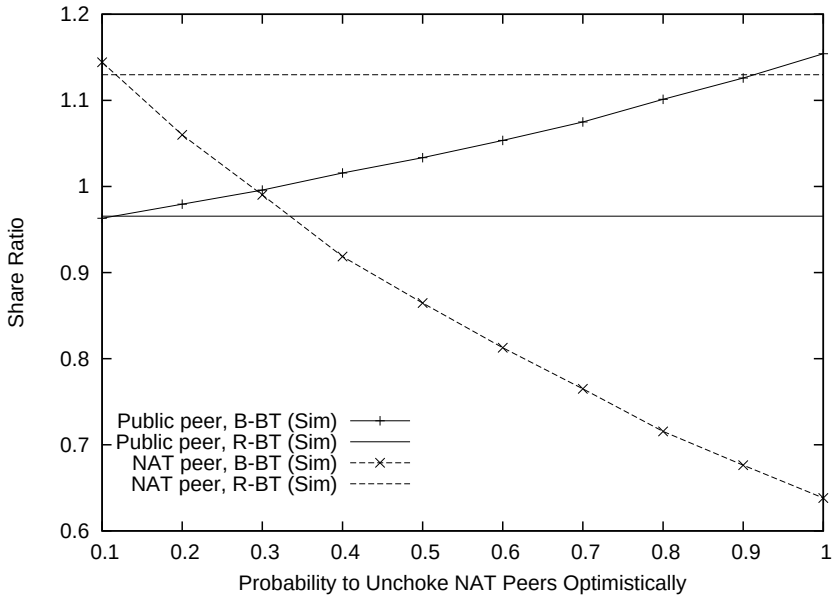

Figure 5.5: Average share ratio in homogeneous scenarios ($\alpha = 0.2$).

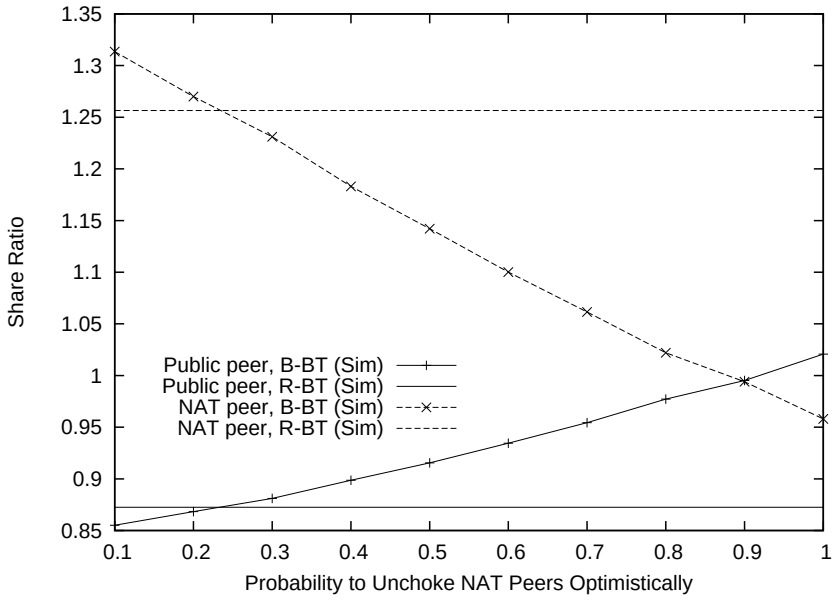

Figure 5.6: Average share ratio in homogeneous scenarios ($\alpha = 0.4$).

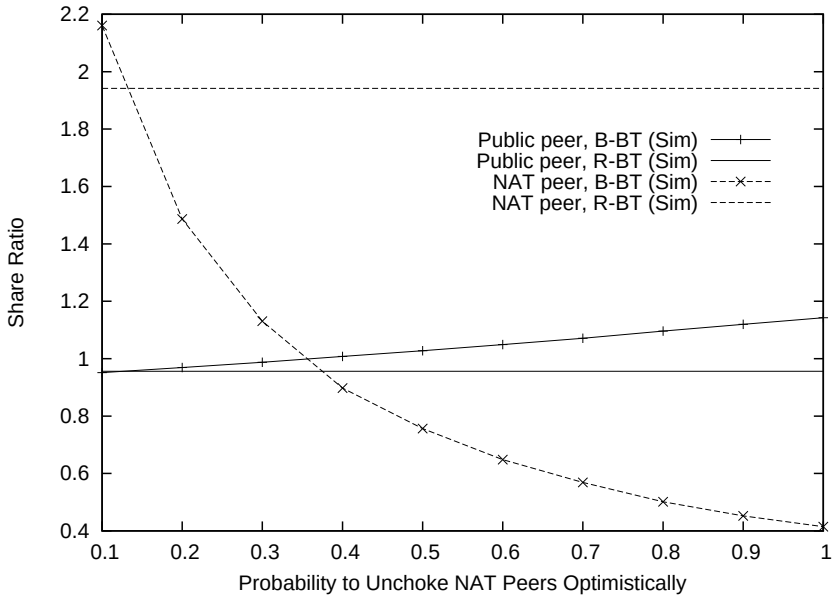

Figure 5.7: Average share ratio in heterogeneous scenarios ($\alpha = 0.2$).

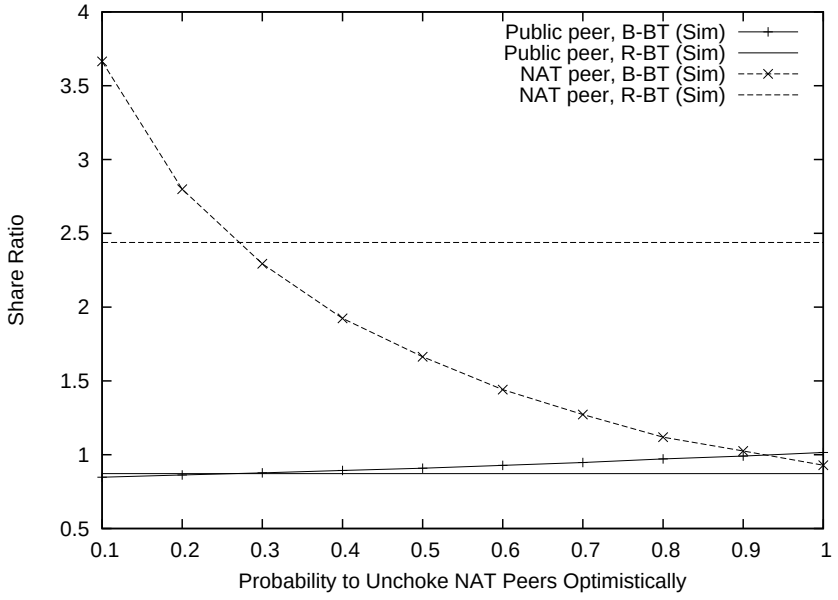

Figure 5.8: Average share ratio in heterogeneous scenarios ($\alpha = 0.4$).

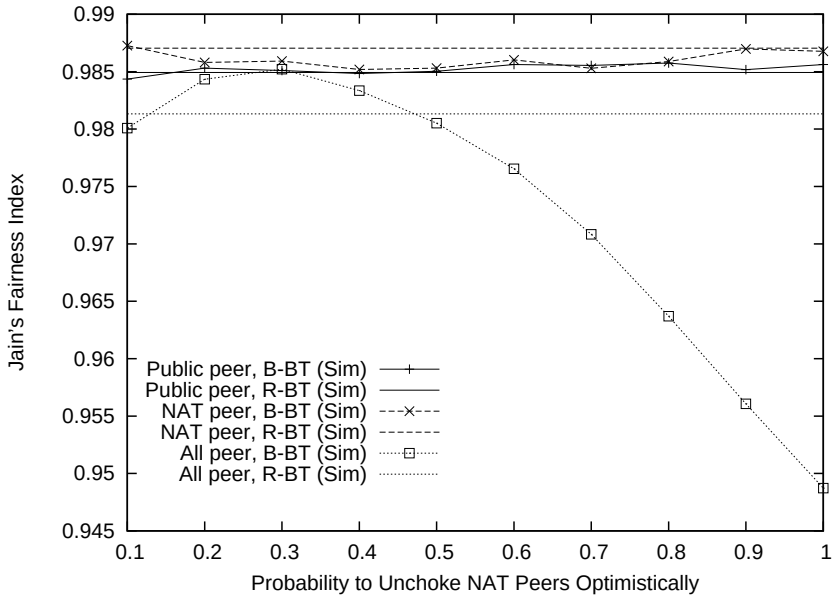

Figure 5.9: Jain's Fairness Index in homogeneous scenarios ($\alpha = 0.2$).

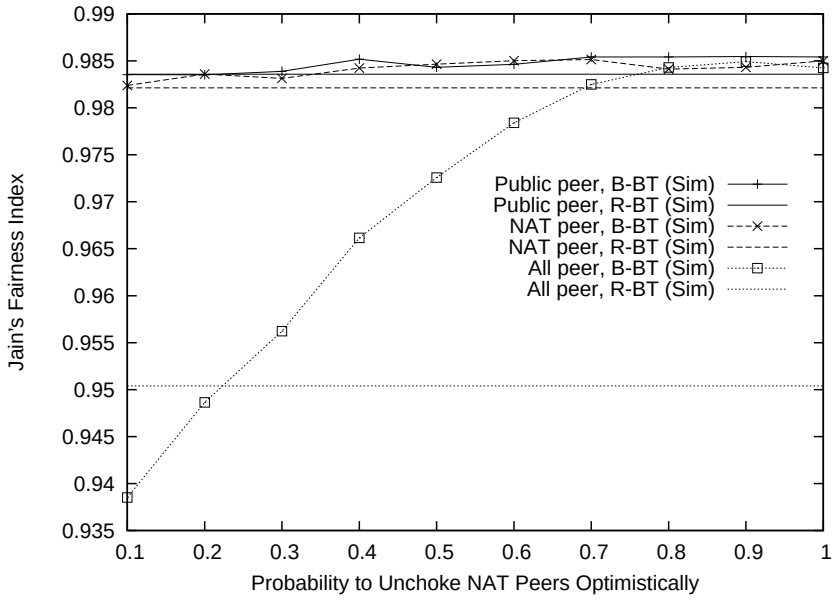

Figure 5.10: Jain's Fairness Index in homogeneous scenarios ($\alpha = 0.4$).

same service to public and NAT peers. Besides, in Fig. 5.9, there is an optimal value of P_o ($P_o = 0.3$) which gives the maximal value of Jain's Fairness Index. In specific, we observe in Fig. 5.5 that when $P_o = 0.3$, public peers and NAT peers have identical share ratios and thus Jain's Fairness Index at this point reaches its peak. Considering the similar file download time at $P_o = 0.3$ in Fig. 5.1, we conclude that with these settings in homogeneous scenarios, particularly when public peers optimistically unchoke NAT peers with probability 0.3, the whole system achieves its optimal performance and fairness.

Heterogeneous Uplink Capacity

The proposed optimistic unchoke strategy can also significantly improve Jain's Fairness Index of the whole system and that of NAT peers in the heterogeneous case as shown in Fig. 5.11 and Fig. 5.12. The optimal value of Jain's Fairness Index, from the view of the whole system, is also obversed in Fig. 5.11 with the corresponding P_o equal to 0.4. We can see in Fig. 5.7 that when $P_o = 0.4$, public peers and NAT peers have very close share ratios and thus the Jain's Fairness Index of the whole system reaches its peak at this point.

5.2.3 Further Discussion

In Fig. 5.1–5.12, we can see that when P_o is very small, the performance of NAT peers in B-BT, in terms of the file download time, share ratio, and Jain's Fairness Index, is not as good as that in R-BT. However, after a critical point, all the aforementioned metrics regarding NAT peers in B-BT surpass those in R-BT. We denote by P_c the value of the critical point in these figures. According to the analysis in Chapter 3, we

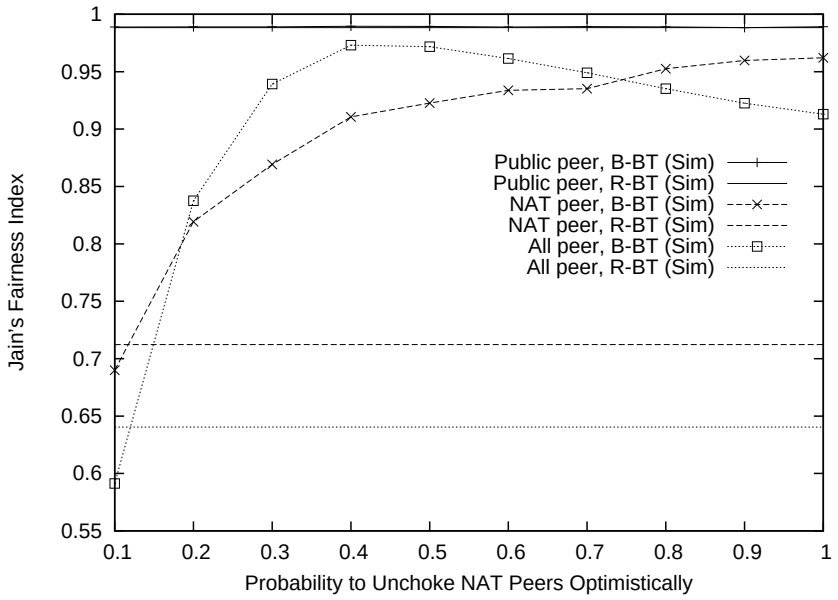

Figure 5.11: Jain's Fairness Index in heterogeneous scenarios ($\alpha = 0.2$).

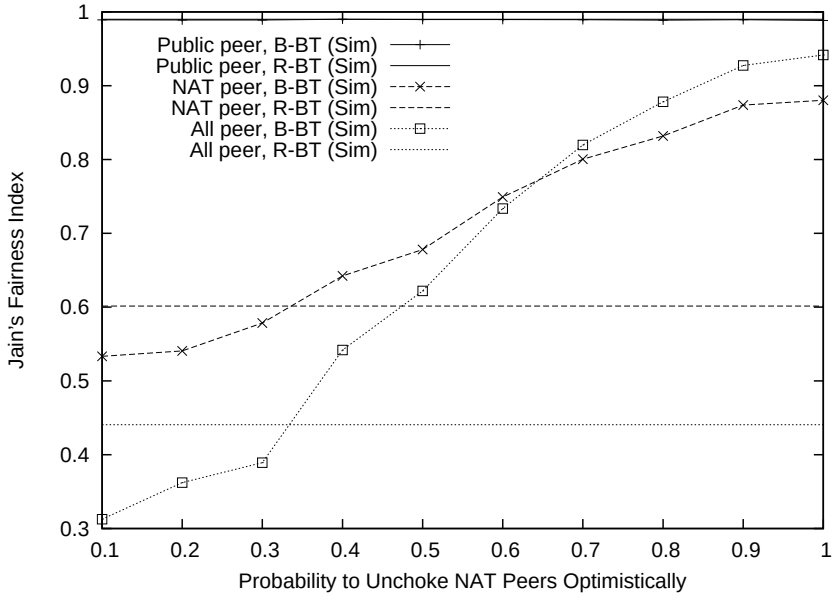

Figure 5.12: Jain's Fairness Index in heterogeneous scenarios ($\alpha = 0.4$).

can easily derive the value of P_c. For the homogeneous case, it is expressed as follows:

$$P_c = \frac{\alpha K - G}{(2-\alpha)K - M + 1}. \tag{5.1}$$

For the heterogeneous case, P_c is calculated by:

$$P_c = \frac{\alpha K}{(2-\alpha)K - M + 1}. \tag{5.2}$$

Obviously, the best value of P_o from the viewpoint of NAT peers is 1. However, if we simply allow public peers to always optimistically unchoke NAT peers, it is likely that NAT peers will have a smaller file download time than public peers, their share ratio is far below 1, and Jain's Fairness Index among all the peers is lower. In some cases, this might not be fair to public peers. Therefore, we introduce a tuning parameter β, and express P_o as follows:

$$P_o = (1-\beta)P_c + \beta, \tag{5.3}$$

where $0 \leq \beta \leq 1$. We indicate that if the new BitTorrent clients want to follow exactly the R-BT system, they can set β to be 0, while if they want to provide better services to NAT peers, they can set β to be a larger number. The modified protocol can be embedded in the client software and users have no control over the value of β. Right now, we are working on the models that can exactly reflect the relationship among α, β, file download time, share ratio, and Jain's Fairness Index. Our goal is to let the system have the ability to dynamically adjust the value of β according to the instant condition of network dynamics and system metrics.

5.3 Summary

In this chapter, we proposed a biased optimistic unchoke strategy with the intent to improve the download performance of NAT peers and fairness metrics of the system. The simulation results demonstrated that the propsoed strategy is quite effective in increasing the quality of service experienced by NAT peers and the overall system fairness, while still maitaining the download performance of public peers within an acceptable range.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

In this thesis, we have studied the impact of NAT devices on BitTorrent-like P2P systems. Specificly, we developed mathematically tractable models to quantify the download performance of the system in the presence of homogeneous and heterogeneous NAT peers. We identified, through modeling and simulation, the existence of NAT peers significantly degrades the download performance of NAT peers and that of the overall system. On the other side, the existence of NAT peers creates an opportunity for public peers to download faster than that in a system where all peers are publicly accessible. This is a counter-intuitive but indeed quite reasonable fact, which is attributed to the inaccessibility of NAT peers and BitTorrent's high efficiency of file sharing among peers.

The fairness issue is another commonly explored topic in P2P systems, which in an ideal case require the total amount of uploaded data from a peer be comparable to the total amount of downloaded data. Using the simulation approach, we investigated the fairness in our scenario. The share ratio and Jain's Fairness Index were of our

main concern. We discovered that when there is a large portion of peers behind NAT, NAT peers usually upload more than they obtain in the same time unit, while public peers download more than they contribute. This implies the existence of unfairness between public peers and NAT peers, which was further proved by applying Jain's Fairness Index.

In order to improve the download performance and fairness metrics of NAT peers and those of the entire system, we further proposed a simple but quite effective strategy. By leveraging the high uplink utilization of peers, we let public peers favor NAT peers a bit more during optimistic unchoke. Our simulation results showed that this easy-to-deploy strategy can bring remarkable improvement in the download performance and fairness metrics of NAT peers and those of the entire system without any noticeable sacrifice of public peers.

6.2 Future work

BitTorrent is notably the most famous P2P application on the Internet. It was originally designed for an efficient large-scale file sharing without the consideration of NAT. Due to its scalability and robustness to peer churn and easiness to deploy, more and more media streaming applications, including IPTV and VoD systems, are now using BitTorrent or its variants as the underlying swarming protocol to coordinate the data distribution. We think there are three main research issues beckon for further attention:

1. In Chapter 5, we explored the entire spectrum of the possible values of the optimistic unchoke probability (P_o) and did find that higher values of P_o can make the system operate much better in terms of the download performance and fairness metrics. We thus believe it is worthwhile to develop models that

can fundamentally reflect the relationship among the download performance of peers, the fairness metrics, the percentage of NAT peers, and the extent of the biased optimistic unchoke strategy;

2. The proposed strategy is obtained by the slight modification on the optimistic unchoke strategy. We think it is also possible to improve the performance of peers by modifying the tit-for-tat strategy like in [17] and [37]. When designing the tit-for-tat based strategies, we need to keep in mind the fairness metrics as well;

3. We also plan to extend our current models to explore the impact of NAT peers on P2P IPTV and VoD services which attract substantial research efforts recently. We anticipate that this is an interesting and challenging topic considering the real-time characteristics and constraints of these two applications and the less synchrony among users especially in P2P VoD systems.

Bibliography

[1] Napster official website, http://www.napster.com/.

[2] Gnutella Development Forum, "The Gnutella protocol specification v0.4," 2004.

[3] B. Cohen, "Incentives build robustness in BitTorrent," http://bitconjurer.org/BitTorrent/bittorrentecon.pdf, 2003.

[4] A. Parker, "The true picture of peer-to-peer file sharing," http://www.cachelogic.com, 2004.

[5] X. Zhang, J. Liu, B. Li, and T. S. P. Yum, "CoolStreaming/DONet: A data-driven overlay network for efficient live media streaming," in *Proc. of IEEE INFOCOM*, 2005.

[6] B. Li, S. Xie, Q. Qu, G. Keung, C. Lin, J. Liu, and X. Zhang, "Inside the new CoolStreaming: principles, measurements and performance implications," in *Proc. of IEEE INFOCOM*, 2008.

[7] X. Hei, C. Liang, J. Liang, Y. Liu, and K. W. Ross, "A measurement study of a large-scale P2P IPTV system," *IEEE Transactions on Multimedia*, vol. 9, no. 8, December 2007.

[8] Y. Huang, T. Z. J. Fu, D.-M. Chui, J. Lui, and C. Huang, "Challenges, design and analysis of a large-scale P2P-VoD system," in *Proc. ACM SIGCOMM*, 2008.

[9] R.K. Jain, D.-M. W. Chiu, and W. R. Hawe, "A quantitative measure of fairness and discrimination for resource allocation in shared computer systems," Technical Report, Digital Equipment Corporation, September 1984.

[10] D. Qiu and R. Srikant, "Modeling and performance analysis of BitTorrent-like peer-to-peer networks," in *Proc. ACM SIGCOMM*, 2004.

[11] A. Legout, G. Urvoy-Keller, and P. Michiardi, "Rarest first and choke algorithms are enough," in *ACM SIGCOMM IMC*, 2006.

[12] M. Izal, G. Keller, E. Biersack, P. Felber, A. Hamra, and L. Erice, "Dissecting BitTorrent: Five months in a torrent's lifetime," in *Proc. of Passive and Active Measurements (PAM)*, 2004.

[13] J. Pouwelse, P. Garbacki, D. Epema, and H. Sips, "The BitTorrent P2P file-sharing system: Measurement and analysis," In *Proc. of 4th International Workshop on Peer-to-Peer Systems (IPTPS)*, 2005.

[14] L. Guo, S. Chen, Z. Xiao, E. Tan, X. Ding, and X. Zhang, "Measurement, analysis and modeling of BitTorrent-like systems," in *Proc. of ACM SIGCOMM Internet Measurement Conference (IMC)*, 2005.

[15] R. Bindal, and P. Cao, "Can self-organizing P2P file distribution provide QoS Guarantees?" *OSR Special Issue on Self-Organizing Systems*, 2006.

[16] A. Legout, N. Liogkas, E. Kohler, and L. Zhang, "Clustering and sharing incentives in BitTorrent systems," in *Proc. of ACM SIGMETRICS*, 2007.

[17] A. Bharambe, C. Herley, and V. Padmanabhan, "Analyzing and improving BitTorrent performance," in *Proc. of IEEE INFOCOM*, 2006.

[18] B. Fan, D.-M. Chiu, and J. Lui, "The delicate tradeoffs in BitTorrent-like file sharing protocol design," in *Proc. of IEEE ICNP*, 2006.

[19] W.-C. Liao, F. Papadopoulos, and K. Psounis, "Performance analysis of BitTorrent-like systems with heterogeneous users," *Performance Evaluation*, vol. 64, no. 9–12, 2007.

[20] A. Chow, L. Golubchik, and V. Misra, "BitTorrent: an extensible heterogeneous model," in *Proc. of IEEE INFOCOM*, 2009.

[21] P. Marciniak, N. Liogkas, A. Legout, and E. Kohler, "Small is not always beautiful," in *Proc. of 7th International Workshop on Peer-to-Peer Systems (IPTPS'08)*, 2008.

[22] Network Address Translation, http://www.wikipedia.org/.

[23] P. Srisuresh and K. Egevang, "Traditional IP Network Address Translator (NAT)," *IETF RFC 3022*, 2001. (Obsoletes: K. Egevang and P. Francis, "The IP Network Address Translator (NAT)," *IETF RFC 1631*, 1994.)

[24] J. Rosenberg, R. Mahy, P. Matthews, and D. Wing, "Session Traversal Utilities for NAT (STUN)," *IETF RFC 5389*, 2008. (Obsoletes: J. Rosenberg, J. Weinberger, C. Huitema and R. Mahy, "STUN - Simple Traversal of User Datagram Protocol (UDP) through Network Address Translators (NATs)," *IETF RFC 3489*, 2003.)

[25] J. Rosenberg, R. Mahy and P. Matthews, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)," *IETF Internet Draft*, 2009.

[26] J. Rosenberg, "Interactive Connectivity Establishment (ICE): A protocol for Network Address Translator (NAT) traversal for offer/answer protocols," *IETF Internet Draft*, 2007.

[27] S. Guha, Y. Takeda and P. Francis, "NUTSS: A SIP based approach to UDP and TCP connectivity," in *Proc. of ACM SIGCOMM'04 FDNA Workshop*, Portland, OR, Aug 2004, pp. 43–48.

[28] S. Guha and P. Francis, "Characterization and measurement of TCP traversal through NATs and firewalls," in *Proc. of ACM SIGCOMM IMC'05*, 2005.

[29] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and S. Shenker, "Making Gnutella-like P2P systems scalable," in *Proc. ACM SIGCOMM*, 2003.

[30] P. Eugster, R. Guerraoui, A. M. Kermarrec, and L. Massoulie, "From epidemics to distributed computing," *IEEE Computer*, vol. 37, no. 5, 2004.

[31] S. A. Baset and H. G. Schulzrinne, "An analysis of the Skype peer-to-peer internet telephony protocol," in *Proc. of IEEE INFOCOM*, 2006.

[32] PPLive official website, http://www.pplive.com/.

[33] E. Adar and B. A. Huberman, "Free riding on Gnutella," in *First Monday*, September 2000.

[34] J. H. Saltzer, D. P. Reed, and D. D. Clark, "End-to-end arguments in system design," in *ACM Transactions on Computer Systems*, vol. 2, pp. 277-288, 1984.

[35] J. J. D. Mol, J. A. Pouwelse, D. H. J. Epema, and H. J. Sips, "Free-riding, fairness, and firewall in P2P file sharing," in *Proc. of IEEE P2P Computing*, 2008.

[36] S. Saroiu, K. P. Gummadi, and S. D. Gribble, "A measurement study of peer-to-peer file sharing systems," in *Proc. of Multimedia Computing and Networking*, 2002.

[37] M. Piatek, T. Isdal, T. Anderson, A. Krishnamurthy, and A. Venkataramani, "Do incentives build robustness in BitTorrent?" in *Proc. of NSDI*, 2007.

Yangyang Liu

EDUCATION

M.Sc.	U. of Victoria, Victoria, BC, Canada	01/2008–Present
B.Eng.	U. of Electronic Sci. & Tech. of China, Chengdu, China	09/2002–06/2006

PUBLICATIONS

- Y. Liu and J. Pan, "The impact of NAT on BitTorrent-like P2P systems," *IEEE P2P 2009*, Seattle, Washington, USA, 2009
- R. Rukhsana, Y. Liu, and J. Pan, "Evaluating video streaming over UWB wireless networks," *ACM WMuNeP 2008*, Vancouver, BC, Canada, 2008

Co-op Work Experience

Software Developer	Research in Motion, Ottawa, ON, Canada	09/2009–12/2009
Software Developer	IBM Canada, Toronto, ON, Canada	01/2010–08/2010