Peer-to-Peer Video-on-Demand System Based on Video-Data Popularity

February 2013

Rei Endo

主　論　文　要　旨

報告番号	㊙乙　第　　号	氏　名	遠藤　伶

主論文題目：

Peer-to-Peer Video-on-Demand System Based on Video-Data Popularity

(動画データの普及度に基づいたピアツーピア動画配信システムに関する研究)

本研究は，動画データの検索や送信を視聴者端末（ピア）間で分散的に行うことで動画配信ノードの配信負荷を低減する，ピアツーピア動画配信システム（P2P VoD システム）の性能向上を目的としたものである．P2P VoD システムは，動画データの検索や送信をピア間で分散的に行うため，動画データ普及度の偏りによりシステムの性能低下や動画配信ノードの負荷増加がおこる．本研究では，P2P VoD システムの主要な機能である，ピア検索，動画データ送信スケジューリング，それぞれについて検討し，動画データが Peer-to-Peer（P2P）ネットワーク中でどれだけ普及しているかを表す，動画データ普及度に着目した新しい方式を提案した．

ピア検索については，P2P ネットワーク上の局所的な普及度をもとに推定したデータ普及度を用いるローカル普及度ベース複合型検索手法 HyDiff を提案した．複数の検索モデルの使い分けに，データの要求度をもとに推定した普及度を利用する既存方式では，適切な普及度推定間隔を決めることが難しく，また，適切な推定間隔を設定できない場合，推定普及度と実際の普及度の差が大きくなり，検索効率が低下することがある．一方，本研究の提案方式では，P2P ネットワーク上のデータの局所的な普及度から求めた推定普及度を用いる．局所的な普及度をもとにした本方式では，推定普及度の精度が普及度推定間隔に対し線形に変化するため，普及度推定間隔の設定が容易になる．さらに，実際の普及度と推定普及度の誤差が小さくなるため，検索モデルをより適切に使い分けることができる．シミュレーション評価により，提案方式の有用性を応答率，応答時間，維持コストなどの点から確認した．

動画データ送信スケジューリングについての検討では，新規参加ピアに普及度の低いデータ・ピースの受信権を貸し付ける仕組みを導入したピース・レンディング方式を提案した．一般の P2P VoD システムでは，ピアに積極的なデータ送信を促すため，ピアごとのデータ送信量に応じた量だけデータ受信を許可する，報酬付けアルゴリズムを用いる．しかし P2P VoD システムでは，動画の先頭部分を構成する断片データの普及度が高くなるため，既存の報酬付けアルゴリズムを用いると，サービス参加直後のピアは動画配信ノードの補助が無ければピース交換に参加できない．一方，本研究の提案方式では，新規参加ピアはシステムに貸し付けられた普及度の低い断片ピースの受信権を使い，一定の期間ピース送信なしに普及度の低いピースを受信できる．これにより，新規参加ピアもピース交換に容易に参加でき，ピアの送信帯域利用率が向上する．シミュレーション評価により，提案方式の有用性をピア送信帯域利用率，再生開始遅延などの点から確認した．

SUMMARY OF Ph.D. DISSERTATION

School Science for Open and Environmental Systems	Student Identification Number 81046658	SURNAME, First name ENDO, Rei

Title

Peer-to-Peer Video-on-Demand System Based on Video-Data Popularity

Abstract

Peer-to-Peer (P2P) Video-on-Demand (VoD) systems are widely researched for the realization of low-cost VoD services. This dissertation discusses methods for peer search and video data transmission scheduling on P2P VoD systems, and proposes novel methods for these two functions with a focus on video-data popularity. Video-data popularity is defined as the data's diffusion ratio in the P2P network and it significantly impacts the efficiency of P2P VoD systems.

This dissertation proposes HyDiff, Hybrid search based on Diffusion ratio for peer search. HyDiff uses search-target popularity estimated by local popularity for choosing search strategy. Existing hybrid search methods use the request rate to estimate search-target popularity. However, it is difficult to retain high efficiency because a large gap between the measured request rate and popularity exists in some cases. The proposed method bases its popularity estimation on local popularity. As a result, the search efficiency is expected to be retained in the proposed method. Efficiency of the proposed method in terms of the response rate, response time, and maintenance cost is evaluated by simulation.

This dissertation also proposes Piece Lending for video data transmission scheduling. Piece Lending adopts a lending approach in which the P2P system lends receiving rights for low-popularity pieces to a newly joined peer, and then later collects contribution from the peer in compensation for using rights after the peer have obtained enough pieces. In existing methods, low-popularity piece pushing from the video provider node is required for newly joined peers. In the proposed method, P2P systems provide receiving rights for low-popularity pieces to newly joined peers. They use these receiving rights to receive low-popularity pieces from other peers without support from the video provider node. Efficiency of the proposed method in terms of the utilization ratio of peer transmission capacity and start-up latency is evaluated by simulation.

Peer-to-Peer Video-on-Demand System Based on Video-Data Popularity

Rei Endo

A dissertation submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

School of Science for Open and Environmental Systems
Graduate School of Science and Technology
Keio University

February 2013

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Video-on-demand (VoD) systems have been researched widely since the 1990s [1–3]. The Digital Audio Visual Council (DAVIC) was founded in 1994 to promote and develop a protocol for standardization of VoD systems. In the 1990s, however, high operating costs of video data transmission networks and video provider nodes pushed VoD service usage fees well above those of video rental services [4–6]. Those costs are much lower today, allowing many users to utilize VoD services; one of the most popular VoD services currently has approximately 800 million users per month [4].

Commercial Internet-based VoD systems have high operating costs associated with the server clusters they require [7]. The content provided by most of the large VoD services includes videos that were commercially produced, as well as those created by users. Content produced and provided by end-users is called user-generated content (UGC), and many services that provide UGC have become quite popular. A characteristic of well-known UGC services is that they offer very large amounts of content [4, 8]. There exist various types of UGC, including videos, music, graphics, and dictionaries. Of these, video files tend to be comparatively large, and therefore place heavy loads on hosting servers. This drives server and network operating costs up, limiting the quality of service provided to VoD system users.

VoD server loads are often high because systems are based on a client-server model,

where servers hold all information related to content and users. This makes systems highly manageable and easily modified. In such systems, video provider nodes send all video content data to users, meaning servers must be relatively high-performance, and thus expensive. Moreover, video provider node performance should scale with the number of users because UGC services generally start small and grow over time. VoD service efficiency on a client-server system is greatly decreased when video provider node performance is insufficient for the number of users, but excessively high-performance systems represent wasted cost. This scalability problem is a major issue for VoD systems.

Recently, peer-to-peer (P2P) networks have attracted interest as a solution to the scalability problem [9–17]. P2P networks have been researched in relation to a wide variety of Internet applications [9–35], particularly file sharing systems [19–21] and live streaming systems [22–25]. P2P VoD systems consist of a video provider node and viewer nodes for each video. Viewer nodes not only receive video data, but also provide resources such as bandwidth, storage, and CPU cycles to the system. Even low-performance video provider nodes can therefore assist in providing high-quality video content to a large number of viewer nodes. Moreover, it is unlikely that P2P VoD system service quality will decrease when the number of users increases rapidly, because more users mean more resources available. VoD providers thus do not need to prepare high-performance video-provider nodes to cover temporary increase in the number of users. Additionally, failures in P2P VoD systems are more localized than in client-server model VoD systems; when failures occur in a video provider node, viewers can keep viewing the video by receiving video data from other viewers of the video.

P2P VoD systems are classified into *peer-assisted* VoD systems and *pure* P2P VoD systems, based on the presence or absence of high-performance servers maintained by the VoD provider. In peer-assisted VoD systems, video provider nodes are high-performance servers that serve multiple video contents to viewer nodes with the assistance of viewer nodes. Video content providers, which can be companies or end-users, put video content on central servers. In pure P2P VoD systems, video provider nodes are peers maintained not by VoD providers, but by video content providers. When the video content provider is an end-user, performance of the video provider node is limited in comparison with high-performance servers.

This dissertation assumes use of pure P2P VoD systems, because pure systems require higher performance than peer-assisted ones. This assumption also means that peers are video provider nodes or viewer nodes, and that the individual node performance is low. Pure P2P VoD systems allow end-users to provide video content using their own PCs instead of video provider nodes offered by commercial service providers. One of the motivations of using pure P2P VoD systems is allowing commercial VoD providers to deliver videos without limitations on quality.

This dissertation focuses on *video-data popularity* and the development of P2P VoD systems. Popularity refers to the number of peers possessing data, a value defined as the data's diffusion ratio in the P2P network. Video data popularity significantly impacts the efficiency of P2P VoD systems; when peers search for desired video data, popular data are easily found but unpopular data are scarce. In P2P VoD systems, video data are divided into small pieces for P2P delivery. Peers must collect all pieces of a desired video for playback. Unpopular data therefore causes many peers to suffer from slow downloading, and can prevent high-quality video playback. It is therefore important to consider video data popularity for designing effective P2P VoD systems.

1.2 Positions of This Dissertation

This dissertation discusses and proposes novel methods for peer search and video data transmission scheduling in P2P VoD systems. To view videos, peers must follow three steps and P2P VoD systems have five major functions: *network entrance search*, *peer search*, *neighbor selection*, *neighbor information gathering*, and *video data transmission scheduling* as shown in Figure 1.1.

The first step is *network participation*. A newly joined peer finds one or more participating peers in the P2P network. This first step requires a network entrance search function. The second step is *neighbor assignment*. A participating peer finds other peers that possess desired video data, and establishes communication connections with them. In P2P networks, such directly communicating peers are called *neighbors*. The second step requires a peer search function and a neighbor selection function. The third step is *video data exchange*, in which a peer gathers information about neighbors' desired video data

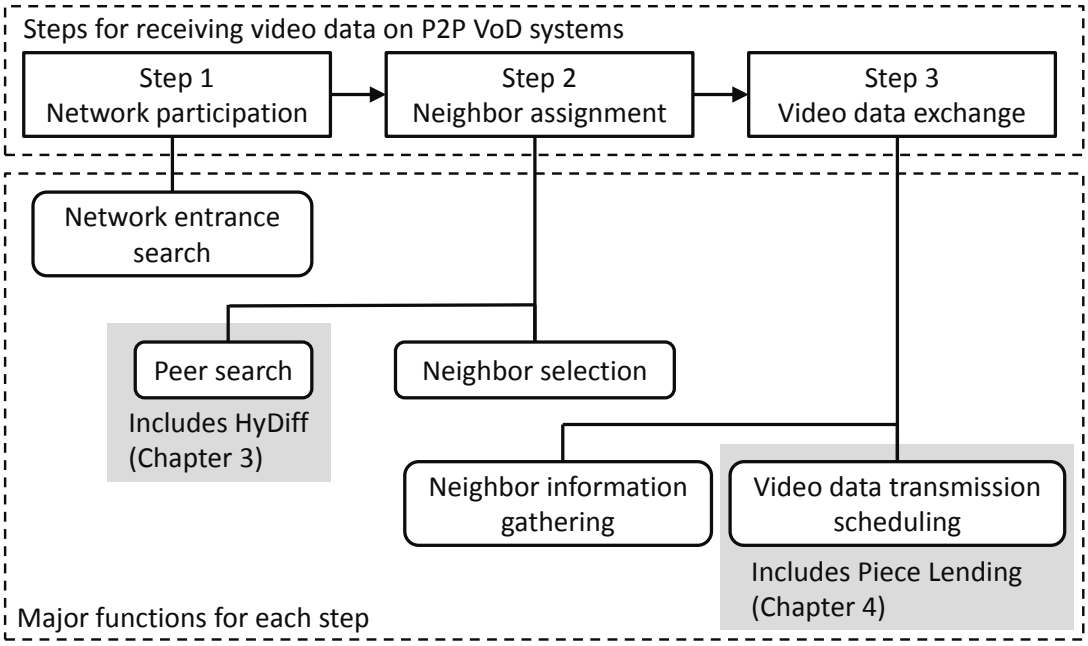

Figure 1.1: Operation flow and major functions on P2P VoD systems

to determine a sequence of video data transmission after the peer assigns its own neighbors. A neighbor information gathering function and video data transmission scheduling function are required for the third step.

This dissertation proposes novel methods for peer search and video data transmission scheduling functions, because improvement of these two functions would significantly impact on the efficiency of P2P VoD systems. Finding distributed objects such as peers or data in P2P networks is a common issue for any P2P system, and search methods for P2P networks have been widely explored [26–35]. The performance of existing methods, however, greatly decreases in some cases. Moreover, scheduling peer transmission of video data is an essential factor for the utilization of peer capacity. The higher capacity utilization is, the lower the video provider node's load is. P2P VoD systems based on poor video data transmission scheduling cannot provide sufficient performance, even when the system uses smart methods for neighbor selection and neighbor information gathering.

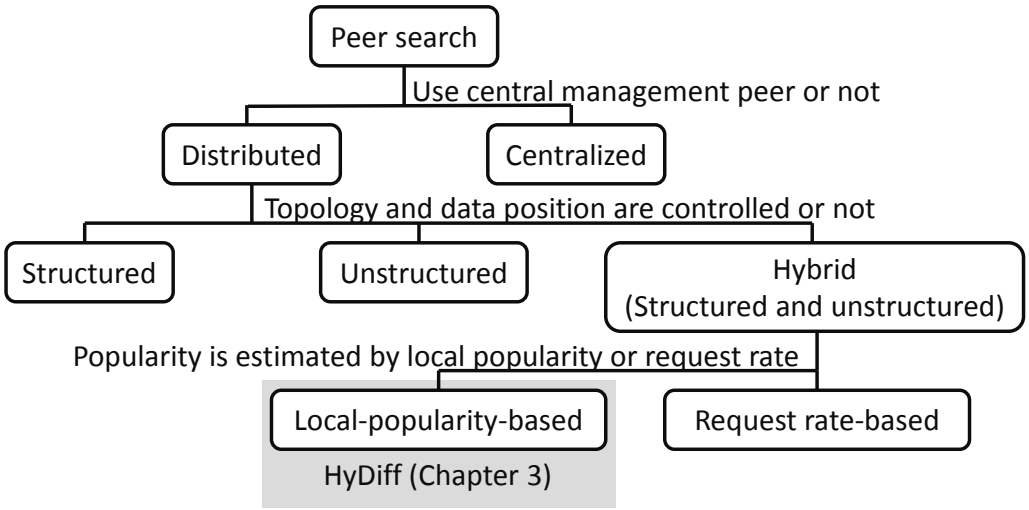

Figure 1.2: Peer search methods classification

1.2.1 Network Entrance Search

Methods for the network entrance search provide ways to join in P2P networks. In general, a newly joined peer is introduced to one or more participating peers by *bootstrappers*. Bootstrappers are particular peers provided by P2P system providers or P2P system users. Users can support this function without a complex procedure because any participating peer, which knows one or more participating peers generally, can operate as the network entrance.. In general, when the system scale is large, this function can be maintained in association with users. Many P2P systems do not pay particular attention to this function because above simple methods do not have serious problems. This dissertation also does not discuss particularly the network entrance search.

1.2.2 Peer Search

Methods for the peer search provide ways to find peers that possess particular data, and they are classified as shown in Figure 1.2. Peer search methods are grouped into two categories: *centralized methods* [36, 37] and *distributed methods* [38–53]. In centralized methods, P2P systems have a central management peer. The search efficiency of

centralized methods is enhanced when performance of the central management peer is unlimited. Search efficiency is greatly affected by a leaving management peer drops and a large number of users. Peers can generally leave P2P networks without advance notification to the system, so unexpected drops of peers with important roles must be considered. Peers joining and leaving a P2P network is called *churn*. In distributed methods, P2P systems can maintain search functions under churn, but search efficiency tends to decrease with the growth of P2P network size. Distributed search methods are classified into *structured* [38–42], *unstructured* [43–51], and *hybrid methods* [48,52,53]. In structured methods, efficiency tends to be high because network topology and data storage locations are strictly controlled, but these methods are sensitive to churn. In unstructured methods churn is unlikely to decrease search efficiency, but the efficiency of unstructured methods is lower than that of structured methods under light churn. Hybrid search methods, combinations of the structured and unstructured approaches, can achieve high efficiency and reduce the influence of churn. Hybrid search methods are classified as *request rate-based methods* [48, 52, 53] and *local-popularity-based methods*. Existing hybrid search methods are generally request rate-based methods, and in some cases it is difficult to retain high efficiency because there exists a large gap between the measured request rate and popularity.

The method proposed in this dissertation is called *HyDiff*, Hybrid search based on Diffusion ratio, and is a local-popularity-based hybrid search method. The challenge of hybrid search methods is how to select an effective method for each search target under given network conditions. HyDiff uses *local popularity* of the search target during search method selection to maintain efficiency when there exists a large gap between measured request rate and popularity. Local popularity refers to the popularity of data in a portion of peers. Conversely, popularity among all peers is called *global popularity*.

1.2.3 Neighbor Selection

Methods for the neighbor selection provide ways to decide which peers select as neighbors. Random-based, peer communication capacity-based, geographical-based and contribution-based selection methods are proposed. Generally, the neighbor selection is discussed with the video data transmission scheduling. Because the video data transmission scheduling

function is operated on the network topology constructed by using the neighbor selection function, moreover, the utilization efficiency of peers' capacity depends on the video data transmission scheduling and the utilization efficiency has big impact on the P2P system performance. This dissertation discusses based on the premise of the simple random-based neighbor selection method. In the method, neighbors are randomly-selected from peers which view the desired video. The random-based neighbor selections can keep high efficiency on average under many kinds of situation, especially, it has high tolerability for churn because the network topology is not strictly-controlled.

1.2.4 Neighbor Information Gathering

Methods for the neighbor information gathering provide ways to gather information of which neighbor possesses which data. Many P2P VoD systems use buffermap-based methods. *A buffermap* of a peer shows which parts of video data the peer possesses. Strength of buffermap-based methods is that peers are able to obtain correct information which neighbor possesses which data. However, the higher the number of each peer's neighbors is, the higher a messaging cost of buffermap-based methods. Some researchers propose low-cost neighbor information gathering methods by a limitation of gathering information. The low-cost methods decrease the messaging cost in exchange of the information's precision. The discussion of this dissertation is based on the premise of buffermap-based neighbor information gathering methods. Because the buffermap-based methods can easily stop the increasing of the messaging cost by a limitation of a maximum number of neighbors.

1.2.5 Video Data Transmission Scheduling

Methods for the video data transmission scheduling provide ways to decide a sequence of video data transmission, and they are classified as shown in Figure 1.3. Video data transmission scheduling methods are classified into *non-incentivized* [9, 11–14] and *incentivized methods* [10, 15–17, 54–58]. In non-incentivized methods, systems efficiently utilize all peer resources, but overall performance decreases in the presence of *free-riders*, peers that try to receive video data without contributing resources to the system. In general

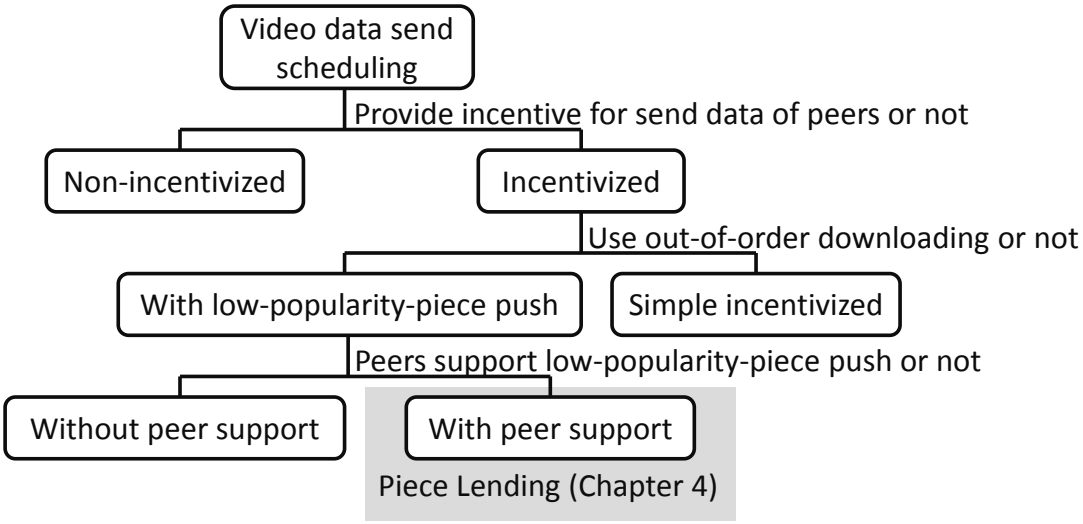

Figure 1.3: Video data transmission scheduling methods classification

video data transmission scheduling in P2P networks, the video data is divided into small *pieces*, and peers collect all pieces to obtain the whole video data. In incentivized methods, peers must contribute resources to gain the right to obtain pieces of video data, decreasing the performance losses due to free-riders since non-contributing users cannot obtain piece transmission rights. Incentivized methods include *simple incentivized methods* [10, 16, 17, 54–57] and *incentivized methods with low-popularity piece push methods* [15, 58]. Incentivized methods require peers to send pieces before receiving any, which makes it difficult for newly joined peers to join the piece exchange incentivized methods because they possess no piece to exchange. This dissertation calls that problem as *negotiation power gap problem*. *Negotiation power* means the amount of pieces that can be sent to other peers. In incentivized methods that implement low-popularity piece push, the video provider node pushes (transmits regardless of previous contributions) low-popularity pieces to newly joined peers to increase their negotiation power. Peers can generally send low-popularity piece to peers because few other peers possess the pieces. Newly joined peers can therefore join the piece exchange by using the received low-popularity pieces in incentivized methods implementing low-popularity piece push. Incentivized methods with low-popularity piece push include *methods without peer support* [15, 58] and *those with peer*

support. Existing incentivized methods with low-popularity piece push are without peer support. Newly joined peers receive low-popularity pieces from only the video provider node, increasing the load on low-popularity piece push servers. Ideally, this load should be distributed to peers.

This dissertation proposes *Piece Lending*, an incentivized method with low-popularity piece push and peer support. The proposed method implements a *lending approach* where the P2P system lends receiving rights of low-popularity pieces to newly joined peers, and collects contributions from *advanced peers* in compensation for using the right when newly joined peers become advanced peers, that are old peers that possess many pieces and thus sufficient negotiation power. The proposed method distributes the load of low-popularity piece pushing to peers using this lending approach.

Table 1.1 summarizes the problems with the existing methods and features of the proposed methods.

1.3 Contributions

This dissertation discusses methods for peer search and video data transmission scheduling on P2P networks, and proposes novel methods for these two functions with a focus on video data popularity. Peer search methods are ways to find a peer possessing needed data, and have widespread applications to key elements of all P2P systems, including VoD systems. Video data transmission scheduling methods are ways to determine a sequence of video data transmission. Video data transmission scheduling methods significantly influence the efficiency of P2P VoD systems, making them a key element in effective P2P VoD systems.

This dissertation proposes HyDiff, Hybrid search based on Diffusion ratio , which uses search-target popularity estimated by local popularity. The proposed method is a hybrid search method that acts as a structured or unstructured search method, depending on search-target popularity. In hybrid search methods, the method of estimating search-target popularity is very important because efficiencies of the structured and unstructured search methods vary depending on the search-target popularity. Additionally, the popularity of the search target changes with time and it is difficult to predict the change.

Table 1.1: Problems of existing methods and achievements of this research

Chapter	Term	Description
Chapter 3	Objective	Effective peer search under churn, large systems, and large gap between request rate and popularity
	Problem	Existing hybrid search methods are ineffective when there exists a large gap between measured request rate and popularity
	Proposal	Hybrid search based on Diffusion ratio (HyDiff) which uses the search-target popularity, estimated from local popularity
	Features	Proposal realizes effective search under situations of churn, large systems, and large gaps between request rate and popularity
Chapter 4	Objective	Effective video data transmission scheduling under situations of churn, large systems
	Problem	Existing incentivized methods need the video provider node's support for newly joined peers joining in the piece exchange
	Proposal	Piece Lending, which lends receiving rights of low-popularity data pieces to newly joined peers and collects contributions from advanced peers
	Features	Proposal improves efficiency by distributing the load of support for newly joined peers to peers

Therefore, in hybrid search, peers must calculate the popularity periodically. Existing hybrid search methods use the request rate to estimate search-target popularity. However, it is difficult to retain high efficiency because in some cases there exists a large gap between the measured request rate and popularity. For example, when peers do not request searching in a period of time, super nodes always select a structured search method in the period of time even when the actual popularity of the search target is high. Therefore, the estimated popularity by request rate does not always properly reflect the actual popularity. The proposed method bases its popularity estimation on local popularity. As a result, search efficiency is expected to be retained in the proposed method. Efficiency of the proposed method in terms of the response rate, response time, and maintenance

cost is shown by simulation.

This dissertation also proposes Piece Lending, which adopts a lending approach in which the P2P system lends receiving rights for low-popularity pieces to newly joined peers, and collects contributions of advanced peers in compensation for using rights when newly joined peers become advanced peers. The proposed method is an incentivized method with low-popularity piece push. Incentivized methods provide incentives (receipt rights) for piece sending, preventing free-riders from receiving pieces. In existing methods, low-popularity piece push from the video provider node is required for newly joined peers to join in the piece exchange, because newly joined peers do not possess transmittable piece, making their negotiation power very low. In the proposed method, P2P systems provide receiving rights for low-popularity pieces to newly joined peers, thus increasing their negotiation powers. Newly joined peers use these receiving rights to receive low-popularity pieces, allowing them to take part in piece exchange without support from the video provider node. Because most peers do not possess low-popularity piece, newly joined peers can send those pieces to peers that do not possess it. Moreover, advanced peers with high negotiation power provide their contribution to the P2P system in compensation for previously borrowing rights; in other words, they send low-popularity pieces to newly joined peers depending on the amount of pieces that they have received using the borrowed rights. As a result, the proposed method distributes the load of low-popularity piece pushing to peers. Efficiency of the proposed method in terms of the utilization ratio of peer transmission capacity and start-up latency is shown by simulation.

1.4 Organization of This Dissertation

This dissertation consists of five chapters. Figure 1.4 shows the chapter flow. Chapter 1 presented the background of P2P VoD Systems, and explained the contributions and focus of the proposed research. Chapter 2 describes functions of the P2P VoD systems and gives brief overviews of several existing methods for peer search and video data scheduling. Chapter 3 proposes HyDiff that uses data popularity estimated from local popularity. The efficiency of the proposed method in terms of response ratio, response time, and maintenance cost is shown by simulation. Chapter 4 proposes Piece Lending.

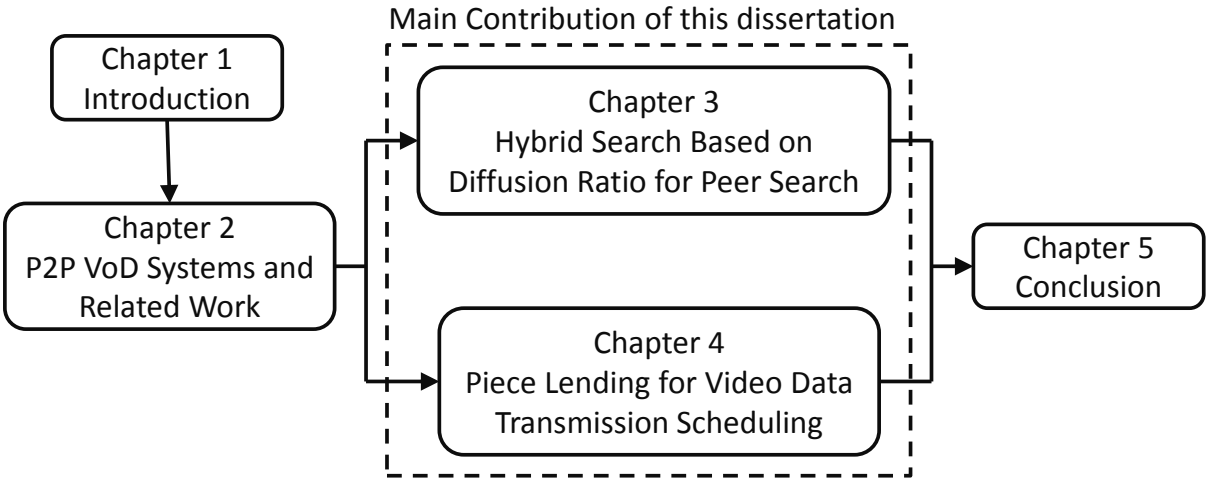

Figure 1.4: Organization of dissertation

The efficiency of the proposed method in terms of the utilization ratio of peer capacity and start-up latency is shown by simulation. Finally, Chapter 5 summarizes the achievements of this research and concludes this dissertation.

Chapter 2

P2P VoD Systems and Related Work

2.1 P2P VoD Systems

P2P VoD systems consist of a large number of peers that are grouped into two categories: video provider nodes and viewer nodes. Each video provider node provides one or more videos to few or many viewer nodes by using video-data delivery method based on P2P networks. Peers construct P2P networks for video-data delivery to provide videos and to view videos. Moreover, in P2P VoD systems, peers construct not only P2P networks for a video-data delivery but also P2P networks for peer search. Peer must search for peers that possess video data to view videos. Those P2P networks are managed independently of each other. Figure 2.1 shows an example of P2P VoD systems. Peers P_1, P_3, and P_9 are video provider nodes and they provide different videos to viewer nodes. Generally, P2P networks for video-data providing are constructed for each video content. Moreover, peers also construct a P2P network for peer search in the figure. Peers search for peers that possess the desired video data by using the P2P network. The P2P network for peer search in the figure is based on Gnutella that is described more fully in Section 2.2.3. In the Gnutella-based P2P network, peers have two roles, super node and leaf node, to realize the peer search function.

Viewers take the following steps in the P2P VoD systems. First, viewers search for videos by using a keyword-based search and wait for the response of the search. Next, they obtain a list of videos related to the search keyword. Then, they choose one video from

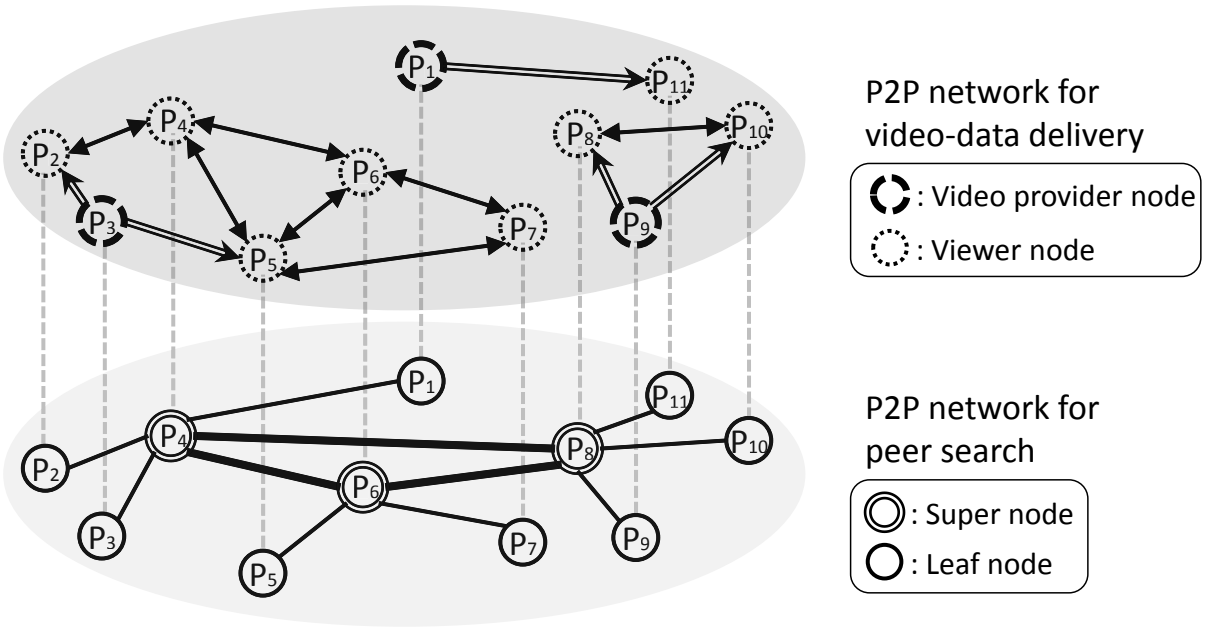

Figure 2.1: Example of P2P networks for P2P VoD systems

the list, and then wait until the video playback is ready in order to view the desired video. Finally, they enjoy viewing the desired video. In the system, viewers wait for the response of the search and wait until the video playback is ready. Therefore, it is important to reduce the waiting times, and peer search function and video-data transmission scheduling have been widely explored to do it. The existing methods of these two functions are described more fully in Sections 2.2 and 2.3.

This dissertation proposes two novel methods: HyDiff and Piece Lending. HyDiff realizes an effective peer search function on a P2P network for peer search, and Piece Lending realizes an effective video-data transmission scheduling on a P2P networks for video-data delivery. P2P VoD systems can adopt only one proposed method because HyDiff and Piece Lending are independent. Moreover, the proposed methods reduce two waiting times. HyDiff decreases the waiting time for the response of the search. Piece Lending decreases the waiting time for the video playback.

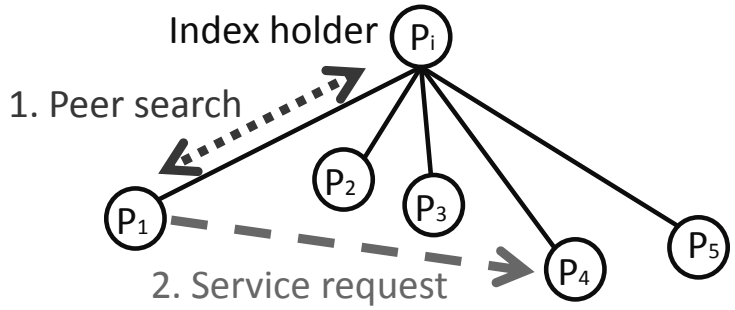

Figure 2.2: Example of centralized method

2.2 Existing Methods for Peer Search

2.2.1 Centralized Search Methods

When a peer tries to receive any kind of services in P2P networks, the peer is required to find out a provider peer that provides the service. General web services adopt systems based on the client-server model. In the client-server-based system, a peer asks the server and the server tell the peer where the provider peer is. P2P networks, however, have no centralized server that knows necessary information of the entire network unlike the networks based on the client-server model. The centralized search methods [36,37] are the one of the solution to fill in the gap. In the centralized search methods, a participating peer has the role like the server.

In the centralized methods, in general, only one peer possesses Central Global Object Index (CGO-Index). The peer called an *index holder* maintains the search function. CGO-Index includes the information of each data possessor. Then, peers maintain connections to the index holder in order to find out any data possessor. Moreover, each peer has to inform the index peer what data it possesses. However, in most cases, the peers in P2P network are end-users' PCs and they are not usually high-performance compared with the general servers. Therefore, it is difficult to maintain a large number of connections between peers. In addition, when the index holder is a high-performance peer, the peer could be a single point of failure. Therefore, the centralized methods would be vulnerable to Denial of Service attack (DoS attack) [59–61].

Figure 2.2 shows an example of the peer search in the centralized search methods. In the figure, circles mean peers in a P2P network and lines among circles mean connections for peer search. When Peer P_1 tries to find out data which Peer P_4 possesses, Peer P_1 asks Index Holder P_i, at first. Then, Index Holder P_i tells Peer P_1 that Peer P_4 possesses the data. After that, Peer P_1 requests the service Peer P_1 desires to Peer P_4. When Index Holder P_i goes down, no peer is able to search for any peer after that.

2.2.2 Structured Search Methods

A single peer such as the index holder of centralized methods is unable to manage search function in the very large-scale P2P networks. Therefore, many researchers have been trying to establish the methods how peers find out the desired peer in the huge P2P networks [62]. One of the solutions is a structured search method. The methods based on the Distributed Hash Table (DHT) are commonly used in the P2P networks; Chord [38], CAN [39], Pastry [40], Tapestry [41], Kademlia [42] and so on. A fundamental strategy of DHT is following; first, the system calculates a hash value of an object, which is data or peer, in P2P networks, the system stores connections of the object (or the object itself) over the P2P network according to the hash value. Peers are able to search for the object with the hash value [62]. Peers only have to calculate the hash value, they can find the object when the object exists in P2P networks. Thus, the structured search methods ensure that peers find existing objects in huge the P2P networks.

In the structured search methods, DHT controls strictly all the positions of the objects and the entire network topology. Therefore, on one hand, peers are able to search over the entire P2P network. On the other hand, DHT reconstructs the network topology every time a peer joins/leaves. Unfortunately, peers usually join/leave in/from the P2P networks without advance notice. When a large number of peers join/leave at the same time, a large number of messages is necessary to reconstruct the topology, and peers could not use the search function.

Distributed Hash Table (DHT)

In DHT algorithms, no index peer exists in the P2P networks. Thus, the P2P systems have no special peer that manages the whole hash space. Each peer locally manages a small part of the hash space and the peers cooperatively behave to achieve the search function. Performance metrics of DHT algorithm are how efficiently peers cooperate and how well the system is resistant to churn.

In DHT algorithms, each peer is assigned a hash value according to its own IP address. Note that, because the system usually calculates the hash value through a 160 bits hash function such as SHA1, the system gains a large enough hash space. The system can uniformly distribute hash values over the hash space due to the characteristics of hash functions. Thus, the hash values in the network are supposed to be uniformly distributed even though each peer individually calculates the hash value.

Each object is also assigned a hash value. The following description is specific to data files, for the purposes of illustration. The hash value of the file is calculated with the hash function that is the same as that used to calculate the peer hash value. Therefore, the hash value of the file and the hash value of the peer are distributed in the same range. A distance is defined in the hash space. The definition of the distance depends on DHT algorithms and they are slightly different, but it means a scalar value of the distance in the space in common; for example, the distance defined in Chord is the difference between the two hash values, the distance defined in Pastry and Tapestry is the number of digits of the common prefix. In those algorithms, each peer manages the files whose hash values are close to its own hash value. Each file's position information must be known to one of the peers in the network. If not, the file is never found out by any peer [63, 64].

When a peer searches for a file, the peer calculates the file hash value according to the file name and uses the search function using the hash value. As peers forward a search request, the next hop's hash value is close to the file's hash value. When the request reaches the closest peer to the file, it means the request reaches the final destination. In most algorithms, the systems tend to gain the redundancy and multiple neighboring peers of a peer possess the information of files managed by the peer.

Each peer has the routing table and forwards the request according to it. The routing table is the application layer's one. The routing table in DHT includes the distance from

Table 2.1: Features of DHT algorithms

Properties	Chord [38]	CAN [39]	Pastry [40]	Tapestry [41]
Parameter	None	Dimen d	Base b	Base b
Logical Path Length	$\log_2 N$	$O(d \times N^{1/d})$	$\log_b N$	$\log_b N$
Neighbor-state	$\log_2 N$	$O(d)$	$b\log_b N + O(b)$	$b\log_b N$
Messages to insert	$O(\log_2^2 N)$	$O(d \times N^{1/d})$	$O(\log_b N)$	$O(\log_b^2 N)$
Mutability	App-dep	Immut	unknown	App-dep
Load balancing	Good	Good	Good	Good

itself, routes to peers between the peer and itself. Each peer refers to the routing table when it forwards the request. Note that, each peer knows many routes to near peers and few routes to far peers. Therefore, this mechanism is effective for topology reconstruction. The more peers forward the request, the smaller the search space become by $1/n$, where n is the number of participating peers. Thus, the retrieval duration converges logarithmically to n. Table 2.1 shows the features comparison among DHT algorithms.

The file registration is needed before the file search and the registration process is almost the same as the search process. Briefly, in the registration process, a requesting peer finds out the peer whose hash value is the closest to that of the file. The requesting peer registers the file on the closest peer.

2.2.3 Unstructured Search Methods

In unstructured search methods, peers do not have to precisely control the topology and object location. Thus, the performance of the search function hardly becomes worse under churn. In general, the unstructured search methods usually adopt the algorithm extended from a flooding algorithm [43–47].

Flooding-based Method

Flooding is one of the simple routing protocols. Every time a peer receives a search query, a peer broadcasts it. In other words, a peer forwards the query to all of the adjacent peers. As a peer forwards the query, the replications of the query are generated in the

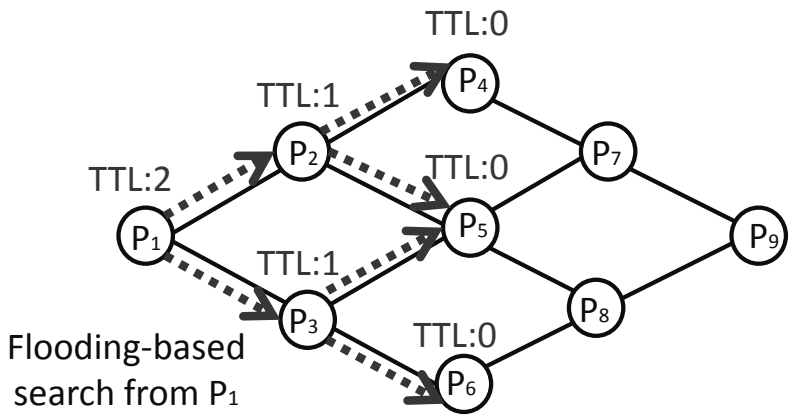

Figure 2.3: Example of flooding-based search

network and they finally spread over the entire network. Those replications, however, are too many to occupy the network resource. Thus, in general, many practical flooding protocols usually limit the number of forwarding hops by Time To Live (TTL). In those manners, each peer decrements the TTL counter and it stops forwarding when the TTL counter equals zero.

Figure 2.3 shows an example of flooding-based search in the unstructured search method. In the figure, circles mean peers and lines mean connections. When Peer P_1 searches with flooding, Peers P_2 and P_3 receive a query respectively. Peer P_2 forwards the query to Peers P_4 and P_5. Peer P_3 also forwards the query to Peers P_5 and P_6. In this case, if the TTL is set at two, Peers P_4, P_5, and P_6 do not forward the query. Therefore, those peers do not forward unlimitedly and the system avoids the suppression of the network bandwidth.

Gnutella

Gnutella is a well-known flooding implementation on P2P networks [48–51]. Gnutella aimed to get rid of any kind of servers from the network in order to localize network failure. A search function of the early Gnutella adopted a simple unstructured method in which every peer searches by flooding.

The followings are five Gnutella messages;

- Ping
  Message sent by a newly joined peer to Gnutella peers when it joins the network
- Pong
  Message sent by a Gnutella peer to a newly joined peer
- Push
  Message sent by peers that are inside the firewall
- Query
  Message sent by a requesting peer when it tries to find out objects
- QueryHit
  Message sent by a peer when it replies to a requesting peer to tell the file position

First, when a peer wants to join the Gnutella network, the peer sends a Ping message to peers that already joined the network. Those peers that receive the Ping message forward the Ping message to other Gnutella peers and, at the same time, reply with a Pong message to the Ping message's original sender. The pong message includes all network interface information such as IP address, port number and so on. In order to join the Gnutella network, the Ping message's original sender collects the Pong messages and creates connections to other Gnutella peers. When a requesting peer searches for an object, the requesting peer broadcasts a Query message and other peers forward it in a flooding manner. When a peer that has the desired object receives the Query message, it sends a QueryHit message to the requesting peer. Then, the requesting peer creates a connection to the QueryHit message's sender and gets the desired object. At that time, if the peer that has the desired object is inside the firewall, the peer creates a connection to the requesting peer with a Push message.

As the number of users increases, however, the network resource would be occupied Ping messages and Query messages. Moreover, the number of users called *free riders* would increases. They receive their desired objects but never contribute to other users. Those problems consequently cause system down. In the Gnutella network, because peers have various performances, low-performance peers became a bottleneck of the network and high-performance peers are not able to utilize their performance. In later Gnutella

network, in order to solve the problem, the system assigns the management node role to high-performance peers and they are called *super nodes*. Super nodes search for objects for the other peers. The other peers are called *leaf nodes* and every leaf node has only one connection to one super node. While, a super node is allowed to create connections to multiple other super nodes and leaf nodes. When a super node receives a Query message from a leaf node, the super node forwards the message to other super nodes in a flooding manner.

Figure 2.4 shows the later Gnutella network. In the figure, double circles mean super nodes. When Leaf Node L_1 tries to obtain an object that Leaf node peer L_{12} possesses, Leaf Node L_1, at first, sends a Query message to Super Node S_1. Super Node S_1 forwards the message to Leaf Nodes L_2, L_3, L_4, Super Nodes S_2, and S_3. Then, Super Nodes S_2 and S_3 forward the message to the other super nodes (to each other, in this case) and their own leaf nodes (from S_2 to L_5, L_6, L_7, L_8, and L_9 and from S_3 to L_{10}, L_{11}, and L_{12} in this case). Next, Leaf Node L_{12} sends a QueryHit to Leaf Node L_1. After that, Leaf Node L_1 creates a connection to Leaf Node L_{12} and the desired object is directly transferred to Leaf Node L_1.

One research using the Gnutella network shows that a peer sometimes could not find out a rare object. If TTL were set at infinity, a peer could search all over the network and would theoretically find out even a rare object. The network traffic, however, would exponentially increase and the network performance would consequently become worse. One of the latest researches of the Gnutella network shows that about 17% of the rare objects could not be found out in the network.

2.2.4 Hybrid Search Methods

Hybrid search methods combine strengths of the structured search methods and the unstructured search methods. Futures of the two types of search methods are shown as follows.

- Structured Search Method
  - Peers are able to find all object on P2P networks.
  - Response times of all search is constant.

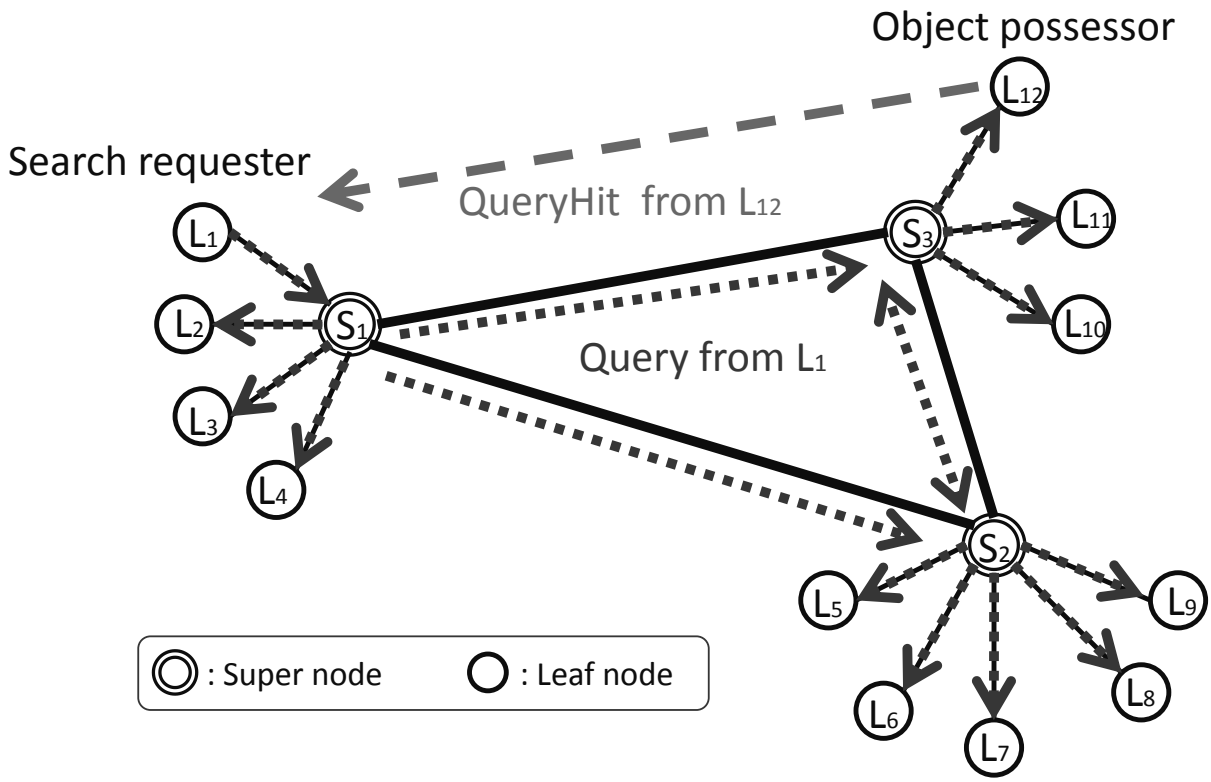

Figure 2.4: Gnutella

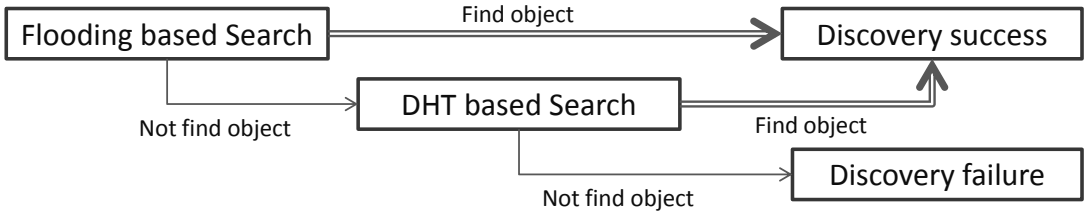

Figure 2.5: Simple Hybrid Search

  - Low efficiency under churn.

- Unstructured Search Method
  - Peers are not able to search entire networks. A response ratio is low.
  - Response time is short when search-target object is popular.
  - Response time is long slow or the search fails when search-target object is unpopular.
  - Search efficiency is not influenced in churn.

A fundamental strategy is that the structured search method is used for unpopular objects and the unstructured search method is used for popular objects. Moreover, the system keeps the minimum search functions using the unstructured search methods when many peers leave networks or the structured search methods search goes down.

Simple Hybrid Search

Simple Hybrid Search [48, 52] is the first hybrid search method. Simple Hybrid Search uses two search methods, Gnutella and PIER. In Simple Hybrid Search, super nodes on the Gnutella network construct a DHT network based on PIER. The search-target is not the object itself but a connection to the object. Therefore, super nodes collects object information of their own leaf nodes, and store the information based on a DHT algorithm. When a peer searches for an object, the peer finds a connection of the object.

Figure 2.5 shows a flow of selecting search methods in Simple Hybrid Search. At first, super nodes use a flooding search method for object searching. Next, super nodes use

a DHT search method when they do not receive responses of the searching for a fixed period. Simple Hybrid Search is able to discover an unpopular object using DHT. A response ratio of Simple Hybrid Search is 99.9%. However, a response time of unpopular object searching is very long because DHT search is used only after failure of flooding search.

Gossip Adaptive HyBrid

Gossip Adaptive HyBrid (GAB) [53] is a hybrid search method based on the Gossip algorithm. GAB estimates object popularity on networks by using the Gossip algorithm [65, 66], and takes on two search methods depending on the estimated popularity of the search target. The method reduces probability of selecting the unstructured search method for unpopular objects, and decreases messaging cost and response time. Generally, the popularity of the search target changes with time and it is difficult to predict the time change of the search-target popularity [67]. Therefore, GAB tries to calculate the popularity of the objects periodically. In large-scale P2P systems, the systems must need very long calculation time to calculate the precise popularity. Thus, GAB calculates approximate values of objects' popularity by using request rate for decreasing the calculation time.

The Gossip algorithm is an effective distributed algorithm for calculating an average value of local values possessed by each peer [65, 66]. In the Gossip algorithm, a peer repeats exchanging local values with a neighbor peer selected at random, and calculating an average value of these. The average value is closer to a precise average value of overall networks with increasing the number of repeating time. A termination condition of that repeating is that a difference between the current average value and the previous average value is smaller than a gossip threshold. Finally, the Gossip algorithm finds an approximate value of an average value of all local values of networks. The average number of exchanges is around 22 when the number of peers is $1,000$ and the Gossip threshold is 0.01.

Each peer must find a search-target popularity by using the Gossip algorithm. In GAB, a popularity of an object is defined as the number of requests in a fixed period. Each super node counts a single word used for searching for keywords every time when the super

node searches for objects instead of leaf nodes. Each super node normalizes the results of the counts, and the normalized value is the local popularity in GAB. When searching for keywords includes multiple words, the total of words' popularity is the popularity of the keywords.

Figure 2.6 shows an example of calculation of local popularity. Each super node counts searching keywords used by their own leaf nodes. In the figure, Super Node S_1 counts the number of each single word included in the searching keywords of Leaf Nodes L_1, L_2, L_3, and L_4. Super Node S_1 makes a local popularity list as shown in Table 2.2. In the table, each value is normalized of the numerical sum of values. Super nodes estimate the popularity of each single word on overall networks by using the local popularity lists and the Gossip algorithm.

Table 2.2: Local popularity list of Super Node S_1

Word	Local popularity
Poem	0.33
of	0.33
Bob	0.25
Alice	0.08

GAB selects the structured search method or the unstructured search method based on the search-target popularity estimated by the Gossip algorithm. A fundamental strategy is that the structured search method is used for unpopular objects and the unstructured search method is used for popular objects. GAB should decide a search threshold for the search method selection. GAB defines a utility function as shown by Equation 2.1 for deciding the search threshold.

$$u = \min(R, 1) \times \omega_1 + \min(R, R_{\max}) \times \omega_2 - T \times \omega_3 - B \times \omega_4, \tag{2.1}$$

where u is a utility of a one search, R is the number of responses of the search, $R_{\max}$ is the maximum number of responses of the search, T is a shortest response time of the search, and B is the number of messages of the search overall networks. The first term means that the presence of a response has large impact on the utility. The second term

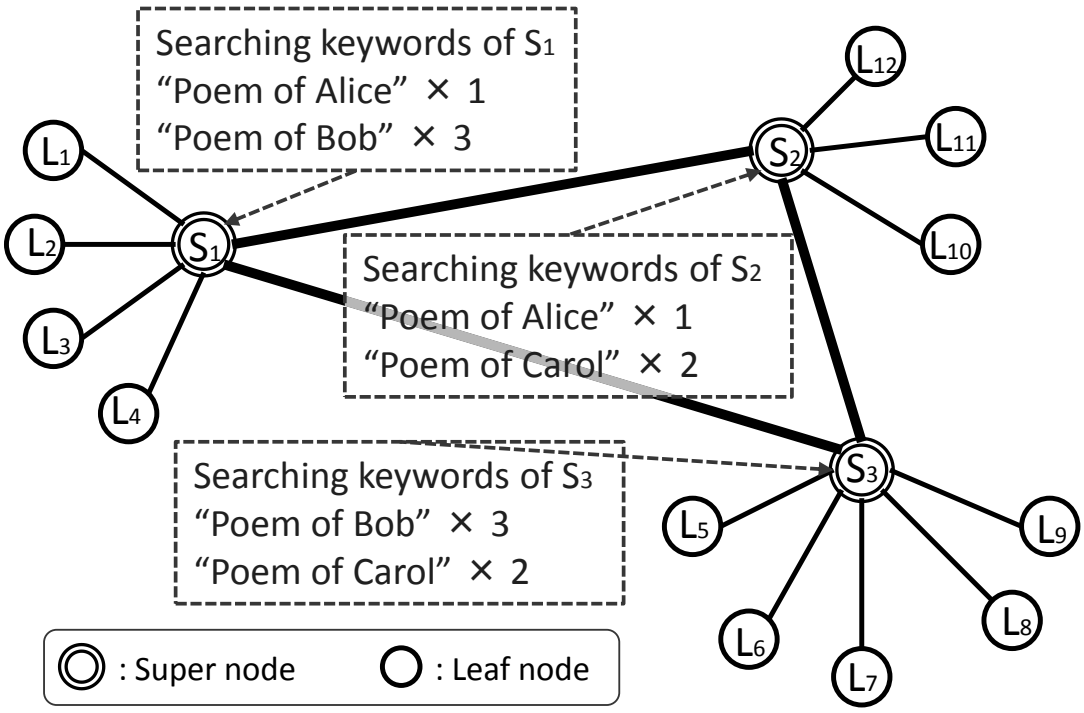

Figure 2.6: Calculation of local popularity

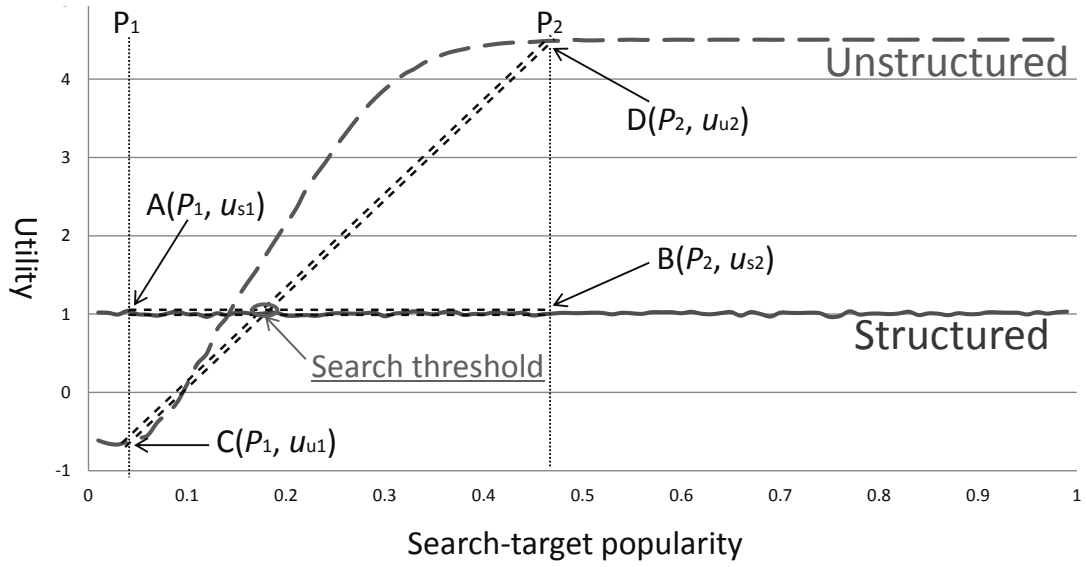

Figure 2.7: Utility versus search-target popularity

Table 2.3: Simulation parameters

Parameter	Value
Number of peers	10,000 peers
Number of objects	20,000 objects
Object distribution	Zipf distribution
$R_{\max}$	25

means that too many responses do not contribute to the utility. The third term means that the shorter the shortest response time is, the better the utility is. The fourth term means that the lower the number of messages is, the better the utility is.

Figure 2.7 shows a utility with changing a search-target popularity. The vertical axis shows a search utility calculated by Equation 2.1. Simulation parameters are shown in Table 2.3. In the simulation, the object popularity does not changes by time. In the figure, a utility of the structured search method is constant, and a utility of the unstructured method increases with increasing the object popularity. Optimal search threshold of this case is about 0.16, but the optimal value depends on network statement and the system

does not calculate the optimal value before searching. In GAB, a super node selects two objects that have different popularities at random, and runs two search methods for the objects. As a result, the super node finds four sets of a search-target popularity and a utility; A(P_1, u_{s1}), B(P_2, u_{s2}), C(P_1, u_{u1}), and D(P_2, u_{u2}) in the figure. GAB calculates a intersection point of line A-B and line C-D as an approximate value of the optimal threshold.

2.3 Existing Methods for Video Data Transmission Scheduling

2.3.1 Non-incentivized Methods

In P2P VoD services, the video provider can deliver videos at low cost, because the video provider load is distributed to peers. Content-unit-based methods are simple methods for video data transmission [68, 69]. A peer receives video data from only one other peer in the methods. Peers in a video data stream are able to enjoy high quality video stably when peers on upstream of the video data stream have high performance. When peers on upstream are low-performance peers, however, video quality of peers on downstream is low. Moreover, peers are not able to view high quality video continuously under churn because a peer does not receive the video data when peers on upstream leave networks.

In piece-unit-based methods, video content data is divided into small pieces and P2P systems deliver the video data in unit of a piece. The piece-unit-based methods are used in P2P VoD services such as [9–14, 54, 55]. Each peer requests the pieces in order to view the video, and caches the viewed pieces. Peers store the pieces using the first-in, first-out (FIFO) method. Then, peers exchange pieces each other in order to collect all pieces and view the video. [54] and [55] applied the BitTorrent file-sharing scheme to VoD streaming and proposed this scheme. Efficiency of the methods for P2P networks has been shown in most of researches [12–14, 67, 70].

In the methods, however, an overall system performance decreases when free riders join in P2P networks because the systems do not provide an incentive for sending pieces to peers. Free riders receive pieces from the video provider or other peers but they send

no or few pieces to other peers. Therefore, the video quality of peers contributing to the systems decreases with increasing the number of free riders. Free riders are classified into *malicious free riders* and *unmalicious free riders*. Malicious free riders use falsified program and they receive pieces without any piece transmission. Unmalicious free riders are motivated to contribute the P2P system but they cannot send many pieces because their transmission capacity is low. To solve the free rider problem, incentivized methods are proposed [10, 15–17, 56–58].

2.3.2 Simple Incentivized Methods

In P2P file sharing systems, peers are motivated to send contents to other peers in order to achieve high receiving rate from the P2P system. By employing the tit-for-tat policy [19], BitTorrent punishes free riders who do not contribute to the system. In P2P VoD systems, peers are motivated to contribute more in order to get higher video quality. Simple incentivized methods, which use scalable video coding, have been proposed [16, 17, 56, 57]. In the methods, the more video data peers send, the higher quality video they can view. The incentivized algorithms like tit-for-tat are not applicable to P2P VoD systems because P2P VoD systems have the asynchronous peers' playback progress and the data flow directionality. iPass [10] is proposed to conform peers to the data flow directionality. In iPass, each peer sends more pieces to less advanced neighbors that sent a larger number of pieces.

The incentivized methods, however, suffer from slow downloading of unpopular pieces. In general, all peers receive pieces in the consecutive order in P2P VoD systems in order to decrease the start-up latency. Therefore, pieces of the last part of the video tend to become unpopular pieces in P2P networks. Moreover, it is hard for newly joined peers to join peers in the piece exchange because newly joined peers do not possess any pieces right after they join. Newly joined peers suffer from long start-up latency.

2.3.3 Incentivized Methods with Low-Popularity Piece Push

To solve the problem, an incentivized method with low-popularity piece push is proposed [15, 58]. In the incentivized methods with low-popularity piece push, the video provider

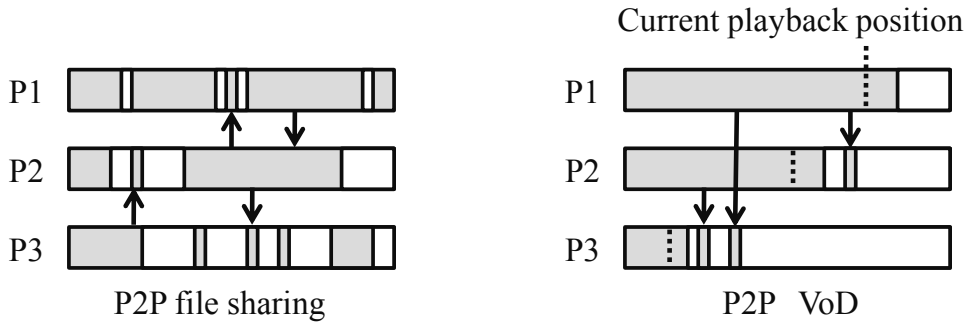

Figure 2.8: Directionality of P2P file sharing and P2P VoD

pushes low-popularity pieces to a newly joined peer. The methods, however, do not utilize the sending capacity of peers because peers receive some pieces without sending pieces from the video provider. In addition, newly joined peers have to wait a long time for the piece pushed by the video provider due to the load concentration problem when the video provider is low-performance server.

Overview of Incentivized P2P VoD with Low-Popularity Piece Push

Tit-for-tat is an incentivized algorithm that is typically used in P2P applications. However, because of the data flow directionality due to the asynchronous peer playback progress, P2P VoD systems to which tit-for-tat is applied simply applied tit-for-tat [16,17] are not efficient for using overall peers' transmission capacity. Figure 2.8 shows directionality of P2P file sharing, which uses tit-for-tat generally, and P2P VoD. Although the data flow is unidirectional in the P2P file sharing, the data flow is directional because newly joining peers have no pieces and collect pieces in order generally in P2P VoD.

To solve this problem, a low-popularity piece pushing approach in tit-for-tat based video data transmission scheduling method is proposed in [15, 58] and improves utility of peers' transmission capacity and receiving quality of video streaming. Additionally, this approach can design the decentralized scalable system by using low-popularity piece push. Ordering peers according to their playback position and, connecting them that way, achieves an optimal throughput as all peers contribute to piece dissemination. This ensures a constant good put over time and a useful piece being disseminated at each

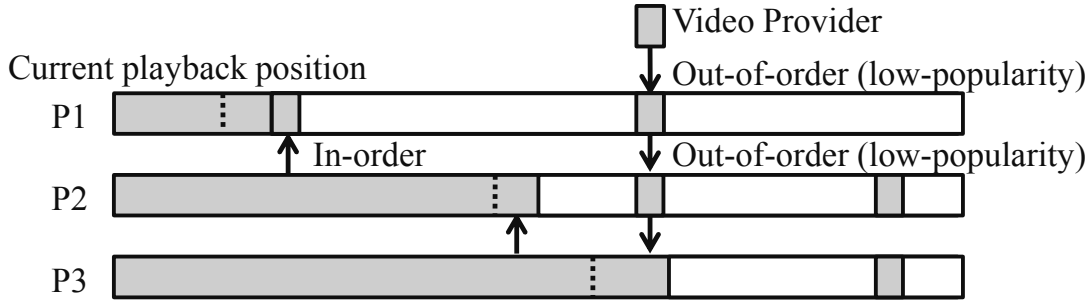

Figure 2.9: Example of piece exchanging among peers

exchange. The intuition is that the most advanced clusters, in terms of playback position, feed the less advanced ones, while the less advanced clusters push useful pieces to the most advanced ones. Those useful pieces, from the standpoint of the most advanced clusters, are provided by the video provider to the less advanced clusters, in-order to provide them with some negotiation power.

Figure 2.9 shows example of piece exchanging among peers. The video provider forwards the most advanced piece to the least advanced peer. This piece is eventually sent to the most advanced peer on a forward path. In return, in order pieces are downloaded on a reverse path.

Algorithm Details of Existing Methods

An important component required for the construction of a connected list at the cluster level is to identify the pieces required by the most advanced cluster. Since the pieces required by the most advanced cluster are not available in the swarm, the video provider provides those pieces. However, instead of directly sending to the most advanced cluster, the video provider provides those pieces to the least advanced cluster. This allows the least advanced cluster to have a good bargaining power in the system. The exchange policy has to be designed accordingly so that these pieces are further fed on the forward path such that every intermediate cluster downloads it and forwards it to the next cluster. Eventually, these pieces reach the most advanced cluster. Note that the video provider can easily obtain a list of peers in the least advanced piece from the tracker. The most

advanced piece can be obtained by the video provider in a distributed fashion by polling peers through the connected list.

Neighbor Selection

The peers in each cluster can exchange pieces among themselves and they can participate in a connected list style feeding process. Therefore, we ensure that the peer set of every peer is limited to the peers either from the same cluster or from the neighboring clusters. When a newly joined peer joins the swarm, it is connected to some peers within the first group. The remainder of the peer set is constructed by connecting to neighbors of neighbors. The structure is maintained during the download through Gossip: peer set is updated periodically by exchanging the set of neighbors with current neighbors similarly to [17]. If the peer remains within the same cluster, it asks its neighbors to return a subset of the peers from their respective clusters. When a peer moves out of a cluster, it should update its neighborhood such that it is now connected to the peers within its new cluster and also to some peers in the clusters neighboring to this new cluster. This can again be done by polling through a neighbor of neighbors using gossip. In this way, we can easily maintain the structure in a decentralized way.

Video Data Exchange Policy

The exchange policy determines whether two peers n_1 and n_2 should exchange pieces upon an encounter and which specific pieces p_1 and p_2 should be exchanged if any. If the peers are in the same group (i.e., their positions in the file lie in the same piece $S_1 = S_2$) then traditional swarming should be performed. Both peers look in a random order for a piece in their common current piece $S = S_1 = S_2$ that they could send to each other. More specifically, they look for a piece in their piece sets that does not belong to the other peer's piece set. To ensure piece diversity inside each piece, and thus efficient intra-group swarming, such pieces are looked for by exploring the piece in a random order. Due to the peer set structure described in the previous paragraph, advanced pieces can be pushed only from a cluster to the immediate next one when a peer connects to a member of the next cluster. In that situation, the less advanced peer, denoted n_1, downloads a randomly chosen useful piece for its current piece in exchange for a piece in the future. Priority

Table 2.4: Problems of existing methods

Function	Method	Problem
Peer search	Centralized	The index holder is a single point of failure and cannot handle a large number of queries at same time
	Structured	The efficiency decreases under churn
	Unstructured	The efficiency decreases when the search-target popularity is low
	Hybrid	The efficiency decreases when there exists a large gap between measured request rate and popularity
Video data transmission scheduling	Non-incentivized	The video quality decreases when free riders join in P2P networks
	Simple incentivized	It is hard for a newly joined peer to join in the piece exchange
	Incentivized with low-popularity piece push	A newly joined peer needs the support from video provider to join in the piece exchange and peers transmission capacity is not utilized

is given to the most advanced pieces in pieces after n_2's piece (denoted $p_2 > S_2$). If no such piece can be exchanged, then n_1 tries to send a random piece in S_2. As explained in the previous sections, the motivation for sending the most advanced pieces with highest priority is two-fold: (i) ensure fast feeding of the most advanced piece and (ii) build an as long as possible forward path and in turn a long reverse path establishing intercluster feeding. If no mutual interesting pieces can be found using this exchange policy, the contract between the two peers is simply broken.

Note that using the transfer strategy presented in the previous paragraph, pieces before a peer's playback position are used only to feed peers in the previous group. Therefore, a peer can drop pieces before the playback position of the peers in the previous group without reducing its feeding ability.

2.4 Problem Statements

The problems in the existing methods are stated. Table 2.4 shows the problems for each type of the existing methods. In terms of the peer search function, the existing hybrid search methods achieve high performance under churn and large-scale systems. In terms of the video data transmission scheduling, the existing incentivized methods also achieve high performance under churn, free rider joining, and large-scale systems. The efficiency of the methods, however, decreases by an influence of video data popularity. Therefore, this dissertation focuses on the video data popularity and proposes novel methods for the peer search function and the video data transmission scheduling function.

2.4.1 Problems of Existing Peer Search

Centralized search methods adopt the CGO-index for the search function. P2P systems select one peer from participating peers in P2P networks and the selected peer is named the index holder that keeps CGO-index. Briefly, only one peer maintains the search function of the method. Therefore, when any failures occur in the index holder, the search function goes down. Moreover, the index holder cannot handle a large number of search queries at the same time because peers are end-user PCs and their performance is not high generally. Thus, the centralized search method is not suitable for the large-scale P2P networks.

Structured search methods and unstructured search methods are superior search methods to centralized methods because the distributed methods have tolerability for the load concentration and do not have a single point of failure. The search efficiency of the two search methods, however, decreases in particular cases. The efficiency of structured search methods decreases under churn. The efficiency of unstructured search methods also decreases when the search-target popularity is low.

Hybrid search methods combine strengths of the structured search method and the unstructured search method. In the hybrid search methods, the P2P systems maintain a structured search method and an unstructured search method. Then, the structured search method is used for unpopular objects and the unstructured search method is used for popular objects. Thus, hybrid search methods achieve high response ratio, short

response time, and high tolerability for churn. Efficiency of the existing hybrid search methods, however, decreases in some cases. The existing hybrid search methods use the request rate of objects in order to estimate the search-target popularity. However, it is difficult to decide how to calculate the request rate because peers' requests are randomly-generated. In the existing hybrid search methods, the request rate is defined as the number of requests in a fixed term. For example, when peers do not request searching in a period of time, super nodes always select a structured search method in the period of time even when the actual popularity of the search target is high. Therefore, the popularity estimated by request rate do not always properly-reflect the actual popularity. Then, an important point is that the efficiency of the structured search methods and the unstructured search methods do not depend on the request rate but depend on the popularity. Therefore, super nodes may select an unreasonable search method when there exists a large gap between the request rate and the popularity.

In Chapter 3, this dissertation proposes HyDiff, Hybrid search based on Diffusion ratio, which uses the search target popularity estimated by local popularities. The proposed method employs not the request rate but a local popularity for the popularity estimation. As a result, the proposed method unlikely decreases the search efficiency when there exists a large gap between the request rate and the popularity.

2.4.2 Problems of Existing Video Data Transmission Scheduling

On one hand, in non-incentivized methods, the overall system performance decreases when free riders join in P2P networks. Free riders receive pieces from a video provider or other peers but they do not send any piece to other peers. Therefore, the video quality of peers contributing the systems decreases with increasing the number of free riders. On the other hand, in incentivized methods, P2P systems provide the receiving right of pieces to a peer depending on the amount of pieces that the peer have sent. Therefore, free riders do not receive pieces because they do not send pieces to other peers and do not obtain the receiving right. As a result, the incentivized method can keep video quality of contributing peers even when many free riders join the P2P networks.

Simple incentivized methods belonging to the incentivized methods adopt the tit-for-tat approach, which is generally used in P2P file sharing systems, to P2P VoD systems.

Efficiency of the simple incentivized methods, however, decreases by a negotiation power gap. Negotiation power means the amount of pieces that can be sent to other peers. In general, all peers receive pieces in consecutive order in P2P VoD systems in order to decrease the start-up latency. Thus, pieces of the early part of video are very popular in P2P networks, and it is difficult for peers to send popular pieces to other peers because the most part of peers have already obtained the pieces. Newly joined peers, however, possess no pieces or possess only pieces of the early part of video right after joining. Therefore, in simple incentivizded methods, it is hard for newly joined peers to join in the piece exchange because the negotiation powers of newly joined peers are low.

An incentivized method with low-popularity piece push is proposed to solve the problem mentioned above. In the incentivized method with low-popularity piece push, the video provider pushes the low-popularity pieces to newly joined peers in order to increase their negotiation powers. The method, however, does not utilize the sending capacity of peers because peers receive some pieces without sending pieces from the video provider. In addition, newly joined peers have to wait long for the piece pushing by the video provider when a video provider is a low-performance peer.

In Chapter 4, this dissertation proposes Piece Lending which lends a receiving right of low-popularity data-pieces to each newly joined peer. In the proposed method, newly joined peers use the receiving right in order to receive low-priority pieces and join the piece exchange. The proposed method distributes the load of low-popularity piece push to peers, and it achieves high utilization ratio of peers' capacity and low start-up latency.

Chapter 3

Hybrid Search Based on Diffusion Ratio for Peer Search

3.1 Overview

This chapter proposes HyDiff, Hybrid search based on Diffusion ratio, which uses search-target popularity estimated by local popularity. The proposed method is a hybrid search method that acts as a structured or unstructured search method, depending on search-target popularity. In hybrid search methods, the method of estimating search-target popularity is very important because efficiencies of the structured and unstructured search methods vary depending on the search-target popularity. Additionally, the popularity of the search target changes with time and it is difficult to predict the time change of the search-target popularity. Therefore, in hybrid search methods, peers must calculate the popularity periodically. Existing hybrid search methods use the request rate to estimate search-target popularity. However, it is difficult to retain high efficiency because in some cases there exists a large gap between the measured request rate and popularity. For example, when peers do not request searching in a period of time, super nodes always select a structured search method in the period of time even when the actual popularity of the search target is high. Therefore, the estimated popularity by request rate does not always properly reflect the actual popularity. The proposed method bases its popularity estimation on local popularity. As a result, search efficiency is expected to be retained in

HyDiff. Efficiency of HyDiff in terms of the response rate, response time, and maintenance cost is shown by simulation.

3.2 Constructing Network Topology

The basic topology of the proposal is based on the Gnutella [71] network. Some super nodes, which are representative nodes, maintain the search function. The super nodes monitor the object information possessed by their own neighbors and keep information of objects' local popularities. Here, popularity means how many peers possess a particular search-target object, and an object means a peer or data. Then, super nodes estimate global popularity of the search-target object from the object's local popularities for the search method selection.

Figure 3.1 shows the topology of HyDiff. In HyDiff, there exist three types of peers: a super node, a leaf node, a DHT node.

- Leaf nodes are normal nodes that only have a connection to one super node. The most part of peers in the network is leaf nodes. When leaf nodes search for any object, they send a search query to their known super nodes.

- Super nodes are representative nodes that have connections to leaf nodes, other super nodes, and DHT nodes. Super node has three tasks: global popularity estimation, search method selection, and unstructured search processing. First, in the global popularity estimation, super nodes calculate the object local popularities in their neighbors and estimate the global popularities by using the Gossip algorithm and the local popularities. Second, in the search method selection, when the super nodes receive search query from leaf nodes, the super nodes select an effective method from a structured search method or an unstructured search method based on the estimated global popularity. Finally, in the unstructured search running, when the unstructured search is necessary, it conducts the unstructured search between super nodes.

- DHT nodes are randomly selected nodes that have connections to other DHT nodes and one super node. When DHT nodes search for any object, they send a search

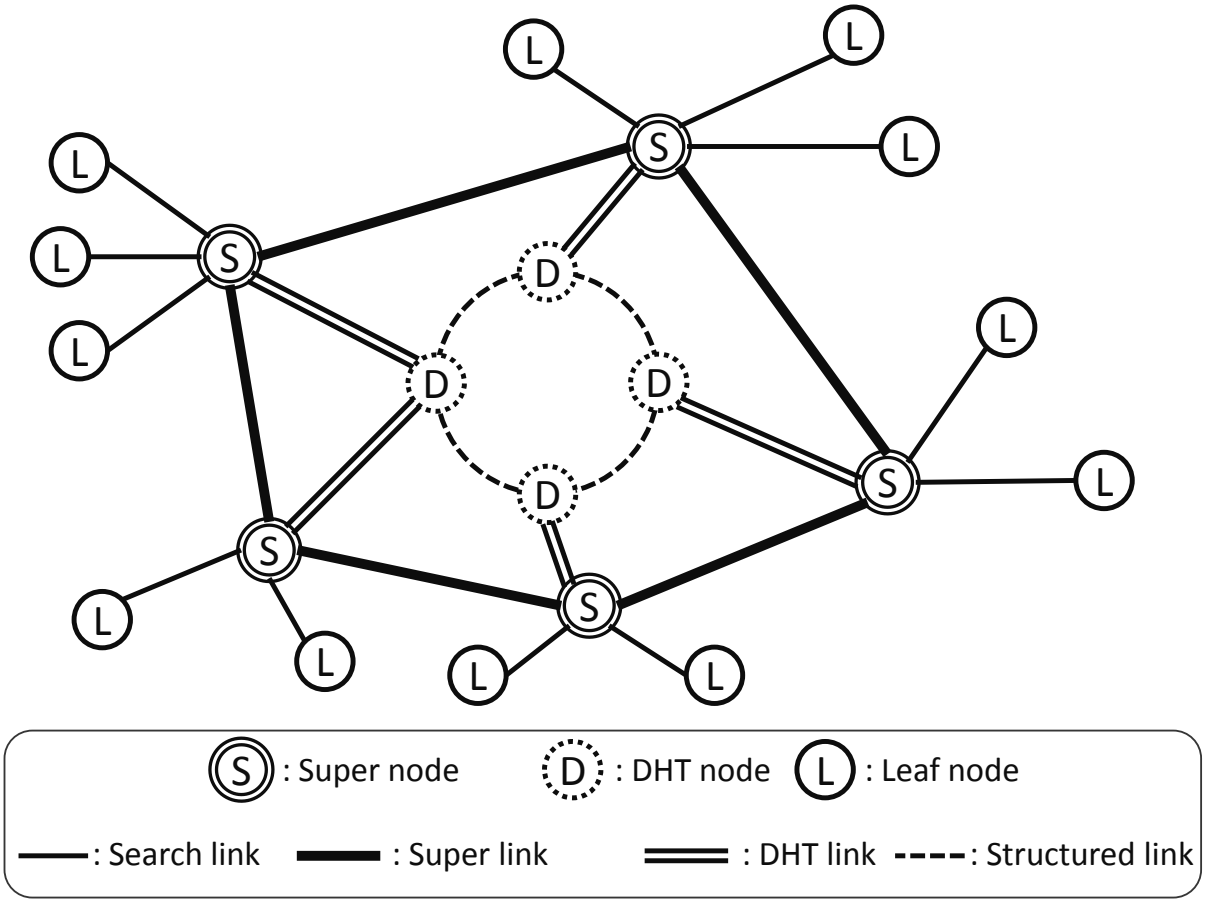

Figure 3.1: Topology of proposed method

query to their known super nodes as well as leaf nodes. DHT nodes have two tasks: structured search maintenance and structured search running. First, in the structured search maintenance, DHT nodes generate a DHT topology and keep object possessor information between DHT nodes. Second, in the structured search running, when DHT search is necessary, it conducts DHT search between DHT nodes.

In addition, each node establishes and maintains connections as following.

- A Search link is a connection between a leaf node and a super node or a connection between a DHT node and a super node. It is used when a search is requested to a super node.
- A super link is a connection between a set of super nodes and used when an unstructured search and the Gossip algorithm are applied.
- A DHT link is a connection between a super node and a DHT node, and used when the super node stores data to DHT and send search queries to DHT nodes for structured search.
- A structured link is a connection between structured nodes, and used for structured search.

This processing uses two messages: Incomig Request and Incomig Reply. At first, a peer sends the Incomig Request to a bootstrapper. Then, the bootstrapper that received the Incomig Request sends the Incomig Reply holding at least one super node's IP address. Here, any nodes can become the bootstrapper because all nodes know one super node at least. The newly joined peer adds the super nodes' IP address within the Incoming Reply to super node list and stores the IP address to its own local storage space. Next, the newly joined peer has to decide which it becomes a super node, a leaf node, or a DHT node. If the system has very strict policy, the role decision process must be followed by the bootstrapper arranged by the system administrator. Otherwise, the newly joined node decides its role based on the probability set by the system administrator. From a research of Gnutella, using past log data when the node joined the network enables selecting a

faster peer for a super node, and structuring an efficient topology. Next behavior of peers depends on the selected role. The initial processing of leaf nodes is shown in Section 3.2.1, that of super nodes is shown in Section 3.2.2, and that of DHT nodes is shown in Section 3.2.3.

3.2.1 Initial Processing of Leaf Node

This section explains the initial processing of a leaf node. At the end, the leaf node has a single search link. When a peer joins the network as a leaf node, the system uses three messages: Search Link Request, Search Link Reply, and Search Link Reject. A message sender of the Search Link Request is a newly joined leaf node. A message sender of the Search Link Reply is a super node and it includes information of the number of established search links of the super node. A message sender of the Search Link Reject is a super node and it includes information of one or more other super node's IP addresses and port numbers.

First, newly joined leaf node sends the Search Link Request to all super nodes known to bootstrappers. Super nodes that received the Search Link Request reply with the Search Link Reply when the number of their established search links is under the limitation. Otherwise, the super nodes reply with the Search Link Reject when the number of their established search links is over the limitation. When a newly joined leaf node receives the Search Link Reply, it adds the message sender's IP address, port number, and the number of established search links to its own a super node list, and establishes a search link with the message sender. When the leaf node receives multiple Search Link Replies, it adds the all messages sender's IP addresses, port numbers, and the numbers of established search links to its own super node list, and it selects the super node whose number of established search links is the lowest. The leaf node establishes a search link with the selected one super node. When the selected super node of the leaf node has any trouble, the leaf node tries to contact other super nodes by using its own super node list. When the leaf node does not receive any Search Link Reply, it sends the Search Link Request to another super node. Each super node must send the Search Link Reply or the Search Link Reject. The Search Link Reject including one or more super node's IP addresses and port numbers, therefore, the leaf node that does not received the Search Link Reply can

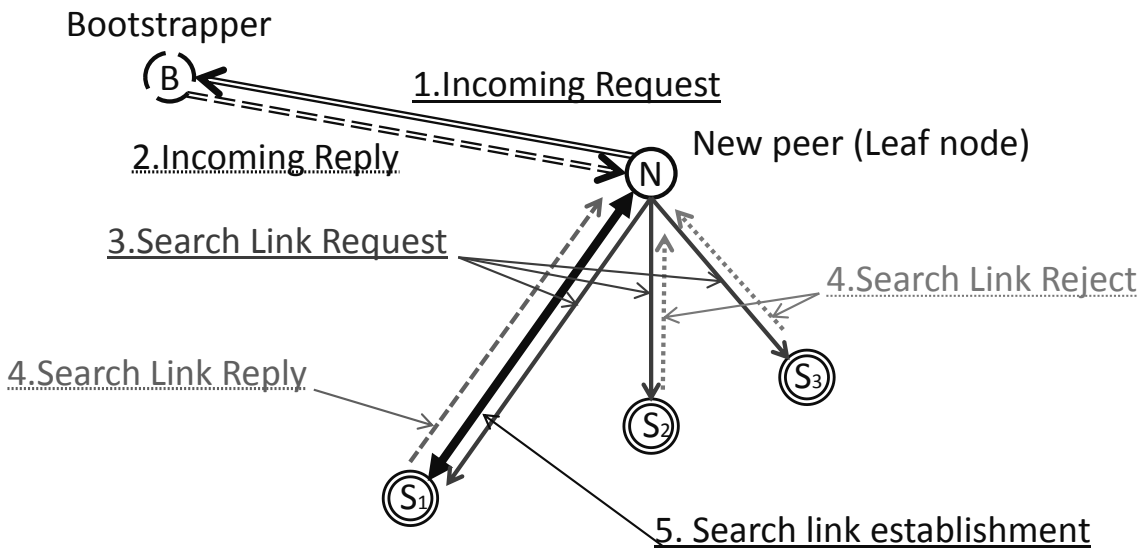

Figure 3.2: Search link establishment by newly joined leaf node

obtain information of other super nodes. To send as many Search Link Requests as retry limitation, it establishes a search link with a super node whose number of established search links is the lowest.

Figure 3.2 shows an example of search link establishment. There exist one bootstrapper, one newly joined leaf node, and three super nodes. At first, Newly Joined Peer N sends the Incoming Request to the bootstrapper in order to join the P2P network. The bootstrapper B replies with the Incoming Reply to Peer N, and the message includes the IP addresses and port numbers of three super nodes. In this case, the newly joined peer selects leaf node as its own role. Referring to the Incoming Reply, Leaf Node N sends the Search Link Request to three super nodes. In the figure, Super Node S_1 replies with the Search Link Reply and Super Nodes S_2 and S_3 reply with the Search Link Reject. Finally, Leaf Node N establishes a search link with Super Node S_1 because only Super Node S_1 sends the Search Link Reply.

3.2.2 Initial Processing of Super Node

This section explains the initial processing of a super node. At the end, the super node has one or more super links and DHT links. When a peer joins the network as a super node, the system uses six messages: Super Link Request, Super Link Reply, Super Link Reject, DHT Link Request, DHT Link Reply, and DHT Link Reject. A message sender of the Super Link Request is a newly joined super node and it includes information of Time to Live (TTL). Message senders of the Super Link Reply and the Super Link Reject are super nodes. A message sender of the DHT Link Request is a newly joined super node and it includes information of a value selected at random. Message sender of the DHT Link Reply and the DHT Link Reply is a DHT node.

Super Link Establishment

First, a newly joined super node sends the Super Link Request to all super nodes based on information from the bootstrapper. The super node that received the Super Link Request sends the Super Link Reply to the newly joined super node and broadcasts the Super Link Request to other super nodes when the number of their established super links is under the limitation. The super node that received the Super Link Request sends the Super Link Reject to the newly joined super node when the number of their established super links is over the limitation. The newly joined super node that received the Super Link Reply adds the message sender's IP address and port number to its super node list and establishes a super link with the message sender. Super nodes can establish multiple super links under the limitation of the number of super node neighbors. When the newly joined super node does not receive any Super Link Reply, the newly joined super node increments TTL of the Super Link Request and sends the message again.

Figure 3.3 shows an example of super link establishment. There exist one bootstrapper, one newly joined super node, and six super nodes. First, Newly Joined Peer N sends the Incoming Request to the bootstrapper in order to join the P2P network. The Bootstrapper B replies with the Incoming Reply to Peer N, and the message includes IP addresses and port numbers of Super Nodes S_2 and S_5. In this case, the newly joined peer selects super node as its own role. Referring to the Incoming Reply, Super Node N sends the Super Link Request to three super nodes. Super Nodes S_2 and S_5 receives the messages and

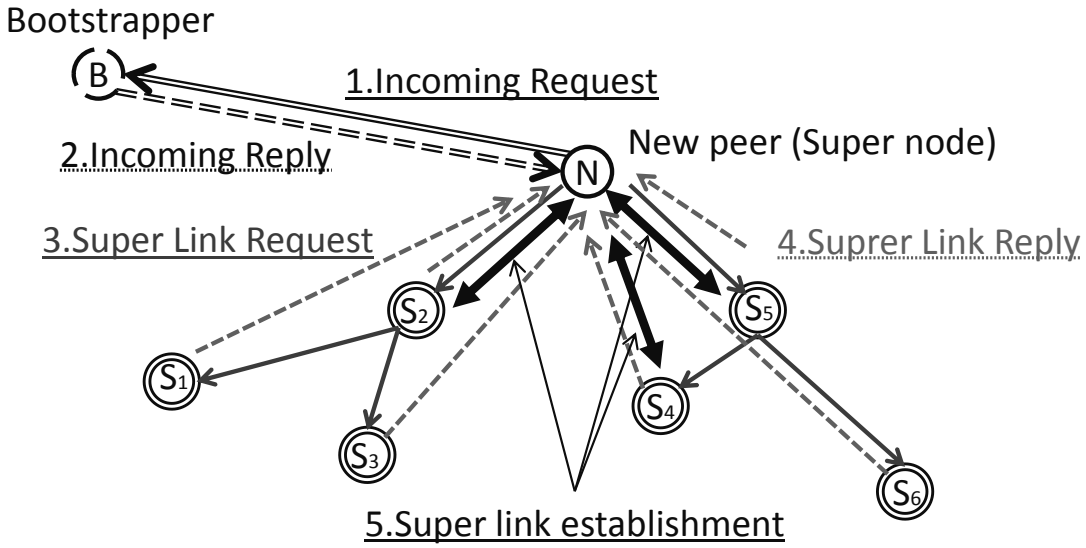

Figure 3.3: Super link establishment by a newly joined super node

broadcast to their neighbors. Next, Super Nodes S_2, S_4, and S_5 reply with Super Link Replies because numbers of their established super links is under the limitation. Finally, Super Node N establishes super links with Super Nodes S_2, S_4, and S_5.

DHT Link Establishment

After super links are established, the newly joined super node selects one super node from its own neighbor super nodes and sends the DHT Link Request with a random value to the selected neighbor super node. The super node that received the DHT Link Request forwards the message to a DHT node through its own DHT link. The DHT node that received the DHT Link Request runs the structured search using the random value included in the message as search-target ID. Next, the DHT Link Reply is sent from the DHT node whose ID is the nearest to the random value of the newly joined super node. The detail of the structured search is the same as Chord [38]. The newly joined super node that received the DHT Link Reply establishes a DHT link with the DHT node in the message.

Figure 3.4 shows an example of DHT link establishment. There exist one newly joined

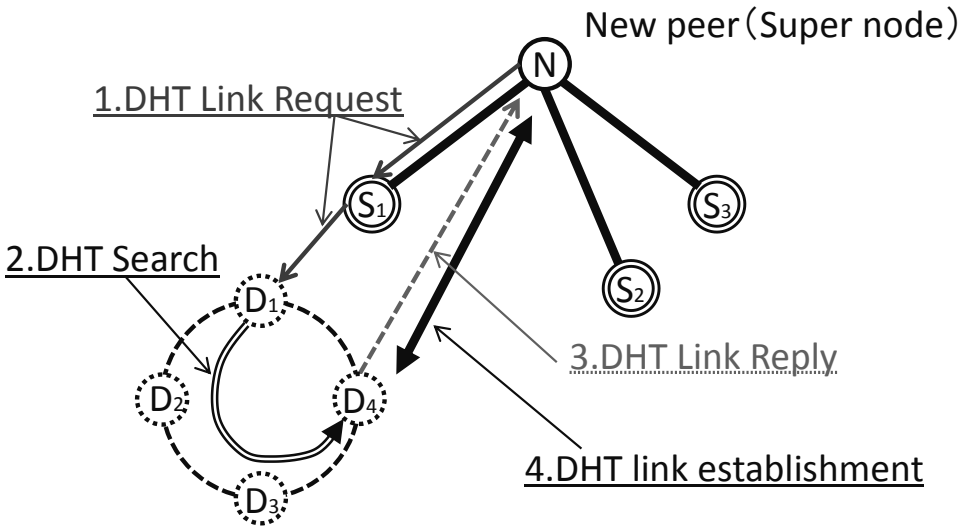

Figure 3.4: DHT link establishment by newly joined super node

super node, three super nodes, and four DHT nodes. First, Newly Joined Peer N sends the Incoming Request to the bootstrapper in order to join the P2P network. Super Node N, which is newly joined peer, selects Super Node S_1 and sends the DHT Link Request to Super Node S_1. Super Node S_1 forwards the message to DHT Node D_1, which is a neighbor of Super Node D_1. DHT Node D_1 runs the structured search by using a random value of the received message. In this case, DHT Node D_4' s ID is the nearest to the random value, and DHT Node D_4 sends DHT Link Reply to Super Node N. Finally, Super Node N establishes a DHT link with DHT Node D_4.

3.2.3 Initial Processing of DHT Node

This section explains the initial processing of a DHT node. At the end, the DHT node has two or more structured links and one or more DHT links. When a peer joins the network as a DHT node, the system uses five messages: Search Link Request, Search Link Reply, Search Link Reject, Structured Link Request, and Structured Link Reply. A message sender of the Search Link Request is a newly joined DHT node. Search Link Reply and Search Link Reject are the same as the initial processing of leaf node. A message sender

of the Structured Link Request is a newly joined DHT node and it includes the DHT node ID of the message sender. A message sender of the Structured Link Reject is a DHT node.

DHT Link Establishment

Similar to a leaf node, the Search Link Request is sent to the super node, and the Search Link Reply or the Search Link Reject will be received. However, different from the leaf node, newly joined DHT node establishes s DHT link not s search link.

Structured Link Establishment

The proposed method uses Chord as the structured search method. When a newly joined peer becomes a DHT node, the node establishes links as well as Chord. In HyDiff, the links based on Chord are called structured links. At first, the peer calculates hash value using its IP address and sets it as its own DHT node ID. Next, the Structured Link Request with the DHT node ID is sent to its known super nodes. The super node that received the Structured Link Request selects one of its DHT links and forwards the message. The DHT node that received the Structured Link Request forwards the message using the joining algorithm of Chord. DHT nodes selected by the Chord algorithm send the Structured Link Reply to the newly joined DHT node. The DHT node that received the Structured Link Reply establishes structured links to two or more DHT nodes contained in the message.

3.3 Search with Estimated Search-Target Popularity

3.3.1 Local Popularity Calculation

This section explains the local popularity calculation procedure. A super node collects the information of objects possessed by child nodes in order to makes a local object list and a popularity list. Here, child nodes mean leaf nodes and DHT nodes in neighbors of the super node. The local object list consists of sets of an object name and an object possessor. A super node uses the local object when the super node receives search queries,

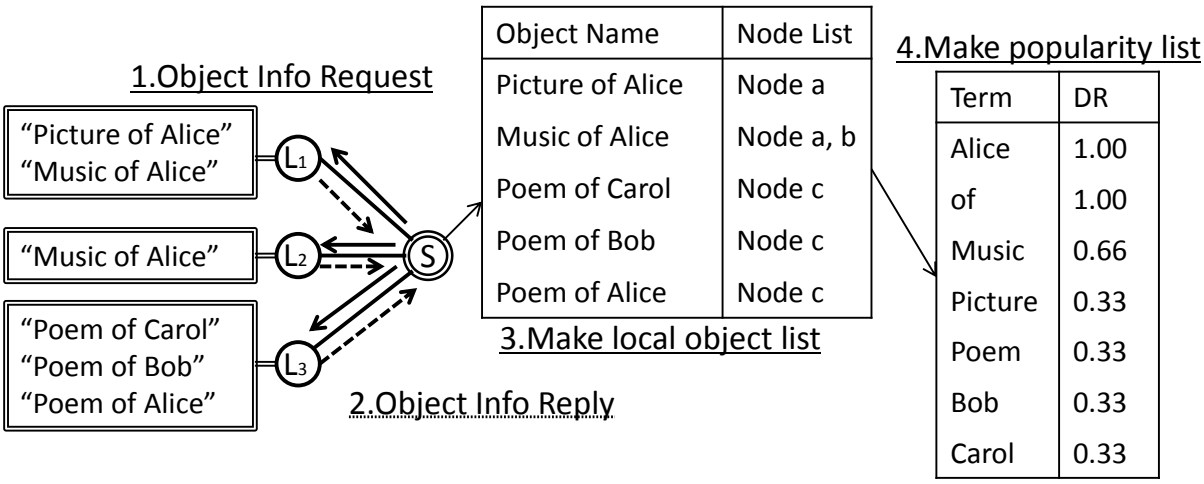

Figure 3.5: Making local popularity list

and the super node replies with requesters of the search queries when its own neighbors possess required objects. The popularity list consists of sets of a word and a popularity of the word, and the list shows how many peers possess objects whose names are included in each word. A starting value of popularity in the list is a local popularity in child nodes of one super node. Repeating update based on the Gossip algorithm, the popularity value moves in closer to a precise popularity entire network. This local popularity calculation process is executed periodically, a cycle of the local popularity calculation is called the popularity estimation interval. In the local popularity calculation, the system uses two messages: Object Info Request and Object Info Reply. Message sender of the Object Info Request is a super node. Message sender of the Object Info Reply is a leaf node or a DHT node and the message includes the information of object names possessed by the message sender.

First, a super node sends the Object Info Request to its own child nodes. The peers that received the Object Info Requests reply with the Object Info Replies including their possessing object names. The super node that received the Object Info Reply adds the object names and the object possessor to its own local object list. Moreover, the super node adds the words included in the object names and duplication numbers of the words to its own popularity list. The duplication number means how many peers possess the

objects whose names include the word.

Note that which peer is a search target of a search query is decided by only whether the peer possesses one or more search-target objects. The number of objects whose names include a word in a single peer has no effect on the popularity of the word. When many objects whose names including a word are possessed by only one peer, the probability of discovery of the objects is equal to the probability of discovery of one peer and the system cannot find such objects by using the unstructured search. Therefore, in HyDiff, the number of objects whose names include a word possessed by a single peer is ignored for popularity calculation.

Figure 3.5 shows an example of local popularity estimation. In the figure, Node S is a super node and Nodes L_1, L_2, and L_3 are child nodes of Node S. First, Super Node S sends the Object Info Requests to three child nodes, and the child nodes reply with the Object Info Replies. The Object Info Reply of Child Node L_1 includes "Picture of Alice" and "Music of Alice," and that of Child Node L_2 includes "Music of Alice," and that of Child Node L_3 includes "Poem of Carol," "Poem of Bob," and "Poem of Alice." Next, Super Node S makes its own local object list and popularity list. Note that Object Info Reply of Child Node L_3 includes three "Poem" and that of Child Node L_1 includes one "Picture," but the popularity of "Poem" and the popularity of "Picture" are the same value in the popularity list of Super Node S.

3.3.2 Popularity Estimation

This section describes how global popularity is estimated by local popularity. To estimate global popularity, the popularity of the whole network, the Gossip algorithm as a traditional method is applied. The Gossip algorithm is an effective distributed algorithm for calculating an average value of local values possessed by each peer [65, 66]. HyDiff supposes that a distribution of object positions is uniformly and adopts the Gossip algorithm to estimate popularity. An error of calculated average value in the Gossip algorithm is expected to be few when a variance of local values is small. This Gossip algorithm is applied based on schedule and to avoid the confusion with previous Gossip exchange, the exchanging values are decided by the Gossip sequence number. Therefore, the local popularity list used to exchange must have enough number of past lists. Moreover, a newly

joined peer does not know the Gossip sequence number. In that case, Gossip Exchange Request received from other super node must be set as initial value to participate in the Gossip algorithm.

First, a super node sends its local popularity list and the Gossip Exchange Request with the Gossip sequence number to randomly chosen super node from its super node list. The super node that received Gossip Exchange Request replies with its local popularity list corresponds to the Gossip sequence number and Gossip Exchange Reply containing the Gossip sequence number, and takes the average value between its own local popularity list corresponds to the Gossip sequence number in the received message and the other local popularity list. The super node that received the Gossip Exchange Reply also calculates the average value between the other's local popularity list in the message and its own local popularity list, and apart from its local popularity list it updates local popularity list based on the Gossip sequence number. At this point, when a word is included in the list of only one side, and then the average value using popularity zero is calculated. In addition to this, when the size of the Gossip local popularity list exceeds the upper limitation size of the Gossip local popularity and then popularity under search threshold are deleted. This continues until the ending condition of message exchange is met. The ending condition, the difference between the values after calculation and before calculation is under the set value of the Gossip threshold value. Moreover, when the Gossip Exchange Request corresponding to the Gossip sequence number is received, the above exchange method is conducted even in the finished Gossip local popularity list.

Figure 3.6 shows an example of popularity list exchange of each sequence number. Super nodes who exchange lists of same sequence update their lists to average of the lists. When a super node receives an old exchange request message, the super node replies with an old list. In the figure, Super Nodes S_1 and S_2 are calculating popularity of round i and Super Node S_3 is calculating popularity of round i-1. Gossip exchange request (i) means message of round i. popularity list (i) means popularity list of round i. When super nodes calculate average of their lists' values, they treat a value of word which is in only one list as zero.

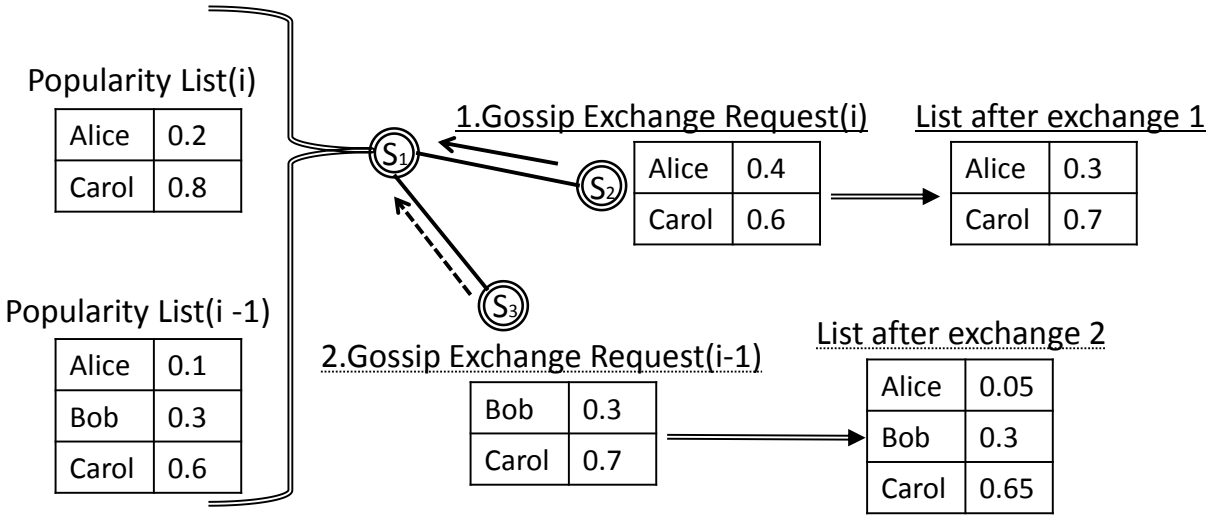

Figure 3.6: Popularity estimation

3.3.3 Search Method Selection

Decision of Utility Threshold for Search Method Selection

This section describes the procedure to calculate a search threshold value for selecting search method. The search threshold value must be calculated for selecting search method. As well as existing hybrid search methods, the search threshold value is calculated based on the utilities of the two types of search methods. The utility is calculated by Equation 2.1. The calculation of search threshold value starts when the calculation of global popularity is finished. First, each super node randomly selects one word with little-higher popularity than present threshold and one word with little-lower popularity than present threshold, and for each of them conducts the structured search and the unstructured search similar to normal search, However, this Search Query Request contains flag for assuming threshold value and transfer number field. The number of transfer is added as a node is transferred, and the value is written in Search Query Reply sent to the last super node that conducts search, When the search is the unstructured search, the transfer number field decreases as many copies of Search Query Request as made, and when TTL becomes zero Search Query Reply with object number set zero is sent to search execution node. From above,

the number of responses, the shortest response time, and the message cost in the search can be measured, and these values give the utility of each search using the equation 2.1 in Section 2.2.4.

The search threshold $U_{\text{threshold}}$ is calculated by Equation 3.1.

$$U_{\text{threshold}} = P_1 + \frac{(P_1 - P_2)(u_{\text{s1}} - u_{\text{u1}})}{u_{\text{u2}} - u_{\text{u1}} - u_{\text{s2}} + u_{\text{s1}}}, \tag{3.1}$$

where P_1 and P_2 are selected popularity value by the super node, u_{s1} is the utility of structured search for popularity P_1, u_{u1} is the utility of unstructured search for popularity P_1, u_{s2} is the utility of structured search for popularity P_2, and u_{u2} is the utility of unstructured search for popularity P_2, The utility values are calculated by Equation 2.1. Equation 3.1 calculates the intersection point of two lines: line $(P_1, u_{\text{s1}})(P_2, u_{\text{s2}})$ and line $(P_1, u_{\text{u1}})(P_2, u_{\text{u2}})$. The intersection point of two lines can be used as an approximate value of optimal threshold. By repeating this calculation, search threshold value lead to a more practical value. Based on the previous study, by repeating the calculation about twenty times not depending on the number of nodes, give assumed value to be closed to practical search threshold value [53].

In HyDiff, super nodes conduct searches for peers other than super nodes, and decide either of two search methods is used. Hence, initial procedure of search differs from nodes. First, when the node is a leaf node, Search Query Request is sent to the other super node of search link. When the node is DHT node, Search Query Request sent to the other super node of DHT link. The super node that received Search Query Request calculates the popularity of searched word as reference to finished Gossip local popularity list with latest Gossip sequence number. When a super node conducts search, it does not use Search Query Request, and it calculates the popularity of the searched word by itself. Moreover, when there exists a word not contained in the list, and the popularity of the word is considered as zero. When the calculated popularity is higher than search threshold value and then unstructured search is chosen, and when it is lower and then structured search is chose by super nodes.

Unstructured Search

To conduct unstructured search, Search Query Request with set TTL is sent to a super link. The super node that received Search Query Request from super link transfers Search Query Request to nodes with the same object through comparing its objects with the other. Otherwise, it transfers Search Query Request to super node with others' super link. The node that received Search Query Request sends Search Query Reply to the search execution node. Also, by decreasing the TTL count as transfer, super node stops transferring when the super node receives Search Query Request with TTL is zero.

Structured Search

To conduct structured search, Search Query Request is sent to DHT link. The DHT node that received Search Query Request, it calculates hash value using search word contained in the message. Then, Search Query Request is transferred to the node with closest hash value of DHT node ID and structured link. Following this, the DHT node that received Search Query Request repeats the same steps, and when it finally reaches the DHT node with exact same search word, Search Query Request is transferred to linking node that has the object from the final DHT node. At the end, when the object holding node that received Search Query Request replies with Search Query Request, and then the search is finished.

3.4 Simulation Setup

The proposed HyDiff is evaluated by comparison of the existing GAB [53] by using simulation. In this simulation, peers construct P2P networks like Gnutella and replicas of an object are distributed uniformly over the entire network. A popularity of one object is changed, and utility of searching for the object is calculated to evaluate. Moreover, in actual P2P networks, there exist *ultrapeer* which is cooperative node to P2P system. This simulation also includes some ultrapeers.

3.4.1 Simulation Parameters

Table 3.1 shows simulation parameters. In this simulation, the churn rate is static and an influence on the churn-rate change is not evaluated, because the churn rate has a similar impact on the proposed HyDiff as the existing GAB. Both methods uses the same utility function 2.1 to calculate efficiency of the structured search method and the unstructured search method. Therefore, in the both methods, the calculated utility values of the unstructured methods decrease similarly when efficiency of the unstructured search method greatly-decreases by heavy churn. Therefore, in this simulation, the churn rate do not changes. Request rate change interval means a degree of request rate change. In this simulation, peers request the search with a normal probability distribution. The standard deviation of the normal probability distribution is one-half of the request rate change interval. 95% of nodes search for an object during 20 hours when the object of request rate change interval is 10 hours, This request-rate change describes a change of contents like News that is requested by a large number of users in a short period of time [67]. This request-rate changes of this simulation is one of the situations that may be encountered. Estimation interval means the request rate estimation interval in GAB and the popularity estimation interval in proposed method. Other parameters are selected corresponding to the existing work [53].

3.4.2 Simulation Scenario

Simulation scenario is shown as follows.

1. Simulation runs over a joining of peer until a number of peers is equal to the maximum number of peers.
2. Simulation runs five hours without changing popularity at first for a converging of the popularity threshold.
3. Simulation selects a peer at random, and then the peer possesses a particular object. The first possessor peer does not delete the object.
4. Each peer searches for and transfers the object depending on the request rate change interval.

Table 3.1: Simulation parameters

Parameter	Value
Simulation time	96 hours
Maximum number of peers	10,000 peers
Node join/leave rate (churn rate)	0.75 peer/second
Ultrapeer ratio	5%
Average number of object	10 objects/peer
Gossip threshold	0.01
Maximum number of reply ($R_{\max}$)	25
Request rate change interval	10-40 hours
Average object keep time	4 hours
Estimation interval	40-3,600 seconds
Super node ratio	0.1
Structured node ratio	0.1

Peer behavior is shown as follows.

- Some peers become ultrapeer, they do not delete their file.
- Peers search for objects depending on Zipf distribution.
- Peers delete the file depending on an average object keep time.
- Peers leave from network and join as newly joined peer depending on node join/leave rate.

In the simulation, request rate starts to increase at five hours and, request rate is maximum at request rate change interval plus five hours. For example, request rate is maximum at fifteen hours when the request rate change interval is ten hours. The reason of that request rate is not changed in five hours is waiting for the convergence of searching threshold calculation. Increasing the request rate means increasing the number of search request by peers. When the request rate change interval is small, a number of searching request per each time is high, and request rate is rapidly changed.

Figure 3.7 shows changes of object popularity when request rate change interval are ten, twenty, and forty hours. As shown by the figure, request rate and object popularity

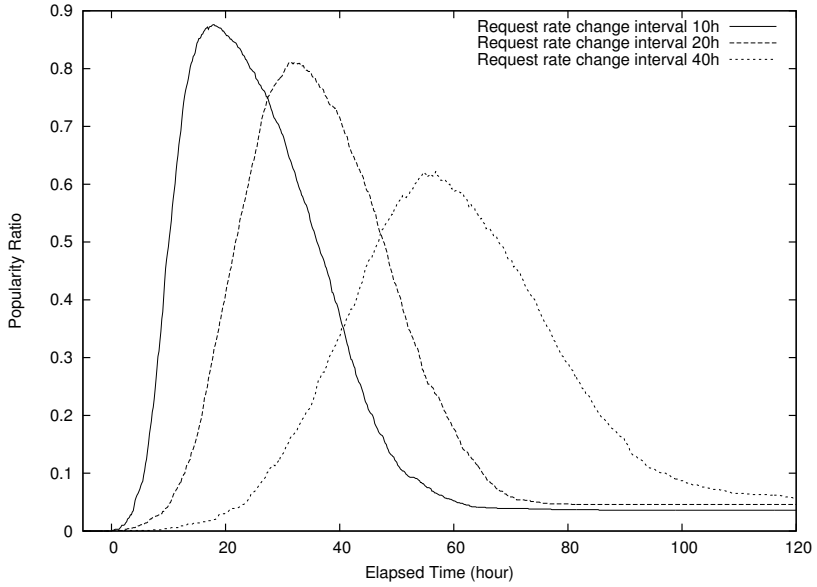

Figure 3.7: Object popularity

is change more rapidly when request rate change interval is smaller. Object popularity is still 0.05 after 120 hours because ultrapeers, whose participating ratio is 0.05, do not delete object. In the figure, there exists the difference between the peak of popularity and the peak of request rate. Because peers search for only peers that possess the complete cash of the desired data in this simulation. In P2P systems, data are divided into the small pieces and peers receive data in a unit of a piece. The complete cash is all pieces of data. When a peer finds peers that possess few pieces, the peer cannot always obtain the complete data. Therefore, only peers that possess the complete cash respond to search queries to ensure that a peer can obtain complete data when the peer receive reply messages [72]. This simulation assumes that peers need around four hours to obtain the complete cash. The difference between the peak of popularity and the peak of request rate decreases when the time of obtaining the complete cash is shortened.

3.5 Performance Evaluation

3.5.1 Evaluation Items

Following evaluation items are used for the proposed HyDiff and the existing method GAB [53] with periodic popularity estimation.

Utility Difference Ratio from Theoretical Value

The utility difference ratio from theoretical value $U_{\text{difference}}$ is found by

$$U_{\text{difference}} = \frac{\sum_{x \in X} u_{\text{theoretical}}(x) - \sum_{x \in X} u_{\text{search}}(x)}{\sum_{x \in X} u_{\text{theoretical}}(x)}, \tag{3.2}$$

where X is a set of all search, proposed search utility of Search $x (x \in X)$ is found by function $u_{\text{search}}(x)$, optimal search utility of Search $x (x \in X)$ is found by function $u_{\text{theoretical}}(x)$. The lower utility difference ratio is, the better search efficiency is.

Average Search Message Cost

The average search message cost $C_{\text{search}}(t)$ is found by

$$C_{\text{search}}(t) = \frac{\sum_{x \in X_t} c_{\text{search}}(x)}{|X_t|}, \tag{3.3}$$

where X_t is a set of all search in the time zone t, $c_{\text{search}}(x)$ is the number of messages related to Search $x (x \in X)$.

Average Shortest Response Time

The average shortest response time $R_{\text{shortest}}(t)$ is found by

$$R_{\text{shortest}}(t) = \frac{\sum_{x \in X_t} r_{\text{shortest}}(x)}{|X_t|}, \tag{3.4}$$

where X_t is a set of all search in the time zone t, $r_{\text{shortest}}(x)$ is the shortest response time for Search $x(x \in X)$. The shortest response time $r_{\text{shortest}}(x)$ is found by

$$r_{\text{shortest}}(x) = \min\{r_{\text{response}}(y) | y \in Y(x)\}, \tag{3.5}$$

where $Y(x)$ is a set of all response for Search $x(x \in X)$, $r_{\text{response}}(y)$ is a response time of Response $y(y \in Y(x))$.

Response Ratio

The response ratio F_{response} is found by

$$F_{\text{response}} = \frac{N_{\text{response}}}{|X|}, \tag{3.6}$$

where X is a set of all search and N_{response} is a number of all search which finds one or more object.

Maintenance Cost

Hybrid search methods have to maintain DHT topology and updating object information when any peers do not use search function. The maintenance cost $C_{\text{maintenance}}$ is a message cost for maintaining DHT topology and updating object information.

3.5.2 Search Utility

Figure 3.8 shows an average utility difference ratio $U_{\text{difference}}$ of each methods when the estimation interval changes. A vertical axis expresses the average utility difference ratio $U_{\text{difference}}$ and horizontal axis is the estimation interval.

At first, GAB that is the existing method is discussed. As shown by the figure, the maximum utility difference ratio $U_{\text{difference}}$ is near 0.8 in any request rate change interval. The existing method counts each search keyword for popularity estimation, and the counts are reset in the next estimation. Therefore, when estimation interval is smaller than a interval of peers' requesting, estimated results of each interval are various and system performance decreases. Moreover, when the estimation interval is around 250 seconds,

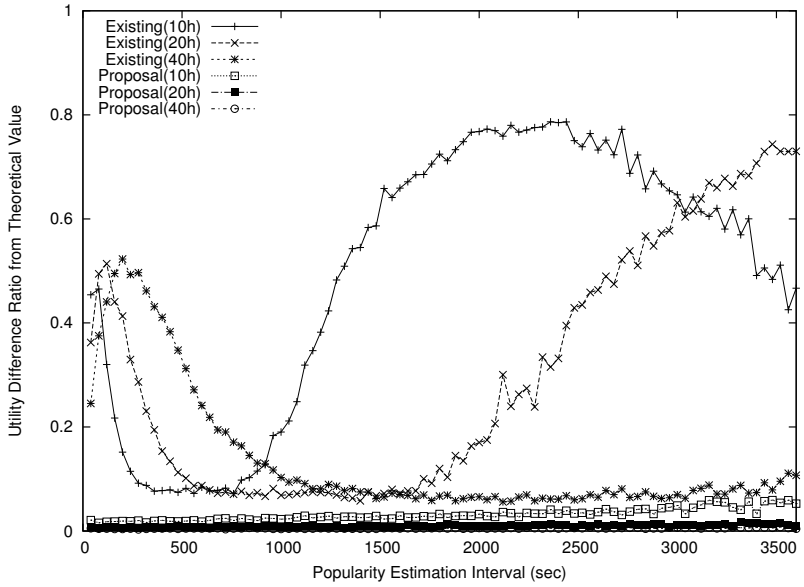

Figure 3.8: Utility difference ratio from theoretical value

utility difference ratio $U_{\text{difference}}$ reaches to 0.5. $U_{\text{difference}}$ of the existing method varies depending on the combination of the request rate change interval and the popularity estimation interval. As shown in the figure, in the existing method, the range of effective estimation interval varies depending on the request rate change interval. It is difficult to decide the value of the estimation interval because the request rate change of the search-target is unpredictable.

Next, the proposed HyDiff is discussed. In the case of any request rate change interval, HyDiff is stable for the existing method and a utility difference ratio becomes small. In addition, utility difference ratio $U_{\text{difference}}$ of proposed method decreases with decreasing the estimation interval in any request rate change interval. When the request rate change interval is 20 hours and the estimation interval is under $1,200$ seconds, when the request rate change interval is 40 hours and the estimation interval is under $2,800$ seconds, utility difference ratio $U_{\text{difference}}$ is under 0.01. Therefore, the estimation interval can be set low value as far as the system performance permits in HyDiff.

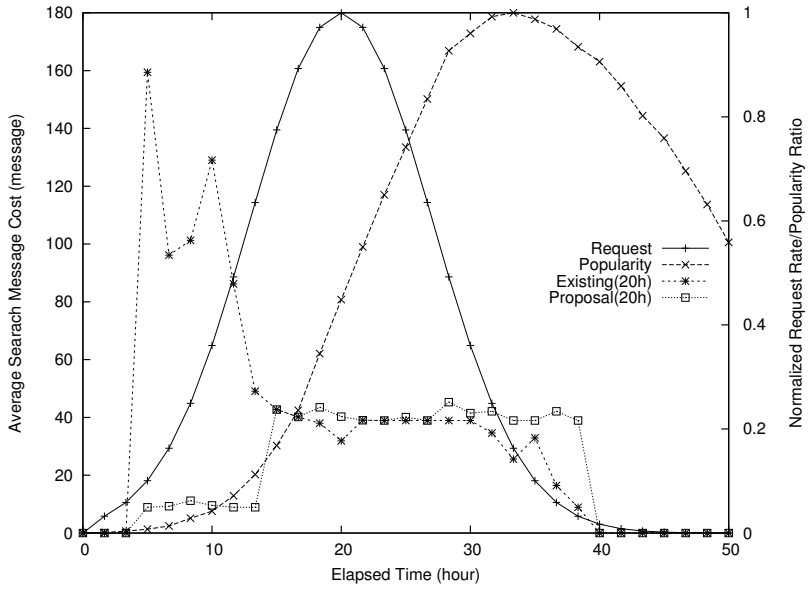

Figure 3.9: Average search message cost $C_{\text{search}}(t)$

3.5.3 Average Search Message Cost $C_{\text{search}}(t)$

Figure 3.9 shows the average search message cost $C_{\text{search}}(t)$ of search queries in each time zone, normalized popularity of the search-target object, and normalized request rate of the search-target object when the request rate change interval is 20 hours and the estimation interval is $1,200$ seconds. The values of the request rate change interval and the estimation interval are selected from a range that the efficiencies of both methods is the highest in Figure 3.8. Therefore, in the figure, it is assumed that the existing method can adopt the appropriate estimation interval. The request rate and the popularity are normalized by the each maximum values.

The average search message cost $C_{\text{search}}(t)$ of the existing method is much higher than that of HyDiff when the simulation time is lower than thirteen hours. Many peers begin to send search queries and the request rate increases. The existing method selects the unstructured search methods for the queries because the request rate of search target is high. Most part of the unstructured searches, however, fails because the popularity of the search target is still low. In both methods, super nodes search again using the structured

search after the unstructured search fails. As a result, the existing method generates a large number of messages. On the other hand, the search message cost $C_{\text{search}}(t)$ of HyDiff is under ten messages when the simulation time is lower than thirteen hour. The reason is that HyDiff selects the structured search because HyDiff can estimate more precise search-target popularity. From thirteen hours to thirty hours, both methods select the unstructured search and it can find the search target because the popularity is much higher. The search message cost $C_{\text{search}}(t)$ of HyDiff is little higher than that of the existing method when the simulation time is from thirty hours to forty hours. Because the request rate decreases and the existing method selects the structured search. The utility function for the search method selection consists not only the search message cost $C_{\text{search}}(t)$ but also others. Therefore, HyDiff still selects the unstructured search. However, as shown in the figure, the difference between message cost $C_{\text{search}}(t)$ is few when the simulation time is from thirty hours to forty hours. Moreover, the average message cost $C_{\text{search}}(t)$ of all search queries for HyDiff is lower than that of the existing method. The average search message cost $C_{\text{search}}(t)$ of all search queries for HyDiff is 32.7 messages and that for the existing method is 49.2 messages. The proposed method decrease the average search message cost $C_{\text{search}}(t)$ of all queries by 33% in this simulation.

3.5.4 Average Shortest Response Time $R_{\text{shortest}}(t)$

Figure 3.10 shows the average shortest response time $R_{\text{shortest}}(t)$ of search queries for each time, normalized popularity of the search-target object, and normalized request rate of the search-target object when the request rate change interval is 20 hours and the estimation interval is $1,200$ seconds. Here, the shortest response time $R_{\text{shortest}}(t)$ is defined as the number of hops because the time of 1 hop depends on under-layer networks topology.

The average shortest response time $R_{\text{shortest}}(t)$ of the existing method is very slow then the simulation time is under fifteen hours. The reason is that the existing method selects the unstructured search as well as Figure 3.9. When the search-target popularity is low, the unstructured search cannot find the object and peers search again using the structured search after failure of the unstructured search. Therefore, the average shortest response time $R_{\text{shortest}}(t)$ of the existing method is higher than that of proposed method quadruple. When the simulation time is from thirty-five hours to forty hours, the existing

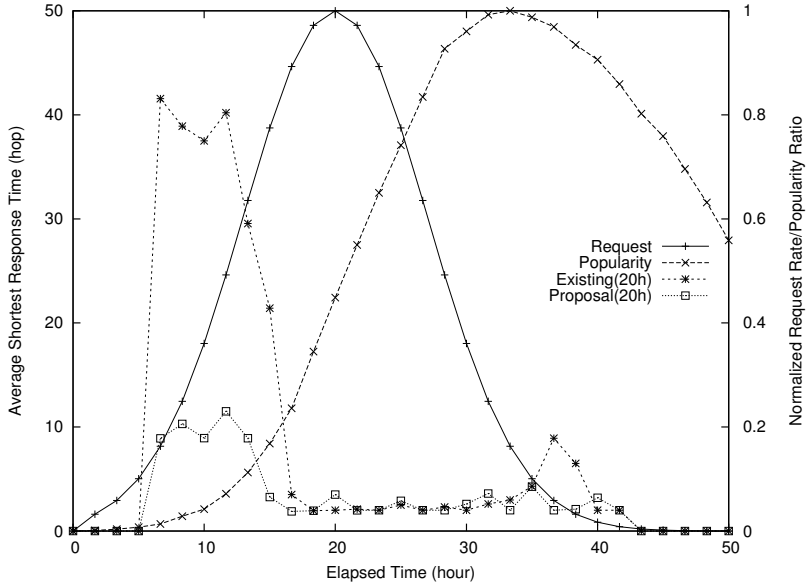

Figure 3.10: Average shortest response time $R_{\mathrm{shortest}}(t)$

method select the unstructured search as well as Figure 3.9. On the other hand, HyDiff select the unstructured method and the average shortest response time $R_{\mathrm{shortest}}(t)$ is lower than that of the existing method because the target-object popularity is enough high. As shown in Figure 3.9 and 3.10, when the simulation time is from thirty hours to forty hours, HyDiff use higher search message cost and faster shortest response time than the existing method. In other words, the shortest response time $R_{\mathrm{shortest}}(t)$ and the search message cost $C_{\mathrm{search}}(t)$ have a trade-off relationship when the request rate is much low and the popularity is enough high. Finally, the average shortest response time of all search queries for proposed method is 3.49 hops and that for the existing method is 11.1 hops. The proposed method decreases the average shortest response time $R_{\mathrm{shortest}}(t)$ of all search queries by 70% in this simulation.

A time length of one hop is related to a topology of under layer network. Assuming the situation that the P2P network is applied on a global scale, a time length of one hop is about 200 milliseconds. In the situation, HyDiff reduces $R_{\mathrm{shortest}}(t)$ from around eight seconds to around two seconds when search-target's request rate is high but its popularity is very low.

3.5.5 Impact on Request Generating with Fixed Probability

When search requests are generated with a fixed probability, the existing method that uses the request rate tends to keep high efficiency. To discuss a weakness of HyDiff, it is evaluated under such disadvantageous condition. The evaluation item is the response ratio, the average shortest response time, the average search cost, and the maintenance cost.

Detailed maintenance cost is shown by follows.

- DHT maintaining cost
  DHT peer leave from networks without notice. Therefore, DHT peer check dead or alive of peers connected by structured link periodically. Moreover, when a leaving of DHT is detected, DHT nodes re-construct DHT topology. The number of messages for this check and re-constructing DHT is DHT maintaining cost.

- Monitoring cost of leaf node
  Super peers collect object information of leaf nodes each estimation interval. The number of messages for this monitoring is monitoring cost.

- Gossip cost
  Super peers exchange local popularity list each estimation interval. The number of messages for this exchange is gossip cost.

- Majoring cost of popularity threshold
  Super nodes major popularity threshold each estimation interval. The number of messages for this majoring is majoring cost of popularity threshold.

Most part of maintenance cost is DHT maintenance cost and gossip cost. Influence of other cost is very small.

Table 3.2 shows results normalized by results of the GAB for the response ratio F_{response}, the average shortest response time $R_{\text{shortest}}(all)$, the maintenance cost, the search cost $C_{\text{search}}(all)$ when object popularity is not changed. As shown in the table, the efficiency of HyDiff and that of the existing method is the same values when object popularity is not changed. The difference between the maintenance cost of HyDiff and that of the

Table 3.2: Evaluation on request generating with fixed probability

Method	Response ratio	Average shortest response time	Maintenance cost	Average search cost
Existing	99.9	1.00	1.00	1.00
Proposal	99.9	0.99	1.01	1.00

existing method is only 1%, and it is very few. The reason is that the DHT maintaining cost is much higher than the monitoring cost of local popularity.

3.6 Summary

This chapter proposes HyDiff, Hybrid search based on Diffusion ratio, which uses search-target popularity estimated by local popularity. The proposed method is a hybrid search method that acts as a structured or unstructured search method, depending on search-target popularity. In hybrid search methods, the method of estimating search-target popularity is very important because efficiencies of the structured and unstructured search methods vary depending on the search-target popularity. Additionally, the popularity of the search target changes with time and it is difficult to predict the time change of the search-target popularity. Therefore, in hybrid search methods, peers must calculate the popularity periodically. Existing hybrid search methods use the request rate to estimate search-target popularity. However, it is difficult to retain high efficiency because in some cases there exists a large gap between the measured request rate and popularity. For example, when peers do not request searching in a period of time, super nodes always select a structured search method in the period even when the actual popularity of the search target is high. Therefore, the estimated popularity by request rate do not always properly reflect the actual popularity. The proposed method bases its popularity estimation on local popularity. As a result, search efficiency is expected to be retained in HyDiff.

The key idea of HyDiff is that super nodes use not the request rate of the search-target object but the popularity of the search-target object for search-target popularity estimation. Hybrid search methods including HyDiff maintain two-difference type of

search methods: the structured search method and the unstructured search method. Then, it is very important to select an effective search method depending on the search target popularity. Existing methods treat the request rate the same way as the popularity implicitly, and use the request rate of objects for the search-target popularity estimation. The request rate and the popularity are, however, different strictly. A request rate of an object means how many peers want to obtain the object, and a popularity of an object means how many peers possess the object. Therefore, a gap between the estimating popularity by the request rate and the actual popularity is generated in some cases. The proposed method adopts the local popularity to bridge the gap between the estimated popularity and the actual popularity. Moreover, super nodes can easily-calculate the local popularities because the network topology is based on the Gnutella network, which use for hybrid search methods generally. The proposed method tends to keep the effective search function with a small increase in the maintenance cost when there exists a large gap between the request rate and the popularity.

The proposed HyDiff is evaluated by computer simulation in terms of the average response rate F_{response}, the average shortest response time $R_{\text{shortest}}(t)$, and the search message cost $C_{\text{search}}(t)$. In the simulation, a distribution of object positions is uniformly. Then, a popularity of one object is changed, and utility of searching for the object is calculated to evaluate. At first, HyDiff is evaluated by using the utility function shown by Equation 2.1. The utility difference ratio from theoretical value $U_{\text{difference}}$ means the difference between the actual utility and theoretical value. The theoretical value is the utility of theoretical optimal hybrid search that always selects a reasonable search method. One hand, $U_{\text{difference}}$ of the existing method varies depending on the combination of the request rate change interval and the popularity estimation interval. In the existing method, it is difficult to decide the value of the estimation interval because the request rate change of the search-target is unpredictable. One the other hand, $U_{\text{difference}}$ of HyDiff decreases with decreasing the estimation interval. Therefore, the estimation interval can be set low value as far as the system performance permits in HyDiff. Additionally, when the search target's request rate change interval is twenty hours and estimation interval is $1,200$ seconds, the utility difference ratio $U_{\text{difference}}$ of HyDiff is under 1%. Therefore, HyDiff selects a reasonable search method in most cases. In addition, this dissertation studies the shift of

the average search message cost $C_{\text{search}}(t)$ for one search and the average shortest response time $R_{\text{shortest}}(t)$ by each time zone in one of the situations that the request-rate change is like News that is requested by a large number of users in a short period of time. In the result, HyDiff greatly-improves the average message cost $C_{\text{search}}(t)$ for one search and the average shortest response time $R_{\text{shortest}}(t)$ from the existing method soon after the request rate of the search-target object starts to increase. Because HyDiff rightly-selects the structured search method for an object whose request rate is high but popularity is low. Moreover, the maintenance cost of HyDiff is evaluated compared with that of the existing method. Then, the simulation result showed that the increase in the maintenance cost is few and it is not serious problem. Finally, those results showed that HyDiff maintains search efficiency under the situation that there exists a large gap between the measured request rate and the popularity.

Chapter 4

Piece Lending for Video Data Transmission Scheduling

4.1 Overview

This chapter proposes Piece Lending, which uses a lending approach in which the P2P system lends receiving rights for low-popularity pieces to newly joined peers, and collects contributions of advanced peers in compensation for using rights when newly joined peers become advanced peers. The proposed method is an incentivized method with low-popularity piece push. Incentivized methods provide incentives (receipt rights) for piece sending, preventing free-riders from receiving pieces. In existing methods, low-popularity piece push from the video provider is required for newly joined peers to join in the piece exchange, because newly joined peers do not possess transmittable piece, making their negotiation power very low. In the proposed method, P2P systems provide receiving rights for low-popularity pieces to newly joined peers, thus increasing their negotiation powers. Newly joined peers use these receiving rights to receive low-popularity pieces, allowing them to take part in piece exchange without support from the video provider. Because the most part of peers do not possess low-popularity piece, newly joined peers can send that pieces to peers that do not possess it. Moreover, advanced peers with high negotiation power pay their contribution to the P2P system in compensation for previously borrowing rights; in other words, they send low-popularity pieces to newly joined peers depending on

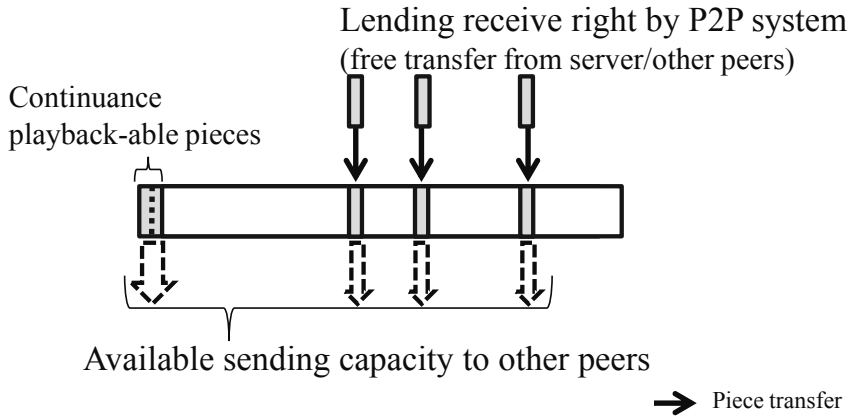

Figure 4.1: Receive buffer of newly joined peer

the amount of pieces that they have received using the borrowed rights. As a result, the proposed method distributes the load of low-popularity piece pushing to peers. Efficiency of the proposed method in terms of the utilization ratio of peer transmission capacity and start-up latency is shown by simulation.

4.2 Piece Lending Approach

4.2.1 Concept of Piece Lending

Figure 4.1 and Figure 4.2 shows receive buffers of two peers: a newly joined peer and an advanced peer. The newly joined peer possesses few pieces and the advanced peer possesses many continuance playback-able pieces. The concept of piece lending is that the P2P system lends a piece receiving right to the newly joined peer and the P2P system collects contribution whose amount is equivalent to the lent right from the advanced peer. Then, the distribution of low-popularity pieces for newly joined peers is supported by not only video provider but also the whole P2P system. Here, the whole P2P system means all peers and the video provider on P2P networks. The P2P system lends piece receiving rights to newly joined peers, in other words, the P2P system allow the newly joined peers to receive low-popularity pieces independently of their own contribution from

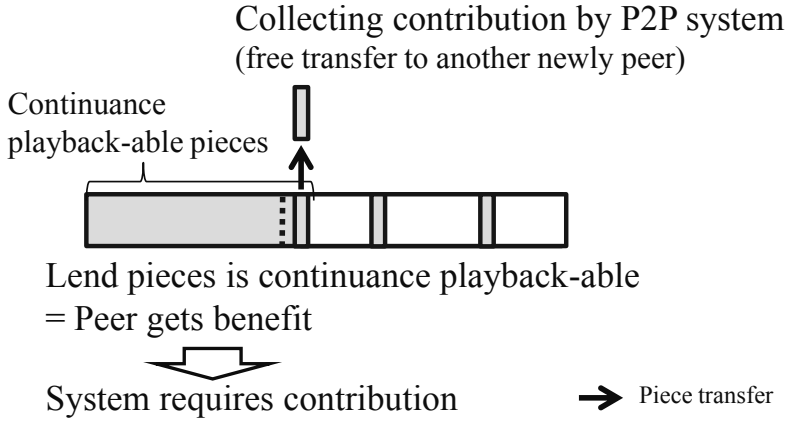

Figure 4.2: Receive buffer of advanced peer

other peers and the video provider. As a result, the newly joined peers receive new pieces and increase the number of their send-able pieces even when they cannot send pieces to other peers. The newly joined peers can easily-join in the piece exchange. When peers receive enough in-order pieces and their received pieces without their own contribution become continuance playback-able pieces, the P2P system collects contribution of them, in other words, the P2P system makes them send pieces without a payment in order to maintain the P2P system. In the contribution collection by the P2P system, the P2P system makes the advanced peers send pieces to the newly joined peers. Therefore, this proposed method realizes the concept of piece lending which is that the distribution of low-popularity pieces for newly joined peers is supported by not only video provider but also the whole P2P system.

4.2.2 System Design

To realize the concept of piece lending, the P2P system needs a structure fulfilling three conditions. First condition is that the P2P system allows only newly joined peers to receive pieces without piece sending. Second condition is that the P2P system requires piece sending whose amount depends on the number of received pieces of peers without newly joined peer. Third condition is that the number of sent pieces of a peer is equivalent to the number of received pieces of the peer when the peer completes collecting all pieces.

The proposed method defines the amount of peer's benefit for the incentivized algorithms as the number of peer's continuous playback-able pieces. The number of peer's continuous playback-able pieces is called a *benefit value*. Additionally, the proposed method defines the amount of peer's contribution to the P2P system as the number of pieces that had sent by the peer. The number of the sent pieces is called a *contribution value*. The proposed method uses the benefit value and contribution value for the incentivized algorithms in order to realize the concept of piece lending. In the proposed method, just receiving pieces of peers is not a benefit of the peer. When the receiving pieces become continuous playback-able, the P2P system see that the peer gets a benefit. As a result, in the propose method, newly joined peers can receive low-popularity pieces even when they do not contribute to the P2P system. Then, they collect in-order pieces by using peer's piece exchange with received low-popularity pieces. When the received low-popularity pieces become continuous playback-able, the benefit value increases. Peers cannot receive new pieces when their contribution value is lower than their benefit value by the control of incentivized algorithm. Therefore, the P2P system can require the piece sending whose amount is equivalent to the number of received low-popularity pieces of the advanced peers.

Figure 4.3 shows a structure of piece lending using the benefit value. In the figure, b_i is the benefit value of peer i and s_i is the contribution value of peer i. Left receive buffer is a newly joined peer's one and right one is an advanced peer's one. Newly Joined Peer i can receive low-popularity pieces without any relation to its own contribution value s_i. Then, Peer i can join peers in piece exchange because the number of the peer's send-able pieces increases by received low-popularity pieces. Additionally, in the proposed method, each peer cannot receive in-order pieces when their contribution value is lower than their benefit value in any case. Therefore, when contribution value $s(j)$ is lower than benefit value b_i, Advanced Peer j sends low-popularity pieces to newly joined peers in order to increase its own contribution value $s(j)$ Advanced Peer j can receive in-order pieces when s_i is larger than b_i.

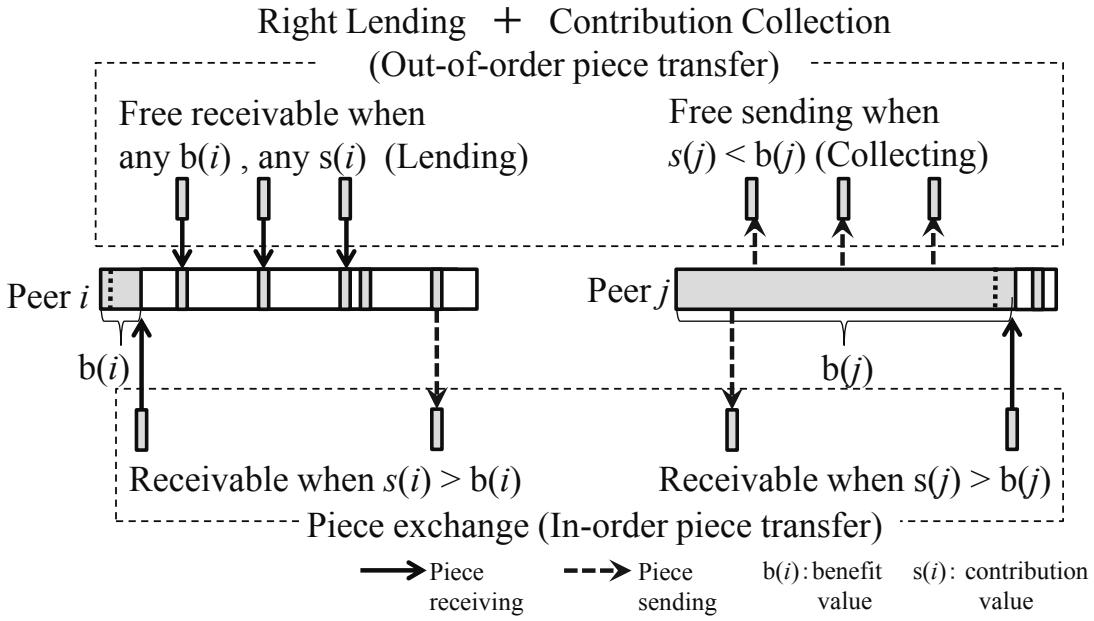

Figure 4.3: Benefit value and lending method

4.3 Neighbor Selection

Each peer is connected with randomly chosen peers as their neighbors in the proposed method. Peers should have heterogeneous neighbors in terms of playback progress because advanced peers should send low-popularity pieces to newly joined peers by rarest-first in order to increase utilization of peer transmission capacity. Moreover, advanced peers can request in-order pieces to various neighbors because all peers can have low-popularity pieces, which are often pieces of the video's last part and it are in-order pieces for advanced peers. Therefore, each peer is connected with randomly chosen peers as their neighbors.

When peers join in the system, they use the peer search function in order to tracker node of desired video contents. Generally, the video provider has also a role of the tracker node. The tracker node introduce randomly chosen peers to the newly joined peers and the peer offer introduced peers linking with each other as neighbors by neighbor offering message. When each peer has neighbors as many as $N_{\max}$, the peer rejects linking with other peers if the peer receives the neighbor offering messages. When each peer has fewer

neighbors than $N_{\min}$, the peer sends the neighbor offering messages to peers who are the neighbors of neighbor. Moreover, each peer periodically changes neighbors into new neighbors whom tracker introduces. In the proposed methods, the tracker only introduces some peers to the newly joined peers. In the assumed P2P VoD systems, any peer which knows some neighbors related to a video content can operate as the tracker of the video content as well as bootstrappers.

4.4 Piece Transmission Scheduling

4.4.1 Piece Transmission Policy of Viewer

Piece Lending uses two types of piece transfer: a low-popularity piece transfer and an in-order piece transfer. The low-popularity transfer is started from a request of a sender side peer, and a transferred piece is low-popularity pieces for receiver side peer. The receiver side peer is always a newly joined peer. In the low-popularity transfer, receiver side peer cannot select which piece is transferred, but contribution of the receiver side peer is not required. On the other hand, the in-order piece transfer is started from a request of a receiver side peer, and a transferred piece is the in-order piece for receiver side peer as well as piece transmission scheduling methods of general P2P VoD. In the in-order piece transfer, the receiver side peer receives its desired in-order pieces, but enough contribution of the receiver side peer to the P2P system is required. Peers divide their own transmission capacity into two capacities: a transmission capacity for in-order piece transfer and a transmission capacity for low-popularity piece transfer. The transmission capacity for in-order piece transfer u_{in} and the transmission capacity for out-order piece transfer u_{low} are follows;

$$u_{\text{in}} = u(1 - r_{\text{o}}), \tag{4.1}$$

$$u_{\text{low}} = ur_{\text{o}}(0 \leq r_{\text{o}} \leq 1), \tag{4.2}$$

where r_{o} is a ratio of u_{low} divided by the total transmission capacity u. r_{o} means peers uses their own resources for the distribution of low-popularity pieces for newly joined peers.

Low-Popularity Piece Transfer

In the low-popularity piece transfer, peers send pieces to peer i only when $b_i < fE$ $(0 \leq f \leq 1)$. Where, E is the divided piece number of video data, f is a ratio of receive-able low-popularity pieces. In other words, peer i is treated as a newly joined peer when contribution value b_i of the peer i is lower than fE in proposed method. Newly joined peer whose b_i is low can join peers in piece exchange early because the newly joined peer can receive low-popularity pieces. Which piece is transferred is decided by the Rarest-first algorithm [73]. Piece sending algorithm of peers in the low-popularity piece transfer is shown in Algorithm 1.

Referring to buffermap, peers send pieces by low-popularity piece transfer when the peers have request-able peers whose contribution value is lower than fE. When the number of the request-able peers is two or more, peers send request message to them when their own transmission capacity for low-popularity piece transfer u_{low} is enable. The transmission capacity for in-order piece transfer u_{in} is not utilized for low-popularity piece sending even when the transmission capacity for in-order piece transfer u_{in} is enabled. The reason is that peers cannot continuous playback of the video when the transmission capacity for in-order piece transfer u_{in} is exhausted. Which piece is transferred is decided by the Rarest-first algorithm. Peers refer to all buffermaps of neighbors and selects lowest-popularity pieces that is not possessed by most part of the neighbors. Newly joined peer can easily-join peers in piece exchange by using the received lowest-popularity piece. Moreover, it is deterrence for free riders because the receiver side peer cannot select a transferred piece in the low-popularity piece transfer. In the video streaming, a peer must collect pieces whose IDs are continuously in order to view high-quality video. Free riders can receive pieces by using the low-popularity piece transfer, but the received pieces are not continuous playback-able. Therefore, free riders cannot view the video continuously even when they use the low-popularity piece transfer.

In-Order Piece Transfer

In the in-order piece transfer, peers send pieces to peer i when the contribution value s_i is higher than the benefit value b_i. A transferred piece is decided based on b_i of receiver

Algorithm 1 Transmission algorithm of Sender Peer s in the low-priority piece transfer

INPUT: u^s_{low} : Transmission capacity of Peer s for low-popularity piece transfer
INPUT: N^s_{all} : Array of all neighbors of Peer s
INPUT: $N_{possessor}$: Array of the number of possessor for each pieces
INPUT: E : The divided number of video data
INPUT: f : The ratio of receive-able low-popularity pieces
INPUT: $n^s_{neighbor}$: The number of neighbors of Peer s
OUTPUT: u^s_{add} : Additional capacity of Peer s for in-order piece transfer

for $i = 0$ to E **do**
  $N_{possessor}[i] \leftarrow 0$
  for $j = 0$ to $n^s_{neighbor}$ **do**
    if Peer $N^s_{all}[j]$ possesses Piece i **then**
      $N_{possessor}[i] \leftarrow N_{possessor}[i] + 1$
    end if
  end for
end for
$u^s_{add} \leftarrow u^s_{\text{low}}$
for $i = 0$ to $n^s_{neighbor}$ **do**
  if $b_{N^s_{all}[i]} < fE$ and $u^s_{add} > 0$ **then**
    $u^s_{add} \leftarrow u^s_{add} - 1$
    $RaresetPiecePossessorNum \leftarrow n_{neighbor} + 1$
    for $j = 0$ to E **do**
      if $N_{possessor}[j] < RaresetPiecePossessorNum$ **then**
        $RarestPieceID \leftarrow j$
        $RaresetPiecePossessorNum \leftarrow N_{possessor}[j]$
      end if
    end for
    Peer p send Piece $RarestPieceID$ to Peer $N^s_{all}[i]$
    $N_{possessor}[RarestPieceID] \leftarrow N_{possessor}[RarestPieceID] + 1$
  end if
end for

side peer i. Algorithm 2 shows the transmission algorithm in the in-order piece transfer.

Proposed method constructs a P2P network base on random mesh topology and peers exchange buffermap with each neighbors periodically as well as the existing method [15]. Referring to the buffermap, peer i sends a request message with contribution value s_i to peer that possesses pieces whose ID is $b_i + 1$. In the proposed method, the system allocates piece ID to pieces in order of playback time. Therefore, when peers receive pieces in ascending order of piece ID, the peers can begin to view the video before completing the collection of all pieces. Peer i should receive a piece which ID is $b_i + 1$ because the benefit value b_i means the number of continuous playback-able pieces. When the number of request-able peers, which possess the piece which ID is $b_i + 1$, is two or more, peer i selects one peer and sends a request message to the selected peer. When the request message is rejected by the selected peer, peer i send a request message to another request-able peer again. In addition, when peer i have no request-able peer or all requests of the peer i are rejected, the peer i send a request message to video provider. Peer received the request message checks the contribution value s_i of the requester peer i and benefit value b_i, and the peer sends the requested piece to peer i only when $s_i > b_i$. When a peer receives multiple request messages, the peer accepts the requests when its own transmission capacity for in-order piece transfer u_{in} is enabled. When the transmission capacity for in-order piece transfer u_{in} is not enabled, the peer rejects requests. Here, when transmission capacity for low-popularity piece transfer u_{low} of a peer is enabled and the peer has no request-able neighbor for low-popularity transfer, the peer can utilize the transmission capacity for low-popularity piece transfer u_{low} for the in-order piece transfer.

Malicious free riders try to use in-order piece transfer by telling a lie that declared benefit value is larger than precise value. If a malicious free rider underclaims the benefit value b_i for in-order piece transfer, the transferred piece's ID is same as previously-received piece. Therefore, the malicious free rider has no motivation to underclaim the benefit value b_i. On the other hand, a malicious free rider tries to tell a lie about the contribution value s_i. As a prevention against the false declaration, the P2P system can use the tracker, which is a special resident node, for monitoring the numbers of sent pieces for each peer, or trust mechanize [74, 75] for referring to degree of each peer's trust. Further discussion is beyond the scope of this dissertation.

4.4.2 Piece Transmission Policy of Video Provider

Most part of video provider's sending algorithm is as same as peer's sending algorithm. The difference of them is that the video provider allows any requests of in-order piece transfer. Peers can receive pieces from the video provider without any contribution. Piece sending algorithm of the video provider in the in-order piece transfer is shown in Algorithm 3. Piece sending algorithm of the video provider in the low-popularity piece transfer is as same as Algorithm 1.

4.5 Simulation Setup

This section evaluates proposed Piece Lending comparing with the existing incentivized method with low-popularity piece push, Server-Push method [15] by using round-based simulation.

4.5.1 Simulation Parameters

Table 4.1 shows basic simulation parameters. Those parameters are set by referring to existing other works for P2P VoD streaming [10, 15, 17]. In this simulation, the video provider transmission capacity u_{p} is equal to the peer transmission capacity u. In P2P VoD systems, video content providers are not only companies but also end-users. When the video content provider is an end-user, performance of the video provider is low and efficiency of P2P VoD systems tends to decrease. Therefore, it is important to assume a strict condition that the video provider transmission capacity u_{p} is equal to the peer transmission capacity u. Moreover, the parameters can apply some environments that fulfill follow requirements.

$$B_{\mathrm{v}} = u, \tag{4.3}$$

$$d = 2u, \tag{4.4}$$

$$jL = j\frac{E}{B_{\mathrm{v}}} = 250, \tag{4.5}$$

$$p = \frac{B_{\mathrm{v}}L}{E}, \tag{4.6}$$

Algorithm 2 Transmission algorithm of Sender Peer s in the in-order piece transfer

INPUT: u^s_{in} : Transmission capacity of Peer s for in-order piece transfer
INPUT: R_s : Array of in-order piece transfer requests to Peer s
INPUT: $R_s[i].requester$: Requester peer of Request $R_s[i]$
INPUT: s_i : Contribution value of Peer i
INPUT: b_i : Benefit value of Peer i
INPUT: u_{add} : Additional capacity of Peer s for in-order piece transfer

1: $u_{blank} \leftarrow 0$
2: **for** $i = 0$ to $u^s_{\text{in}} + u_{unalloc}$ **do**
3: **if** $R_s[i]$ not equal $null$ and $s_{R_s[i].requester} > b_{R_s[i].requester}$ **then**
4: Peer p send Piece $b_i + 1$ to $R_s[i].requester$
5: **end if**
6: **end for**

Algorithm 3 Transmission algorithm of Video Provider p in the in-order piece transfer

INPUT: u^p_{in} : Transmission capacity of Peer p for in-order piece transfer
INPUT: R_p : Array of in-order piece transfer requests to Peer s
INPUT: $R_p[i].requester$: Requester peer of Request $R_p[i]$
INPUT: s_i : Contribution value of Peer i
INPUT: b_i : Benefit value of Peer i
INPUT: u^p_{add} : Additional capacity of Peer s for in-order piece transfer

1: **for** $i = 0$ to $u^p_{\text{in}} + u^p_{add}$ **do**
2: **if** $R_p[i]$ not equal $null$ **then**
3: Peer p send Piece $b_i + 1$ to $R_p[i].requester$
4: **end if**
5: **end for**

Table 4.1: Simulation parameters

Parameter	Value
Peer transmission capacity u	10 pieces/round
Peer receive capacity d	20 pieces/round
Video provider transmission capacity u_{p}	10 pieces/round
peers' joining rate j	5 peers/round
Number of divided pieces E	500 pieces
Bit rate of video streaming B_{v}	10 pieces/round
Maximum number of neighbors	40 peers
Minimum number of neighbors	20 peers
Simulation time	$1,500$ rounds
Ratio of receive-able low-popularity piece f	0.05
Ratio of transmission capacity for low-popularity transfer r_{o}	0.9

Table 4.2: Examples of specific bit rate parameters

Pattern	u	d	u_{p}	B_{v}
i	384 kbps	768 kbps	384 kbps	384 kbps
ii	512 kbps	1 Mbps	512 kbps	512 kbps
iii	1 Mbps	2 Mbps	1 Mbps	1 Mbps

where p is the piece size and L is the length of providing video. Tables 4.2, 4.3, and 4.4 show examples of applicable parameters. In round-based simulators, a round is a measure of time. In this simulation, when L is assumed as fifty seconds, the length of a round is one second.

4.5.2 Simulation Scenario

In this simulation, peers send low-popularity pieces by using u_{low} as many as possible before the rest of its transmission capacity is used for in-order piece sending. Moreover, peers join in the P2P system according to the peers' joining rate j. Then, peers leave from the P2P system when they complete all piece collection. This scenario shows a strict condition that many peers join in the P2P system but peers that have all pieces are not exist in the P2P network. In the real environment, there exist some ultrapeers [53]

Table 4.3: Examples of specific time parameters

Pattern	Length of 1 round	L
A	1 second	50 seconds
B	30 second	$1,500$ seconds
C	60 seconds	$3,000$ seconds

Table 4.4: Examples of piece size p

Bit rate parameter pattern	Time parameter pattern	p
A	i	38.4 kb
A	ii	51.2 kb
A	iii	102.4 kb
B	i	1.152 Mb
B	ii	1.536 Mb
B	iii	3.072 Mb
C	i	2.304 Mb
C	ii	3.072 Mb
C	iii	6.144 Mb

which greatly contribute to the system, but the assumed strict condition often occurs. For example, the number of ultrapeers is few or zero right after starting to provide the video. Additionally, in P2P data providing systems, an increase of the number of ultrapeers and an increase of the video provider transmission capacity u_{p} have almost the same meaning. Therefore, an influence on the number of ultrapeers is shown in Section 4.6.5, that discuss an influence on video provider transmission capacity u_{p}.

Note that we assume that the transmission capacity for messages is ensured on the side. Thus, message overhead is zero in the simulation experiments because message overhead is typically negligible compared to bandwidth consumption by video pieces in P2P streaming. Additionally, peers are obedient to unveil their truthful information to each other.

4.6 Performance Evaluation

4.6.1 Evaluation Items

- Average utilization ratio of peer transmission capacity
  The average utilization ratio of peer transmission capacity, F_{u}, is denoted as follows;

$$F_{\mathrm{u}} = \frac{\sum_{i \in \psi} \frac{U_{\mathrm{send}}(i)}{u(i)t(i)}}{|\psi|}, \tag{4.7}$$

  where ψ is a set of all leaved peers and $u(i)$ is transmission capacity of peer i and $t(i)$ is life time of peer i and $U_{\mathrm{send}}(i)$ is the data size which peer i sent in life time.

- Average start-up latency
  The average start-up latency before starting continuous playback, σ, is denoted as follows;

$$\sigma = \frac{\sum_{i \in \psi} \sigma(i)}{|\psi|} \tag{4.8}$$

$$= \frac{\sum_{i \in \psi} \max_{k=0}^{E}(t(i,k) - \frac{kp}{B_{\mathrm{v}}})}{|\psi|}, \tag{4.9}$$

  where $\sigma(i)$ is start-up latency of peer i and E is the number of pieces in the whole video and $t(i,k)$ is the time when peer i receives piece k since the peer joins the network. $\sigma(i)$ can be calculated only when Peer i has received all pieces. Therefore σ is just a evaluation index and it means a potential to shorten the waiting time of viewers. σ is same as sum of stopping time for playback in the peers life time even if the peer starts play as soon as joining network.

  An example of start-up latency σ is as shown in Figure 4.4. The figure shows piece arrival times and a start-up latency of a peer. The ratio of receive-able low-popularity piece f is set as 0.2 for understandability. The peer joins the P2P network at round $1,376$, and receives pieces using low-popularity transfer from $1,376$ round to $1,386$ round. In this case, the start-up latency is five rounds. The peer can view the video continuously when it waits five rounds at the beginning.

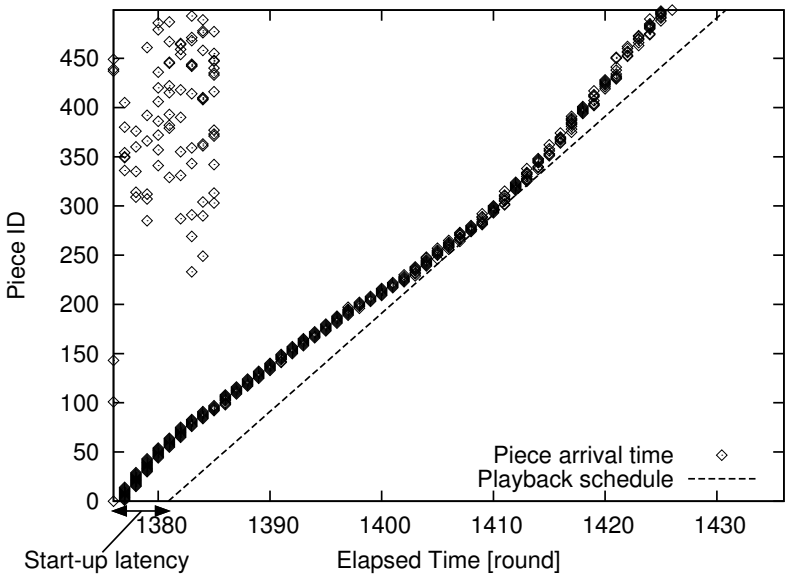

Figure 4.4: Piece arrival time and start-up latency σ

4.6.2 Low-Popularity Piece Transmission Capacity Ratio r_{o} and Receive-able Low-Popularity Piece Ratio f

Figure 4.6 and Figure 4.5 shows the influence of the low-popularity piece transmission capacity ratio r_{o} and the receive-able low-popularity pieces ratio f on the average utilization ratio of peer transmission capacity F_{u} and the average start-up latency σ in the proposed Piece Lending. The average utilization ratio of peer transmission capacity F_{u} increases with increasing the low-popularity piece transmission capacity ratio r_{o}. There exist two reasons. First reason is that newly joined peers tend to receive pieces by using low-popularity piece transfer easily. Second reason is that transferred pieces are selected by using Rarest-first in the low-popularity piece transfer. In P2P data providing, aggressive transfer of low-popularity data increases the system efficiency [73]. With increasing of the low-popularity piece transmission capacity ratio r_{o}, however, peers tend not to receive pieces in order because the transmission capacity for in-order piece transfer decreases. Therefore, the average start-up latency σ increases. When the receive-able low-popularity pieces ratio f is enough low, however, the average start-up latency σ does

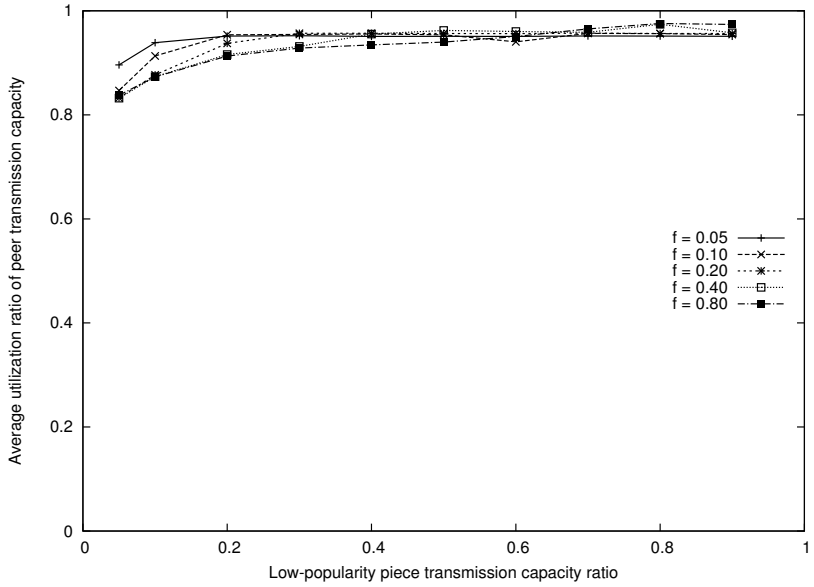

Figure 4.5: Average utilization ratio of peer transmission capacity F_{u} versus low-popularity piece transmission capacity ratio r_{o} with various receive-able low-popularity pieces ratio f

not increase even when the low-popularity piece transmission capacity ratio r_{o} is high. As a result, the proposed method should set the low-popularity piece transmission capacity ratio r_{o} as a high value in order to increase the average utilization ratio of peer transmission capacity F_{u}, and it should set the receive-able low-popularity pieces ratio f as a low value in order to decrease the average start-up latency σ. In the scope of the evaluation, efficiency of the Piece Lending is highest when $f = 0.05,\ r_{\mathrm{o}} = 0.9$. Therefore, Piece Lending uses $f = 0.05,\ r_{\mathrm{o}} = 0.9$ after that.

4.6.3 Average Utilization Ratio of Peer Transmission Capacity

Table 4.5 shows the average utilization ratio of peer transmission capacity ,F_{u}, for the proposed Piece Lending and existing Push method. The proposed method improves F_{u} by 41.7%, and it utilizes 95.1% of peer transmission capacity although existing method utilizes 67.1% of that. In the proposed method, newly joined peer can receive low-popularity pieces from not only video provider but also other peers. The newly joined peers join peers

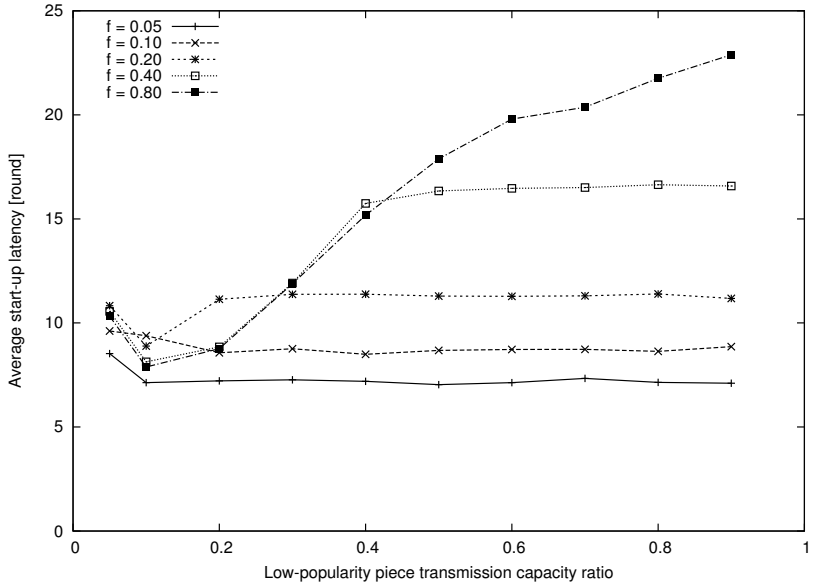

Figure 4.6: Start-up latency σ versus low-popularity piece transmission capacity ratio r_{o} with various receive-able low-popularity pieces ratio f

Table 4.5: Average utilization ratio of peer transmission capacity F_{u}

Piece Lending (Proposal)	Serve Push (Existing)
95.1%	67.1%

in piece exchange early by using the received low-popularity pieces. Moreover, after the peers collect enough in-order pieces, they send pieces whose number is equivalent to number of received low-popularity pieces to other newly joined peers. As a result, the proposed method realizes the concept of piece lending and improves F_{u}.

4.6.4 Average Start-Up Latency δ with Network Stress ϱ

Figure 4.7 shows the average start-up latency σ with various network stress ϱ for the proposed method and the existing method. The network stress is a ratio of the bit rate

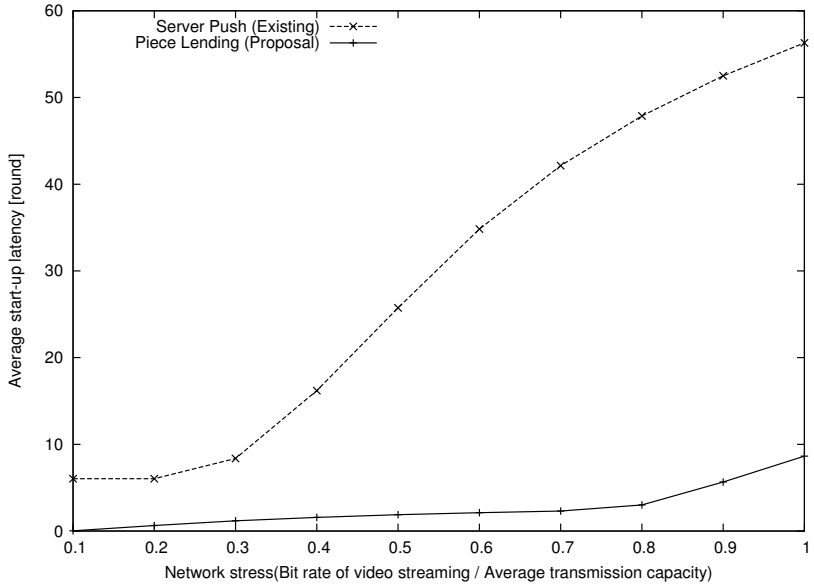

Figure 4.7: Average start-up latency σ versus network stress ϱ

of video streaming B_{v} to the average peer transmission capacity u as following equation.

$$\varrho = \frac{B_{\mathrm{v}}}{u}. \tag{4.10}$$

The network stress ϱ decreases with increasing the average peer transmission capacity u. In addition, in this model, when ϱ is lower than one and the average peer transmission capacity u is much higher than the bit rate of video streaming B_{v}, the average start-up latency σ is zero if peers receive pieces in order at a fixed receiving rate. In the figure, the average start-up latency σ of the propose method is always lower than that of the existing method. When the network stress ϱ increases, the average start-up latency σ decreases on both methods by a lack of peer transmission capacity in the P2P network. In the proposed method, newly joined peers can receive low-popularity pieces from other peers without the video provider. Therefore, the transmission capacity of the video provider does not become a bottleneck and the average start-up latency σ is low. Especially, when ϱ is one, the average start-up latency σ of the proposed method is shorter than that of the existing method by 87.4%. In practical situations, ϱ is much smaller than one. For

example, assuming the situation that a peer whose transmission capacity is 10 Mbps tries to view a video whose bit rate is 2 Mbps and whose length is 100 minutes, Piece Lending reduces σ from 12.1 minutes to 1.28 minutes.

4.6.5 Influence on Video Provider Transmission Capacity u_{p}

Figure 4.8 shows the average utilization ratio of peer transmission capacity versus the video provider transmission capacity u_{p} for lending and existing method. Although when the system uses existing method, utilization ratio of peer transmission capacity is lower as video provider transmission capacity u_{p} decreases, the decreasing of the utilized transmission capacity is restricted when the system uses lending. That is because although existing method relies on video provider to push pieces to newly joined peers, lending let also peers send pieces to newly joined peers. Moreover, although existing method relies on video provider to deliver the comparatively rare pieces to the network, lending restricts pieces from being rare by rarest-first low-popularity piece sending policy.

Figure 4.9 shows average start-up latency σ versus video provider transmission capacity u_{p} for lending and existing method. Although when the system uses existing method, the start-up latency σ is larger as video provider transmission capacity u_{p} decreases, this increasing of the latency is comparatively restricted when the system uses lending. That is because lending utilizes more peer transmission capacity as described above. Therefore, lending method more effectively improves average receiving quality achieved by existing method when video provider's transmission capacity u_{p} is small, for example, due to deliver multiple videos by single provider which may be a general end-user PC.

4.6.6 Influence on Free Riders

Figure 4.10 shows average utilization ratio of peer transmission capacity of normal peers versus participating ratio of free riders for lending and existing method. As participating ratio of free rider increases, utilized transmission capacity of normal peers is decreasing in either case that the system uses lending or existing method. That is because of three reasons. One is consumption of provider's transmission capacity by free rider. Secondary, each peer has insufficient number of neighbors who actually send pieces to them because

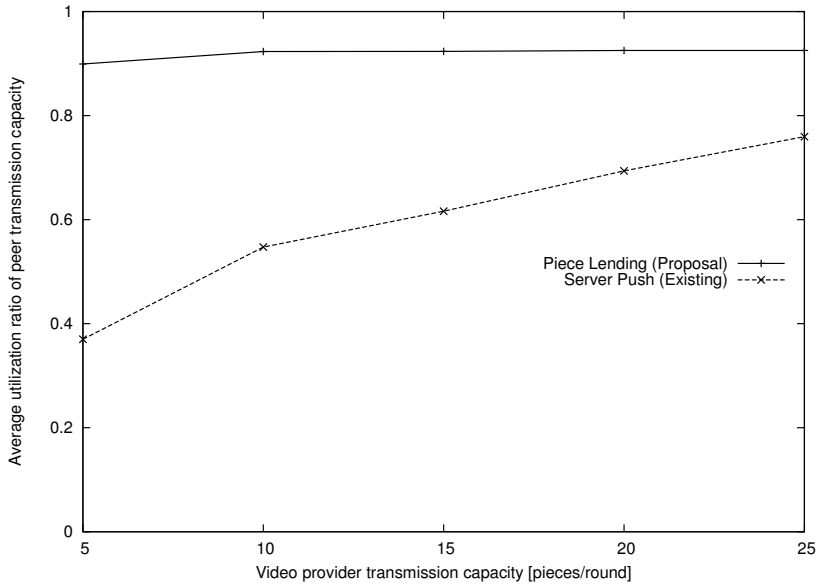

Figure 4.8: Average utilization ratio of peer transmission capacity F_{u} versus video provider transmission capacity u_{p}

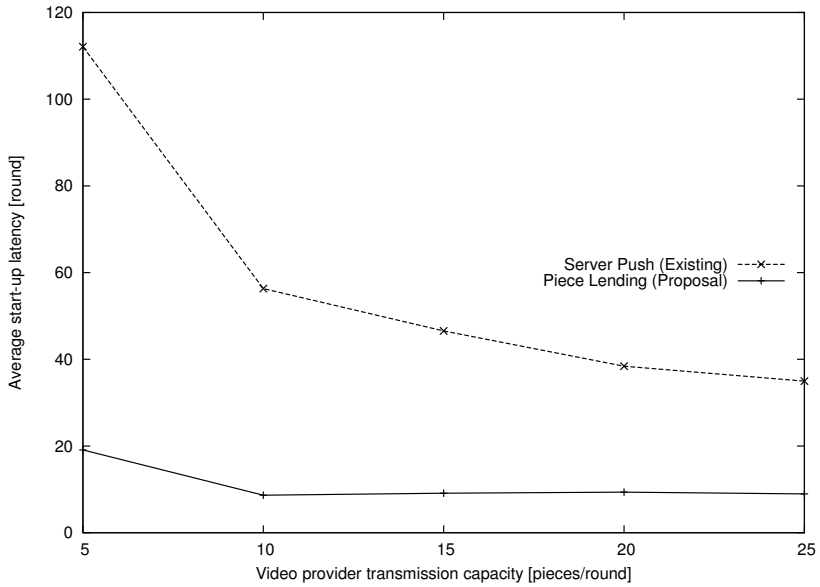

Figure 4.9: Average start-up latency σ versus video provider transmission capacity u_{p}

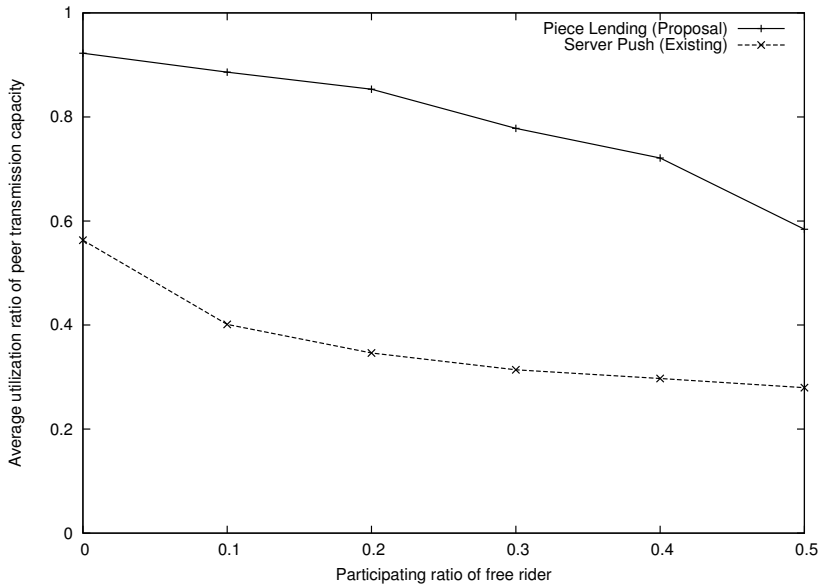

Figure 4.10: Average utilization ratio of peer transmission capacity of normal peers versus participating ratio of free rider

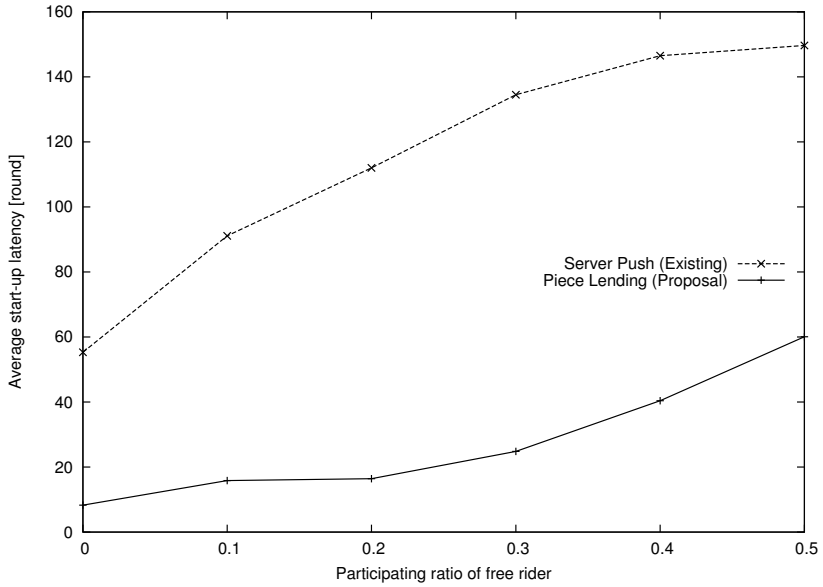

Figure 4.11: Average start-up latency σ versus participating ratio of free rider

their neighbors include free riders. Another is consumption of peer transmission capacity by free rider and this is only when the system uses lending. Although existing method using tit-for-tat based piece exchanging algorithm avoids consumption of peers transmission capacity by free rider, lending uses more peer transmission capacity for normal peers than existing method even if participating ratio of free rider is 0.5.

Moreover, there exists no motivation for becoming free rider also when the system uses lending method. That is because they cannot obtain in-order pieces and their obtained rate of low-popularity pieces is same as receive-able low-popularity piece ratio f even at a maximum and playback position of the pieces are dispersed.

Figure 4.11 shows average start-up latency σ versus participating ratio of free rider for lending and existing method. Lending method can achieve smaller latency compared to existing method at any participating ratio of free rider under 0.5. That is because lending utilizes more peer transmission capacity as described above.

4.7 Summary

This chapter proposes Piece Lending, which uses a lending approach in which the P2P system lends receiving rights for low-popularity pieces to newly joined peers, and collects contributions of advanced peers in compensation for using rights when newly joined peers become advanced peers. The proposed method is an incentivized method with low-popularity piece push. Incentivized methods provide incentives (receipt rights) for piece sending, preventing free-riders from receiving pieces. In existing methods, low-popularity piece push from the video provider is required for newly joined peers to join in the piece exchange, because newly joined peers do not possess transmittable piece, making their negotiation power very low. In the proposed method, P2P systems provide receiving rights for low-popularity pieces to newly joined peers, thus increasing their negotiation powers. Newly joined peers use these receiving rights to receive low-popularity pieces, allowing them to take part in piece exchange without support from the video provider. Because most peers do not possess low-popularity piece, newly joined peers can send that pieces to peers that do not possess it. Moreover, advanced peers with high negotiation power pay their contribution to the P2P system in compensation for previously borrowing

rights; in other words, they send low-popularity pieces to newly joined peers depending on the amount of pieces that they have received using the borrowed rights. As a result, the proposed method distributes the load of low-popularity piece pushing to peers.

The key idea of the lending approach is that the P2P system defines the benefit of peers is as the number of received continuous playback-able pieces. Then, the benefit value is used as a reward for contribution in the incentivized algorithm. This idea realizes the concept of the lending approach. The incentivized algorithm is used for preventing free riders from enjoying services without contributing to the P2P systems. In the proposed method, free riders can receive non-continuous pieces, which are low-popularity pieces, although it cannot receive continuous in-order pieces without sending pieces to other peers. The fact that free riders can receive non-continuous pieces is not serious problem. In general P2P VoD systems, the video data is divided into small pieces and peers receive the video data in unit piece. Therefore, when free riders collect some non-continuous pieces, they can view only fragmentary video. In addition, even when free riders tray to collect all pieces over time, they cannot complete it by the limitation of the receive-able low-popularity piece ratio f in the proposed method. As above, the proposed method realizes the concept of the lending approach while preventing free rider joining.

The proposed Piece Lending is evaluated by computer simulation in terms of the utilization ratio of peer transmission capacity F_u and the start-up latency σ. At first, this dissertation investigates the low-popularity piece transmission capacity ratio r_{o} and the effect on the receive-able low-popularity piece ratio f. Simulation results showed that newly joined peers can easily-join in the piece exchange when they obtain only few low-popularity pieces. In the scope of the evaluation, efficiency of the Piece Lending is highest when $f = 0.05,\ o = 0.9$. Therefore, Piece Lending uses $f = 0.05,\ o = 0.9$ after that. Simulation results showed that the proposed method improves F_{u} by 41.7%, and it utilizes 95.1% of peer transmission capacity although existing method utilizes 67.1% of that. As the start-up latency, when the network stress ϱ is one, σ of the proposed method is shorter than that of the existing method by 87.4%. Then, assuming practical situation that a peer whose transmission capacity is 10 Mbps tries to view a video whose bit rate is 2 Mbps and whose length is 100 minutes, Piece Lending reduces σ from 12.1 minutes to 1.28 minutes. Moreover, this dissertation studies the effects of the video provider transmission

capacity u_{p} and the participating ratio of free riders. The proposed method tends not to decrease the F_u and σ when u_{p} is low because the low-popularity piece push is supported by not only the video provider but also peers. In addition, the efficiency of the proposed method is higher than that of the existing method even when free-riders join in the P2P network. Finally, those results showed that proposed method is effective even when the video provider's performance is low.

Chapter 5

Conclusion

Recently, P2P VoD systems have been widely discussed, but peer search and video data transmission scheduling functions in existing methods are ineffective. This dissertation focused on video data popularity, meaning how many peers possess given data, and challenged conventional methods of peer search and video data transmission scheduling.

Chapter 1 described the background of P2P VoD systems and explained the position of this dissertation.

Chapter 2 described major functions of the P2P VoD systems and gave brief overviews of several existing methods for peer search and video data scheduling.

Chapter 3 proposed HyDiff, a hybrid search method that estimates data popularity by using local popularity. Hybrid search methods are effective methods that take on different search models depending on search target popularity. Conventional hybrid search methods use a request rate to estimate the popularity, but it is difficult to retain high efficiency because the estimated popularity based on request rate does not always properly reflect the actual popularity. HyDiff uses local popularity for the popularity estimation to maintain the search efficiency when there exists a large gap between measured request rate and popularity.

The efficiency of HyDiff in terms of the utility that contains the number of response, the shortest response time, and the messaging cost was evaluated by simulation. In the simulation, replicas of an object are distributed uniformly over the entire network. Then, a popularity of one object is changed, and utility of searching for the object is

calculated to evaluate. HyDiff can maintain a low $U_{\text{difference}}$, utility difference ratio from the theoretical value, in the simulation. $U_{\text{difference}}$ of the existing method varies depending on the combination of the request rate change interval and the popularity estimation interval. In the existing method, it is difficult to decide the value of the estimation interval because the request rate change of the search-target is unpredictable. On the other hand, $U_{\text{difference}}$ of HyDiff decreases with decreasing the estimation interval. Therefore, the estimation interval can be set low value as far as the system performance permits in HyDiff. Additionally, when the search target's request rate change interval is twenty hours and estimation interval is $1,200$ sec, the utility difference ratio $U_{\text{difference}}$ of HyDiff is under 1%. Moreover, assuming a situation that the P2P network is applied on a global scale, HyDiff reduces $R_{\text{shortest}}(t)$ from around eight seconds to around two seconds when search-target's request rate is high but its popularity is very low. This indicates that the proposed method maintains search efficiency when there exists a large gap between request rate and popularity.

Chapter 4 proposed Piece Lending that decreases the negotiation power gap without the support of the video-providing node. In this lending approach, the P2P system lends receiving rights for low-popularity piece to newly joined peers, and collects contributions from advanced peers in compensation for using the right when newly joined peers become advanced peers. In existing methods, newly joined peers require low-popularity piece push from the video-providing node to participate in piece exchange. Piece Lending distributes the load of low-popularity piece push to peers.

The efficiency of the Piece Lending in terms of utilization ratio of peer transmission capacity and start-up latency was evaluated by simulation. At first, this dissertation investigates the low-popularity piece transmission capacity ratio r_{o} and the effect on the receive-able low-popularity piece ratio f. Next, the average utilization ratio of peer transmission capacity F_{u} is evaluated. The Simulation results showed that newly joined peers can easily-join in the piece exchange when they obtain only few low-popularity pieces. Next, the other results showed that Piece Lending improves F_{u} by 41.7%, and utilizes 95.1% of peer transmission capacity. Utilization in the existing method is 67.1%. Moreover, when the network stress ϱ is one, the average start-up latency σ of Piece Lending is shorter than that of the existing method by 87.4%. Then, assuming practical

situation that a peer whose transmission capacity is 10 Mbps tries to view a video whose bit rate is 2 Mbps and whose length is 100 minutes, Piece Lending reduces σ from 12.1 minutes to 1.28 minutes. Additionally, the efficiency of Piece Lending is hardly affected by the video provider's performance. The results showed that Piece Lending distributes the load of low-popularity piece push to peers. These results shows that the proposed method is effective even when the video provider's performance is low.

This dissertation proposed two novel methods, HyDiff and Piece Lending, for peer search and video data transmission scheduling on P2P VoD systems, and showed the effectiveness of the proposed methods by simulation. One of the strengths in adopting this proposed P2P VoD system is that viewers can expect decreasing two types of waiting times: latency for the response of searching and start-up latency before starting continuous playback. Additionally, video providers also can expect providing high-quality videos without high-performance and high-cost servers. Therefore, this research will contribute to the improvement of P2P VoD systems.

Acknowledgments

First of all, I would like to express my heartfelt gratitude toward Professor Hiroshi Shigeno for his great supports and advices for six years to obtain this doctoral degree. I would also like to thank Professor Ken-ichi Okada, Professor Naoaki Yamanaka, and Professor Fumio Teraoka for reviewing this dissertation and giving me valuable suggestions.

I have been supported and helped by many respectable fellows in Shigeno Laboratory. Especially, Masaki Suzuki gave me various gifts such as the ability of thinking logically and skills in writing technical papers in English. I have greatly benefited from stimulating discussions with SOON group members: Nozomu Kita, Kenji Takagi, Kei Matsumoto, Takaya Fujimoto, Kazuyuki Takayama, Yoshiki Sakata, and Sho Hatakeyama. Among them, Takaya Fujimoto provided me great inspirations for interesting researches. I have enjoyed spending time with all laboratory members. I am glad I shared many rewarding experiences with them.

Finally, I would like to sincerely thank my parents for supporting me for years. I could not accomplish this work without their support and understanding.

Rei Endo
January 2013

Bibliography

[1] H. Kasahara, H. Okuda, and K. Shimamura, "DAVIC : Interoperability solution for video-on-demand systems," *IEICE transactions on information and systems*, vol. 79, no. 6, pp. 647–652, June 1996.

[2] Y. Kawanami, "In platform for DAVIC video on demand service," in *The IEEE Intelligent Network Workshop*, 8 pages, April 1996.

[3] D. King, "DAVIC-interoperability for market success," in *IEE Coloqioi, on Interactive Television*, pp. 4/1–4/8, Novenber 1995.

[4] "YouTube," 2012. [Online]. Available: http://www.youtube.com/

[5] "Netfix," 2012. [Online]. Available: http://www.netfix.com/

[6] J. Choi, A. S. Reaz, and B. Mukherjee, "A survey of user behavior in VoD service and bandwidth-saving multicast streaming schemes," *Journal of IEEE Communications Surveys and Tutorials*, vol. 14, no. 1, pp. 156–169, April 2012.

[7] Z. Chena, K. Xue, P. Hong, and H. Lu, "Differentiated bandwidth allocation for reducing server load in P2P VOD," in *The 8th International Conference on Grid and Cooperative Computing (GCC 2009)*, pp. 31–36, August 2012.

[8] M. Cha, H. Kwak, P. Rodriguez, A.-Y. Yong, and S. Moon, "Analyzing the video popularity characteristics of large-scale user generated content systems," *IEEE/ACM Transactions on Networking*, vol. 17, no. 5, pp. 1357–1370, October 2009.

[9] Q. Xuanjia, W. Chuan, L. Xiaola, and C. M. L. Francis, "InstantLeap: Fast neighbor discovery in P2P vod streaming," in *The ACM 19th international workshop on Network and operating systems support for digital audio and video (NOSSDAV '09)*, pp. 19–24, June 2009.

[10] L. Chao, F. Zhenghua, L. Yong, and W. W. Chai, "Incentivized peer-assisted streaming for on-demand services," *IEEE Transactions on Parallel and Distributed Systems*, vol. 21, no. 9, pp. 1354–1363, January 2010.

[11] W. Danqi and K. Y. Chai, "Superchunk-based efficient search in P2P VoD system," *IEEE Transactions on Multimedia*, vol. 13, no. 2, pp. 376–387, January 2012.

[12] Y. Shan and S. Kalyanaraman, "Hybrid video downloading/streaming over peer-to-peer networks," in *The 2003 IEEE International Conference on Multimedia and Expo (ICME 2003)*, vol. 2, pp. II–665–8, July 2003.

[13] A. Sharma, A. Bestavros, and I. Matta, "dPAM: A distributed prefetching protocol for scalable asynchronous multicast in P2P systems," in *The 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2005)*, vol. 2, pp. 1139–1150, March 2005.

[14] C. Yi, L. Baochun, and K. Nahrstedt, "oStream: Asynchronous streaming multicast in application-layer overlay networks," *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 1, pp. 91–106, January 2004.

[15] K. Huguenin, A.-M. Kermarrec, V. Rai, and M. V. Steen, "Designing a tit-for-tat based peer-to-peer video-on-demand system," in *The 20th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV 2010)*, pp. 93–98, June 2010.

[16] A. R. Bharambe, C. Herley, and V. N. Padmanabhan, "Analyzing and improving a bittorrent networks performance mechanisms," in *The 25th IEEE International Conference on Computer Communications (INFOCOM 2006)*, pp. 1–12, April 2006.

[17] S. Annapureddy, S. Guha, and C. Gkantsidis, "Is high-quality vod feasible using P2P swarming?" in *The 16th International Conference onl World Wide Web (WWW 2007)*, pp. 903–912, May 2007.

[18] R. Endo, K. Matsumoto, and H. Shigeno, "GBLT-VG for high user densities by user group behavior and hot point in mmo virtual environment," *Springer Journal of Network*, pp. 1–19, October 2011.

[19] "BitTorrent," 2012. [Online]. Available: http://www.bittorrent.com/

[20] "Download music - music downloads and mp3 downloads from kazaa.com," 2012. [Online]. Available: http://www.kazaa.com

[21] Y. Z. Xin, Z. Qian, Z. Zhensheng, S. Gang, and Z. Wenwu, "A construction of locality-aware overlay network: mOverlay and its performance," *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 1, pp. 18–28, January 2004.

[22] Y. Chu, S. G. Rao, and H. Zhang, "A case for end system multicast," in *ACM international Conference on Measurement and Modeling of Computer Systems (SIGMETRICS 2000)*, pp. 1–12, January 2000.

[23] Z. Xinyan, L. Jiangchuan, L. Bo, and Y.-S. P. Yum, "CoolStreaming/DONet: a data-driven overlay network for peer-to-peer live media streaming," in *The 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2005)*, pp. 2102–2111, March 2005.

[24] M. Zhang, J. Luo, L. Zhao, and S. Yang, "A peer-to-peer network for live media streaming - using a push-pull approach," in *The 13th Annual ACM international Conference on Multimedia (MULTIMEDIA 2005)*, pp. 6–12, November 2005.

[25] "PPLive," 2012. [Online]. Available: http://www.pplive.com/

[26] Y. Liu, Z. Zhuang, L. Xiao, and L. M. Ni, "AOTO: Adaptive overlay topology optimization in unstructured P2P systems," in *IEEE Global Telecommunications Conference (GLOBECOM 2003)*, pp. 4186–4190, December 2003.

[27] Y. Liu, X. Liu, L. Xiao, L. M. Ni, and X. Zhang, "Location-aware topology matching in P2P systems," in *The 23rd AnnualJoint Conference of the IEEE Computer and Communications Societies (INFOCOM 2004)*, vol. 4, pp. 2220–2230, March 2004.

[28] Y. Liu, L. Xiao, and L. M. Ni, "Building a scalable bipartite P2P overlay network," in *The 18th IEEE International Parallel and Distributed Processing Symposium (IPDOS 2004)*, pp. 1–10, April 2004.

[29] Y. Liu, Z. Zhuang, L. Xiao, and L. M. Ni, "A distributed approach to solving overlay mismatching problem," in *The 24th International Conference on Distributed Computing Systems*, pp. 132–139, May 2004.

[30] D. Qiu and R. Srikant, "Modeling and performance analysis of bittorrent-like peer-to-peer networks," in *The 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM 2004)*, pp. 367–378, August 2004.

[31] C. Wang, L. Xiao, Y. Liu, and P. Zheng, "Distributed caching and adaptive search in multilayer P2P networks," in *The 24th International Conference on Distributed Computing Systems*, pp. 219–226, March 2004.

[32] J. Han, Y. Liu, L. Lu, L. Hu, and A. P. Patil, "A random walk based anonymous peer-to-peer protocol design," in *The 3rd International Conference on Networking and Mobile Computing (ICCNMC 2005)*, pp. 143–152, August 2005.

[33] J. Han, Y. Liu, L. Xiao, R. Xiao, and L. M. Ni, "A mutual anonymous peer-to-peer protocol design," in *The 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS 2005)*, pp. 1–10, April 2005.

[34] Y. Liu, L. Xiao, A.-H. Esfahanian, and L. M. Ni, "Approaching optimal peer-to-peer overlays," in *The 13th International Symposium on Modeling, Analysis, and Simulation of Computer Systems (MASCOTS 2005)*, pp. 407–414, September 2005.

[35] X. Liao, H. Jin, Y. Liu, L. M. Ni, and D. Deng, "AnySee: Peer-to-peer live streaming," in *The 25th IEEE International Conference on Computer Communications (INFOCOM 2006)*, pp. 1–10, April 2006.

[36] K. Nagaraja, S. Rollins, and M. Khambatti, "Looking beyond the legacy of napster and gnutella," *IEEE Distributed Systems Online*, pp. 1–5, March 2006.

[37] D. Manini, R. Gaeta, and M. Sereno, "Performance modeling of P2P file sharing applications," in *The Workshop on Techniques, Methodologies and Tools for Performance Evaluation of Complex Systems (FIRB-Perf 2005)*, pp. 34–43, September 2005.

[38] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," in *The 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM 2001)*, pp. 149–160, August 2001.

[39] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A scalable content-addressable network," in *The ACM 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM 2001)*, pp. 161–172, August 2001.

[40] A. Rowstron and P. Druschel, "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems," in *The IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, pp. 329–350, November 2001.

[41] K. Hildrum, J. D. Kubiatowicz, S. Rao, and B. Y. Zhao, "Distributed object location in a dynamic network," in *The 14th Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA 2012)*, pp. 41–52, August 2002.

[42] P. Maymounkov and D. Mazieres, "Kademlia: A peer-to-peer information system based on the xor metric," in *The 1st International Workshop on Peer-to-Peer Systems (IPTPS 2002)*, pp. 53–65, October 2002.

[43] S. Jiang and X. Zhang, "Floodtrail: an efficient file search technique in unstructured peer-to-peer systems," in *The IEEE Global Telecommunications Conference (GLOBECOM 2003)*, pp. 2891–2895, December 2003.

[44] H. Kim, Y. Kim, K. Kim, and S. Kang, "Restricted path flooding scheme in distributed P2P overlay networks," in *The 2008 International Conference on Information Science and Security (ICISS 2008)*, pp. 58–61, January 2008.

[45] Y. Hu, M. Feng, L. N. Bhuyan, and V. Kalogeraki, "Budget-based self-optimized incentive search in unstructured P2P networks," in *INFOCOM 2009. The 28th Conference on Computer Communications. IEEE*, pp. 352–360, April 2009.

[46] W. Ai, L. Xinsong, and L. Kejian, "Efficient flooding in peer-to-peer networks," in *The 7th International Conference on Computer-Aided Industrial Design and Conceptual Design (CAIDCD 2006)*, pp. 1–6, November 2006.

[47] L. Jiaqing, Z. Shijie, W. Chunjiang, D. Yiyi, and Y. Xiaoqian, "Adaptive flooding routing algorithm in unstructured p2p," in *The IEEE 2006 International Conference on Communications, Circuits and Systems (ICCCAS 2006)*, pp. 1557–1561, june 2006.

[48] B. T. Loo, J. M. Hellerstein, R. Huebsch, S. Shenker, and I. Stoica, "Enhancing P2P file-sharing with an internet-scale query processor," in *The 30th International Conference on Very Large Data Bases (VLDB 2004)*, pp. 432–443, August 2004.

[49] S. S. ans P. K. Gummadi and S. D. Gribble, "A measurement study of peer-to-peer file sharing systems," in *Multimedia Computing and Networking (MMCN 2002)*, pp. 1–15, January 2002.

[50] M. Ripeanu, "Peer-to-peer architecture case study: Gnutella network," in *The IEEE 1st International Conference on Peer-to-Peer Computing (P2P 2001)*, pp. 99–100, August 2001.

[51] F. Yuan, J. Liu, and C. Yin, "A scalable search algorithm on unstructured P2P networks," in *The 8th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (SNPD 2007)*, pp. 199–204, August 2007.

[52] B. T. Loo, R. Huebsch, I. Stoica, and J. M. Hellerstein, "The case for a hybrid P2P search infrastructure," in *The 3rd International Conference on Peer-to-Peer Systems (IPTPS 2004)*, pp. 141–150, February 2004.

[53] M. Zaharia and S. Keshav, "Gossip-based search selection in hybrid peer-to-peer networks," *Concurrency and Computation: Practice and Experience - Recent Advances in Peer-to-Peer Systems and Security (P2P 2006)*, vol. 20, no. 2, pp. 139–153, Feburary 2008.

[54] Y. R. Choe, D. L. Schuff, J. M. Dyaberi, and V. S. Pai, "Improving VoD server efficiency with bittorrent," in *The ACM 15th International Conference on Multimedia (MULTIMEDIA 2007)*, pp. 117–126, September 2007.

[55] C. Dana, D. Li, D. Harrison, and C.-N. Chuah, "BASS: Bittorrent assisted streaming system for video-on-demand," in *The IEEE 7th Workshop on Multimedia Signal Processing (MMSP 2005)*, pp. 1–4, Octobar 2005.

[56] Y.-W. Sung, M. Bishop, and S. Rao, "Enabling contribution awareness in an overlay broadcasting system," in *ACM SIGCOMM*, pp. 411–422, September 2006.

[57] Z. Liu, Y. Shen, K. W. Ross, S. S. Panwar, and Y. Wang, "Substream trading: Towards an open P2P live streaming system," in *The 16th IEEE International Conference on Network Protocols (ICNP 2008)*, pp. 94–103, October 2008.

[58] M.-L. Champel, A.-M. Kermarrec, and N. L. Scouarnec, "Phosphite: Guaranteeing out-of-order download in P2P video on demand," in *The IEEE 9th International Conference on Peer-to-Peer Computing (P2P 2009)*, pp. 281–290, September 2009.

[59] J. Liedtke, N. Islam, and T. Jaeger, "Preventing denial-of-service attacks on a μ-kernel for webOSes," in *The 6th Workshop on Hot Topics in Operating Systems (HotOS-VI)*, pp. 73–79, May 1997.

[60] F. Lau, S. H. Rubin, M. H. Smith, and L. Trajkovic, "Distributed denial of service attacks," in *The IEEE 2000 International Conference on Systems, Man, and Cybernetics*, pp. 2275–2280, October 2000.

[61] Y. Bouzida, F. Cuppens, and S. Gombault, "Detecting and reacting against distributed denial of service attacks," in *The IEEE International Conference on Communications 2006 (ICC 2006)*, pp. 2394–2400, June 2006.

[62] H. Balakrishnan, M. F. Kaashoek, D. Karger, R. Morris, and I. Stoica, "Looking up data in P2P systems," *Communications of the ACM*, vol. 46, no. 2, pp. 43–48, February 2003.

[63] K. Junemann, P. Andelfinger, and H. Hartenstein, "Towards a basic dht service: Analyzing network characteristics of a widely deployed DHT," in *The 20th International Conference on Computer Communications and Networks (ICCN 2011)*, pp. 1–7, July 2011.

[64] J. S. Kong, J. S. A. Bridgewater, and V. P. Roychowghury, "A general framework for scalability and performance analysis of DHT routing systems," in *The International Conference on Dependable Systems and Networks (DSN 2006)*, pp. 343–354, June 2006.

[65] D. Kempe, A. Dobra, and J. Gehrke, "Gossip-based computation of aggregation information," in *The 44th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2003)*, pp. 482–491, October 2003.

[66] S. Nath, P. B. Gibbons, S. Seshan, and Z. R. Anderson, "Synopsis diffusion for robust aggregation in sensor networks," in *The ACM 2nd International Conference on Embedded Networked Sensor Systems (SenSys 2004)*, pp. 250–262, November 2004.

[67] J.-G. Luo, Q. Zhang, Y. Tang, and S.-Q. Yang, "A trace-driven approach to evaluate the scalability of P2P-based video-on-demand service," *IEEE Transactions on Parallel and Distributed Systems*, vol. 20, no. 1, pp. 59–70, January 2009.

[68] M. Saitoh, M. Isogai, T. Koumoto, and N. Funabiki, "A development of peercast-based multimedia disribution system peerstream and its evaluations," in *IEICE Techinical Report*, vol. 105, no. 562, pp. 5–8, Janualy 2006.

[69] K. Yasuhara, T. Koumoto, N. Funabiki, and Y. Sugiyama, "Design and implementation of dynamic autonomous distributed connection switching function in peercast," in *IEICE Technical report*, vol. 108, no. 258, pp. 7–12, October 2008.

[70] T. Fujimoto, R. Endo, K. Matsumoto, and H. Shigeno, "P2P video-on-demand streaming using caching and reservation scheme based on video popularity," *International Journal of Grid and Utility Computing (IJGUC)*, vol. 3, no. 2/3, pp. 188–199, July 2012.

[71] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and S. Shenker, "Making gnutella-like P2P systems scalable," in *The ACM SIGCOMM 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pp. 407–418, August 2003.

[72] I. Kaneko, *The Technology of Winny (in Japanease).* ASCII MEDIA WORKS, 2005.

[73] K. Matsumoto, R. Endo, and H. Shigeno, "Efficient P2P file sharing method considering rarity of blocks," *IPSJ Journal*, vol. 51, no. 6, pp. 1310–1319, January 2010.

[74] A. Srinivasan, J. Teitelbaum, and J. Wu, "DRBTS: Distributed reputation-based beacon trust system," in *The 2nd IEEE International Symposium on Dependable, Autonomic and Secure Computing (DASC 2006)*, pp. 277–283, October 2006.

[75] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina, "The eigentrust algorithm for reputation management in P2P networks," in *The 12th International Conference onl World Wide Web (WWW 2003)*, pp. 640–651, May 2003.

Publications

Journals Related to this Dissertation

1. Rei Endo, Kazuyuki Takayama, Hiroshi Shigeno, "Piece Lending Scheme Using Benefit Value for Incentivized P2P VoD Streaming," *IPSJ Journal*, Vol. 54, No. 2, 9 pages, February 2013.

2. Rei Endo, Hiroshi Shigeno, "HyDiff: P2P Hybrid-search Based on Diffusion Rate for Objects with Varying Request Rate," *IPSJ Journal*, Vol. 52, No. 2, pp. 368-375, February 2011.

Journals not Related to this Dissertation

1. Takaya Fujimoto, Rei Endo, Kei Matsumoto, Hiroshi Shigeno, "P2P Video-on-Demand Streaming Using Caching and Reservation Scheme Based on Video Popularity," *International Journal of Grid and Utility Computing (IJGUC)*, Vol. 3, Nos. 2/3, pp. 188-199, July 2012.

2. Rei Endo, Kei Matsumoto, Hiroshi Shigeno, "GBLT-VG for High User Densities by User Group Behavior and Hot Point in MMO Virtual Environment," *Springer Journal of Network and System Management*, p. 19, DOI: 10.1007/s10922-011-9215-z, October 2011.

3. Kei Matsumoto, Rei Endo, Hiroshi Shigeno, "Efficient P2P File Sharing Method Considering Rarity of Blocks," *IPSJ Journal*, Vol. 51, No. 6, pp. 1310-1319, January 2010.

International Conferences

1. Yoshiki Sakata, Kazuyuki Takayama, Rei Endo, and Hiroshi Shigeno, "A Chunk Scheduling based on Chunk Diffusion Ratio on P2P Live Streaming," in *The 15th International Conference on Network-Based Information Systems(NBiS 2012)*, pp. 74-81, September 2012.

2. Rei Endo, Kazuyuki Takayama, Yoshiki Sakata, and Hiroshi Shigeno, "Neighbor Selection Method based on Sending Capacity for P2P Live Streaming with Layer Coding," in *The 9th IEEE International Conference on Ubiquitous Intelligence and Computing(UIC 2012)*, pp. 264-271, September 2012.

3. Kazuyuki Takayama, Takaya Fujimoto, Rei Endo, Hiroshi Shigeno, "Neighbor Selection Based On Transmission Bandwidth on P2P Live Streaming Service," in *The 8th International Symposium on Frontiers of Information Systems and Network Applications(FINA 2012)*, pp. 105-110, March 2012.

4. Takaya Fujimoto, Rei Endo, Kei Matsumoto, Hiroshi Shigeno, "Video-Popularity-based Caching Scheme for P2P Video-on-Demand Streaming," in *The 25th IEEE International Conference on Advanced Information Networking and Applications(AINA 2011)*, pp. 748-755, March 2011.

5. Rei Endo, Kenji Takagi and Hiroshi Shigeno, "GBLT: Load Distribution method for High Density by User Behavior in MMO Virtual Environment," in *The IEEE 23rd International Conference on Advanced Information Networking and Applications(AINA 2009)*, pp. 792-799, May 2009.

Technical Reports

1. Rei Endo, Kei Matsumoto, Takaya Fujimoto, Hiroshi Shigeno, "Piece Deposit Scheme to Decrease Load of Video Server in P2P VoD Streaming," in *IPSJ DICOMO 2012*, pp. 1543-1550, July 2012.

2. Kazuyuki Takayama, Takaya Fujimoto, Rei Endo, Hiroshi Shigeno, "Examination of Neighbor Selection based on Peer Transmission bandwidth in P2P Live Streaming Service," in *IPSJ Sig Technical Report*, Vol. 2011, No. 9, pp. 1-8, June 2011.

3. Takaya Fujimoto, Rei Endo, Kei Matsumoto, Hiroshi Shigeno, "Effective utilization of upload bandwidth considering video popularity in P2P VoD Streaming," in *IEICE Technical Report*, Vol. 110, No. 372, pp. 63-68, January 2011.

4. Rei Endo, Kei Matsumoto, Takaya Fujimoto, Hiroshi Shigeno, "P2P Hybrid-Search based Diffusion Rate Considering Request Rate Change," in *IPSJ DICOMO 2010*, pp.2015-2022, July 2010.

5. Takaya Fujimoto, Rei Endo, Kei Matsumoto, Hiroshi Shigeno, "Proposal of Cashing Based on Video Popularity for P2P VoD Service," in *IPSJ Sig Technical Report*, Vol. 2010, No. 24, pp. 1-8, May 2010.

6. Takaya Fujimoto, Rei Endo, Kei Matsumoto, Hiroshi Shigeno, "Investigation of Cache Based on Popularity of Data for P2P VoD Service," in *The 72th National Convention of IPSJ*, pp. 3_7-3_8, March 2010.

7. Kei Matsumoto, Rei Endo, Kenji Takagi, Hiroshi Shigeno, "P2P file sharing method controlling high rarities of blocks," in *IPSJ Sig Technical Report*, Vol. 2009 No. 10, pp. 1-6, September 2009.

8. Kei Matsumoto, Rei Endo, Kenji Takagi, Hiroshi Shigeno, "CAS: P2P file share method considering rarities of parts," in *IPSJ Sig Technical Report*, Vol. 2009 No. 20, pp. 265-270, March 2009.

9. Kenji Takagi, Rei Endo, Hiroshi Shigeno, "A Study of Bloom-Filter-Based Semantic Overlay Networks," in *IPSJ DICOMO 2008*, pp. 424-429, June 2008.

10. Rei Endo, Kenji Takagi, Hiroshi Shigeno, "Load distribution technique for high density of user in MMO Virtual Environment," in *IPSJ Sig Technical Report*, Vol. 2008, No. 21, pp. 213-218, March 2008.

11. Rei Endo, Kenji Takagi, Nozomu Kita, Hiroshi Shigeno, "Load Distribution Technique Corresponding to Object Density in Virtual Environment," in *The 70th National Convention of IPSJ*, pp. 3_55-3_56, March 2008.

Awards

1. Rei Endo, Kazuyuki Takayama, Yoshiki Sakata, Hiroshi Shigeno, Best Paper Award, *IPSJ DICOMO 2012*, "Piece Deposit Scheme to Decrease Load of Video Server in P2P VoD Streaming," 30 August 2012.

2. Rei Endo, Outstanding Presentation Award, *IPSJ DICOMO 2012*, "Piece Deposit Scheme to Decrease Load of Video Server in P2P VoD Streaming," 6 July 2012.

3. Takaya Fujimoto, Rei Endo, Kei Matsumoto, Hiroshi Shigeno, Best Student Paper Award, *The 25th IEEE International Conference on Advanced Information Networking and Applications (AINA 2011)*, "Video-Popularity-based Caching Scheme for P2P Video-on-Demand Streaming," 24 March 2011.

4. Rei Endo, Outstanding Presentation Award, *IPSJ DICOMO 2010*, "P2P Hybrid-Search based Diffusion Rate Considering Request Rate Change," 9 July 2010.