A Scheduling Scheme for Continuous Media Data Broadcasting With a Single Channel

Tomoki Yoshihisa, Masahiko Tsukamoto, and Shojiro Nishio

Abstract—Recently, various schemes for broadcasting continuous media data such as audio and video have been studied. Some of them have focused on reducing the waiting time of clients under the condition that clients can play data without interruption from beginning to end. These schemes usually employ multiple channels to broadcast continuous media data. However, clients for most broadcast systems such as wireless LAN, DVB and ISDB-T cannot receive data from multiple channels concurrently. In this paper, we propose and evaluate a scheduling scheme to reduce the waiting time of clients with a single channel. Preliminary results suggest that this scheme outperforms existing schemes.

Index Terms—Broadcasting, continuous media data, multimedia communication, streaming, video on demand.

I. INTRODUCTION

RECENTLY, broadcasting continuous media data such as audio and video has become increasingly popular. In a typical broadcast system, a server repetitively broadcasts data. Although the server can simultaneously deliver data to many clients, each client has to wait until its desired data is broadcast. On the other hand, in continuous media data broadcasting, it is important for clients to be able to play data without interruption from beginning to end. Accordingly, various schemes are proposed to reduce clients' waiting time under the condition that clients can play data continuously from start to finish ([1]–[3], [5]–[9], [16]). These schemes reduce the waiting time by dividing data into several segments and broadcasting segments via multiple channels. However, with multiple channels, clients are forced to receive data from multiple channels concurrently, and the server has to control all these multiple channels; hence, their mechanisms are more complicated than broadcasting data via a single channel. In practical systems, unfortunately, many clients cannot receive data from multiple channels for such hardware reasons as complexity or mounting space. For example, most of the commercial DVB (Digital Video Broadcasting) systems cannot receive data from multiple channels concurrently. Moreover, since almost all set-top-boxes, i.e.,

clients of DVB systems, have only one tuner, they cannot receive data from multiple channels concurrently. Accordingly, it is more realistic to assume that clients can receive data only from a single channel.

In this paper, we propose a scheduling scheme to reduce the waiting time of clients under the condition that clients can play data without interruption from beginning to end. Our scheme divides data into several segments and broadcasts them via a single channel. By broadcasting precedent segments frequently, the waiting time is reduced. In our scheme, the number of segments and the broadcast schedule affect the waiting time. We present a schedule that provides the minimum average waiting time in schedules where data is divided into two segments of equal size. In the case where it is necessary to divide data into more than two segments, we propose a heuristic scheduling method.

The remainder of this paper is organized as follows. Section II explains our basic idea. We show how to reduce the waiting time by broadcasting precedent segments frequently with a single channel. Our assumed system environment and proposed scheme are explained in Section III. In Section IV, we analyze schedules where data is divided into two segments of equal size and present a schedule that provides the minimum average waiting time. In the case of dividing data into more than two segments, we propose a heuristic scheduling method in Section V. Our scheme is discussed in Section VI. In Section VII, we make remarks on related work. Finally, we conclude the paper in Section VIII.

II. BASIC IDEA

In this section, first we calculate the average waiting time in the case of broadcasting data repetitively without division. Next, we explain how broadcasting precedent segments frequently with a single channel reduces the waiting time.

Suppose that it takes 180 seconds to broadcast a continuous media data for 30 minutes. Note that this case is suitable for broadcasting an MPEG2 (5 Mbps)-encoded video for 30 minutes on 50 Mbps satellite broadcast systems. In the case of broadcasting the data repetitively without division as shown in Fig. 1 (a simple repetition scheme), a client which demands the data should wait for 90 seconds on average to start playing the data (max. 180 seconds, min. 0 seconds). Note that the data is streaming media data, i.e., the client can start playing the data as soon as it starts receiving it.

Next, we explain our basic idea. Suppose the case of dividing the data into two segments of equal size. The playing time of each segment is 15 minutes and it takes 90 seconds to broadcast it. Let S_1 be the first segment and S_2 the last segment. The server

T. Yoshihisa is with the Academic Center for Computing and Media Studies, Kyoto University, Kyoto 606-8501, Japan (e-mail: yoshihisa@media.kyoto-u.ac.jp).

M. Tsukamoto is with the Department of Electrical and Electronics Eng., Faculty of Engineering, Kobe University, Kobe, Japan (e-mail: tuka@kobe-u.ac.jp).

S. Nishio is with the Department of Multimedia Eng., Grad. School of Information Science and Technology, Osaka University, Osaka, Japan (e-mail: nishio@ist.osaka-u.ac.jp).
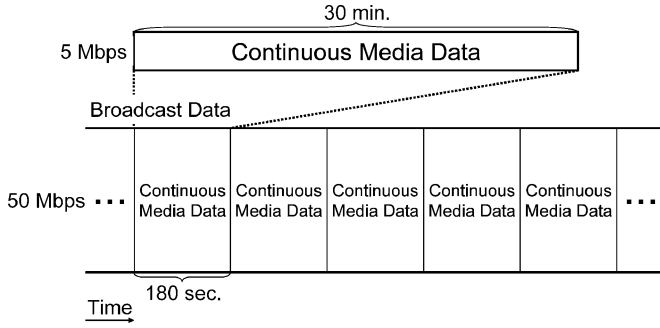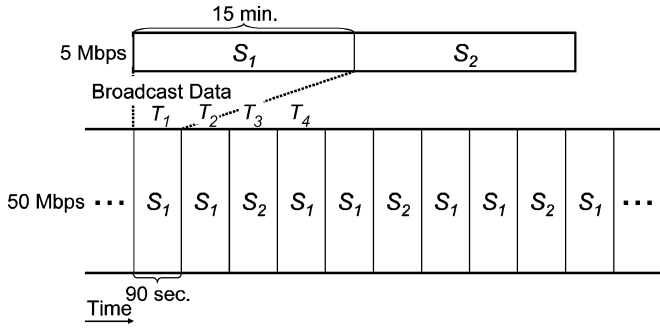
Fig. 1.   A simple repetition scheme.

Fig. 2.   An example for explaining our basic idea.

broadcasts these segments in the order of $S_1 S_1 S_2$ as shown in Fig. 2, and in this case, the broadcast cycle is 270 seconds. Let $T_i (i = 1, 2, \cdots)$ be the time slot. The broadcast cycle starts from T_1. That is, S_1 is broadcast from T_1 and T_2, and S_2 is broadcast from T_3. The average waiting time of clients is calculated as follows.

- In the case of demanding the data in T_1:

  Since S_1 is broadcast from T_2, the client can start playing the data from the beginning of T_2. The time to finish playing S_1 is after T_3, because the playing time of S_1 is 15 minutes. Hence, the client can store S_2 broadcast from T_3 while it plays S_1. After the client finishes playing S_1, it can play the data without interruption by playing stored S_2 immediately. In this case, the average waiting time of clients is $90/2 = 45$ seconds.

- In the case of demanding the data in T_2:

  Since S_2 is broadcast from T_3, the client can store S_2 before it starts playing the data. In this case, the average waiting time is 135 seconds.

- In the case of demanding the data in T_3:

  In this case, the average waiting time is 45 seconds.

As a result, the average waiting time becomes $(45 + 135 + 45)/3 = 75$ seconds. This is 17% shorter than that of the simple repetition scheme.

Moreover, when the server broadcasts segments in the order of $S_1 S_1 S_1 S_1 S_1 S_1 S_1 S_1 S_1 S_1 S_2$, it is similarly shown that the average waiting time falls to 53 seconds, 42% shorter than that of the simple repetition method. This is the minimum average waiting time in this example.

In this way, by broadcasting precedent segments frequently with a single channel, clients' waiting time can be reduced.

III. Our Approach

Let D be the playing time of a continuous media data and D' the time required to broadcast the data. We define the *playback ratio a* by the following equation:

$$a = \frac{D}{D'}. \tag{1}$$

For example, an MPEG2 (5 Mbps)-encoded video lasting 30 minutes is 1,125 Mbytes. If the data is broadcast via a 50 Mbps satellite broadcast system, it takes 180 seconds to broadcast it. In this case, the playback ratio is $1800/180 = 10$. Moreover, MPEG1-encoded video is usually assumed to be 1 Mbps. If the data is broadcast via a 20 Mbps DVB system, the playback ratio is 20 [10]. Not only video, but also audio can be applied to this discussion. An MP3 (Moving Picture Experts Group Audio Layer 3)-encoded audio is usually 128 Kbps. If the data is broadcast via a 1 Mbps wireless LAN system, the playback ratio is 7.8. Note that the playback ratio is derived while encoding the data.

In this paper, we assume that a is constant and known. However, if a varies, our proposed scheme can be applied by using the maximum value of the playback ratio as a.

A. Assumed System Environment

- The client can receive data only from a single channel.
- The client can play data without interruption from beginning to end.
- The client can start playing data as soon as it starts receiving it.
- The client has its own buffer. The client can receive and store new data while playing data. The buffer has enough capacity to store the data.
- The client starts receiving broadcast data after it demands playing the data.
- Since a segment includes additional information such as the segment number in front of it, the client cannot receive segments midstream.
- The server does not update data.
- a is larger than 1.

We assume guarantee-type networks that guarantee steady bandwidth and reliable data receiving. The Internet is assumed to be a guarantee-type network by applying a bandwidth reservation protocol such as RSVP (Resource ReSerVation Protocol).

In the case where the server broadcasts some contents, it broadcasts them one by one. Since clients do not receive a data broadcast prior to demanding the data, they do not always have to observe broadcast data.

B. Proposed Scheme

We propose a scheduling scheme called the *division-based broadcasting* scheme. In this scheme, a continuous media data is divided into N segments S_i $(i = 1, \cdots, N)$ of equal size. The necessary time to broadcast a segment is D'/N (Fig. 3). The server broadcasts c segments in a broadcast cycle.

The problem is, how we produce a broadcast schedule that effectively reduces the waiting time? For example, suppose the
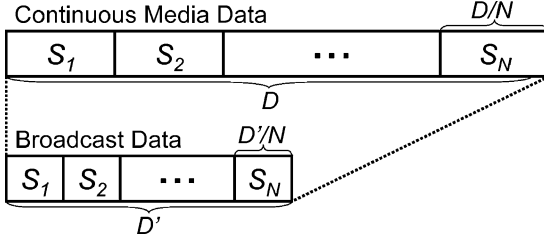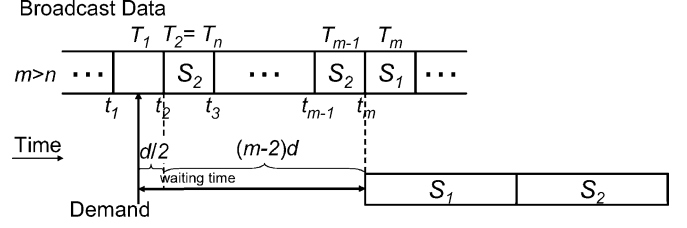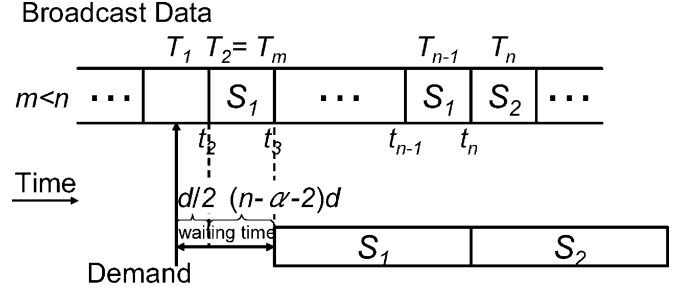
Fig. 3. A division of a continuous media data.

Fig. 4. An example of the value of m and the value of n ($n \leq a + m$).

Fig. 5. An example of the value of m and the value of n ($n > a + m$).

TABLE I
BROADCAST SCHEDULES THAT INCLUDE FOUR OR FIVE SEGMENTS IN ONE BROADCAST CYCLE ($N = 2$, $D = 30$ minutes, $a = 10$)

$c = 4$		$c = 5$	
Broadcast Schedule	Average Waiting Time (seconds)	Broadcast Schedule	Average Waiting Time (seconds)
$S_1 S_1 S_1 S_2$	67.5	$S_1 S_1 S_1 S_1 S_2$	63
$S_1 S_1 S_2 S_2$	112.5	$S_1 S_1 S_1 S_2 S_2$	99
$S_1 S_2 S_2 S_2$	180	$S_1 S_1 S_2 S_1 S_2$	81
		$S_1 S_1 S_2 S_2 S_2$	153
		$S_1 S_2 S_1 S_2 S_2$	117
		$S_1 S_2 S_2 S_2 S_2$	225

TABLE II
BROADCAST SCHEDULES THAT INCLUDE FOUR SEGMENTS IN ONE BROADCAST CYCLE ($N = 3$, $D = 30$ minutes, $a = 10$)

broadcast schedule	the average waiting time (seconds)
$S_1 S_1 S_2 S_3$	75
$S_1 S_1 S_3 S_2$	75
$S_1 S_2 S_1 S_3$	60
$S_1 S_2 S_2 S_3$	120
$S_1 S_2 S_3 S_2$	120
$S_1 S_3 S_2 S_2$	120
$S_1 S_2 S_3 S_3$	120
$S_1 S_3 S_3 S_2$	120
$S_1 S_3 S_2 S_3$	120

case where it takes 180 seconds to broadcast a video for 30 minutes. Table I shows broadcast schedules that include four or five ($c = 4, 5$) *schedule segments* in one broadcast cycle when the data is divided into two segments. Since $S_1 S_2 S_1 S_2$ is regarded as $c = 2$, this broadcast schedule is not shown in the table. The minimum average waiting time in Table I is 63 minutes for the broadcast schedule $S_1 S_1 S_1 S_1 S_2$. Table II shows broadcast schedules that include four schedule segments in a broadcast cycle when the data is divided into three segments. In this case, the minimum average waiting time is 60 seconds for the broadcast schedule $S_1 S_2 S_1 S_3$. Thus, the average waiting time is calculated from given a, N, and c. However, since these values are given voluntarily and the number of broadcast schedules is infinity, finding a broadcast schedule that provides the minimum average waiting time in all schedules is not realistic. Hence, in Section IV, we analyze schedules in the case of dividing data into two segments of equal size and present a schedule that provides the minimum average waiting time. In the case of dividing data into more than two segments, since the large number of broadcast schedules makes analysis difficult, we propose a heuristic scheduling method in Section V.

IV. DIVIDING DATA INTO TWO SEGMENTS

In this section, we analyze schedules in the case of dividing data into two segments and present a schedule that provides the minimum average waiting time.

A. Classification of the Average Waiting Time

To find a broadcast schedule that provides the minimum average waiting time, we classify equations. Let d be the necessary time to broadcast a segment. Since we assume that the additional information data size of segments is small, $d = D'/2$. We call the time in which a segment is broadcast a time slot $T_i (i = 1, 2, \cdots)$. The start time of T_i is denoted by t_i. Let T_1 be the time slot in which the client demands the data.

The timing for a client to receive S_1 or S_2 after the demand is essential for calculating the average waiting time. Therefore, we term the time slot in which the first S_1 is broadcast after t_1 as T_m, and the time slot in which the first S_2 is broadcast as T_n. By comparing m with n, the situation can be divided into two cases, as shown in Figs. 4 and 5. Since clients cannot receive the segment broadcast from T_1, the segment broadcast from T_1 is not represented in Fig. 4. m and n must be different numbers. When the value of either m or n is 2, the other value must be more than 2.

Suppose that the client starts playing the data as soon as it receives S_1. Let τ be the time at which the client finishes playing S_1. τ is expressed by $t_m + ad$. In the case of $n \leq a + m$, since $t_n \leq \tau$, the client can receive S_2 while it plays S_1. In the case of $n > m + a$, since $t_n > \tau$, the client cannot receive S_2 while it plays S_1. Below, we show the average waiting time of each case.

- Where $n \leq a + m$:
  After the client finishes playing S_1, it can play S_2 continuously. The average waiting time of this case $W^{\mathrm{I}}(m)$ is given by the following equation:

$$W^{\mathrm{I}}(m) = \frac{d}{2} + (m - 2)d. \tag{2}$$

TABLE III
EXAMPLES OF SEGMENT SETS

The broadcast schedule	Segment sets
$S_1 S_2$	$U_1 = \{S_1 S_2\}$
$S_1 S_1 S_2$	$U_1 = \{S_1 S_1 S_2\}$
$S_1 S_2 S_2$	$U_1 = \{S_1 S_2 S_2\}$
$S_1 S_1 S_1 S_2$	$U_1 = \{S_1 S_1 S_1 S_2\}$
$S_1 S_1 S_1 S_2 S_2 S_2$	$U_1 = \{S_1 S_1 S_1 S_2 S_2 S_2\}$
$S_1 S_2 S_2 S_1 S_1 S_2$	$U_1 = \{S_1 S_2\}, U_2 = \{S_1 S_1 S_2\}$
$S_1 S_2 S_2 S_2 S_1 S_2$	$U_1 = \{S_1 S_2 S_2\}, U_2 = \{S_1 S_2\}$
$S_1 S_2 S_1 S_1 S_1 S_2$	$U_1 = \{S_1 S_2\}, U_2 = \{S_1 S_1 S_1 S_2\}$
$S_1 S_2 S_1 S_2 S_2 S_1$	$U_1 = \{S_1 S_2 S_2\}, U_2 = \{S_1 S_1 S_2\}$
	(By considering the periodicity)
$S_1 S_2 S_1 S_2 S_1 S_1 S_2$	$U_1 = \{S_1 S_2\}, U_2 = \{S_1 S_2\}$
	$U_3 = \{S_1 S_1 S_2\}$

Fig. 6. An example of segment sets.

The meanings of terms are shown in Fig. 4. Since $W^{\mathrm{I}}(M)$ are the average waiting time, we showed the case which a client demands the data at the center of a time slot in the figure.

- Where $n > m + a$:

In this case, if the client starts playing the data from t_m, it cannot play S_2 continuously after S_1. Hence, the client has to wait before it starts playing S_1 to play the data continuously. To calculate the average waiting time, we suppose that the client starts playing the data from time t_w. Let τ_w be the time at which the client finishes playing S_1. That is, $\tau_w = t_w + ad$. Therefore, S_2 is broadcast until or from τ_w, the client can play S_2 after S_1. Hence, $t_n \le \tau_w$ must be established. That is, $w \ge n - a$. In this case, when $w = n - a$, the average waiting time is the minimum. The average waiting time of this case $W^{\mathrm{II}}(n)$ is given by the following equation:

$$W^{\mathrm{II}}(n) = \frac{d}{2} + (n - a - 2)d. \qquad (3)$$

The meanings of terms are shown in Fig. 5.

Here, we consider the order of the above average waiting times. Since $\alpha \le a < \alpha + 1$ (α denotes the integral part of a, i.e., $\alpha = \lfloor a \rfloor$), the following inequality is derived.

$$\begin{aligned} W^{\mathrm{I}}(2) &< W^{\mathrm{II}}(\alpha + 3) \le W^{\mathrm{I}}(3) \\ &< W^{\mathrm{II}}(\alpha + 4) \le W^{\mathrm{I}}(4) \\ &< W^{\mathrm{II}}(\alpha + 5) \le W^{\mathrm{I}}(5) < \cdots \end{aligned} \qquad (4)$$

By using the order of $W^{\mathrm{I}}(M)$ and $W^{\mathrm{II}}(N)$ in (4), we will find the optimal schedule in Section IV-C.

B. Segment Sets

In the case of dividing the data into two segments, the broadcast schedule consists of S_1 and S_2. Hence, we divide the broadcast schedule into segment sets at the point between S_2 and S_1, i.e., let $U_i = S_1 \cdots S_1 S_2 \cdots S_2$ ($i = 1, \cdots, l$) be segment sets. The broadcast schedule is represented by $U_1 \cdots U_l$. T_{p_i} indicates the time slot in which the first S_1 of U_i is broadcast, T_{q_i} indicates the time slot in which the first S_2 of U_i is broadcast, and T_{r_i} indicates the time slot in which the last S_2 of U_i is broadcast. Then, $r_i + 1 = p_{i+1}$ holds for all $i = 1, \cdots, l - 1$. Examples of segment sets are shown in Table III. As the table demonstrates, all broadcast schedules can be divided into segment sets. Since
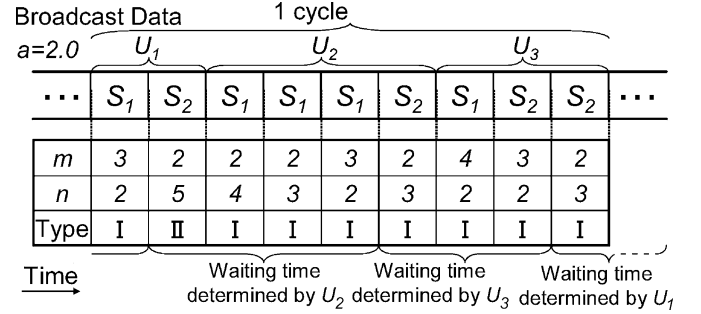
the first segment of a segment set is S_1, and the last segment of the segment set is S_2, each combination dividing a broadcast schedule into segment sets is unique.

As for segment sets, the next theorem is established.

Theorem 1: A segment set U_i defines the waiting time of clients that demand the data in $T_{p_i - 1}, \cdots, T_{r_i - 1}$.

For example, a client that demands the data in $T_{p_i - 1}$ can receive S_1 from T_{p_i}, and receive S_2 from T_{q_i}. Hence, the client's waiting time is decided by the value of p_i and q_i of U_i. Moreover, a client that demands the data in $T_{r_i - 1}$ can receive S_2 from T_{r_i} while it waits for S_1. In this case, since the first segment of the next segment set is S_1, the client can receive S_1 from $T_{p_i + 1}$. From the reasons described above, segment sets included in a broadcast schedule are independent of the waiting time. Hence, the average waiting time is calculated by the average of the average waiting time of each segment set.

That is, we can determine the waiting time of a request by looking ahead and seeing which U_i is present. This is simple and can be implemented for actual systems. For example, by calculating the waiting time for each time slot beforehand, the server can get the waiting time needed for clients to play the data without interruption. The server adds the information about the necessary waiting time in front of the segment included in each time slot and broadcasts them. Since clients can get the necessary waiting time as soon as they start receiving the segment, they do not have to look ahead and see which U_i is there. This can be implemented for actual systems and is realistic.

Fig. 6 shows how the average waiting time is calculated. In this case, the broadcast schedule is divided into 3 segment sets. Since the second S_1 is broadcast after 3 time slots from the first time slot, the value of m of the first time slot is 3. Also, since the first S_2 is broadcast after 2 time slots from the first time slot, the value of n of the first time slot is 2. When $a = 2$, since $n \le a + m$ is established, the type is I (Section IV-A). Therefore, the average waiting time of a client which demands the data in the first time slot is $3d/2$. Moreover, that of a client which demands the data in the last time slot is $d/2$. As a result, in the case of broadcasting U_1 as a broadcast schedule, the average waiting time is d. In the case of the second time slot, since $m = 2$ and $n = 5$, $n > a + m$ is established. Accordingly, the type is II and the average waiting time is $3d/2$. In the same way as U_1, the average waiting time of U_2 is d. The average waiting time of U_3 is $3d/2$. To calculate the average waiting time of the schedule, we have to consider the number of segments included in each segment set. Since the number of segments included in U_1, U_2,

TABLE IV
THE AVERAGE WAITING TIME FOR A SEGMENT SET

	U	$W^{\mathrm{I}}(2)$	$<$	$W^{\mathrm{II}}(\alpha+3)$	$\leq$	$W^{\mathrm{I}}(3)$	$<$	$W^{\mathrm{II}}(\alpha+4)$	$\leq$	$W^{\mathrm{I}}(4)$	$<$	$\cdots$
	S_1S_2	1		0		1		0		0		
	$S_1S_2S_2$	1		0		1		0		1		
	$S_1S_1S_2$	2		0		1		0		0		
	$S_1S_1S_1S_2$	3		0		1		0		0		
(a)	$\overbrace{S_1\cdots S_1}^{\alpha} S_2$	α		0		1		0		0		
(b)	$\overbrace{S_1\cdots S_1}^{\alpha} S_1 S_2$	α		1		1		0		0		
	$\overbrace{S_1\cdots S_1}^{\alpha} S_1 S_1 S_2$	α		1		1		1		0		

U_3 is 2, 4, 3, respectively, the average waiting time becomes $(2 \times d + 4 \times d + 3 \times 3d/2)/9 = 7d/6$.

Let U_{min} be the segment set that gives the minimum average waiting time of all segment sets. Here, the following theorem is established.

Theorem 2: A broadcast schedule that repeats U_{min} provides the minimum average waiting time.

To prove the theorem, suppose that a broadcast schedule consists of the segment sets $U_1 \cdots U_l$. When all of $U_i(i = 1, \cdots, l)$ are not be the same as U_{min}, the average waiting time of U_i is longer than U_{min}. Therefore, if one of these segment sets is changed to U_{min}, the average waiting time decreases. By doing this over the same arguments, we can see that a broadcast schedule that provides the minimum average waiting time is the schedule that consists of repetitions of U_{min}. Hence, by finding U_{min}, we can find a broadcast schedule that provides the minimum average waiting time in the case where the data is divided into two segments of equal size.

C. Detection of u_{min}

To detect U_{min}, we can easily see the number of $W^{\mathrm{I}}(m)$ and $W^{\mathrm{II}}(n)$ in a segment set U. The result is shown in Table IV. Since U is a segment set, the form of U is $S_1 \cdots S_1 S_2 \cdots S_2$. In Table IV, equations of average waiting times are lined up in the first line. According to (4), $W^{\mathrm{I}}(2)$ is the shortest of all. A equation gives longer average waiting time along with right. The number of equation which gives the average waiting time of each time slot is counted in the right side of schedules shown in the table. Suppose that U is S_1S_2 as shown in the first line of Table IV. This schedule is same as the simple method. In this case, the number of $W^{\mathrm{I}}(2)$ is one and $W^{\mathrm{I}}(3)$ is also one. First, by adding S_2 to U, U becomes the second line, and the number of $W^{\mathrm{I}}(4)$ becomes one. Since $W^{\mathrm{I}}(3) < W^{\mathrm{I}}(4)$, the average waiting time of U increases. By increasing S_2, the number of $W^{\mathrm{I}}(5)$ becomes one. Hence, the more the number of S_2 increases, the longer the waiting time becomes. Accordingly, when U includes only one S_2, the average waiting time is shorter. Next, by adding S_1 to U, U becomes the third line. The number of $W^{\mathrm{I}}(2)$ becomes two. Since the average waiting time by $W^{\mathrm{I}}(2)$ is the minimum, the average waiting time decreases. As S_1 increases, the average waiting time is reduced while the number of S_1 is less than α. In the case where the number of S_1 equals α, U becomes (a) in Table IV. By adding S_1 to U, the number of $W^{\mathrm{II}}(\alpha+3)$ becomes one, and U becomes (b). If the average waiting time of (a) is less than $W^{\mathrm{II}}(\alpha+3)$, the average

waiting time of (a) is less than the average waiting time of (b). Otherwise, the average waiting time of (a) is greater than the average waiting time of (b). By adding S_1 to (b), the number of $W^{\mathrm{II}}(\alpha+4)$ becomes one. Since $W^{\mathrm{II}}(\alpha+3) < W^{\mathrm{II}}(\alpha+4)$, the average waiting time increases. Hence, a schedule that provides the minimum waiting time (i.e., U_{min}) can be (a) or (b) in Table IV. In the case of (a), the average waiting time W_a is given by the following equation:

$$W_a = \frac{\alpha+3}{2\alpha+2}d. \tag{5}$$

In the case of (b), the average waiting time W_b is given by the following equation:

$$W_b = \frac{3\alpha+6-2a}{2\alpha+4}d. \tag{6}$$

From (5) and (6), we compare the average waiting time of (a) with that of (b).

$$\begin{aligned} W_a - W_b &= \frac{\alpha+3}{2\alpha+2}d - \frac{3\alpha+6-2a}{2\alpha+4}d \\ &= \frac{(\alpha+1)(a-\alpha)-\alpha}{(\alpha+1)(\alpha+2)}d \end{aligned} \tag{7}$$

Here, let A be the numerator of (7).

$A =$ (the minimum integer larger than a)
$\times$(the decimal part of a) $-$ (the integer part of a)

A broadcast schedule that provides the minimum average waiting time is given as follows.

- In the case of $A < 0$:

$$\overbrace{S_1 \cdots S_1}^{\alpha} S_2 \tag{8}$$

  and the average waiting time is obtained by (5).
- In the case of $A > 0$:

$$\overbrace{S_1 S_1 \cdots S_1}^{\alpha+1} S_2 \tag{9}$$

  and the average waiting time is obtained by (6).
- In the case of $A = 0$:
  the above two schedules have the same average waiting time.

TABLE V
COMPUTATION TIMES TO FIND THE OPTIMAL BROADCAST
SCHEDULE (MINUTES)

N	$c = 7$	$c = 8$	$c = 9$	$c = 10$	$c = 11$
3	1	2	8	32	114
4	2	17	93	531	2817
5	6	56	503	3691	25753

A broadcast schedule that provides the minimum average waiting time in the case of dividing data into two segments in equal size is found above.

V. DIVIDING DATA INTO MORE THAN TWO SEGMENTS

Dividing data into more than two segments makes it difficult to analyze schedules. A solution is to find the optimal broadcast schedule that provides the minimum average waiting time in all broadcast schedules through computer programming. Hence, we produced a computer program which finds the optimal schedule that includes c segments in a broadcast cycle. This program enables us to find the optimal broadcast schedule when $a = 10$ and $c = 7, \cdots, 11$. The computation time is shown in Table V (CPU: Pentium III 1.2 GHz, Memory: 512 MBytes). Since the computation time is $O(N^c)$, the computation time for finding the optimal schedule of $c = 24$ is estimated at 3,000 years. Clearly then, finding the optimal broadcast schedule with this program is not realistic. Moreover, since we can choose arbitrary N and c, the combinations of segments are infinity. Accordingly, we cannot find the optimal solution for all N and c. For above reasons, finding the optimal solution is impossible. Therefore, in this paper, we propose a heuristic scheduling method.

A. Alternative Broadcasting Method

Our proposed method, the *alternative broadcasting* method for single data, is based on the division-based broadcasting scheme. As a result of finding the schedule which gives the minimum average waiting time for values that require relative low computation power, we got the tendency that the waiting time is effectively reduced when the broadcast schedule includes precedent segments frequently. Hence, in the alternative scheme, the server divides a continuous media data into several segments of equal size and schedules segments in order to broadcast precedent segments frequently. segments of equal size, and by scheduling segments in order to broadcast precedent segments frequently.

In the alternative broadcasting method, the server broadcasts S_n in the following time slot. T_1 is a time slot from which a broadcast cycle starts:

- In the case of $n = 1$:
  S_1 is broadcast in $T_{2j-2}(j = 1, \cdots, N - 1)$.
- In the case of $n \neq 1$:
  S_n is broadcast in T_{2n-2}.

Examples of schedules produced by the alternative broadcasting method are shown in Fig. 7. As the figure demonstrates, the first segment and the other segments are broadcast alternatively in the alternative broadcasting method. The segment number next to S_1 becomes larger, and the number of segments broadcast in one broadcast cycle is $2(N - 1)$.
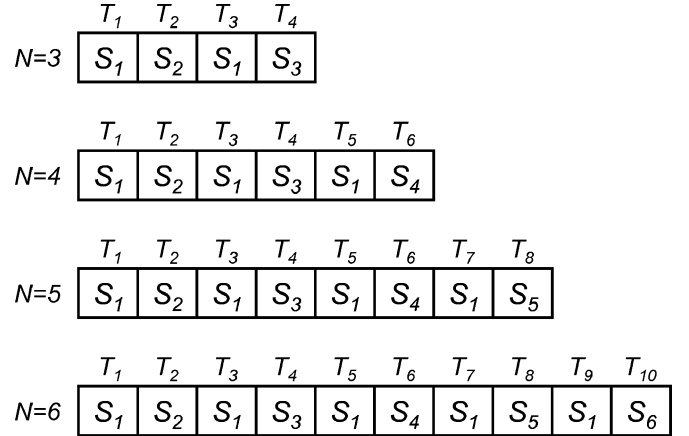

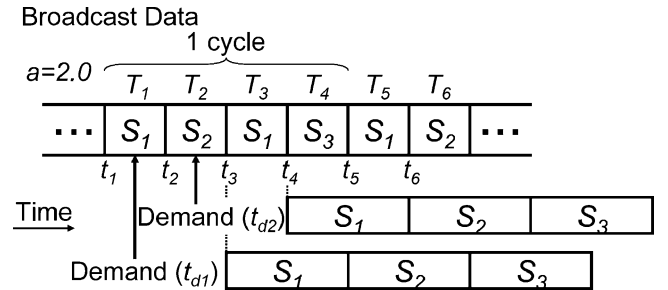Fig. 7. Examples of schedules produced by the alternative broadcasting method.

Fig. 8. An example of implementation of the alternative broadcasting method ($a = 2.0$, $N = 3$).

B. Scenario Description

After a client demands playing a continuous media data, it starts receiving the broadcast data. The client ceases to start playing the data until it can play the data without interruption from beginning to end. The client stores the data into its buffer while it plays the data. After the client finishes playing $S_n(n = 1, \cdots, N - 1)$, it immediately starts playing S_{n+1}, which is in the buffer or is being broadcast. For example, suppose the case of $a = 2.0$ and $N = 3$ (Fig. 8). When a client demands the data at t_{d1}, it receives the S_1 broadcast from T_3. Hence, the client can start playing the data from t_3. The client finishes playing S_1 at t_5. Since the client can store S_2 broadcast in T_2 until t_5, it can play S_2 continuously after it finishes playing S_1. Also, the client can store S_3 until the time to start playing S_3. In this way, the client can play the data continuously. In another case, when a client demands the data at t_{d2}, it receives the S_1 broadcast from T_3. However, the client has to start playing S_1 from t_4, because if the client starts playing S_1 from t_3, the time to finish playing S_1 is t_5. Since the client cannot receive S_2 until t_5, it has to wait before playing S_2. Hence, the client waits until t_4 to be able to play the data continuously. By starting to play the data from t_4, the client can store S_2 and S_3 until the time to start playing them. Hence, the client can play the data without interruption from beginning to end.

C. The Number of Segments and the Waiting Time

Fig. 9 shows the average waiting time by a computer simulation. Since the average waiting time is proportional to the playing time, the value of the average waiting time divided by
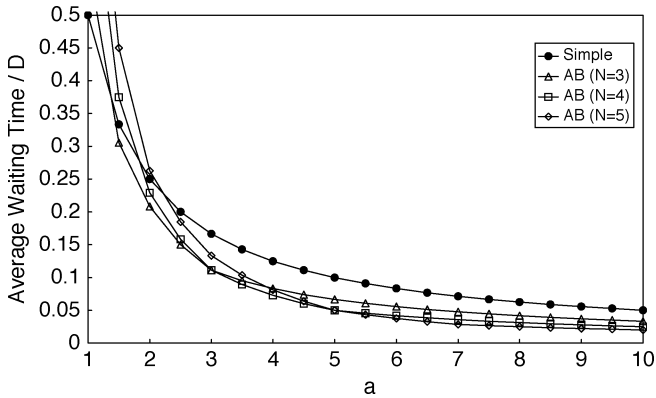
Fig. 9.   The number of segments and the waiting time ($N = 3, \cdots, 7$).

Fig. 10.   The average waiting time under our proposed schemes.

the playing time is shown on the vertical axis. The horizontal axis shows playback ratio a. "Simple" is the average waiting time under the simple repetition scheme, which I explained in Section II. "AB $(N = i)$" $(i = 3, \cdots, 7)$ is the average waiting time under the alternative broadcasting method.

Although the average waiting time under the alternative broadcasting method of $N = 3$ is the shortest at $1.5 \leq a \leq 3.0$, that of $N = 4$ is the shortest at $3.0 \leq a \leq 5.0$. Thus, the N that gives the shortest average waiting time depends on the playback ratio. Hence, in this paper, we propose some methods for determining N.

D. Methods for Determining N

We propose two methods for determining N.

- The *mechanism dominance* (MD) method:

  In this method, even if a client starts playing the data as soon as it receives S_1, it can play the data without interruption from beginning to end. Hence, the advantage of the MD method is that clients need not get information on the broadcast schedule and that the client can start playing the data as soon as it receives S_1. That is, cases such as demanding the data at t_{d2} in Fig. 8 do not occur. To satisfy the above condition, in the MD method, N is given by

$$\left\lfloor \frac{a-1}{2} + 2 \right\rfloor. \tag{10}$$

This equation is discussed in Section VI-C. The average waiting time under the MD method is sometimes longer than that under the other method below (Section VI-B).

- The *waiting-time dominance* (WD) method:

  The average waiting time under this method is sometimes shorter than that under the MD method. However, when a client starts playing S_1 as soon as it receives S_1, it cannot always satisfy the continuity condition. Therefore, the client has to wait for the data to start playing after it receives S_1 to play the data without interruption from beginning to end. Hence, the disadvantage of the WD method is that clients need to obtain information on this waiting time. Clients can get this information by adding it to the header of segments. In the WD method, N is given by

$$\left\lceil \frac{a+3}{2} \right\rceil. \tag{11}$$
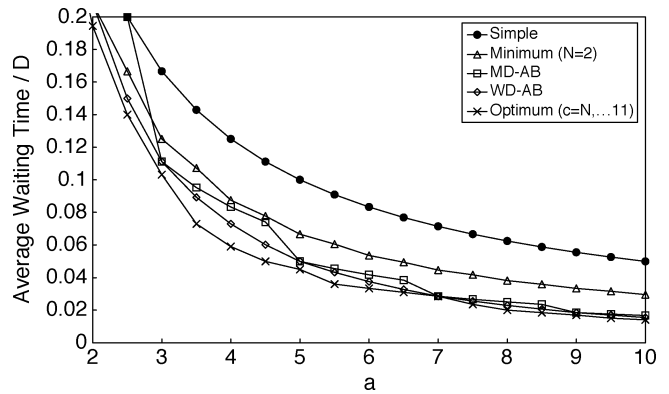
This is discussed in Section VI-D.

VI. DISCUSSION

A. Comparison

To demonstrate the effectiveness of our proposed scheme, we compare the average waiting time under the division-based broadcasting scheme with that under the simple repetition scheme. Fig. 10 shows the average waiting time. To make the figure clearly understandable, the ordinate range is different from that in Fig. 8. "Minimum $(N = 2)$" is the minimum average waiting time under the division-based broadcasting scheme of $N = 2$. These minimum average waiting times are discussed in Section IV. "MD-AB" is the average waiting time under the mechanism dominance determination method of the alternative broadcasting method. "WD-AB" is the average waiting time under the waiting-time dominance determination method of the alternative broadcasting method. "Optimum $(c = N, \cdots, 11)$" is the minimum average waiting time under schedules when c is changed from N to 11. N is the same value as WD-AB selects. This waiting time is calculated by the computer program explained in Section V.

From these figures, we can see that the alternative broadcasting scheme gives close waiting time to the optimum for values that require relative low computation.

For example, suppose that a 5 Mbps video for 30 minutes is broadcast via a 50 Mbps satellite broadcast system ($a = 10$). In this case, the average waiting time under the simple repetition scheme is $0.05 \times 30 = 1.5$ minutes. When dividing the data into two segments, the minimum average waiting time becomes $0.030 \times 30 = 0.9$ minutes. This is 42% shorter than that under the simple repetition scheme. Moreover, the average waiting time under the MD-AB method is $0.017 \times 30 = 0.51$ minutes, and that under the WD-AB method is $0.015 \times 30 = 0.45$ minutes. In the case of broadcasting using the WD-AB method, the average waiting time is 69% shorter than that under the simple repetition scheme.

B. The Mechanism or the Waiting-Time Dominance Determination Method

The WD-AB method determines N in order to further reduce the average waiting time. For example, the average waiting time under the WD-AB method in Fig. 10 is the same as the minimum average waiting time in Fig. 9. However, since a client has to wait after receiving S_1 to play the data continuously, it must obtain information on how long this waiting time will be.

In the MD-AB method, a client that starts playing S_1 as soon as it receives S_1 can always play the data without interruption from beginning to end. However, the average waiting time can be longer than that under the WD method. For example, in Fig. 10, the average waiting time is longer at $a \leq 2.5$, $3.5 \leq a \leq 4.5$, and so on. The MD method is used to simplify the mechanism, and the WD method is used to further reduce the waiting time.

We show the mathematical argument of each determination method below.

C. Argument for the Mechanism Dominance Method

The following theorem is established in the alternative broadcasting method,

Theorem 3: The condition for clients to play the data without interruption between S_n and S_{n+1} is equivalent to the following inequities.

- In the case of $n = 1$:

$$D'(2N - 3) \leq D. \tag{12}$$

- In the case of $n = 2, \cdots, N - 1$:

$$D' \leq D. \tag{13}$$

Proof is shown in Appendix I.

From Theorem 3, $N \leq (a - 1)/2 + 2$ must be established to play the data without interruption from beginning to end. Considering that a larger N further reduces the average waiting time, the equation becomes $N = \lfloor (a - 1)/2 + 2 \rfloor$. This is same as (10). Hence, in the MD method, a client that starts playing the data as soon as it receives S_1 can play the data continuously. In this case, the average waiting time is equal to the time for clients to receive S_1. Since the interval of S_1 is $2D'/N$, the average waiting time W is given by the following equation:

$$W = \frac{2D'}{2N} = \frac{D'}{N}. \tag{14}$$

D. Argument for the Waiting-Time Dominance Determination Method

It is supposed that $D'(2N - 3) < D_m \leq D'(2N' - 3)$ is established for all N when the average waiting time under N becomes longer than that under $N' = N + 1$. This is supposed from Fig. 9. Namely, when the number of segments is N, a client that starts playing S_1 as soon as it receives S_1 can play the data without interruption, but when the number of segments is N', it cannot. For example, in Fig. 9, the playback ratio at which the average waiting time under $N = 4$ becomes shorter than that of $N = 3$ is 9.6. This satisfies $7 < a \leq 13$. Accordingly, we can obtain the average waiting time in this bound. When the number of segments is N, $D'(2N - 3) \leq D$ is established. Hence, by Theorem 3, the average waiting time W_N is

$$W_N = \frac{D'}{N}. \tag{15}$$

Next, we show the average waiting time under N'. Here, the following theorem is established.

Theorem 4: In the case of $D'(2N - 2i + 1) \leq D \leq D'(2N - 2i + 3)(i = 2, \cdots, N - 1)$, the average waiting time W is

$$W = \frac{1}{2N(N - 1)} (D(-2i + 5)$$
$$- D'(2i^2 - 4i(N + 1) + 8N - 1)). \tag{16}$$

Proof is shown in Appendix 2. By Theorem 4, in the case of $D' + 2D'(N - 2) < D \leq D' + 2D'(N' - 2)$, the average waiting time $W_{N'}$ is:

$$W_{N'} = \frac{1}{2N'(N' - 1)} (-D + D'(4N' - 5)) \tag{17}$$

Compared with W_N, when the following equation is established, the average waiting time under $N + 1$ is shorter than that under N:

$$N \leq \frac{a + 3}{2}. \tag{18}$$

Considering that N is incremented when (18) is established, N is given by $N = \lceil (a + 3)/2 \rceil$ to reduce the average waiting time effectively. This is same as (11). Hence, a shorter average waiting time than the MD method is given by (11).

VII. RELATED WORK

Various strategies have been proposed to reduce the waiting time under the condition that clients can play data without interruption from beginning to end. In pyramid broadcasting (PB) [16], a continuous media data is divided into N segments, and N channels are used. Let D_i be the time to broadcast the ith segment. The data is divided so that D_i is $D_1 \times \alpha(i - 1)$, where, α is 1 or more. The server broadcasts the ith segment via the ith channel. Since the interval of broadcasting precedent segments is shorter than the other, the first segment is broadcasted frequently. Hence, clients' waiting time is reduced, because they have many opportunities to receive the first segment. In this scheme, however, the greater the value of N becomes, the larger the data size of the Nth segment is. Therefore, a larger buffer capacity is required.

Skyscraper broadcasting (SB) [6] considers this problem. Segments with a longer playing time than $D_1 \times W$ are divided so that its playing time becomes $D_1 \times W$. By modifying W, the buffer requirement and the average waiting time are voluntarily given.

Dynamic skyscraper broadcasting [3] dynamically changes broadcast data based on the SB scheme. This improves the performance of SB when clients infrequently request data.

The permutation-based pyramid broadcasting scheme [1] divides N channels into p sub-channels. A server broadcasts the ith segment using p sub channels. The beginning of the broadcast cycle of each sub-channel is shifted. As a result, the storage requirements, the disk transfer rate, and the waiting time are improved compared to PB. This scheme proposes time-division multiplexing, which employs a single channel. However, the paper does not detail a technique for reducing the waiting time of clients with a single channel.

To further reduce the waiting time of clients, harmonic broadcasting (HB) [8] has been proposed. In this scheme, the ith segment is divided into i sub-segments. A server broadcasts i sub-segments of the ith segment of each data via the ith channel. The bandwidth of the ith channel is $1/i$ of the first channel, while the bit rate of the first channel is the same as the consumption rate. Hence, we can say that the playback ratio of the ith channel is $1/i$. However this work did not cover a method of reducing the waiting time for clients with a single channel. This scheme always requires a buffer on the client side.

In fast broadcasting (FB) [9], clients that do not have a buffer can play data by waiting longer. Regarding FB, methods for channel transition have been studied [15].

These strategies reduce the waiting time by broadcasting continuous media data with multiple channels. Because of this, clients are forced to receive data from multiple channels concurrently, and the server has to control multiple channels. Hence, their mechanisms are more complicated than broadcasting data via a single channel. Regarding constructing an actual system, neither wireless LAN nor satellite broadcast systems can receive data from multiple channels concurrently.

VIII. CONCLUSION AND FUTURE WORK

In this paper, we proposed the division-based broadcasting scheme to reduce the waiting time of clients with a single channel. In our approach, the number of segments N and the broadcast schedule affect the waiting time.

In the case of dividing data into two segments of equal size, we presented a schedule that provides the minimum average waiting time. The number of the first segment, S_1, included in the broadcast schedule is decided by the playback ratio (the data playing time divided by the time required to broadcast the data), and the last segment S_2 is included only once. In the case of broadcasting a 5 Mbps video for 30 minutes via a 50 Mbps satellite broadcast system, the minimum average waiting time under $N = 2$ is 53 seconds. This is 42% shorter than that in the case of broadcasting the data without division. Hence, dividing the data into only two segments is sufficiently realistic.

For the case where it is necessary to divide data into more than two segments, we propose the alternative broadcasting method. In the alternative broadcasting method, the N that gives the shortest average waiting time depends on the playback ratio. Hence, we proposed two methods for determining N. For example, in the aforesaid case, the average waiting time under the mechanism dominance determination method of alternative broadcasting method (MD-AB) is 30 seconds, and that under the waiting-time dominance determination method (WD-AB) is 28 seconds. In the case of broadcasting using the WD-AB method, the average waiting time is 69% shorter than that under the simple repetition scheme.

In the future, we plan to study other heuristics and delivering multiple data with a single channel.

APPENDIX

A. Proof of Theorem 3

Let l_n be the time from starting broadcasting $S_n (n = 1, \cdots, N - 1)$ to starting broadcasting S_{n+1} (Fig. 11). l_n are given by the following equations:

- In the case of $n = 1$:

  Let $l_1(i)$ be the time from starting broadcasting S_1 from $T_{2i-1} (i = 1, \cdots, N - 1)$ to starting broadcasting S_2:

$$l_1(i) = \begin{cases} \frac{D'}{N} & (i = 1) \\ \frac{D'}{N} + \frac{2D'(N-i)}{N} & (i = 2, \cdots, N - 1) \end{cases} \quad (19)$$

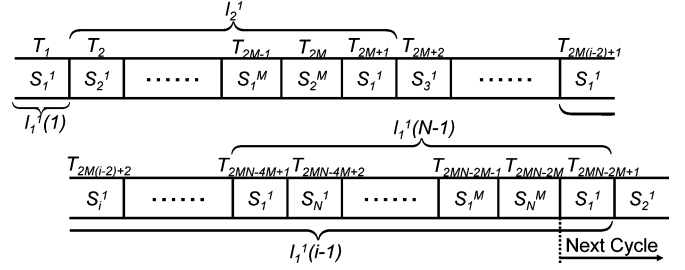- In the case of $n = 2, \cdots, N - 1$:

$$l_N = \frac{D'}{N} \quad (20)$$

Fig. 11. Examples of l_i.

To play the data without interruption between S_n and S_{n+1}, $l_n \leq D/N$ must be established. The maximum of l_n is $l_1(2)$. Hence, when $l_1(2) \leq D/N$, the client can play the data without interruption from beginning to end. □

B. Proof of Theorem 4

In the case of $D'(2N - 2i + 1) \leq D \leq D'(2N - 2i + 3)(i = 2, \cdots, N - 1)$, (13) is always established. Hence, by Theorem 3, if the client can play S_2 after S_1 continuously, it can play the data without interruption from beginning to end. In this case, the average waiting time is calculated by the following three cases:

- In the case of demanding data in T_1:
  Since a client can store S_2 broadcast from T_2, the client that starts playing S_1 as soon as it receives S_1 from T_3 can play the data without interruption from beginning to end. The average waiting time is $D'/(2N)$.
- In the case of demanding data in from T_2 to T_{2i-4}:
  A client that starts playing S_1 as soon as it receives S_1 cannot play S_2 continuously. Hence the client has to wait. In this case, the average waiting time is $-D/N + (4N - 2i + 1)D'/(2N)$.
- In the case of demanding the data in from T_{2i-3} to T_{2N-2}:
  A client that starts playing S_1 as soon as it receives S_1 can play the data continuously. The average waiting time is D'/N.

Accordingly, the average waiting time W is:

$$W = \frac{1}{2N(N-1)} \left(D(-2i + 5) - D' \right.$$
$$\left. \times \left(2i^2 - 4i(N + 1) + 8N - 1 \right) \right) \quad (21)$$

□

REFERENCES

[1] C. C. Aggarwal, J. L. Wolf, and P. S. Yu, "A permutation-based pyramid broadcasting scheme for video-on-demand system," in *Proc. of the IEEE Int. Conf. on Multimedia Computing and Systems (ICMCS '96)*, June 1996, pp. 118–126.

[2] A. Dan, D. Sitaram, and P. Shahabuddin, "Scheduling policies for an on-demand video server with batching," in *Proc. of the ACM Int. Multimedia Conf. (Multimedia '94)*, Oct. 1994, pp. 15–23.

[3] D. L. Eager and M. K. Vernon, "Dynamic skyscraper broadcasts for video-on-demand," in *Proc. of the 4th Int. Workshop on Multimedia Information Systems (MIS '98)*, Sep. 1998, pp. 18–32.

[4] G. B. Horn and J. E. Rasmussen, "A scalable and reliable paradigm for media on demand," *IEEE Computer*, vol. 34, no. 9, pp. 40–45, Sep. 2001.

[5] C.-H. Hsu, G. Lee, and A. L. P. Chen, "A near optimal algorithm for generating broadcast programs on multiple channels," in *Proc. of the Int. Conf. on Information and Knowledge Management (CIKM '01)*, Nov. 2001, pp. 303–309.

[6] K. A. Hua and S. Sheu, "Skyscraper broadcasting: a new broadcasting scheme for metropolitan video-on-demand systems," in *Proc. of the ACM SIGCOMM '97*, Sep. 1997, pp. 89–100.

[7] R. Janakiraman and M. Waldvogel, "Fuzzycast: efficient video-on-demand over multicast," in *Proc. of the IEEE INFOCOM '02*, June 2002, pp. 920–929.

[8] L.-S. Juhn and L. M. Tseng, "Harmonic broadcasting for video-on-demand service," *IEEE Trans. on Broadcasting*, vol. 43, no. 3, pp. 268–271, Sep. 1997.

[9] ——, "Fast data broadcasting and receiving scheme for popular video service," *IEEE Trans. on Broadcasting*, vol. 44, no. 1, pp. 100–105, Mar. 1998.

[10] Moving Picture Experts Group, MPEG Home Page, [Online] Available: http://www.chiariglione.org/mpeg/.

[11] J.-F. Paris, "An interactive broadcasting protocol for video-on-demand," in *Proc. of the IEEE Int. Performance, Computing, and Communications Conference (IPCCC '01)*, Apr. 2001, pp. 347–353.

[12] J.-F. Paris, S. W. Carter, and D. D. E. Long, "Efficient broadcasting protocols for video on demand," in *Proc. of the Int. Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS '98)*, July 1998, pp. 127–132.

[13] ——, "A hybrid broadcasting protocol for video on demand," in *Proc. of the SPIE Multimedia Computing and Networking Conf. (MMCN '99)*, Jan. 1999, pp. 317–326.

[14] J.-F. Paris, D. D. E. Long, and P. E. Mantey, "Zero-delay broadcasting protocols for video-on-demand," in *Proc. of the ACM Int. Multimedia Conf. (Multimedia '99)*, Nov. 1999, pp. 189–197.

[15] Y.-C. Tseng, C.-M. Hsieh, M.-H. Yang, W.-H. Liao, and J.-P. Sheu, "Data broadcasting and seamless channel transition for highly-demanded videos," in *Proc. of the IEEE INFOCOM '00*, Mar. 2000, pp. 727–736.

[16] S. Viswanathan and T. Imilelinski, "Pyramid broadcasting for video on demand service," in *Proc. of the SPIE Multimedia Computing and Networking Conf. (MMCN '95)*, Feb. 1995, pp. 66–77.

Masahiko Tsukamoto received the B.E., M.S., and Dr.E. degrees from Kyoto University, Kyoto, Japan, in 1987, 1989, and 1994, respectively. From April 1989 to February 1995, he was a research engineer at Sharp Corporation. In March 1995, he joined the Department of Information Systems Engineering of Osaka University as an assistant professor and in October 1996, he became an associate professor at the same department. From October 2004, he has been a full professor at the Faculty of Engineering, Kobe University. He has been also the president of NPO Wearable Computer Research and Development Organization since February 2004. His research interests include wearable computing and ubiquitous computing.

Tomoki Yoshihisa received the Bachelor's, Master's, and Doctor's degrees from Osaka University, Osaka, Japan, in 2002, 2003, 2005, respectively. From April 2005, he has been a research associate at Academic Center for Computing and Media Studies, Kyoto University. His research interests include broadcast computing, and wearable computing.

Shojiro Nishio received his B.E., M.E., and Dr.E. degrees from Kyoto University, Japan, in 1975, 1977 and 1980, respectively. From 1980 to 1988 he was with the Department of Applied Mathematics and Physics of Kyoto University. In October 1988, he joined the faculty of the Department of Information and Computer Sciences, Osaka University, Japan. In August 1992, he became a full professor in the Department of Information Systems Engineering of Osaka University. Since April 2002, he has been a full professor in the Department of Multimedia Engineering of Osaka University. From April 2000 to July 2003, he served as the director of Cybermedia Center of Osaka University. In August 2003, he has been serving as the dean of the Graduate School of Information Science and Technology of Osaka University. His current research interests include database systems and multimedia systems. Dr. Nishio has served on the Editorial Board of *IEEE Transactions on Knowledge and Data Engineering*, and is currently involved in the editorial board of *Data and Knowledge Engineering*, *Data Mining and Knowledge Discovery*. He is a fellow of IPSJ and he is a member of nine learned societies, including ACM and IEEE.