Distributed Multimedia Streaming over Peer-to-Peer Networks

Jin B. Kwon[1] and Heon Y. Yeom[2]

[1] Sunmoon University, Dept. of Computer Science,
Asan, Chungnam, 336-708, South Korea
jbkwon@sunmoon.ac.kr
[2] Seoul National University, Dept. of Computer Science,
Seoul, 151-742, South Korea
yeom@dcslab.snu.ac.kr

Abstract. A peer-to-peer model is very useful in solving the server link bottleneck problem of a client-server model. In this work, we discuss the problems of distributing multimedia content over peer-to-peer network. We focus on two problems in peer-to-peer media content distribution systems. The first is the transmission scheduling of the media data for a multi-source streaming session. We present a sophisticated scheduling scheme, which results in minimum buffering delay. The second problem is on the fast distribution of media content in the peer-to-peer system that is self-growing. We propose a mechanism accelerating the speed at which the system's streaming capacity increases.

1 Introduction

It is generally believed that the streaming media will constitute a significant fraction of the Internet traffic in the near future. Almost all of the existing works on multimedia streaming is based on client-server models. Since multimedia streaming requires high bandwidth, server network bandwidth runs out rapidly if unicast client-server model is used. Single source multicast is one of the solutions that use a single stream to feed all the clients. The deployment of IP multicast has been slowed by difficult issues related to scalability, and support for higher layer functionality like congestion control and reliability. A peer-to-peer(P2P) model is ideal as a model to solve the server link bottleneck problem. In the P2P model, multimedia contents are distributed by using the bandwidth of the clients themselves.

The clients in P2P systems contribute resources to the community and in turn use the resources provided by other clients. More specifically, supplying peers holding a certain media file may stream it to requesting peers. Thus, data traffic is not localized on a specific site since the peers cooperate for sharing contents. It is typical that there is no central server that holds the contents, and peers work on an equal footing. How the contents exist at first in the P2P system is another question. We assume that there are seed peers. Examples of P2P content distribution systems include Napster[2], Gnutella[1], and so on.

There has been much work on P2P systems in recent years[3,6,7,8]. Those works dealt mainly with data lookup and storage management in a general P2P system. The problems on distributing *streaming media* over P2P network have also been studied in [4,5,9]. However, we dare say that the work on P2P media streaming systems is still in the early stage, and there is still some room for improvement of performance and generalization of model.

In this paper, we focus on two problems in P2P media content distribution systems. We first concentrate on the transmission scheduling of the media data for a multi-supplier P2P streaming session. More specifically, given a requesting peer and a set of supplying peers with heterogeneous out-bound bandwidth, the problem is how to schedule the segments of the media data using multiple channels respectively established with each supplying peer. A buffering delay is determined by the streaming schedule of each channel. The transmission scheduling has already been studied in [9], where Xu et al. presented OTS as its solution, which is optimal when the length of segments, the scheduling units, is identical. We present another scheduling scheme called *fixed-length slotted scheduling*(FSS), which further reduces the buffering delay. Unlike OTS, FSS employs variable length segments whose size is determined based on the bandwidth with which each segment is transmitted. The second problem is the fast distribution of media contents. Initially, there are only a few seed peers holding a content, and non-seed peers request the content as requesting peers. The P2P system is self-growing since the requesting peers can become supplying peers after they receive all the data. However, in the beginning, since there is only a small number of peers holding the content, the system can accommodate only a limited request arrival rate. The number of supplying peers grows as the content spreads out, and the system would eventually be able to service all the requests from other peers. Therefore, for a given arrival rate, it is very important to convert the requesting peers to the supplying peers in a short time so that all the incoming requests can be serviced. We have come up with a mechanism accelerating the speed at which the P2P system capacity increases, called FAST. It accelerates the distribution speed by allowing requesting peers that satisfy a condition to supply contents to other peers.

At first, we define a P2P media streaming model and state our assumptions. Our model is more practical and more general than the models of the previous works. A set of peers that can supply a media data is defined as the *candidate* set of the media content. A requesting peer selects its supplying peers from the set, opens a channel with each selected supplying peer, and requests the data segments from them according to a scheduling mechanism. The requesting peer receives the data from multiple channels and store them in its local storage, and then the peer may become a candidate of the media content. Note that a supplying peer can supply the media data to multiple requesting peers. Since the content searching problem in P2P network is another research issue, we assume that a requesting peer can get the candidate peer set and the information about resource usage of each peer using an appropriate searching mechanism. Let γ denote the playback rate of the media data. Each requesting peer P_r has an in-bound bandwidth of $R_{in}(r)$ and an out-bound bandwidth of $R_{out}(r)$. Peers are heterogeneous in their in-bound and out-bound bandwidth. We assume that $0 < R_{in}(r) \leq \gamma$ and $R_{out}(r) > 0$.

2 Transmission Schedule

In this section, we study the problem of media data transmission scheduling. The problem is stated as follows: For a requesting peer P_r and n channels, to determine the data segments to be transmitted over each channel and the transmission order of the segments. The goal is to minimize buffering delay while ensuring a *continuous playback* at P_r, with *minimum* buffering delay. The buffering delay is defined as the time interval between the start of streaming and the start of playback at P_r. How to select the supplying peers of P_r is dealt with in Section 3. If the data is transmitted over multiple channels of various bandwidth lower than γ, the buffering delay is inevitable. That is because the transmission order of data is not the same as the playback one. Therefore, a well-devised data transmission schedule is essential in reducing the buffering delay.

A function $p(t)$ is defined as the amount of data being played for t seconds since the beginning of the playback. And a function $d(t)$ is defined as the amount of *consecutive data* from the beginning of the media file received for t seconds since the beginning of the streaming, at P_r. Since the data is assumed to be encoded in CBR, we express the amount of data in seconds. The data amount of k means the amount to be played for k seconds. To ensure the continuous playback, the following condition must be satisfied:

$$\forall t \geq 0, \quad d(t) \quad \geq \quad p(t). \tag{1}$$

Fig. 1(a) illustrates the increasing shape of $p(t)$ and $d(t)$ for the example of OTS[9]. In the figure, there are four channels with bandwidth of $\frac{\gamma}{2}, \frac{\gamma}{4}, \frac{\gamma}{8}$ and $\frac{\gamma}{8}$, respectively, and the requesting peer schedules the transmission of eight $\frac{\gamma}{8}$-second segments over the channels, according to OTS. As shown in the figure, the buffering delay δ is required to satisfy the condition of Eq.(1). δ is $\frac{3L}{8}$ in this example. Hence, $p(t)$ can be expressed as follows:

$$p(t) = \min\{\max\{t - \delta, 0\}, L\}, \quad t \geq 0, \tag{2}$$

where δ is the maximum of the difference between the dashed line($y = t$) and $d(t)$. That is, $\delta = \max_{t \geq 0}\{t - d(t)\}$. The closer $d(t)$ is to the straight dashed line, the smaller the buffering delay. A well-designed scheduling can make $d(t)$ close to the straight line. We propose the *fixed-length slotted scheduling*(FSS) as such a scheduling scheme.

To increase $d(t)$ linearly, the requesting peer P_r should receive the data in a sequential order. Based on this idea, FSS assigns the data to fixed-length slots of each channel, and the one-slot data chunks are defined as the data segments of FSS. Since the bandwidth of each channel is variable, the segment length also varies according to the channel bandwidth to which the data segment is assigned. And, the variable-length segments are assigned to the channels in a

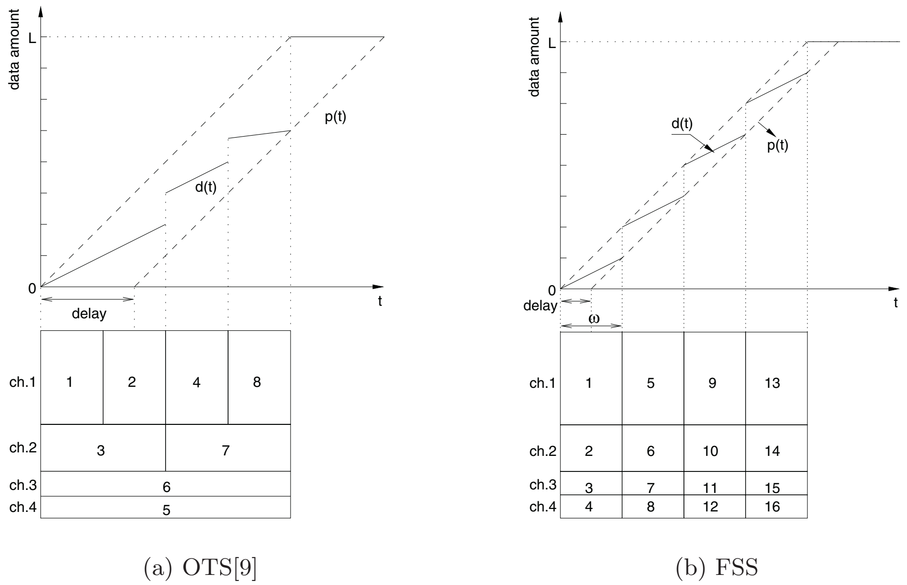


Fig. 1. Transmission Schedules

round-robin fashion. When the slot length is ω and i-th channel bandwidth is B_i, the segment length of channel i is ωB_i. Fig. 1(b) illustrates the concept of FSS, where the number of channels and the bandwidth of each channel are the same as those of Fig. 1(a). The slot length ω is $\frac{L}{4}$. In this example, the buffering delay of FSS is $\frac{L}{8}$, which is only a third of that of OTS.

Let us find the buffering delay, δ, of FSS. When the number of channels is n, the aggregated in-bound bandwidth $B^* = \sum_{i=1}^{n} B_i$. Then, $d(t)$ is:

$$d(t) = \min\{\frac{B_1}{\gamma}t + \frac{B^* - B_1}{\gamma}\left\lfloor\frac{t}{\omega}\right\rfloor\omega,\ L\}. \tag{3}$$

By expanding Eq.(1) with the $d(t)$ of Eq.(3) and the $p(t)$ of Eq.(2) and solving for δ, we can obtain the minimum δ. The detailed derivation is omitted here. The minimum buffering delay δ is:

$$\delta \ = \ \left(\frac{\gamma}{B^*} - 1\right)L + \left(\frac{B^* - B_1}{\gamma}\right)\omega, \quad \text{if} \ \ B^* \le \gamma. \tag{4}$$

From the above equation, δ depends on B_1 and ω. This relationship is such that δ becomes smaller as B_1 gets greater and ω gets smaller. Thus, FSS chooses the channel with the maximum bandwidth as the first channel to minimize the buffering delay. When the aggregated bandwidth is equal to the playback rate, the shorter the slot length is, the shorter the buffering delay gets. However, the slot length is required to be long enough to cover the fluctuation of bandwidth and the overhead of transmission, delivery processing, and so on. Also, if it is too smaller than the maximum packet size of the underlying physical network, the network utilization would be low and the bandwidth would be wasted. Therefore, the slot length is a system parameter that should be carefully determined.

3 Fast Distribution

In [9], the candidate set consists of only the peers holding the whole media file. However, some of the peers while downloading the media data may be able to supply the data to requesting peers. Since the system capacity is proportional to the size of the candidate set, it would be beneficial to make the candidate set as large as possible. If it has a large candidate set, it would be easier for a requesting peer, P_r, to acquire sufficient bandwidth, and P_r would become a candidate peer within a short time. Thus, our goal is to find the peers that have enough data to act as supplying peers, in order to make the candidate set as large as possible.

A peer holding the whole media file is defined as a *mature* peer, and a peer being downloading the media data is defined as an *immature* peer. Fig. 2 shows P_r and its supplying peers. The shaded ones indicate mature peers, and the white ones indicate immature peers. Let $d_i(t)$ be $d(t)$ of a peer P_i, which we defined above, and we define another function as follows.

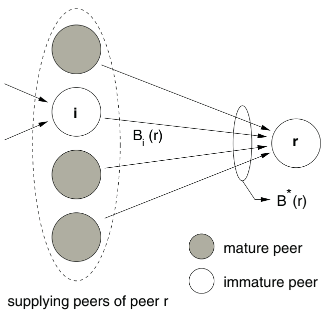
– $x_i(t, r)$: *when* P_i *is assumed to be selected as a supplying peer of a requesting peer* P_r*,* the position within the media file *of the data to be requested to transmit at time* t*.*

According to FSS, a supplying peer does not transmit the data continuously in a consecutive order, but transmits some data segments and skips some data segments in a predetermined pattern. Considering the instance of Fig. 1(b). The second channel transmits four segments of $\frac{\gamma}{4}\omega$ with the distance of $\frac{3\gamma}{4}\omega$. Therefore, as shown Fig. 2(b), $x_i(t, r)$ is a slanted staircase-shaped function, and the rate of increase of the solid line is $\frac{B_i(r)}{\gamma}$, where $B_i(r)$ is the i-th channel bandwidth of P_r. Also, the width of a step in the function corresponds to the slot length. $d_i(t)$ is L in case that P_i is a mature peer, and is determined by Eq.(3) in case that P_i is an immature peer. Fig. 2(b) shows $d_i(t)$ when P_i is an immature peer. That $x_i(t, r)$ crosses $d_i(t)$ as shown in the figure means that P_r would request the data which P_i would not have already received. Thus, P_i cannot be a supplying peer of P_r in this case. However, for an immature peer P_i to be a supplying peer of P_r, the following conditions must be satisfied:

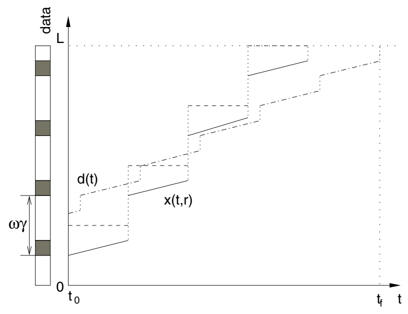
$$\forall t \geq t_0, \quad d_i(t) \ \geq \ x_i(t, r). \tag{5}$$

The immature peers satisfying this condition are called *semi-mature* peers of P_r. Consequently, the candidate set of P_r can consist of mature peers and semi-mature peers of P_r.

Although $d_i(t)$ has already been determined at the present time t_0, $x_i(t, r)$ has not. That is because $x_i(t, r)$ cannot be determined until P_r finishes selecting its supplying peers. For this reason, we use an upper bound function of $x_i(t, r)$, $\bar{x}_r(t)$, instead. The upper bound function $\bar{x}_r(t) = R_{in}(r) \cdot \lfloor (t - t_0)/\omega \rfloor \omega$. In Fig. 2(b), the staircase-shaped dashed line indicates $\bar{x}_r(t)$. Therefore, a sufficient condition of Eq.(5) is: $\forall t \geq t_0, d_i(t) \geq \bar{x}_r(t)$. It can be used for a criterion to determine whether an immature peer P_i can be a semi-mature peer. However, since it is a sufficient condition, not satisfying it does not mean that P_i is not



(a) Semi-mature Peer



(b) Data Growth of Peer i

Fig. 2. Candidate Peers

a semi-mature peer. But, if it is satisfied, we only ensure it is a semi-mature peer. The procedure to select supplying peers is as follows: P_r determines its candidate set $\mathbb{C}_r$ such that each peer in $\mathbb{C}_r$ is a mature peer or a semi-mature peer and it has some available out-bound bandwidth [1]. To determine $\mathbb{C}_r$, P_r should know whether or not each immature peer is a semi-mature peer, by testing $\forall t \geq t_0, d_i(t) \geq \bar{x}_r(t)$. Since the higher the first channel bandwidth is, the shorter the buffering delay in FSS(Eq.(4)), the peer with the maximal $v_{out}(i)$ among $\mathbb{C}_r$ is chosen as the first supplying peer of P_r. This procedure is repeated until the aggregated bandwidth $B^*(r)$ is equal to $R_{in}(r)$. The formal algorithm is omitted here due to the space limitation.

In case that there are so many requesting peers that the P2P system is beyond its capacity, it may not acquire enough bandwidth. However, in the latter case, P_r can have a choice among three policies. The first is to start downloading with the acquired channels(FAST1), the second is to withdraw the request and retry after σ(FAST2), and the third is to start downloading with the acquired channels and retry to acquire the remainder after T minutes(FAST3). In FAST3, unlike the others, the number of in-bound channels of P_r may be changed during a session. This means the change of $d_r(t)$, which accordingly affects the transmission schedule and the condition for a semi-mature peer. The details of this dynamic session is omitted here due to the space limitation.

4 Performance Study

First, we compare the buffering delay of FSS with that of OTS. However, since OTS has the restrictions on channel bandwidth and FSS has the concept of slots, it is not simple to compare them directly. For a fair comparison, we set a criterion on the slot length and evaluate the two schemes under the condition satisfying the restrictions of OTS. The restrictions of OTS is the following: 1) A single

[1] P_r is assumed to be able to know all the information needed to determine $d_i(t)$ for each immature peer P_i

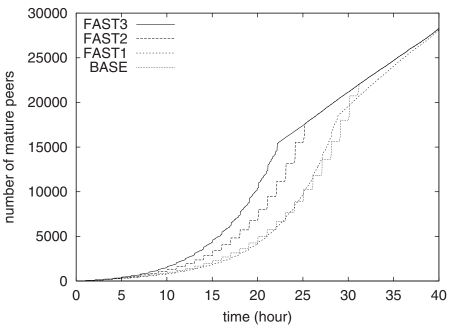


Fig. 3. Distribution Speed

channel bandwidth has one of $\frac{\gamma}{2}$, $\frac{\gamma}{4}$, $\frac{\gamma}{8}$, ..., $\frac{\gamma}{2^N}$. 2) The aggregate bandwidth of a session $B^*(r) = \gamma$. Let m be the number of segments. Given a set of n supplying peers and a requesting peer P_r, OTS achieves $\delta_{OTS} = (n-1)L/m$[9], if we assume that the content can be played while being downloaded. Under the same condition, the buffering delay of FSS is: $\delta_{FSS} = (1-B_1(r)/\gamma)\cdot\omega$, according to Eq.(4). Since δ_{OTS} depends on m and δ_{FSS} on ω, the relation between m and ω should be drawn for the comparison. In OTS, the time taken for a segment to be transmitted over the channel with the highest bandwidth is corresponding to the slot length. That is, $\frac{\gamma}{B_1(r)} \cdot \frac{L}{m} = \omega$, and then, by applying it to δ_{OTS}: $\delta_{OTS} = (n-1)\frac{B_1(r)}{\gamma} \cdot \omega$. Here, let $c = B_1(r)/\gamma$. Then, since $B_1(r)$ is the highest bandwidth, n must be greater than or equal to $1/c$ to satisfy that $B^*(r) = \gamma$. Hence, $n \geq 1/c$. For example, if $B_1(r) = \gamma/4$, $n \geq 4$. Therefore,

$$\delta_{OTS} = (n-1)c\cdot\omega \quad \geq \quad (1/c-1)\,c\cdot\omega = \delta_{FSS}.$$

Finally, the buffering delay of FSS is smaller than or equal to that of OTS. In the example used in Fig. 1(a) and 1(b), $\delta_{OTS} = 3\omega/2$ and $\delta_{FSS} = \omega/2$. FSS has only a third the buffering delay of OTS in that case.

Second, we study the performance of FAST using a simulation. We simulate a P2P system with total of 50,100 peers. Initially, there are only 100 'seed' peers, while the other 50,000 peers request the media data according to a given request arrival rate. The request arrival follows a Poisson distribution with mean $1/\theta$. Each seed peer possesses a copy of a popular video file. The running time of the video is 60 minutes. The in-bound bandwidth of all the peers is γ, equal to the playback rate. The out-bound bandwidth of the seed peers is $\frac{\gamma}{2}$, and that of the others is $\frac{\gamma}{2}$, $\frac{\gamma}{4}$, $\frac{\gamma}{8}$, or $\frac{\gamma}{16}$ with the distribution of 10%, 10%, 40%, and 40%, respectively. The time interval T for retrying to acquire bandwidth has an uniform distribution with a mean of 10 minutes. And, the slot length ω is set also to be 10 minutes. A BASE scheme for comparison to FAST considers only mature peers as supplying peers, and it allows a requesting peer to start downloading when it acquires a sufficient bandwidth like FAST2.

As the content is distributed to more peers, the P2P system's capacity increases. Thus, when requests arrive as Poisson process with a rate of $1/\theta$, not all of the requests may be serviced at the beginning stage when there are only a small number of mature peers and semi-mature peers. This congestion due to the high arrival rate lasts for a while, until the number of mature peers grows sufficiently. We define *θ-capacity time* as the time to reach a capacity being able to service all the requests arriving at a rate of $1/\theta$. The smaller θ-capacity time means the faster distribution of the media content. Fig. 3 shows the number of mature peers of BASE, FAST1, FAST2, and FAST3 when $\theta = 5$ seconds. If the system reaches θ-capacity, the number of mature peers increases linearly at a rate of $1/\theta$. However, as shown in the figure, the lines increase in a concave fashion in the early stage, and then increase in a straight line. The concave curves indicate that the system is congested. The 5-capacity time of FAST3, about 21 hours, is shorter than those of FAST2, FAST1, and BASE, which are about 24, 28, and 30 hours, respectively. And, FAST3 has almost more than twice the number of mature peers than that of BASE.

5 Conclusion

We have discussed the problems of distributing multimedia content over P2P network. The first problem is the transmission scheduling of the media data for a multi-source streaming session. We have presented a sophisticated scheduling scheme, which results in minimum buffering delay. The second one is on the fast diffusion of media content in the P2P system that is self-growing. We have also proposed a mechanism accelerating the speed at which the system's streaming capacity increases.

References

1. Gnutella. `http://gnutella.wego.com`.
2. Napster. `http://www.napster.com`.
3. I. Clake, O. Sandberg, B. Wiley, and T. Hong. Freenet: A Distributed Anonymous Information Storage and Retrieval System. In *Proc. of Workshop on Design Issues in Anonymous and Unobservability*, July 2000.
4. T. Nguyen and A. Zakhor. Distributed Video Streaming Over Internet. In *Proc. of Multimedia Computing and Systems*, San Jose, California, January 2002.
5. Venkata N. Padmanabhan, Helen J. Wang, Philip A. Chou, and Kunwadee Sripanidkulchai. Distributing Streaming Media Content Using Cooperative Networking. In *Int. Workshop on Network and Operating Systems Support for Digital Audio and Video*, Miami Beachi, FL, May 2002.
6. S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A Scalable Content-Addressable Network. In *ACM SIGCOMM*, August 2001.
7. A Rowstron and P. Druschel. Pastry:Scalable Distributed Object Location and Routing for Large-Scale Peer-to-Peer Systems. In *Proc. of IFIP/ACM Middleware*, November 2001.
8. I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan. Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications. In *Proc. of ACM SIGCOMM*, August 2001.
9. Dongyan Xu, Mohamed Hefeeda, Susanne Hambrusch, and Bharat Bhargava. On Peer-to-Peer Media Streaming. In *Proc. of Int. Conf. on Distributed Computing Systems 2002*, Austria, July 2002.