國立臺灣大學電機資訊學院資訊網路與多媒體研究所

碩士論文

Graduate Institute of Networking and Multimedia
College of Electrical Engineering & Computer Science
National Taiwan University

Master Thesis

經由選擇性排程面對同儕網路系統中的連接限制

Resolving Peer-to-Peer Connection Barriers by Selective Scheduling

張朝覺

Chao-Chueh Chang

指導教授：陳宏銘 博士

Advisor: Homer H. Chen, Ph.D.

中華民國 98 年 6 月

June, 2009

國立臺灣大學碩士學位論文
口試委員會審定書

經由選擇性排程面對同儕網路系統中的連接限制

Resolving Peer-to-Peer Connection Barriers by Selective Scheduling

本論文係張朝覺君（學號 R96944037）在國立臺灣大學資訊網路與多媒體研究所完成之碩士學位論文，於民國九十八年六月十三日承下列考試委員審查通過及口試及格，特此證明

口試委員：

陳宏銘（簽名）

蘇志宏（指導教授）　施吉昇

[illegible]　陳奕廷

所　長：　[illegible]（簽名）

誌謝

在研究所兩年的期間中，首先要感謝的是我的指導教授陳宏銘教授，從我的指導教授身上，我學到了作學問的嚴謹態度，以及對於自己的工作永不懈怠、精益求精的精神，這些都讓我印象深刻。也感謝老師能讓我在他的指導下進行碩士班的學業，有了老師的指導，我才能順利地完成我的碩士論文。再來要感謝黃寶儀教授，在研究所求學的這段期間，也在黃教授的指導以及協助之下，才能順利的完成碩士論文，除了學業上的討論，黃教授也給予了我很多作人處事上的觀念以及面對困難時的心態調整，在此致上最誠摯的感謝。

感謝施吉昇教授、陳寬達教授以及蔡志泓教授撥空參加我的口試，並且在口試期間提出許多寶貴的意見，這些意見都使得我的碩士論文更加完善。

感謝 MPAC 實驗室的各位學長、同學以及學弟妹們在這段期間的幫助，感謝呂孟庭學長帶領我進入這個研究領域，並且對於我許多的問題都能夠不厭其煩的回答，感謝梁家愷學長以及翁秉義同學在我心情低落時表演打電動的神技，感謝楊奕軒學長對於求學以及待人處事上各式各樣的幫助。感謝蘇亞凡同學在我心情低落的時候帶來歡樂，感謝黃翊鑫同學在我遇到困難的時候給予鼓勵，感謝林佑璟同學在我雜事煩身的時候提供幫助，感謝歐道聖以及李文甫學弟平常的聊天解悶。也感謝 NSLab 的各位成員們，尤其是同屆的夥伴們，江怡萱、陳冠名、陳宥融、黃介廷以及唐彬雲，若沒有你們大家，我想我的碩士生涯會過的淡然無味。

最後要感謝我的家人，一直以來扮演著最支持我的角色，無論我的心情起伏，總是能夠提供我最大的協助。

張朝覺 2009 年 7 月

中文摘要

在現今網際網路架構下，正面臨著網路位址不足的問題，此問題已由使用網路位址轉譯器暫時得到紓解。網路位址轉譯器在導引網路流量時，若是不知封包目的地，則會將連線阻擋起來，此現象在需要成員能夠接受外來連線要求的同儕系統中造成了系統效能的降低。此篇論文研究了使得同儕系統中能夠接受外來連線的成員先取得資料的方法，我們期望透過此方法，可以使得這些能夠接受外來連線的成員可以幫助同儕網路系統中資料的傳送。在檔案傳輸同儕系統中，我們的模擬呈現了百分之二十的效能增進。

此論文另一重點為考慮網際網路服務提供者(Internet service provider)的同儕系統傳輸設計。考慮到在同一網際網路服務提供者之下的傳輸效能較好，此篇論文的方法在資料傳輸排程時考慮資料傳送者的 ISP，在模擬中，此方法於檔案傳輸以及多媒體串流的同儕系統均能大幅的減少跨網際網路服務者的網路流量，並且服務品質仍與原來未考慮 ISP 即進行排程時一致。

關鍵字：同儕網路系統(Peer-to-peer networking system), 減少跨網路之網路傳輸量

ABSTRACT

Recent study indicates that a large portion of peers in a P2P system do not contribute their out-going bandwidth due to the usage of network address translator (NAT) that blocks the incoming traffic. Recent study also shows that many internet service providers (ISPs) simply choose to block P2P connections because they create too much cross-ISP traffic. In this paper, we present a public-first approach to resolve the NAT-related connectivity constraint for P2P content delivery systems. Unlike STUN, which attempts to resolve the NAT traversal problem directly, our approach targets better utilization of available connectivity between peers and achieves this goal by selective content delivery scheduling. Experimental results show that it improves the P2P file transmission time by 20%. To reduce the cross-ISP traffic of P2P systems, we propose a light-weight, distributed method to identify peers of the same ISP. The proposed method reduces cross-ISP traffic without affecting the P2P system performance.

Index Terms — P2P systems, ISP-Friendly transmission

CONTENTS

口試委員會審定書 .....#

誌謝 i

中文摘要 iii

ABSTRACT iv

CONTENTS v

LIST OF FIGURES vii

Chapter 1 Introduction.....1

1.1 Problem Statement.....2

1.1.1 Network Address Translator (NAT) 2

1.1.2 Cross-ISP Traffic 4

1.2 Contribution.....6

1.3 Organization of this Study 7

Chapter 2 Literature Survey.....8

2.1 NAT Traversal Techniques.....8

2.1.1 STUN (Simple Traversal of User Datagram Protocol through Network Address Translators).....8

2.1.2 STUNT.....10

2.1.3 UPnP 11

2.2 ISP-Friendly Mechanism.....12

2.1.1 Network Distance Measurement.....12

2.1.2 Dedicate Neighbor List. 13

2.1.3 Cross-ISP Traffic Bandwidth Limitation. 14

Chapter 3 Proposed Methods 16

3.1 Public-First Delivery 16

3.1.1 Characteristics of the public peer 17

3.1.2 Public peer detection process & the protocol stack 18

3.1.3 Priority request queue 21

3.2 ISP-Friendly Scheduling 22

3.2.1 Identification of domestic peers 23

3.2.2 ISP-aware rarest-first segment scheduling 24

Chapter 4 Simulation 26

4.1 Simulation Model 26

4.2 Effect of Public-First Delivery 26

4.3 ISP-friendly Scheduling 35

4.4 Effect of ISP-Friendly with Public-first Delivery 38

Chapter 5 Conclusion 40

REFERENCE 42

LIST OF FIGURES

Fig. 1-1. Global Consumer Internet Traffic.....2

Fig. 1-2. Ratio of Peers behind NAT. 3

Fig. 2-1. STUN probing algorithm. 8

Fig. 3-1. Public peer detection protocol. 19

Fig. 3-2. ISP-friendly scheduling algorithm.....24

Fig. 4-1. Performance plot of peer-to-peer file sharing system simulation for different percentage of public peers in the network 28

Fig. 4-2. Performance plot of peer-to-peer streaming system simulation for different percentage of public peers in the network 30

Fig. 4-3. Average buffer utilization of peer-to-peer streaming system simulation for different percentage of public peers in the network 31

Fig. 4-4. Performance plot of peer-to-peer streaming system simulation for different start-up time of private peers in the network.....33

Fig. 4-5. Performance plot of peer-to-peer streaming system simulation for different start-up time of private peers in the network.....33

Fig. 4-6. Traffic between ISPs not Using ISP-Friendly Scheduling in a Peer-to-peer File Sharing System Simulation 36

Fig. 4-7. Traffic between ISPs Using ISP-Friendly Scheduling in a Peer-to-peer File Sharing System Simulation 37

Fig. 4-8. Traffic between ISPs using ISP-friendly Scheduling in a Peer-to-peer Streaming Simulation 37

Fig. 4-9. Traffic between ISPs Using ISP-friendly Scheduling and Public-First Delivery in a Peer-to-peer File Sharing Simulation 38

LIST OF TABLES

Table 3-1. The Probability for Finding a Lacking Data Segment in a Partner Located in the Same ISP.....22

Chapter 1 Introduction

In the Internet environment nowadays, numerous applications are proposed to offering services needed. Peer-to-peer technology had been proposed to solve the massive bandwidth requirement in the service like file transmission and multimedia content delivery. Traditionally, these services were provided under a client-server network scenario. That is, a dedicate server responds for handling requesting from numerous clients. With Internet user grows dramatically, such a network environment scenario may not be able to serve massive clients. The application server may need more available bandwidth for serving the need of massive data transmission.

Peer-to-peer technology had been adopted in many applications in the Internet. In a peer-to-peer network, a peer not only downloads but also uploads contents. That makes the peers are able to share the massive bandwidth demand of data transmission. The peer-to-peer technology indeed reliefs the bandwidth requirement brought by services like file distribution and multimedia content streaming. This self-sustaining nature makes Internet transmission services can be easily carried out without massive bandwidth requirement. Numerous applications were proposed to offer services. BitTorrent [11] was proposed to offer P2P file transmission service. CoolStreaming [6], PPLive[18], and PPStream [19] are famous commercial P2P multimedia streaming systems.

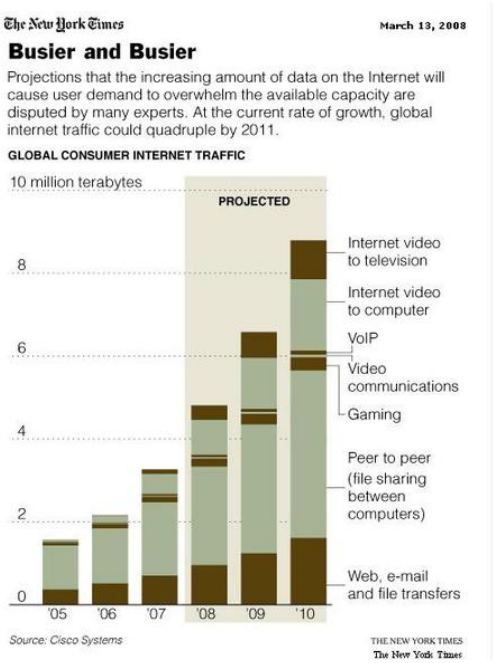


Fig. 1-1. Global Consumer Internet Traffic

Peer-to-peer (P2P) techniques have been adopted to offer various content delivery services including file sharing and multimedia streaming. These services thus become the majority stream of the Internet traffic. Figure 1-1 is the global consumer Internet traffic usage measurement and its projection to the future. We may see that the peer-to-peer application brought the main portion of the Internet traffic.

Even if the peer-to-peer technology is widely used in many applications, it is still facing challenges. The challenges come from the underlying networking infrastructures. The author will describe the problem in the following sections.

1.1 Problem Statement

1.1.1 Network Address Translator (NAT)

P2P systems face many challenges. One of them is related to the usage of the

network address translators, which drop explicit incoming packets. Due to IPv4 address shortage problem, user may connect to the Internet through a NAT, which allows many users share a public IP address. Recent research shows over 70% users of P2P systems connect to the Internet through NAT [1], [10]. Figure 1-2 shows the observed ratio of peers in the P2P streaming system [10]. A NAT routes out-going connections and forwards incoming ones to the clients. However, explicit incoming packets are dropped if the NAT

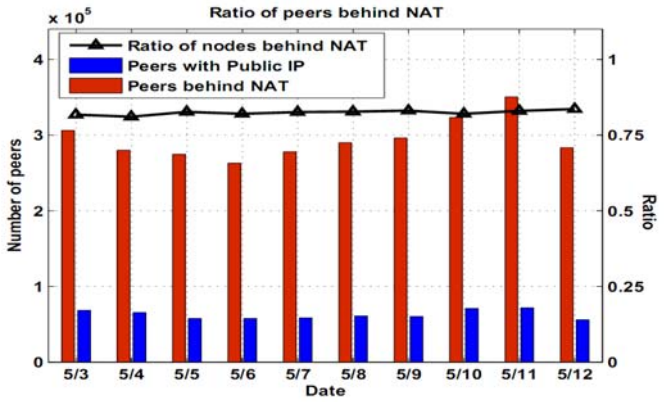


Fig. 1-2. Ratio of peers behind NAT

cannot figure out where the packets should go.

A Peer shall be able to accept incoming data requests for contributing its bandwidth. A peer behind a NAT may fail to accomplish that because of the NAT. That makes NAT traversal critical to P2P system performance. Numerous NAT traversal

techniques were proposed. STUN [2] is a well-known UDP NAT traversal method. It probes how the NATs perform packet forwarding and opens an available port for further usage. However, some NATs perform port forwarding randomly, which makes it impossible for STUN to build connection. Until the NAT traversal problem is solved effectively, it is difficult for peers behind NAT to contribute their uplink bandwidth and disk space to the P2P system. As opposed to proposing a more sophisticated NAT traversal mechanism or an incentive mechanism to encourage sharing, we take a very different stand. We openly recognize and accept the connectivity constraint as a result of fast expansion of the Internet. This leads us to seek an alternative solution to deal with the common existence of peers behind the NAT.

1.1.2 Cross-ISP Traffic

Most existing P2P systems employ arbitrary neighbor selection and build up overlay content delivery network without considering the underlying network architecture. As such, a peer may request data from a distant peer located in another Internet service provider (ISP), even through the data can be easily obtained from a peer nearby in the same ISP. Also, requesting data from a domestic peer obviously costs less than that from a foreign ISP. This has motivated us to make the P2P systems more ISP-friendly.

Besides, P2P traffic is relative massive. Figure 1-1 shows the internet consumer

traffic record and its projection to the future. We may see that the traffic mainly comes from P2P applications. The massive traffic has motivated numerous ISPs to block P2P connections, which may contain massive cross-ISP traffic. While cross-ISP traffic costs ISPs, this behavior of ISP is understandably selfish but hinders data sharing over the Internet. Prior work [9] has proposed to select peers from local ISPs to minimize cross-ISP traffic. Though the question remains is whether such a peer selection policy will also benefit the users.

These connective constraint issues indeed affect P2P services. In this thesis, we seek to deal with the performance degradation due to these issues. To solve the performance degradation caused by peers behind NATs, we propose a public-first content delivery policy that makes better use of peers capable of serving the data, achieves better out-going bandwidth utilization, and improves transmission efficiency. Our method can be easily implemented and applied to most P2P systems. To solve the performance degradation caused by connectivity constrained ISPs, we develop a distributed, easy to implement scheduling method that reduces cross-ISP traffic without sacrificing the P2P system performance. In this ISP-friendly scheduling method, each peer takes the underlying network architecture into consideration when selecting the data sources. Cross-ISP data request is made only if no one in the domestic ISP has the data.

1.2 Contribution

In this thesis, the author proposed a delivery method that discriminates peers that are able to receive data requests from others and prioritizes data requests from them. The peers who are able to receive requests are called public peers in this thesis. By sending data to public peers first, performance of peer-to-peer systems can be improved without extra infrastructures. The author also stated the process of differentiating public peers and prioritizing requests from public peers as public-first delivery. Simulations of peer-to-peer file sharing systems and peer-to-peer streaming systems were conducted to see if the public-first delivery benefits or not. Simulation result shows the public-first delivery benefits file sharing system. It improves the performance by 20% in terms of reducing file transmission time. Simulations for peer-to-peer streaming systems were also conducted. The simulation result shows the performance of peer-to-peer streaming is highly sensitive to available upload bandwidth in the system.

To deal with the constraint comes from the ISPs who are not willing to allow peer-to-peer systems make cross-ISP traffic, the author proposed ISP-friendly scheduling scheme. In this ISP-friendly scheduling scheme, each peer will rearrange its neighbor list by the ISP information. That makes peers contact neighbors who are located in the same ISP first when they are going to request data segments. The author had conducted series of simulations to see if the proposed method effective or not.

Simulation result shows the cross-ISP traffic is controlled and the performance is not affected.

1.3 Organization of this Study

In chapter 2, literature survey is performed to see several topics about NAT and ISP-friendly mechanisms. We may see how the NAT traversal techniques are preceded and how ISP-friendly mechanisms take effect. In chapter 3, the author has introduced the mechanism of public-first delivery and ISP-friendly scheduling. Simulation setting and result are shown in the chapter 4.

Chapter 2 Literature Survey

Recent study has shown that over 70% users of P2P systems connect to the Internet through NAT [1], [10]. However, requesting data from a peer behind a NAT can be blocked by the NAT. Furthermore, the ISPs may block P2P connections due to the billing problem between ISPs caused by the massive P2P traffic. This selfishness definitely affects the P2P system performance. In this chapter, I would like to survey the related works that are fighting these behaviors which are negative to P2P system performance.

2.1 NAT Traversal Techniques

Due to the IP address shortage problem, many computers connect to the Internet through a NAT box. However, the NAT boxes would drop the incoming packets if the route is not established properly. Popular NAT traversal techniques will be introduced in the following sections.

2.1.1 STUN (Simple Traversal of User Datagram Protocol through Network Address Translators)

STUN performs NAT traversal by probing how the port mapping is established and establishing a port mapping that can be used in further applications. STUN probes how the NATs build port mapping by sending UDP packets to a third-party server. After the

probing process, STUN will show if the NAT is traversable or not. If the NAT is traversable, the port forwarding information shall be established according to the type of the NAT. The port forwarding information can be built by sending UDP packets to the connection destination. After establishing the port forwarding, the NAT knows how to route the incoming packets and the user behind the NAT is able to accept incoming packets. The probing protocol is shown in Figure 2-1. In STUN, NATs are categorized into four types by how the NATs recognize the UDP packet destination and port mapping information established: 1) Full cone 2) restricted cone 3) restricted port 4) symmetric. Full cone NAT reserves the port mapping information that is established by the previously sent packets. Restricted cone NAT builds port mapping information by recognizing the destination IP address of out-going packets and filtering incoming packets which are not coming from the recognized IP address. Restricted port NAT acts familiar to restricted cone NAT. Besides recognizing the IP address, the restricted port NAT even recognizes the destination port. These types of NATs mentioned above are traversable. The symmetric NATs, however, are not traversable. Symmetric NATs will establish a new port mapping information once a new session is initialized. Also, the port will be dispatch randomly. Because of the port mapping is established randomly, the user behind the NAT may fail to know which port is usable and thus fail to traverse the NATs.

The disadvantages of STUN are that it requires third-party server's affiliation for the probing process and that it cannot guarantee 100% successful connection.

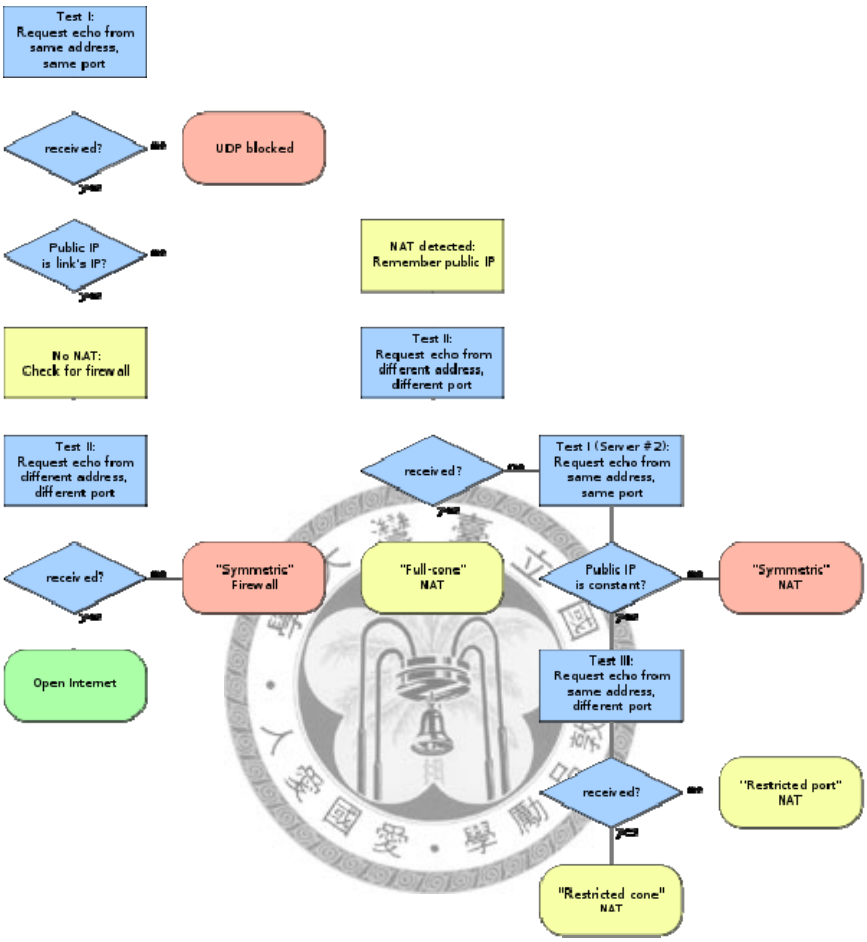


Fig. 2-1. STUN probing algorithm.

2.1.2 STUNT

STUNT is the TCP variation of STUN. STUNT probes how the NATs perform port mapping while establishing TCP connections. Moreover, the author of [12] probes into symmetric NAT, which is not traversable in STUN. The author extended the port mapping probing process and found out the possibility to traverse. In their probing

process, they had found out that the symmetric NAT may accept incoming connection if the destination IP address and port were recognized. Also, NATs may block the connection if abnormal packet sequence is observed. The author of [12] has shown the successful connection rate and the port mapping behavior of NATs that are available in the market [13]. However, if the port mapping is established randomly, the prediction will fail and it cannot build connection successfully. Since this method does not guarantee 100% successful connection rate, the author of this thesis is looking for another method that deals with performance degradation caused by NATs.

2.1.3 UPnP (Universal Plug-and-play)

UPnP [15] is another popular NAT traversal method that makes use of configurable port forwarding. UPnP sets series of protocol that discovers the usability functionality of machines that connect to the same network domain. Clients probe the functionality of the NAT devices and configure how the NAT performs port forwarding. With proper port forwarding settings, the client work behind a NAT will be able to accept incoming packet.

Even UPnP was not designed for NAT traversal; it's still the most popular NAT traversal method. For this method to be effective, the NAT has to be UPnP compatible. The author has performed a market survey for the UPnP compatibility. The result shows that over 80% of NATs are compatible to UPnP.

2.2 ISP-Friendly Mechanism

Most existing P2P applications randomly assign peers to neighbors that are potential data source. This result in traffic randomness: Data path between data source and data sink may cross different ISPs. Mass cross-ISP traffic is not favored by ISPs because of the operation cost. Many ISP-friendly traffic control mechanism have been proposed.

2.2.1 Network Distance Measurement

Ji Li *et al* [3] proposed a scheme that uses proximity neighbor selection for structured P2P systems to estimate the autonomous system (AS) hops between peers and their neighbors. By filtering the neighbors whose distance is larger than a threshold, the cross-ISP traffic is reduced. They propose a hybrid proximity neighbor selection algorithm which uses the AS-path length as a proxy for network latency to reduce the number of nodes to be probed. In their simulation, by using Autonomous System information effectively, P2P systems can achieve lookup performance approaching that based on proximity neighbor selection, but with much less network traffic. Simulations were conducted to show the traffic log. Their simulation results demonstrate a 92% reduction in probing traffic with only a 2% increase in the lookup latency on a synthetic topology. They also present a heuristic approach using simple AS topology and scoping information to improve replication in structured P2P networks.

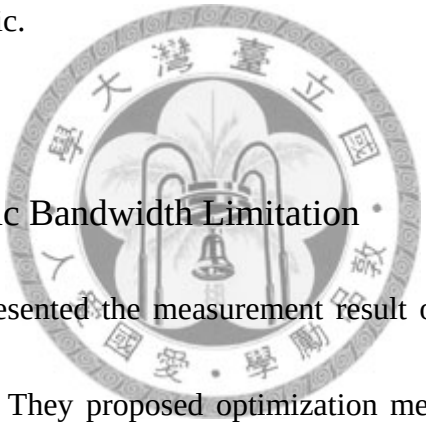
D. R. Choffnes and F. E. Bustamante [5] suggested measuring networking distance before requesting data from neighbors. By reusing the information collected from various content distributing network (CDN) system services, peer may find peers that are relatively closer to it. In their mechanism, peer retrieves the domain name of the replica server of CDN services. Under the assumption that close peer will be serve by the same CDN replica server. The peers that are close to each other may retrieve similar CDN replica server name. By comparing the similarity of the observed domain name, peer may identify if its neighbor is relatively closer to it. They had implemented their method in the BitTorrent client plug-in. Using results collected from a deployment in BitTorrent with over 120,000 users in nearly 3,000 networks, we show that our lightweight approach significantly reduces cross-ISP traffic and, over 33% of the time, it selects peers along paths that are within a single AS.

2.2.2 Dedicate Neighbor List

Bindal *et al* [15] proposed a peers' ISP information collection scheme in BitTorrent tracker. The tracker, which is used to transmit peer list and peers' buffer maps, prepares a dedicate neighbor list for each peer who is requesting peer list. The peer list consists of a certain portion of domestic peer. Given such a peer list, the connection is spontaneously controlled in the domestic ISP. In this paper, the author also showed the effect of different portion of domestic peer in the peer list. The simulation result shows

the transmission delay is decreasing and file transmission time is reduced while the portion of domestic peer increases.

Aggarwal *et al* [9] suggested an ISP-supported neighbor selection mechanism. In [9], the authors suggested that ISPs should provide portal servers for the peer-to-peer systems. The portal servers collect the geographical information of peers and offer the dedicate peer list for every peer. With such a peer list, the peer may requests data from the peer that located closer to it. Simulation results were provided to see how this mechanism affects the traffic.



2.2.3 Cross-ISP Traffic Bandwidth Limitation

In [17], the authors presented the measurement result of a peer assisted video on demand streaming system. They proposed optimization mechanisms, including server bandwidth and cross-ISP traffic optimization, for a video on demand streaming system. The ISP-friendly mechanism was proposed and the traffic measurement results were shown.

In their measurement result, the cross-ISP traffic takes over 60% of the total traffic. While cross-ISP traffic may incur payment, the massive cross-ISP may not be welcomed by ISPs. The authors probed into the relationships between ISPs. Based on the ISP relationships, ISPs can be grouped together to form economic entities, whereby no

payment is involved for traffic within an entity but traffic crossing entity boundaries does incur payment.

To minimize cross boundaries payment, the peers within the same entity will be distributed into the same group. The peers are limited to build connection with peer of the same group. Obviously, the proposed mechanism shall prevent any payment incurred traffic. The result showed the bandwidth requirement is increased accordingly. However, the bandwidth saving is still significant compared to traditional server-client model.

Chapter 3 Proposed Methods

In this section, we describe the feasibility, design philosophy, and implementation of the proposed mechanisms for public-first and ISP-friendly scheduling. The public-first delivery mechanism is designed to resolve the barrier created by NATs, and the ISP-friendly mechanism to reduce cross-ISP traffic. A peer that serves data to another node of the network is called a partner of the node in this section. Section 3.1 offers descriptions about the public-first delivery, which is used to handle the barriers created by NATs. Section 3.2 describes the proposed ISP-friendly scheduling scheme, which is used to fight the barriers created by ISPs.

3.1 Public-first Delivery

The public-first delivery mechanism is a selective delivery mechanism that distinguishes between public peers and private peers. A peer of a P2P system is called a public peer if it is freely accessible by any other peer of the system; otherwise, it is called a private peer. The distinction is made because these two types of peers have profoundly different impacts on the performance of a P2P system. An incoming packet is blocked by a NAT and becomes inaccessible to any peer behind the NAT if it is unrecognizable to the NAT, resulting in a connection barrier of the P2P system. In this case, the peers behind the NAT are unable to receive the incoming packet and are

private peers. Peers behind a firewall are also private peers. Since private peers do not receive incoming requests for content, they cannot serve as partner peers. In other words, they do not contribute resource to the system. Besides, data transmissions are made without considering if the data sink is able to contribute it to the others. This is obviously negative to the system performance. Our mechanism discriminates the public peers from the private peer and grants higher priority to the public peers. By sending the data to the public peers as soon as possible, these peers may contribute their bandwidth and improve system performance.

The proposed mechanism consists of two parts, which are the public peer detection process and the priority queue of data request. We establish the protocol stack for the detection process to detect the capability of accepting incoming packets. Another part is the priority queues for the data requests. Each peer has two data request queues and grants the higher priority to the queue that is prepared for the public peers. By doing this, the data may be delivered to the public peers and the public peers may contribute their bandwidth as soon as possible. The following explains the detail of the proposed mechanism.

3.1.1 Characteristics of the public peer

A public peer is defined as a peer who is able to accept unrecognized incoming packets in this paper. Since a peer who works behind the NAT may accept incoming

connections through proper configuration, definition of the public peers should not limit to ones who own a public IP address. A public peer here could be 1) a peer owns a public IP address and the incoming connections are not blocked by a firewall, or 2) a peer adopts proper port forwarding configuration while it works behind the NAT. There are many ways to make a peer able to accept incoming packet even the peer works behind the NAT. This can be done through proper configuration, which is setting the port forwarding rules. The port forwarding rules tell the NAT how the incoming packets forwarded to the private clients. UPnP [19] established the protocols for users to configure the port forwarding rules. Also, the user may manually configure the port forwarding rules.

Even if there exists mechanisms that turn more private peers into the public peers. However, it is possible that a public peer may refuse to accept incoming connections at anytime and thus become a private peer. For ensuring capability of the peers, the repetition of the detection process is needed.

3.1.2 Public peer detection process & its protocol

The public peer detection process aims to detect if the peer is able to accept incoming packets. The process is performed when a peer is receiving a data request. We term the peer who sends the data request as the child peer, and the peer who receives the data request as the parent peer.

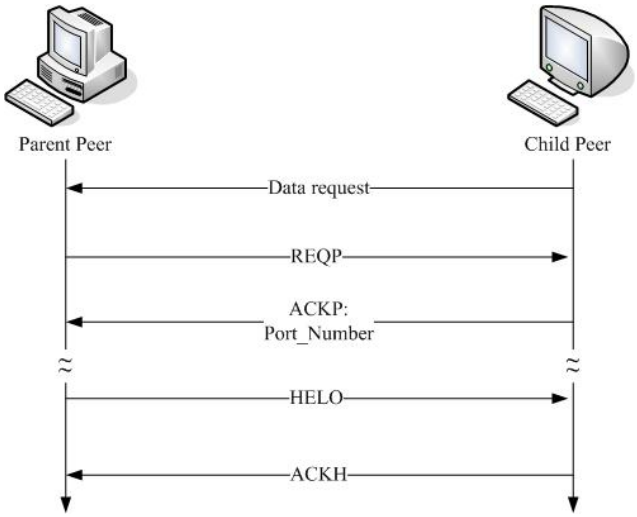


Fig. 3-1. Public peer detection process protocol

The detection process consists of messages that are requesting a listening port and confirming the availability of the port. Four messages are defined in the detection process. Peers indentify the messages with the messages name. The following explains the messages used in the detection process:

- REQP: The parent peer requests a listening port from the child peer by performing REQP. The child peer opens an arbitrary listening port upon it receives the REQP message. This method is initiated when the parent peer receives data request.
- ACKP: ACKP is the acknowledging message to the REQP. It contains the information of the listening port number which is generated by the child peer. Normally, the port number will be an integer within 0-65535. When the child peer fails to initiate listening socket, the port number will be filled with -1.

- HELO: HELO is used to test the availability of the port requested in the REQP. It initiates a new session with the child peer, sends a HELO message, and waits for the response from the child peer.
- ACKH: ACKH is used to confirm the availability of the listening port. Corresponding to HELO method, the child peer acknowledges the parent peer with the ACKH message. The parent peer then confirms the child peer as a public peer upon receiving the ACKH message.

An example of the detection process flow is shown in Fig. 1. The process starts when a peer sends the data request to another peer. The parent peer first sends REQP message to the child peer for requesting a listening port. The child peer then initiates the listening socket and responds the parent peer with the ACKP message. The parent peer checks if the port number in the ACKP message is legal or not. The legal port number will be located in the range of 0-65535. Otherwise, the detection process terminates and the child peer will not be identified as a public peer. If the process continues, the parent peer tests the port number provided in the ACKP message. The parent peer initiates a new session and sends the HELO message to the child peer. If the child peer receives the HELO message, it will respond with an ACKH message. The detection process terminates when the parent peer receives the ACKH message timely. Note that the child peer may works behind a NAT, which drops incoming packets silently. Therefore, the

parent peer sets a timer for the responding message. If the HELO message is not responded timely, the detection process terminates and the child peer is identified as a private peer. To make a brief conclusion, a child peer will be identified as a public peer if the port number in the ACKP message is legal and the ACKH message is timely responded.

3.1.3 Priority request queue

In addition to the public peer detection process, we prioritize the data requests. The data requests from public peers hold higher priority than those from the private peers. Each peer prepares two queues for storing the data requests from the public peers and from the private peers respectively. In our mechanism, a peer serves the data requests from the public peers before serving the data requests from the private peers. In other words, a peer checks the queue for the public peer first when it is able to serve data.

We may make the public peers preempt the ongoing transmission of private peers. However, transmission preemption, which may lead to data starvation, is not adopted in our mechanism. The data starvation may occur and leads to performance degradation. If the preemption is adopted in such a system, the public peers may join the network and send data requests in any time. In this case, private peers may keep waiting for public peers to complete their transmission. This is obviously a data starvation and is negative to the P2P system performance.

3.2 ISP-Friendly Scheduling

The ultimate goal here is to build connections with partners located in the ISPs that do not charge for the cross-ISP traffic. However, the economics of ISP relationship remains unclear due to the ISPs are not willing share the relationship among them. That makes people do not know how the connections incur payment. To reduce the crossing boundary payment, the best strategy here is make all the connection restricted in the same ISP [16]. Whereas such a restriction to the connections degrades the performance, we are seeking an alternative solution.

Our ISP-friendly mechanism reduces cross-ISP traffic by considering the ISP origin while performing data scheduling. In the rarest-first segment scheduling scheme, it is possible that multiple rarest segments exist. In this situation, a peer randomly chooses a segment from the rarest ones and thus results in the traffic randomness. We suggest that a peer to schedule a data source with considering the ISP origin of the data source. Besides, a peer should schedule a transmission with the domestic peers as much as possible. This can be done by sorting the partner list before scheduling. Table 1 show

TABLE I. THE PROBABILITY FOR FINDING A LACKING DATA SEGMENT IN A PARTNER LOCATED IN THE SAME ISP

	File sharing system	Streaming system
Probability	95.8%	59.5%

the probability of seeing a lacking data segment that is hold by a domestic partner while performing scheduling procedure in our simulation. Theoretically, the cross-ISP traffic can be reduced to 4.8% for a P2P file sharing system and 40.5% for a P2P streaming system. However, the available bandwidth sets the limit for the transmission. Though the proposed mechanism may reduce the cross-ISP traffic, the question remains is whether such a peer selection scheme will also benefit the users. Since the transmission capability remains unchanged, the performance of the system should not be changed. Simulations are conducted for further verification.

3.2.1 Identification of domestic peers

We use the nslookup service for querying the ISP information of the peers by the IP address. Nslookup service was originally designed for querying the ownership information of the IP address. It also can be used to query the ISP origin of the IP address. When we use the nslookup service, the IP address is sent to the nslookup service server and the information of the IP address returns. One of the information is the netname, which is the abbreviation of the ISP. The abbreviation of each ISP is unique and can be used as the identifier of the ISPs. Therefore, the domestic peer is identified by comparing the netname of the peer.

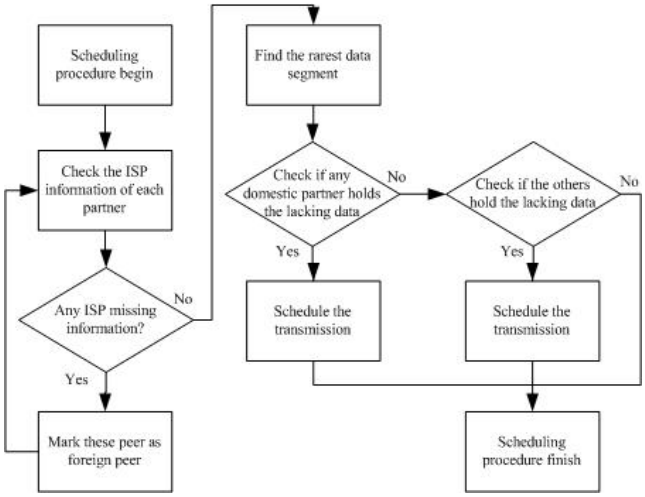


Fig. 3-2. ISP-friendly scheduling algorithm

3.2.2 ISP-aware rarest-first segment scheduling

Our scheduling scheme is modified from the state-of-the-art rarest segment first scheduling scheme. The rarest-first scheduling sorts the number of the segments and schedules the transmission for retrieving the rarest ones. This scheme guarantees high diversity of the segments. In particular, it prevents reappearance of the last segment problem and of the rare segment problem. However, our observation shows that there are multiple rarest segments in the network and the transmission is scheduled to randomly retrieve one of them. While some of these rarest segments may be retrieved within the same ISP, we can make the traffic more ISP-friendly by requesting these segments first.

We propose an ISP-aware rarest-first scheduling scheme which schedules the domestic transmission first. The algorithm is illustrated in Fig. 2. In our method, the

scheduling procedure begins with discriminating the domestic peers. Then we sort the segment list by the number of the segments to find the rarest segment. After the sorting, the peer sequentially asks the domestic partner for the lacking data segments. The transmission will be scheduled if any lacking data segment exists. It is possible that the domestic partners do not have the data segments requested. In this situation, cross-ISP traffic occurs for retrieving these data segments. However, other peers may request the peer who just made cross-ISP traffic for the lacking data and reduce the cross-ISP traffic.

Chapter 4 Simulation

The author has conducted a series of simulations to evaluate how scheduling policies, public-first delivery and ISP-friendly scheduling affect download time for P2P file sharing and video continuity for P2P multimedia streaming system. The simulations are round-based in that, for each round, every peer completes the scheduling computation and data downloading at once.

4.1 Simulation Model

In the simulation, unless otherwise specified, every peer will perform the following actions in every single round: (1) Receiving buffer map of other peers, (2) Sorting the segment availability and request data by rarest first, (3) Randomly choosing a data source from those who holds data segment needed, (4) Respond incoming requests if any.

4.2 Effect of Public-First Delivery

To examine the efficiency of the public-first delivery policy, we implement the FIFO policy for comparison, in which, the data requests will be processed based on the time. Using the public-first policy, a peer will process the data requests from private peers only after requests from public peers are fulfilled.

To evaluate the file download time for a P2P file sharing system, we configure the

simulations as follows. There are 101 peers in the simulation, including a source peer and 100 peers to download a file from the source peer. All peers stay on throughout the simulations. The file is divided into 200 segments. The source peer is capable of serving 10 segments each round. The other peers are capable of serving 3 segments and requesting 6 segments each round. Every peer is connected with every other peer. Before a peer requests a segment, the peer solicits the segment availabilities from other peers and determines the segment to request next based on the above mentioned rarest first policy. When a peer is requesting a segment, it will search by the rarest first order and ask its neighbors sequentially. This makes the peer may not fetching the rarest segment when it holds all the rarest ones. However, this is close to how the real peer-to-peer transmission systems work.

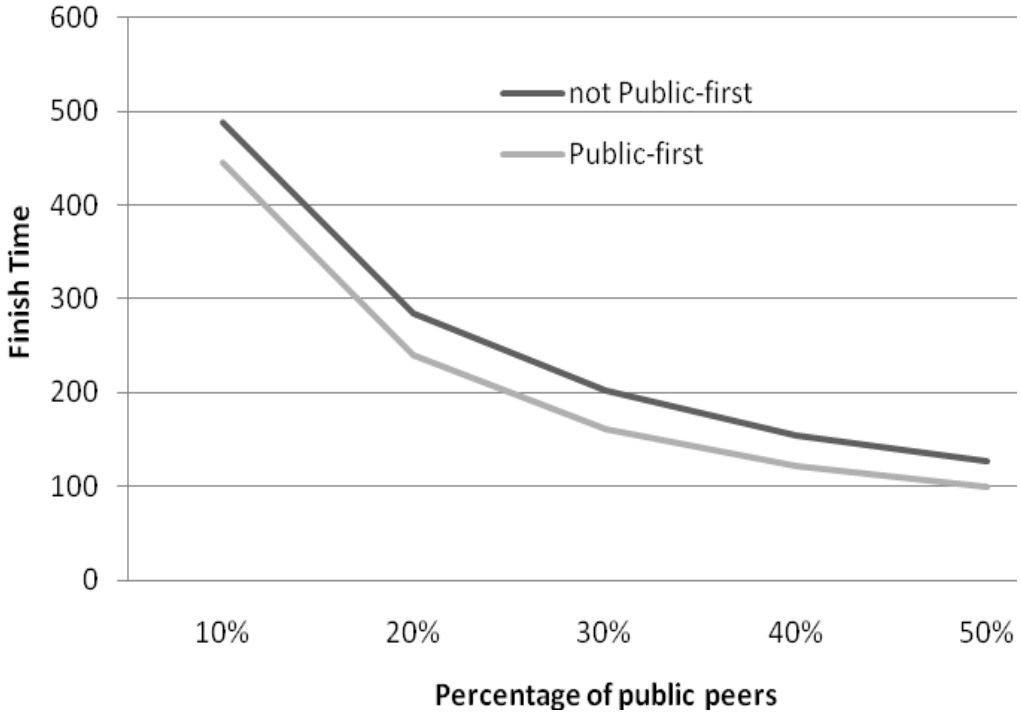


Fig. 4-1. Performance plot of peer-to-peer file sharing system simulation for different

The performance metric in this simulation is the elapsed time to complete the file transmission. We vary the percentage of public peers in the network and see how the proposed policy scales to the amount of public peers in the network. **Fig. 4-1.** is the simulation result.

Before a peer requests a segment, the peer solicits the segment availabilities from other peers and determines the segment to request next based on the above mentioned rarest first policy. **Fig**. 4-1 shows the performance of P2P file transmission using public-first policy vs. not using the public-first policy. Since public-first delivery policy has better utilization on upload bandwidth. We can see the public-first policy reduces

elapsed rounds by about 20%. The benefit of public-first policy is significant even when the percentage of public peers is low. The performance improvement comes from the public-first makes data segments distributed effectively.

To evaluate the continuity of the video stream, we extend the setting to capture operations of a P2P multimedia stream broadcasting system. Unlike P2P file transmission systems, peers in the multimedia streaming system will not hold entire downloaded file. Peer will, usually, maintain a buffer that stores partial streaming data. With streaming buffer, peer will not store entire file that may take hundreds of MB.

A larger amount of buffer benefits the quality of video playback. We have configured in this set of simulations that every peer holds a 60 segments buffer space. Note that the video is divided into segments and each segment contains one second of video. When peer begins playing, it waits 20 rounds of initial buffering. The simulations run for 1000 rounds. Other settings are like those in the file sharing simulation. Segment continuity is considered the main performance metric in the simulation. Segment continuity, which is called continuity index in [6], indicates the percentage of timely received data segments. A higher score means a better video streaming quality. **Figure 4-2** has shown the simulation result of a peer-to-peer streaming system. We may see that the public-first delivery does not benefit in terms of video segment continuity index. In **Figure 4-3**, the average buffer utilization shows the

public-first delivery benefits buffer utilization. However, the average continuity index doesn't improve much. More simulations were conducted for explanation of this phenomenon.

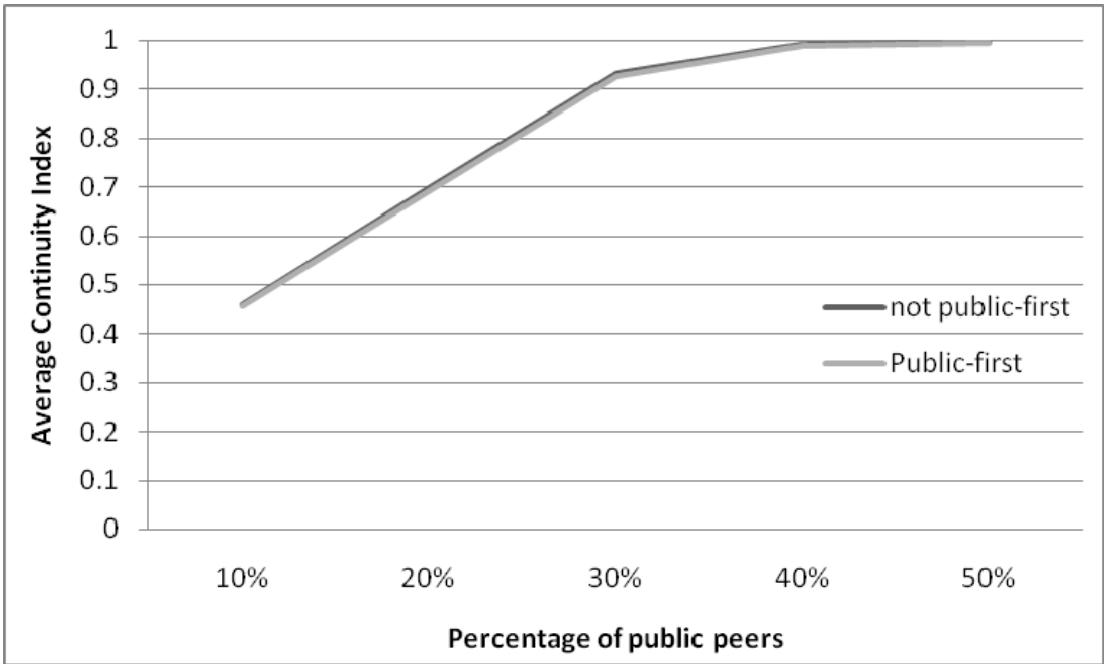


Fig. 4-2. Performance plot of peer-to-peer streaming system simulation for different percentage of public peers

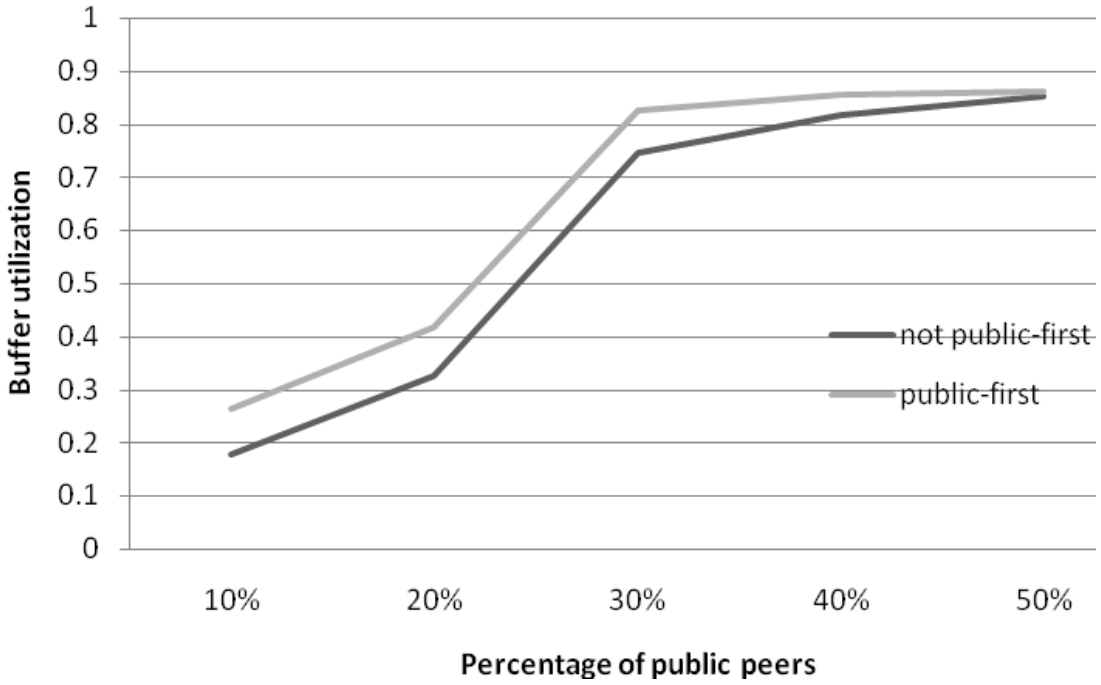


Fig. 4-3. Average buffer utilization of peer-to-peer streaming system simulation for different percentage of public peers in the network

It might be puzzling that the fact of using public-first delivery or not does not benefit peer-to-peer streaming systems performance. The reason comes from how the peer-to-peer system performs scheduling. In this simulation, when a peer is scheduling a data source, it will search its neighbors they hold the segments it lacks in a rarest first fashion. That makes the transmission happens when the requested peer has the ability of uploading a segment and holds the segment. In the previous simulation, the buffer size is 60 segments, which is large enough for a requesting peer finds a segment that it lacks in. That makes the performance bottleneck is only the available upload bandwidth, which remains unchanged in both scenarios.

To illustrate this, we may think of how the peer-to-peer networking transmission works. The transmission in peer-to-peer networking systems fails when the data provider does not has the upload bandwidth or the requested data. Using public-first delivery make public peers has better chance of getting data. However, the performance is limited by available upload bandwidth in the system.

The author has conducted simulations that are illustrative to how the public-first delivery affects the peer-to-peer streaming system performance. The reason of having identical continuity index comes from how the scheduling works. The peer will request a segment which is short of. The requesting peer can always find a deficient segment that is hold by another public peer when the buffer size is large enough. That is, when the buffer size is small, the video segment cannot be distributed effectively. When the public peer cannot receive video before the segment is shifted out of the buffer, it obviously cannot distribute the segment. Public-first, by definition, sends video segment to public peers first. That makes public peers able to contribute their bandwidth as soon as possible.

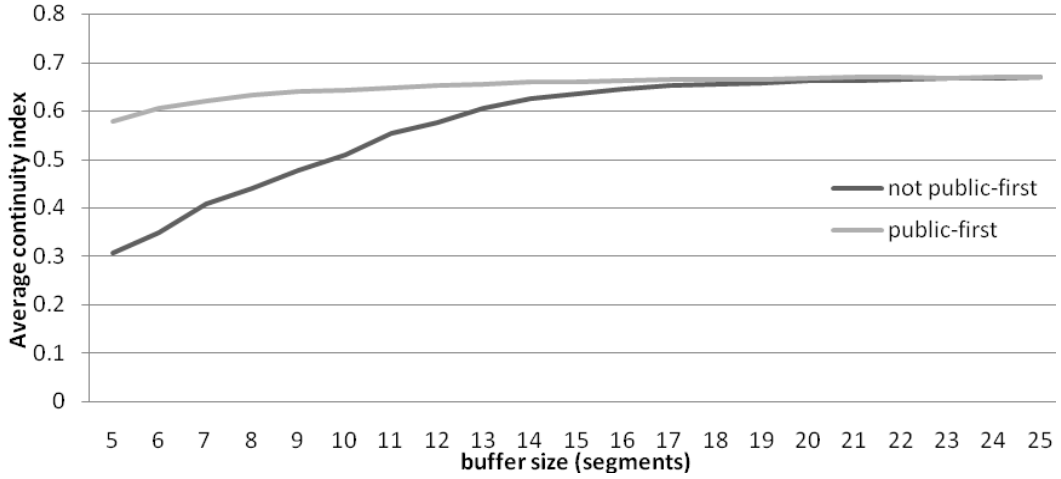


Fig. 4-4. The effect of varying buffer size

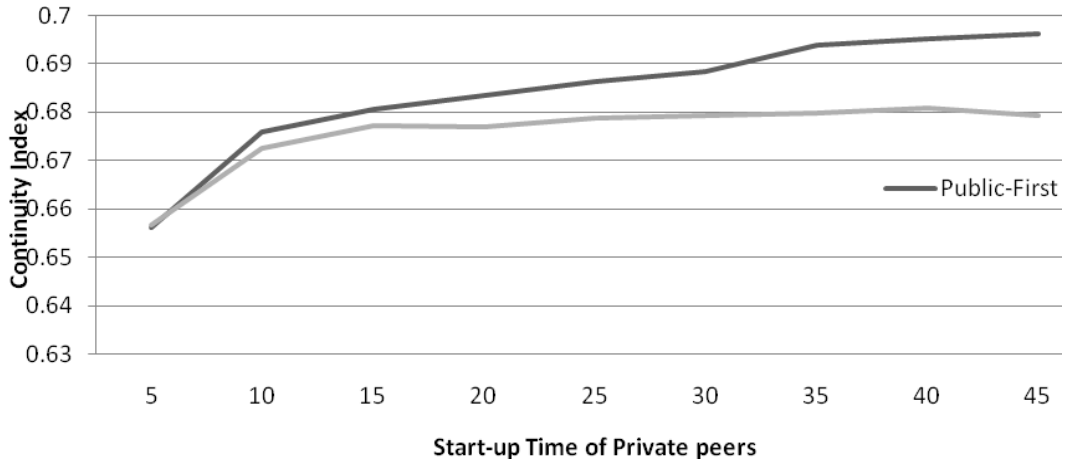


Fig. 4-5. Performance plot of peer-to-peer streaming system simulation for different start-up time of private peers in the network.

Hence, using public-first would not suffer from the performance degradation brought by smaller buffer size. **Figure 4-4** has shown the effect of varying the buffer size.

In this buffer size varying simulation, the percentage of the public peers is 20%. Previous simulation result shows that when the buffer size is large (60 segments), the

continuity index is around 0.68. Reducing buffer size results in continuity index decreasing. We may see the continuity index drops dramatically when the buffer size shrinks. The reason comes from that it takes time for public peers to get the data and forward to others. Since the public-first delivery sends the data to the public peers first, the public peers may contribute their bandwidth as soon as possible. This makes performance difference in terms of continuity index.

The author cannot stop wondering if there are other scenarios that the public-first benefits peer-to-peer streaming systems. In the previous peer-to-peer file sharing system simulation, public-first outperforms 20% in terms of file transmission time. That may indicates if the sliding window is not moving, public-first delivery has better transmission efficiency. Inspired by this, the author has conducted simulation that is changing the start-up time of private peers.

In **Figure 4-5**, the start-up time is changed to see if the public-first benefits. In this simulation, the start-up time for public peers is fixed to 5 rounds and the start-up time for private peers varies. The buffer size for each peer is 60 segments. The percentage of public peers is 20 percent. The goal for this simulation is to see if delay the start-up time of private peers benefits.

In this simulation, the performance of using public-first delivery is better by around 3-5% in terms of continuity index. The reason for this phenomenon comes from

the public-first delivery has better delivery efficiency while the sliding window is not moving. Due to the start-up delay is relatively small to the entire streaming playback time, the improvement is relatively small.

4.3 ISP-Friendly Scheduling

For this set of simulations, we assumed that there are 4 ISPs and 25 peers in each ISP. 12 out of the 25 peers in each ISP are the public peers and the others are the private ones. The source peer is not categorized in any ISP. To simulate ISP-friendly scheduling, every peer will sort the neighbor by the ISP information. Other simulation settings remain the same as the previous sets of simulations. Traffic between ISPs is logged to see if the traffic is well-controlled.

Traffic between ISPs is logged in the tables below. Numbers in the tables represents the number of transported segments. The first row stands for the segment destination ISP and first column stands for segment source ISP. **Figure 4-6** represents the traffic flow between ISPs in the simulation while not using ISP-friendly scheduling. We may see that the traffic is fairly distributed. While using ISP-friendly scheduling, in **Figure 4-7**, log shows the traffic is mostly concentrated in the same ISP. The performance metric, which is the elapsed time for downloading a file, shows the ISP-friendly scheduling does not affect performance. Using the ISP-friendly scheduling takes 130.3 rounds for completing a file transmission in average and not using that takes

130.1 rounds.

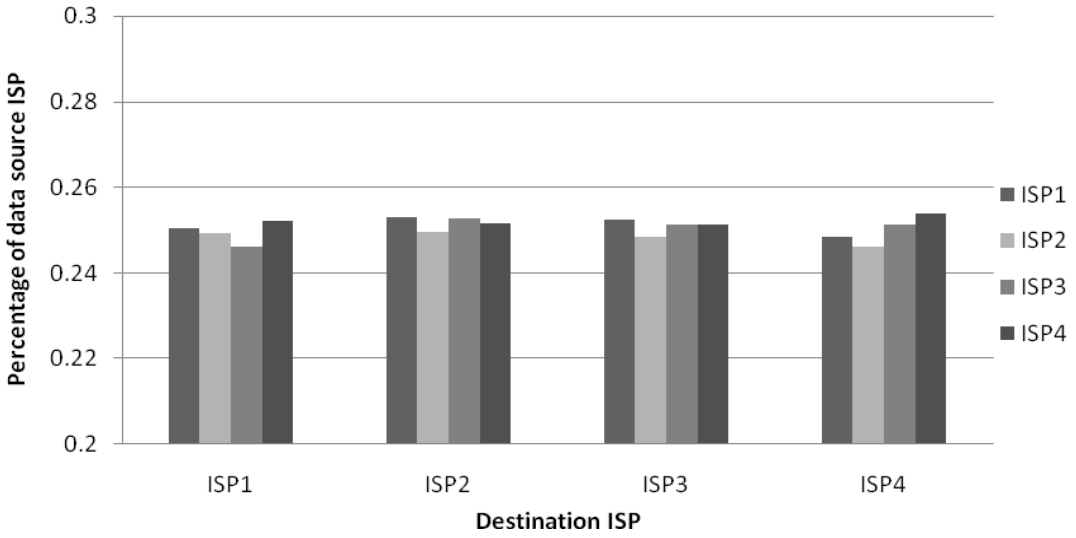


Fig. 4-6. Traffic between ISPs not Using ISP-Friendly Scheduling in a Peer-to-peer File Sharing System Simulation

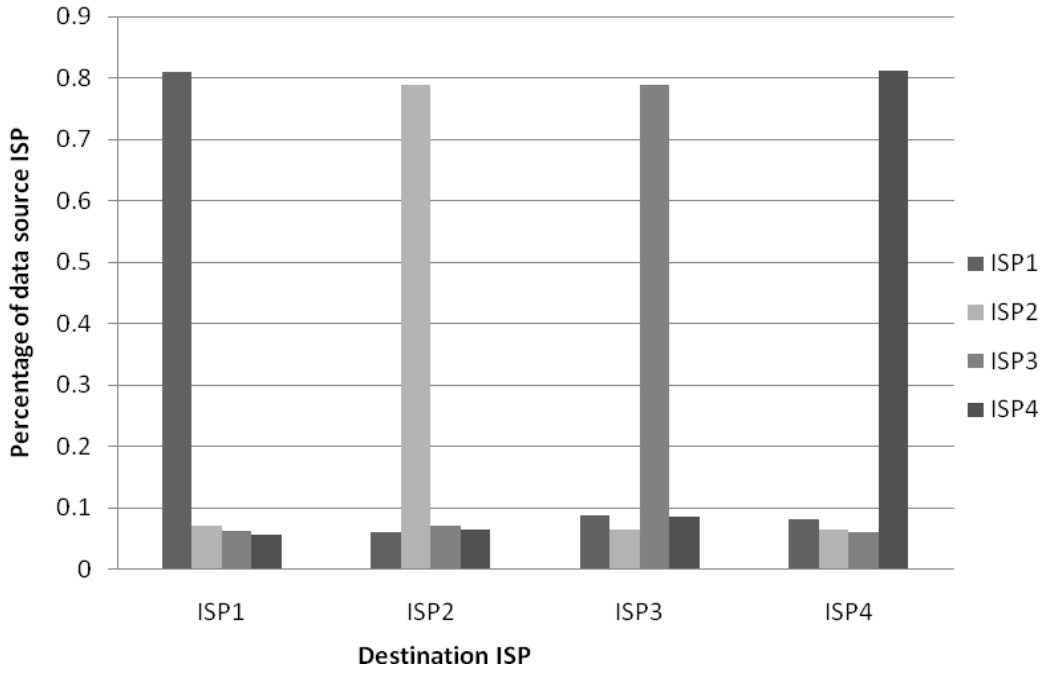


Fig. 4-7. Traffic between ISPs Using ISP-Friendly Scheduling in a Peer-to-peer File Sharing System Simulation

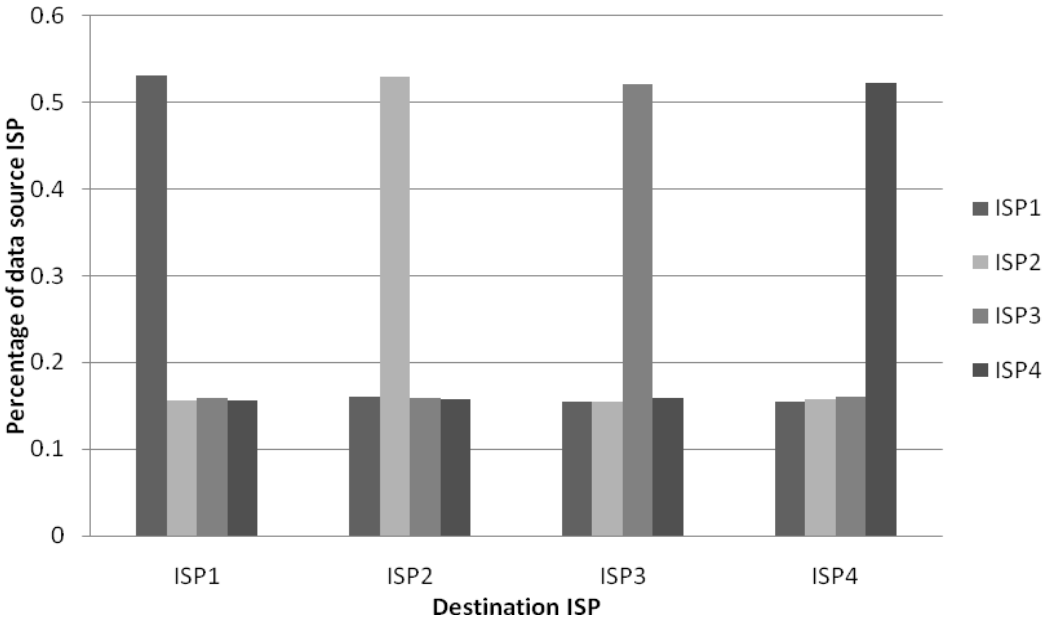


Fig. 4-8. Traffic between ISPs using ISP-friendly Scheduling in a Peer-to-peer

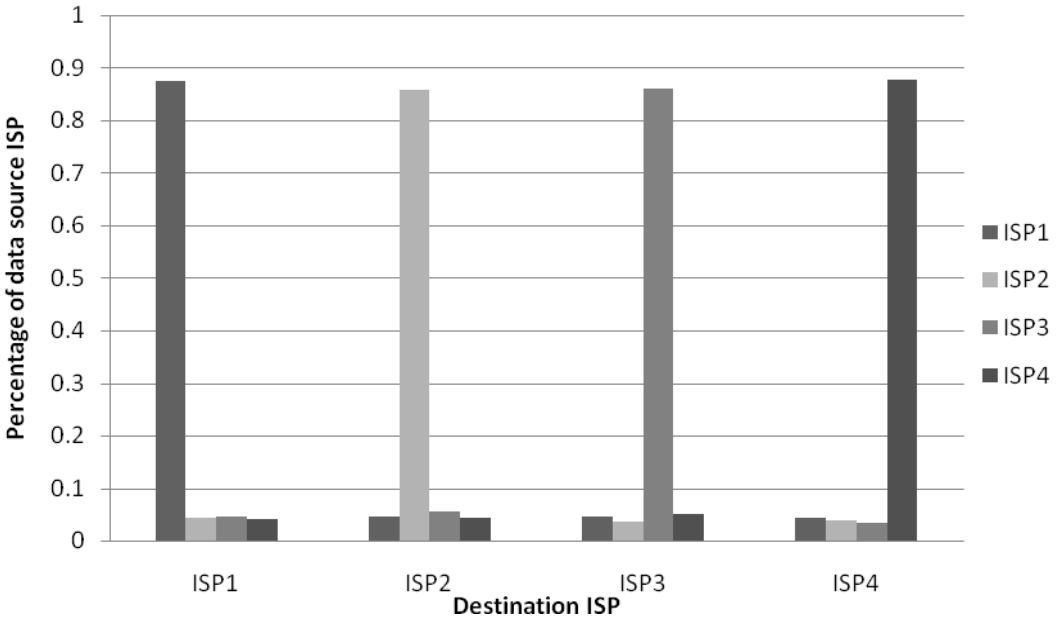


Fig. 4-9. Traffic between ISPs Using ISP-friendly Scheduling and Public-First Delivery in a Peer-to-peer File Sharing Simulation

4.4 Effect of ISP-Friendly with Public-first Delivery

The combination the segment decision scheme, rarest first, and the data source decision strategy, the sorted neighbor list, is how we deal with reducing cross-ISP traffic problem. The rarest first scheme makes peers within the same ISP having better chance to get a complete copy of the file. The sorted partner list will reduce sending the data request to peers in the other ISPs. Since rarest first scheme makes peers in the same ISP have better chance to get a complete copy of file, peers don't have to send request to peers in the other ISP for lacking data segments.

Simulation for ISP-friendly scheduling for P2P streaming systems is also

conducted. In this simulation, we have the same ISP geographic setting in the previous file transmission simulation. Besides, peers will sort the partner list by the ISP information before requesting data. The simulation result shows most of the segment transmission is bounded in the same ISP. The segment continuity shows similar performance with the simulation that is not using ISP-friendly scheduling. Simulation result is shown in **Figure 4-8**.

We had simulation for the mixture for public-first delivery and ISP-friendly scheduling. In this simulation, peer requests data from the peers in the same ISP first and processes requests from the public peers first. **Figure 4-9** shows simulation result on file sharing model.

With the combination of public-first delivery and ISP-friendly scheduling, we may see the cross-ISP traffic is further reduced. Since public-first delivery makes public peers getting data sooner, the peers in the same ISP would be able to request deficient data from the public peers in the same ISP.

Chapter 5 Conclusion

In this thesis, the author presented simulation results and implementation of the public-first delivery policy and ISP-friendly scheduling for the P2P systems. Public-first delivery improves P2P system performance and ISP-friendly scheduling reduces cross-ISP traffic without sacrificing performance. The public-first delivery improves P2P system performance by making public peers getting the data and allowing the public peers helping disseminating data to other peers earlier. The simulation result shows public-first delivery policy improves performance of P2P file sharing systems by 20%. The simulation result also shows how the performance scales to the percentage of public peers in the P2P systems.

The author also provided the simulation result of how the public-first delivery and the buffer size affect the performance of peer-to-peer streaming systems. The performance of peer-to-peer streaming system is highly relative to buffer size. The performance of a peer-to-peer streaming system degrades when the buffer size is small. With public-first delivery, peer-to-peer streaming system shows low performance degradation while the buffer size is small. The reason comes from that the public-first could make use of the upload bandwidth effectively. Simulation results are provided for seeing how the buffer size affects the continuity index.

Another topic in this thesis is the ISP-friendly traffic control mechanism. To achieve this, each peer tends to acquire data from peers within the domestic ISP. Our paper suggests a simple and distributed method that can be easily implemented in P2P systems. The simulation results show the cross-ISP traffic is reduced without sacrificing performance of P2P systems. Moreover, we combine the public-first delivery and ISP-friendly scheduling policy in our simulation. The simulation result shows the combination of two policies further reduce the cross-ISP traffic.

REFERENCE

[1] S. Agarwal *et al-* , “Performance and quality-of-service analysis of a live P2P video multicast session on the Internet,” in *Proc. of IEEE IWQoS,* 2008.

[2] STUN, RFC 3489.

[3] J. Li and K. Sollins, “Exploiting autonomous system information in structured peer-to-peer networks,” in *Proc. of The 13th. IEEE Int. Conf. on Computer Communications and Networks (ICCCN2004)*, Chicago, IL, Oct. 11-13, 2004.

[4] H. Xie *et al-* ,”P4P: Portal for (P2P) Applications,” in *Proc. of ACM SIGCOMM* Aug. 2008.

[5] D. Choffnes and F. Bustamante, “Taming the torrent: a practical approach to reducing cross-ISP traffic in P2P systems”, In *Proc. of ACM SIGCOMM,* Aug. 2008.

[6] 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*, Mar. 2005.

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

[8] A. R. Bharambe *et al-*, “Analyzing and improving a BitTorrent network’s

performance mechanisms," in *Proc. of IEEE INFOCOM*, 2006.

[9] V. Aggarwal, A. Feldmann and C. Scheideler, "Can ISPs and P2P users cooperate for improved performance?," in *Proc. of SIGCOMM Computer Communication Review*, vol. 37, no.3, pp. 29–40, 2007.

[10] Y. Huang *et al*-, "Challenges, design and analysis of a large-scale P2P-VoD system," in *Proc. of ACM SIGCOMM*, 2008.

[11] B. Cohen, "Incentives build robustness in BitTorrent," in *Proc. of Workshop on Economics of Peer-to-Peer Systems*, 2003.

[12] S. Guha and P. Francis. "Characterization and measurement of TCP traversal through NATs and firewalls," in *Proc. of Internet Measurement Conference (IMC2005)*, Oct. 2005.

[13] A. Madhukar and C. Williamson "A Longitudinal Study of P2P Traffic Classification," in *Proc. of IEEE MASCOTS*, Sept. 2006.

[14] S. Ren, L. Guo and X. Zhang, "ASAP: an AS-Aware Peer-Relay Protocol for High Quality VoIP," in *Proc. of The 26th Int. Conf. Distributed Computing Systems (ICDCS2006)*, July, 2006.

[15] R. Bindal *et al*- ,"Improving Traffic Locality in BitTorrent via Biased Neighbor Selection," in *Proc. of The 26th Int. Conf. Distributed Computing Systems (ICDCS2006)*, July, 2006.

[16] “Global Consumer Internet Traffic,” 2008, http://www.nytimes.com/imagepages/2008/03/13/business/20080313_NET_GRAPHIC.html

[17] Cheng Huang *et al*- ,”Can Internet Video-on-Demand be Profitable?,” in *Proc. of SIGCOMM’07,* August, 2007, Kyoto, Japan.

[18] PPLive, Online: http://www.pplive.com

[19] PPStream, Online: http://www.ppstream.com