Journal of Mobile Multimedia, Vol. 5, No.3 (2009) 255-270
© Rinton Press

A SCHEDULING METHOD TO REDUCE WAITING TIME FOR P2P STREAMING SYSTEMS

YUSUKE GOTOH KENTARO SUZUKI

Kyoto University, Kyoto
{gotoh, suzuki}@ais.sys.i.kyoto-u.ac.jp

TOMOKI YOSHIHISA

Cybermedia Center, Osaka University, Osaka
yoshihisa@cmc.osaka-u.ac.jp

MASANORI KANAZAWA

Academic Center for Computing and Media Studies, Kyoto University, Kyoto
bwv147@mbox.kudpc.kyoto-u.ac.jp

Received July 21, 2008
Revised March 26, 2009

Recently, live streaming systems with peer-to-peer (P2P) technology have attracted much attention and are changing how we watch movies. In P2P streaming systems, a peer that plays the movie receives data from other peers. By sequentially playing the received data, users can watch the entire movie from beginning to the end. We previously proposed a method to reduce waiting time in P2P streaming. In conventional methods, by receiving the first chunk of data sequentially from a peer with large bandwidth and making a delivery schedule that considers the finishing time to delivery each content, waiting time is reduced effectively. However, these methods do not consider the case where heterogeneous peers deliver data to multiple peers. Also, since selected peers deliver the data to a peer using all the bandwidth, the number of available peers that deliver data decreases. In this paper, we propose a scheduling method to reduce the waiting time for selecting peers in P2P streaming by selecting peers and considering the available bandwidth. By designing and implementing P2P streaming systems, we consider situations in which our proposed system is effective. Our evaluation shows that our proposed method reduces the average waiting time 62.5% more than conventional methods at maximum.

Key words: Peer-to-peer, streaming, scheduling, waiting time, selecting peers
Communicated by: D. Taniar

1 Introduction

Recently, streaming systems with Peer-to-Peer (P2P) technology have attracted much attention and are changing how we watch movies. In P2P streaming systems, a peer playing a movie receives data from other peers. By sequentially playing the received data, users can watch the data from the beginning to the end. To distribute the network load, in conventional methods, peers from which clients receive data are selected at random. However, when the bandwidth of the selected peer is small, the time required to finish receiving the data is too long. To reduce the waiting time, we must select peers effectively.

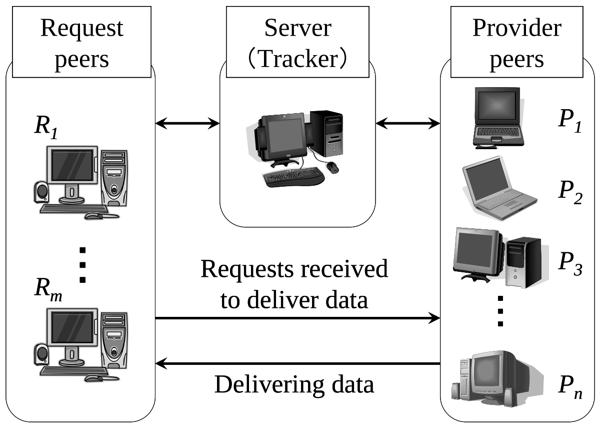


Figure 1 Assumed structure of peers in P2P streaming environment.

There are several methods to select a peer in P2P streaming systems. We previously proposed a method to reduce waiting time for a P2P streaming system that selected peers by considering the available bandwidth. Conventional methods are effective when a peer delivers the data to only one peer. But conventional methods fail to consider the case where a peer concurrently delivers data to multiple peers. Conventional methods are simple and give solid performance. However, in actual systems, peers deliver the data to multiple peers to distribute the data rapidly. When a peer starts delivering the data to other peers, the available bandwidth changes. Therefore, conventional methods do not work well since they assume that bandwidth is stable. By considering access to multiple peers, waiting time can be further reduced. The main contribution of this paper is that it considers more actual P2P streaming systems than conventional studies.

In this paper, we propose a scheduling method to reduce waiting time for selecting peers in P2P streaming systems. In our proposed method, the waiting time is reduced effectively by receiving the first data segment from peers with large bandwidth. By designing and implementing P2P streaming systems, we consider situations in which our proposed system is effective. Since it can introduce conventional scheduling methods, we can construct a delivery system based on the type of clients.

The remainder of the paper is organized as follows. Our assumed P2P streaming systems are explained in Section 2, where the reason our proposed method acquires channel bandwidth identical as the data consumption rate is explained. Related works are introduced in Section 3. Our proposed method is explained in Section 4, where we make a simulation model. Design and implementation are explained in Section 5. Our proposed method is evaluated in Section 6 and discussed in Section 7. Finally, we conclude the paper in Section 8.

2 P2P Streaming

In this section, we explain P2P streaming systems.

2.1 Assumed Network Structure of Peers

Our assumed P2P network structure is shown in Figure 1. In P2P networks, there are two types of peers: request and provider. In Figure 1, the request peer is set in the center of the network and is

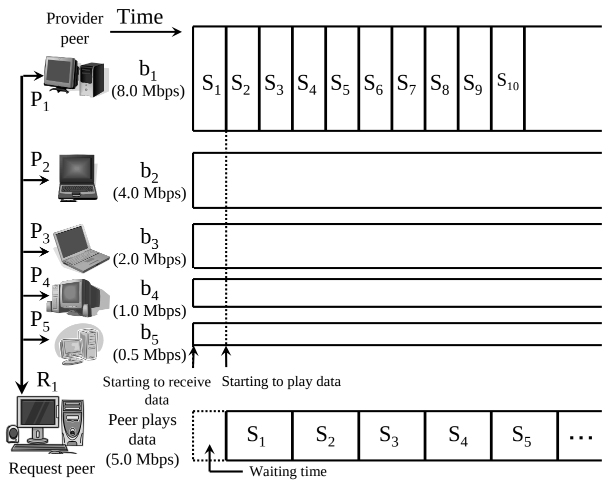


Figure 2 Example of broadcast schedule under simple method.

connected to many provider peers. It demands data and receives them from provider peers using the P2P network. The request peer finishes receiving the initial part of the data and plays it. When the request peer finishes receiving all the data, it becomes the provider peer. When the request peer wants data from the provider peers, it receives data separated into segments. The remarkable points of our paper are summarized below:

(1) Selecting request peers: The request peer receives data from provider peers. In conventional methods, the request peer selects provider peers randomly. In our proposed method, waiting time is reduced effectively because provider peers are selected based on available bandwidth.

(2) Bandwidth of peers: The available bandwidth between the request and provider peers is different. In our proposed method, available bandwidth is set by Pareto distribution, which is used for bandwidth distribution [1].

(3) Data size of each segment: In P2P streaming, the playing time of the data often becomes too long. The request peer receives the data separated into segments. When the data size of each segment is small, the receiving time of each segment is reduced. However, since the processing overhead to play the data grows, the optimal data size of each segment has to be set carefully based on the system [2]. For example, for BitTorrent [3], the data size of each segment is 256 bytes.

When the request peer receives a segment from provider peers, waiting time occurs, which often annoys users.

2.2 Waiting Time

In this subsection, we explain the mechanism for waiting time generation. Since there is only one server, in the streaming of on-demand delivery, waiting time is in inverse proportion to the available bandwidth. However, in conventional P2P streaming, many provider peers exist. Since the request peer separates the data into several segments and delivers them, waiting time greatly changes based on the available bandwidth of each provider peer.

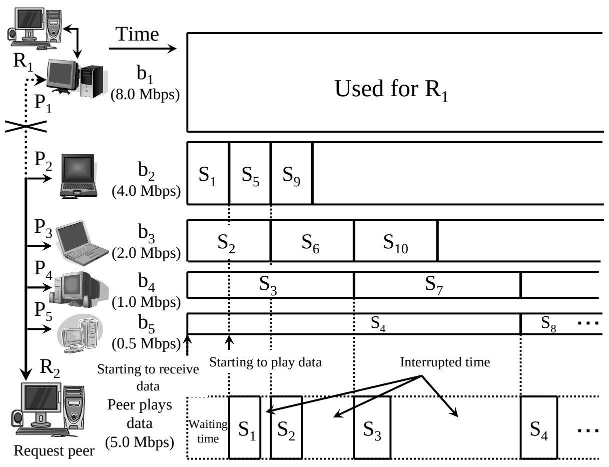


Figure 3 Example of broadcast schedule under simple method (using P_2, …, P_5).

Table 1 Waiting time from receiving data to starting to play them.

	Waiting time (sec.)
Time for receiving S_1	5
Interrupted time between S_1 and S_2	1
Interrupted time between S_2 and S_3	7
Interrupted time between S_3 and S_4	15
Interrupted time between S_7 and S_8	20
Total	48

In conventional methods, the request peer chooses provider peers sequentially from a peer with large bandwidth. For example, when the request peer wants data from the provider peers and receives segments, the delivery schedule is shown in Figure 2. The request peer is R_1, and the available bandwidth is 10 Mbps. The provider peers are P_1, …, P_5. The bandwidth of b_1 is 8.0 Mbps, b_2 is 4.0 Mbps, b_3 is 2.0 Mbps, b_4 is 1.0 Mbps, and b_5 is 0.5 Mbps. The data consumption rate is 5.0 Mbps. When the data are separated into n segments, the separated segments are S_1, …, S_n. When the total data size is 25.6 MB and the data size of each segment is 2.56 MB, n = 25.6 / 2.56 = 10. In Figure 2, when the provider peer receives S_1 for b_1, waiting time is merely the receiving time of S_1, which is 2.56 * 8 / 8.0 = 2.56 sec.

Next, we explain the case where the request peer cannot receive the data from P_1. In conventional methods, a request peer receives the data from a provider peer using all available bandwidth. Therefore, when R_1 receives data from P_1, R_2 can not receive them from P_1. For example, when R_2 requests data from P_1, …, P_5 and R_1 requires data from P_1, the delivery schedule is shown in Figure 3. In this case, since R_2 cannot receive data from P_1, R_2 receives them from P_2, …, P_5. In the simple method, when R_2 wants data from P_2, …, P_5, it receives segment $S_{(i-1)+4j}$ (j=0, …, 2) by channel b_i (i=2, …, 5). In Table 1, when the provider peer receives S_1, …, S_{10} for b_2, …, b_5, the waiting time is 48 sec. The waiting time increases by a factor of 48 / 2.56 = 18.8 compared to selecting b_1.

In P2P streaming, if the request peer can accept a provider peer with more available bandwidth than the consumption rate, it can play the data without interruption, and waiting time is only the time for receiving S_1. Therefore, by acquiring a provider peer with more available bandwidth than the consumption rate, waiting time is reduced effectively.

3 Related Works

Several P2P delivering methods have been proposed [5, 6]. In BitTorrent [7], clients receive from peers the divided data of each segment called a piece. By providing a piece of data to other peers, the client can receive other data from them. Provider peers whose available bandwidth is small can also deliver data. Since many provider peers deliver popular content, many peers can receive it in P2P networks.

Gnutella [8, 9] is an application that shares data between clients. In this application, the client called a servant sends a message to other clients that it knows and waits for a response from them. Since the client receives the message by delivering it to other clients, the range of P2P networks is expanded. However, in these methods, since data must be downloaded, clients cannot play the data until they have finished receiving them.

CoolStreaming [10] is a data-driven overlay network for P2P streaming. By using an efficient scheduling algorithm to fetch video segments from each peer and a buffering system, CoolStreaming achieves smooth video playback and good scalability. PRIME [11] suggests that each P2P connection in a mesh streaming overlay should have roughly the same bandwidth to maximize the utilization of the available bandwidth in each provider peer. Zhang et al. proposed an optimal scheduling method for non-layered streaming, where the Min-Cost Flow Problem (MCFP) is employed for scheduling [12]. Gehlen et al. evaluate the performance mobile P2P Web Services be means of an analytical analysis [13].

In our paper, we focus on algorithm of selecting peers and scheduling segments. Conventional methods construct P2P networks such as mesh-based network and tree-based network and evaluate it. Our proposed method differ in making the delivery scheduling that reduces waiting time based on available bandwidth compared with conventional methods.

Several P2P streaming methods have also proposed [14, 15, 16, 17, 18]. Xu et al. proposed a P2P streaming concept where the request peer receives data from provider peers and analyzed the data capacity of P2P networks [19]. Shah et al. proposed a scheduling method to reduce waiting time [20] in which clients receive the divided data of each segment called a piece from peers using BitTorrent. In Narada [21], when the provider peer delivers data to request peers in P2P streaming, receiving time is reduced by reconstructing a P2P network and making a tree structure.

We previously proposed a scheduling method to reduce the waiting time in P2P streaming called the ``Waiting time Reduction for P2P Streaming (WRPS)" method [22]. In this method, waiting time is effectively reduced by sequentially receiving the first bit of data from a peer with large bandwidth. For example, in the case where the server delivers data shown in Figure 1, the delivery schedule is produced as in Figure 4. The request peer is R_1, and the available bandwidth is 10 Mbps. The provider peers are P_1, …, P_5. The bandwidth of b_1 is 2.0 Mbps, b_2 is 1.4 Mbps, b_3 is 0.9 Mbps, b_4 is 0.6 Mbps, and b_5 is 0.4 Mbps. The data consumption rate is 5.0 Mbps, and the playing time is 60 sec. First, the

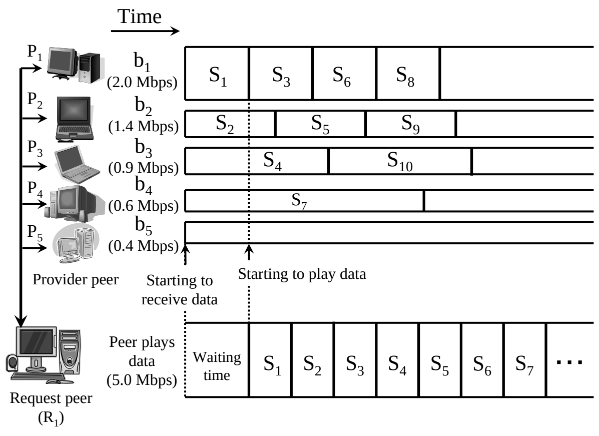


Figure 4 Example of broadcast schedule under WRPS method.

request peer receives S_1 for b_1. Next, the request peer receives S_2 for b_2. When the request peer receives S_3 for the channel that finished receiving the data, the time is the earliest. In this case, the request peer receives S_3 for b_1. Finally, as shown in Figure 4, all segments are scheduled. In this case, the average waiting time in the WRPS method is 1.0 sec. and 2.6 sec. in the simple method. Therefore, average waiting time under the simple method is reduced 61.5 %.

However, WRPS does not suppose the case where a provider peer concurrently delivers data to many request peers. Since the number of channels is reduced whose available bandwidth is the same as the consumption rate, waiting time increases.

4 Proposed Method

We reduce the waiting time for selecting peers in P2P streaming by proposing the ``Waiting time reduction considering Peer bandwidth for P2P Streaming (WPPS)" method. In WPPS, waiting time is reduced effectively by selecting peers that consider the available bandwidth.

4.1 Assumed Environment

Our assumed system environment is summarized below:

- The request peer receives data from one or more provider peers.
- Provider peers have all the data segments.
- Provider peers can be connected concurrently with request peers.
- Bandwidth is stable while delivering data.

In actual environments, several peers may exist between request and provider peers, and network delay can occur by hopping such peers. However, we ignore such peers because network delay can be considered a decrease of the bandwidth. By decreasing b_i, we can consider network delay.

Table 2 Variables for formulation.

Valuable	Explanation
D	Data size
n	Number of segments
w	Data size of each segment, $w = D / n$
r	Consumption rate
p	Number of connected peers, $p \le m$
S_i	State of playing segment, $i = 1, \ldots, n$
P_j	Provider peer, $j = 1, \ldots, p$
b_j	Bandwidth of P_j, $i = 1, \ldots, n$
$t_s(i)$	Starting time to deliver S_i
$t_f(i)$	Finishing time to deliver S_i

4.2 Scheduling Process

The following is the scheduling process under the WPPS method. Formulation values are summarized in Table 2.

(1) When R_i is connected to other request peers,

(a) $b_i \geq 2r$.

The request peer receives data using r out of b_i.

(b) $r < b_i < 2r$.

The request peer receives data using b_i - r from b_i.

(c) $b_i \leq r$.

The request peer does not receive data from R_i.

(2) When R_i is used, the request peer sequentially selects p peers that have large available bandwidth.

(3) $t_f(i)$ is calculated as follows:

$$t_f(i) = t_s(i) + \left(\frac{w}{r}\right) \times b_i \ . \qquad (1)$$

(4) S_i is scheduled from P_j, where the finishing time is the fastest when delivering the data from each provider peer.

(5) The values of $t_s(i)$ and $t_f(i)$ are updated.

(6) The request peer repeats processes 1 to 5 until it has received all of the data.

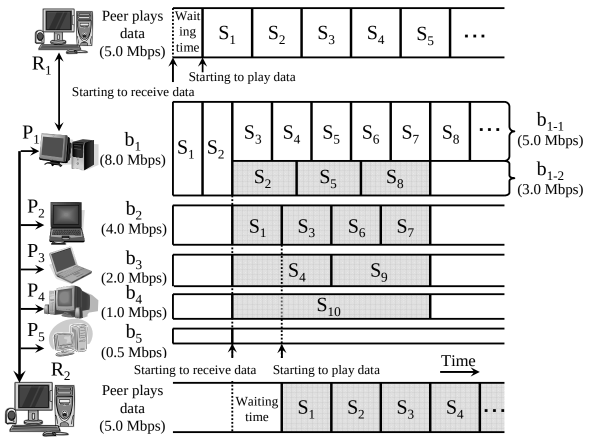


Figure 5 Example of broadcast schedule under proposed method.

4.3 Practical Example

A situation delivering data using WPPS by our proposed method is shown in Figure 5. The waiting time is reduced effectively by receiving the first data segment from provider peers that have larger bandwidth. The request peer schedules each segment by acquiring the available bandwidth that is the same as the consumption rate.

The provider peer produces a delivery schedule based on the procedure explained in Subsection 4.2 and delivers segments based on this schedule. For example, when the provider peer delivers segments explained in Subsection 2.2 and the available bandwidth is 10 Mbps, the delivery schedule produced by our proposed method is shown in Figure 5. The consumption rate is 5.0 Mbps. First, when R_2 requires data after R_1 finishes receiving S_2, since $r < b_1 < 2r$, the request peer receives it using a sub-channel ($b_{1\text{-}2}$) from b_1, whose available bandwidth is 8.0 - 5.0 = 3.0. In step 2, the request peer sequentially selects $b_{1\text{-}2}$, b_2, b_3, b_4 that have larger bandwidth. In step 4, S_1 is scheduled from P_2, where the finishing time is the fastest when delivering the data from each provider peer. In step 5, the values of $t_s(2)$ and $t_f(2)$ are updated. In step 4, S_2 is scheduled from $P_{1\text{-}2}$. Thus, the remaining S_3, …, S_{10} are scheduled. In Figure 5, when the provider peer receives S_1 for b_1, waiting time is identical as the receiving time of S_1, which is 2.56 * 8 / 4.0 = 5.1 sec.

5 Design and Implementation

To evaluate the availability of P2P streaming systems, implementing a P2P streaming system is important. In this research, we designed a P2P streaming system, and its design details are given below.

5.1 System configuration

In this subsection, we explain the details of the processes in the P2P system. As shown in Figure 1, a tracker plays the role of the server and manages the lists of contents and peers in the P2P networks. The list of contents is composed of a file name, a title, an abstract, a search tag, data size, and a list of

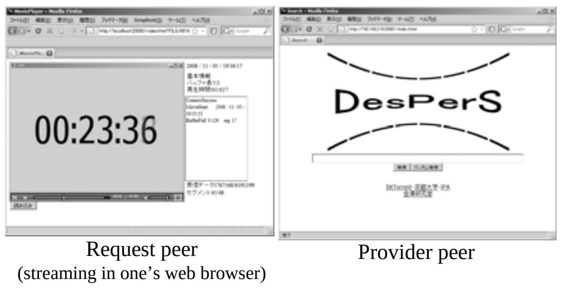


Figure 6 Screenshot for P2P streaming system.

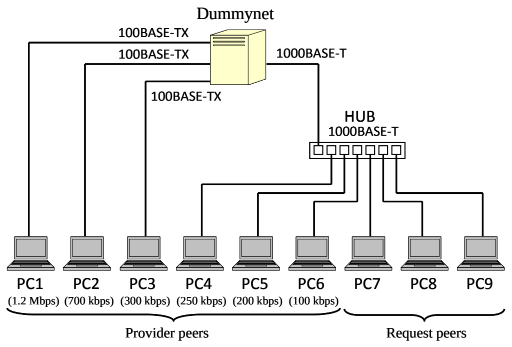


Figure 7 Experiment environment for P2P streaming system.

the peers with appropriate data. The list of peers is composed of an IP address, a port number, and the available bandwidth of each peer.

5.2 P2P Streaming System

We implemented a P2P streaming system based on the system design. A screenshot is shown in Figure 6. In our implementation, we use a WPPS method that can easily construct a delivery schedule.

The system configuration of our P2P system is shown in Figure 7. To control the available bandwidth of each provider peer, we used an artificial bandwidth control machine called FreeBSD Dummynet [23]. By setting Dummynet between the request and provider peers, the tracker can control the available bandwidth of each provider peer based on the network configuration.

Depending on the P2P streaming system, network configuration in actual environments can have many patterns. However, evaluating the performance of our proposed system for all of these patterns is not realistic. Therefore, we used a network configuration in which the tracker controls the available bandwidth of each provider peer using a Dummynet. In this system, we used nine machines: three request peers and six provider peers.

6 Evaluation

Actually, there are many network structures for P2P streaming systems. However, since the number of patterns is excessive, evaluating the performance of our proposed method for all of these patterns is not realistic. Therefore, in this paper, we use the network configuration shown in Figure 1. Also, the WPPS method is compared with the simple and WRPS methods.

In the simple method, to distribute the network load, peers from which the user receives data are selected at random. In WRPS [22], the request peer chooses provider peers sequentially from a peer with large bandwidth.

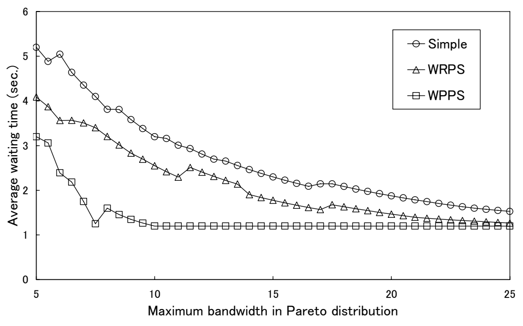


Figure 8 Waiting time and bandwidth.

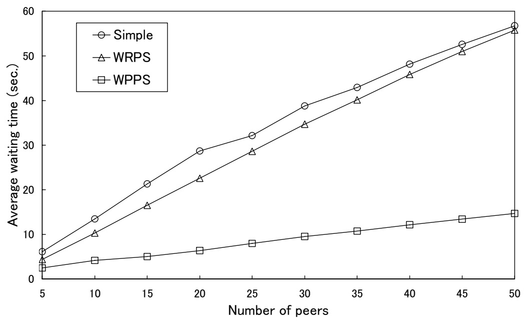


Figure 9 Waiting time and number of peers.

6.1 Simulation Evaluation

6.1.1 Available Bandwidth of each Provider Peer

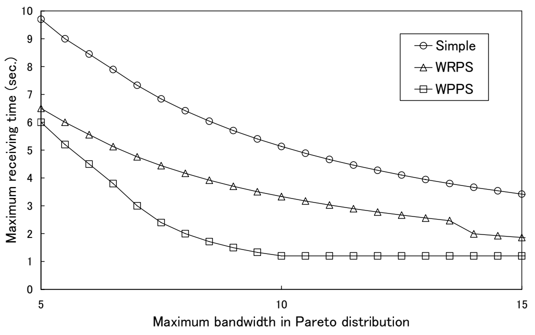
As explained in Section 2, the bandwidths of the provider peers follow a Pareto distribution. When k is a location parameter and $\alpha (1 < \alpha < 2)$ is a shape parameter, distribution function $F(x)$ is calculated as follows:

$$F(x) = 1 - \left(\frac{k}{x} \right)^{\alpha} \tag{2}$$

In our evaluation, available bandwidth follows a Pareto distribution, which is k = 1 and $\alpha = 1.5$ because evaluation by Pareto distribution is generally used in many researches for P2P streaming systems.

6.1.2 Bandwidth Influence

Since available bandwidth influences the average waiting time, the request peer may determine the bandwidth by considering the waiting time. Hence, we calculate the average waiting time under different bandwidths. The result is shown in Figure 8. The horizontal axis is the maximum bandwidth in a Pareto distribution. The vertical axis is the waiting time. The playing time is 60 sec., the number of provider peers is 50, and the number of receivable peers n is 10. The values of the variables assume a P2P network in which the number of connectable clients is a maximum of 100. The consumption rate is 5.0 Mbps. ``Simple" denotes the waiting time under the simple method, ``WRPS" denotes the waiting time under the WRPS method, and ``WPPS" denotes the waiting time under the WPPS method. In this graph, the average waiting time under WRPS is shorter than the simple and WRPS methods. In WPPS, waiting time is reduced effectively by scheduling the data by considering the available bandwidth for each provider peer. When the available bandwidth is more than 10 Mbps, since the request peer can receive the data without interruption, the waiting time is merely the receiving time of S_1. For example, when the total available bandwidth is 10 Mbps, the waiting time under the simple, WRPS, and WPPS methods is 3.2, 2.5, and 1.2 sec., respectively.



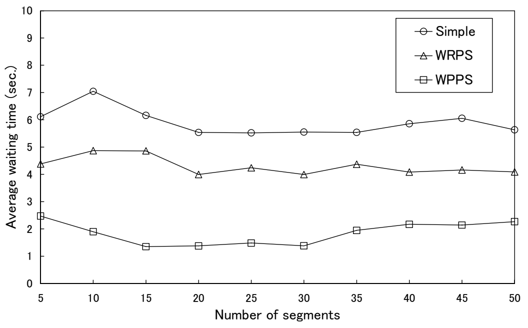


Figure 10 Maximum receiving time and bandwidth.

Figure 11 Average waiting time and number of segments.

6.1.3 Effect of Number of Peers

When several request peers demand data concurrently from several provider peers, the number of request peers increases where waiting time occurs until starting to receive the data. When the total available bandwidth is limited, since the available bandwidth of each provider peer is reduced, waiting time increases. Hence, we calculate the average waiting time under the number of provider peers for each method. The result is shown in Figure 9. The horizontal axis is the number of provider peers to which the request peer can connect. The vertical axis is the waiting time. For evaluation, we used the network configuration shown in Figure 1. The playing time is 60 sec., the available bandwidth is 10 Mbps, the number of provider peers is 50, and the consumption rate is 5.0 Mbps. In this graph, the average waiting time under the WPPS method is shorter than the simple and WRPS methods. In WPPS, waiting time is reduced by acquiring available bandwidth that is identical as the consumption rate at maximum. In conventional methods, since the request peer receives data from one provider peer using all the bandwidth, the provider peer whose available bandwidth is more than the consumption rate cannot concurrently deliver data to other request peers.

6.1.4 Maximum Waiting Time

In P2P streaming broadcasts, since the maximum waiting time is also a factor of system performance, we calculate it under different bandwidths. The result is shown in Figure 10. The horizontal axis is the maximum bandwidth in a Pareto distribution. The vertical axis is the maximum waiting time. For evaluation, we used the network configuration shown in Figure 1. The playing time is 60 sec., the available bandwidth is 10 Mbps, the number of provider peers is 50, and the consumption rate is 5.0 Mbps.

In this graph, the maximum waiting time under the WPPS method is shorter than under the simple and WRPS methods. In WPPS, by acquiring the bandwidth of each provider peer, waiting time is reduced effectively. Also, when the total available bandwidth is more than 10 Mbps, the maximum waiting time becomes constant. As explained in Subsection 2.2, this is because waiting time is only the receiving time of S_1.

6.1.5 Influence of Number of Segments

We calculated the waiting time under the number of segments. The result is shown in Figure 11. The horizontal axis is the number of segments. The vertical axis is the average waiting time. The playing time is 60 sec., the number of provider peers is 50, and the number of receivable peers *n* is 10. The consumption rate is 5.0 Mbps. For example, when the number of segments is 100, the data size of each segment is 5.0 * 8 * 60 / 100 = 24 Mbytes.

In this graph, the average maximum waiting time under the WPPS method is shorter than the simple method. In WRPS, when the number of segments increases, since the request peer schedules data effectively by considering the conclusion of the delivery time, waiting times do not increase compared to the conventional methods. Otherwise, in the simple and WRPS methods, since the request peer receives segments from provider peers whose available bandwidth is small compared to the WPPS method, waiting time increases. For example, when the number of segments is 25, waiting time under the simple, WRPS, and WPPS methods is 5.5, 4.2, and 1.5 sec., respectively. Therefore, the average waiting time under the WPPS method is reduced 73.2 % compared to the simple method and 65.1 % compared to the WRPS method.

6.2 Evaluation in Actual Environment

In this subsection, we evaluate our P2P streaming system.

6.2.1 Evaluation Environment

In our evaluation, the available bandwidth of each provider peer is 1.2 Mbps, 700, 300, 250, 200, and 150 kbps. The consumption rate is 1.0 Mbps. The playing time of the data was 60 sec., and the data size of each segment was 128 Kbytes. The request peer can play the data after it finishes receiving the initial part, which is 10 sec. When there is no more data in the buffer, the request peer stops playing them and waits until it finishes receiving the data, which takes 10 sec. The data size of each segment is 128 KBytes.

6.2.2 Waiting time

We calculated the waiting time under several methods. The result is shown in Figure 12. The horizontal axis is the number of selected machines, which are PC7, PC8, and PC9. The vertical axis is the waiting time. To compare the conventional methods with the proposed method, we calculated the waiting time in each method. ``Simple" denotes the waiting time under the simple method, ``WRPS" denotes the waiting time under the WRPS method, and ``WPPS" denotes the waiting time under the WPPS method.

In this graph, when PC7, since the request peer selects the same provider peers, waiting time is only slightly different among the simple, WRPS, and WPPS methods. When PC9, the waiting time under the WPPS method is reduced compared to the simple and WRPS methods because peers were selected by considering the available bandwidth. When PC7, the number of selected provider peers under the simple and WRPS methods is PC1 and 768 kbps under the WPPS method, which is identical as the consumption rate. Therefore, when PC9, the number of selected provider peers under the simple and WRPS methods is PC4, PC5, and PC6, whose total available bandwidth is 250 + 200 + 150 = 600 kbps.

Meanwhile, under the WPPS method it is PC3, PC4, PC5, and PC6, and the total available bandwidth is 300 + 250 + 200 + 150 = 900 kbps. For example, when the number of selected machines is PC9, the waiting time under the simple, WRPS, and WPPS methods is 81.2, 51.4, and 24.3 sec., respectively. Therefore, the average waiting time under the WPPS method is reduced 70.1 % compared to the simple method and 52.7 % compared to the WRPS method.

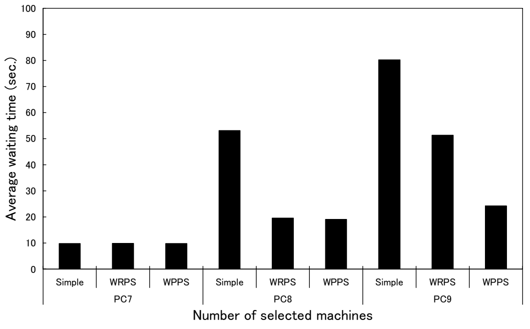


Figure 12 Comparison of waiting time in actual environment (video rate: 1.0 Mbps).

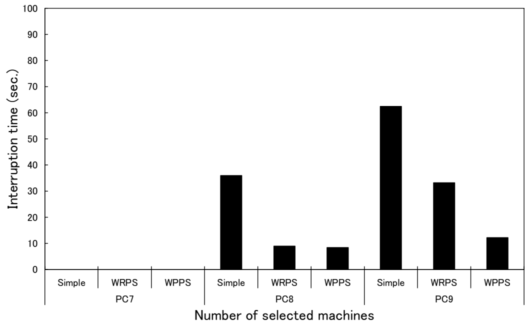


Figure 13 Comparison of interruption time in actual environment (video rate: 1.0 Mbps).

6.2.3 Interruption Time

We calculated the interruption time under several methods. The result is shown in Figure 13. The horizontal axis is the number of selected machines, which are PC7, PC8, and PC9. The vertical axis is the interruption time. To compare the conventional methods with the proposed method, we calculated the waiting time in each method.

In this graph, as the number of provider peers increases, interruption time is reduced. When the number of provider peers increases, since their available bandwidth increases, the number of empty segment of data in the buffer decreases. For example, when the number of selected machine is PC9, the interruption time under the simple, WRPS, and WPPS methods is 62.5, 33.3, and 12.2 sec., respectively. Therefore, the average waiting time under the WPPS method is reduced 46.7 % compared to the simple method and 63.4 % compared to the WRPS method.

7 Discussion

7.1 Comparison with Conventional Methods

By acquiring a provider peer whose available bandwidth is more than the consumption rate, clients can play the data without interruption after receiving them. In the P2P streaming schedule, many channels are necessary where the available bandwidth is the same as the consumption rate. In conventional methods, the request peer receives data from one provider peer using all the bandwidth. Since the number of channels is reduced whose available bandwidth is the same as the consumption rate, waiting time increases. Our proposed method acquires channels whose available bandwidth is the same as the consumption rate. Since the number of channels where the available bandwidth is the same as the consumption rate is reduced, we can the reduce waiting time.

7.2 Optimal Number of Provider Peers

In P2P networks, the total available bandwidth is limited for which provider peers are used. When the number of provider peers connected concurrently increases, since the bandwidths of each provider peer increase, the average waiting time also increases. In the WPPS method, by acquiring channels whose bandwidth is the same as the consumption rate, the number of request peers that cannot use channels to receive data decreases. Therefore, in WPPS, the increasing rate of the average waiting time based on increasing the number of receivable peers is smaller than the simple and WRPS methods. When the number of provider peers increases whose available bandwidth is more than the consumption rate, the increasing rate of the average waiting time is reduced.

7.3 Comparison of Simulation Environment with Actual Environment

In Subsection 6.2, we confirmed that waiting time under the WPPS method is reduced to a greater extent than under the simple and WRPS methods. In the actual environment, the following must be considered: interruption in playing data, load balance in delivering segments, and overhead for receiving the data.

7.3.1 Interruption in Playing Data

In streaming delivery, since the request peer can play data after receiving a given amount of buffer size, we can reduce waiting time more than the download type. When the buffer size is large, if the receiving speed of the data is slower than the playing speed, the request peer can play the data while its playing time is stored in its buffer. However, since the data size becomes large, the waiting time from starting to receive data to starting to play them becomes long.

7.3.2 Load Balance in Delivering Segment Data

In P2P streaming, the size of each data segment is set based on the software. When the provider peer delivers a large amount of data at once, the time to deliver the data using available bandwidth becomes long. In the actual environments, the data size becomes smaller by delivering the data in segments. In the simulation environment, since load balance in delivering the segment data is not considered, the waiting time is reduced by receiving data that are equally divided by segments from provider peers. On the other hand, in the actual environment, when the provider peer delivers data that are equally divided by segments from provider peers, data are loaded each time and each segment is divided. In this case, since the load balance of the provider peer increases, processing time to deliver the data increases.

7.3.3 Overhead in Receiving the Data

In the actual environment, each peer is connected by TCP/IP. For delivering data, since the provider peer appends a header to the data packet, the data size increases, and delivering time is long.

7.4 Scheduling Methods in Actual Environment

In the WPPS method, the delivery schedule is set before starting to receive the data. When the provider peer, whose receiving rate is low, delivers the next segment during the interruption, since an interruption might occur, the receiving time increases, and the possibilities of an interruption become higher.

When an interruption occurs in playing the data, playing time increases. If the request peer finishes receiving the data before the starting time of playing them, it can play all of the data without interruptions. In our implemented P2P streaming system, we can play a piece of data by buffering it for about 10 sec. of playing time. Therefore, when the request peer finishes receiving all of the data within 60 + 10 = 70 sec., it can play the data without interruptions until the end.

8 Conclusion

In this paper, we proposed a scheduling method to reduce the waiting time for selecting peers in P2P streaming. In our proposed method, waiting time is effectively reduced by sequentially receiving the first segment of data from a peer with larger bandwidth. For example, when the total available bandwidth is 10 Mbps, waiting time under the simple, WRPS, and WPPS methods is 3.2, 2.5, and 1.2 sec., respectively. Therefore, the average waiting time under the WPPS method is reduced 62.5 % compared to the simple method and 52.0 % compared to the WRPS method. Also, we designed and implemented a P2P streaming system. With our implemented system, we consider situations in which our proposed method is effective.

In the future, we will make a scheduling method where the request peer delivers to many provider peers who sequentially watch several segments of data.

Acknowledgments

This research was supported in part by Research and Development Program of Ubiquitous Service Platform (2009), The Ministry of Internal Affairs and Communications, Japan. Also, this research was supported in part by Grant-in-Aid for Scientific Research (S) numbered 21220002, by Grant-in-Aid for Young Scientists (B) numbered 21700108 and a grant for Kyoto University Global COE Program, “Informatics Education and Research center for Knowledge-Circulating” from the Japanese Ministry of Education, Culture, Sport, Science and Technology of Japan.

References

1. Fujimoto K., Ata S. and Murata M., Statistical analysis of packet delays in the Internet and its application to playout control for streaming applications. IEICE Trans. on Communications. Vol.E84-B, No.6, 2001, 1504-1512.
2. Gotoh Y., Yoshihisa T. and Kanazawa M., d-Cast : A Division Based Broadcasting System for IP Networks. Proc. of IEEE Int. Conf. on Advanced Communication Technology (ICACT'07), 2007, 1902-1907.
3. BitTorrent, http://www.bittorrent.com.
4. Bonald T., Massoulie L., Mathieu F., Perino D. and Twigg A., Epidemic Live Streaming: Optimal Performance Trade-Offs. Proc. of ACM SIGMETRICS Int. Conf. on Measurement and modeling of computer systems, 2008, 325-336.
5. Hildrum K., Kubiatowicz J., Rao S. and Zhao B., Distributed Object Location in a Dynamic Network. Proc. of 14th annual ACM symposium on Parallel algorithms and architectures, 2003, 41-52.

6. Liu J., Rao S.G., Li B. and Zhang H., Opportunities and Challenges of Peer-to-Peer Internet Video Broadcast. Proc. of IEEE, Special Issue on Recent Advances in Distributed Multimedia Communications, Vol.96, Issue 1, 2008, 11-24.
7. B. Cohen, Incentives build robustness in BitTorrent. Proc. of 1st Workshop on Economics of Peer-to-Peer Systems, 2003.
8. Jnutella.org, http://www.jnutella.org.
9. Gnutella, http://www.gnutella.com.
10. Zhang X., Liu J., Li B. and Yum T.P., CoolStreaming/DONet: A data-driven overlay network for peer-to-peer live media streaming. Proc. of 24th IEEE INFOCOM Conference, Vol.3, 2005, 2102-2111.
11. Magharei N. and Rejaie R., PRIME: Peer-to-Peer Receiver-drIven MEsh-based Streaming. Proc. IEEE INFOCOM '07. 2007, 1415-1423.
12. Zhang M., Xiong Y., Zhang Q. and Yang S., On the Optimal Scheduling for Media Streaming in Data-Driven Overlay Networks. Proc. IEEE Global Telecommunications Conference, GLOBESCOM 06. 2006.
13. Gehlen G., Aijaz F., Zhu Y. and Walke B., Mobile P2P Web Services using SIP. Mobile Information Systems 3(3-4), 2007, 165-185.
14. Padmanabhan V.N., Wang H. and Chou P., Resilient peer-to-peer streaming. Proc. of 11th IEEE Int. Conf. on Network Protocols (ICNP'03), 2003, 16-27.
15. Cui Y. and Nahrstedt K., Layered peer-to-peer streaming. Proc. of 13th int. workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'03), 2003, 162-171.
16. Tran D., Hua K. and Do T., Zigzag: an efficient peer-to-peer scheme for media streaming. Proc. of 22nd IEEE INFOCOM Conference, Vol.2, 2003, 1283-1292.
17. Guo Y., Suh K., Kurose J. and Towsley D., P2Cast: Peer-to-peer Patching Scheme for VoD Service. Proc. of 12th Int. Conf. on World Wide Web (WWW), 2003, 301-309.
18. Guo Y., Suh K., Kurose J. and Towsley D., A Peer-to-Peer on-demand streaming service and its performance evaluation. Proc. of 2003 IEEE Int. Conf. on Multimedia & Expo (ICME 2003), 2003, 649-652.
19. Xu D., Hefeeda M., Hambrusch S. and Bhargava B., On peer-to-peer media streaming. Proc. of 22nd Int. Conf. on Distributed Computing Systems (ICDCS 2002), Vol.1, 2002, 363-371.
20. Shah P. and Paris J.-F., Peer-to-Peer Multimedia Streaming Using BitTorrent. Proc. of 26th Int. Performance of Computers and Communication Conf. (IPCCC 2007), 2007, 340-347.
21. Chu Y., Rao S. and Zhang H., A Case for End System Multicast. IEEE Journal on SAC, Special Issue on Networking Support for Multicast, 2002, 1456-1471.
22. Gotoh Y., Yoshihisa T. and Kanazawa M., Method to Select Peers to Reduce Waiting Time in P2P Streaming Broadcasts. IADIS Int. Conf. Telecommunications, Networks and Systems 2008 (TNS-CONF 2008), 2008, 120-124.
23. L. Rizzo, dummynet. http://info.iet.unipi.it/˜luigi/ip_dummynet/.