A Broadcasting Scheme Considering Units to Play Continuous Media Data

Tomoki Yoshihisa, Masahiko Tsukamoto, and Shojiro Nishio

***Abstract*—Due to the recent proliferation of digital broadcasting systems, various schemes for broadcasting continuous media data such as music or movies have been studied. In general broadcasting systems, since clients have to wait until their desired data are broadcast, these schemes reduce the waiting time by dividing the data into several segments of equal size. However, continuous media data often have units for playing portions of the data. For example, data encoded by MPEG2 can be played every GOP (Group of Pictures). In this paper, we propose a scheme to reduce the waiting time considering the units. Our proposed scheme divides a continuous media data at every unit. By scheduling divided data so that clients finish receiving a unit before starting to play the unit, waiting time is effectively reduced.**

***Index Terms*—Broadband communication, broadcasting, interactive TV, protocols, satellite broadcasting.**

I. Introduction

BROADCASTING systems have recently been digitized. Since the influence of noise on digital broadcasting is fainter than that on analog broadcasting, there is room for the system's bandwidth to expand, e.g. up to approximately 24 Mbps [14]. Moreover, since it is easy for digital broadcasting to multiplex data, many channels are available. Accordingly, various schemes for delivering continuous media data such as music or movies using attractive digital broadcasting systems have been studied [1], [3]–[7], [9]–[13], [15], [16]. Detail is explained in Section II. In general broadcasting systems, a server broadcasts data repetitively. Although the server can concurrently deliver the data to many clients, clients have to wait until their desired data are broadcast. In continuous media data broadcasting, it is usually postulated that clients can play the data without any interruptions until the end of the data. Hence, these schemes reduce clients' waiting time under the postulate.

Studies listed in the references usually assume that a client can play data as soon as all the data is received. However, in many cases, clients can play the data after receiving units for playing portions of the data. For example:

- A unit for playing data encoded by MPEG is a GOP (Group of Pictures) [8]. Clients can play a GOP only after finishing receiving the GOP.
- Continuous media data are often encoded using intra-frame compression. In this case, a unit for playing data must be a frame.
- In the case where data are encoded using inter-frame compression, a unit for playing data must be a group of these interconnected frames. MPEG is one of these cases.

Accordingly, clients must wait until a unit is received. This problem cannot be solved only by artificially delaying the start of playback by the largest size of units. Because a comparison with conventional schemes, which are extended to wait the start of playback, reveals that our proposed scheme gives shorter waiting time than these extended conventional schemes.

In this paper, we propose a broadcasting scheme to reduce waiting time considering units to play continuous media data. By dividing data into each unit to be played, and scheduling them so that clients finish receiving a unit before starting to play the unit, our proposed scheme reduces the waiting time.

The remainder of this paper is as follows: In Section II, we discuss related researches. We explain our proposed scheme in Section III. We analyse the scheme in Section IV and evaluate it in Section V. In Section VI, we discuss our proposed scheme and, finally, we conclude the paper in Section VII.

II. Related Works

In continuous media data broadcasting, clients have to wait until their desired data are broadcast. Accordingly, various schemes have been proposed to reduce this waiting time. These studies assume that a client can play the data as soon as it starts receiving the data. Here, we introduce two major schemes for continuous media data broadcasting.

The HB (Harmonic Broadcasting) scheme [6] divides continuous media data into N segments of equal size. Segments indicate divided data. Here we indicate segments by $S_1, \ldots, S_N$, where the subscript increases along with time. In addition, a segment S_i $(i = 1, \ldots, N)$ is divided into i sub-segments $S_{i,1}, \ldots, S_{i,i}$. Segments or sub-segments can be played as soon as clients start receiving them and cannot be received midstream. The server uses N channels $C_1, \ldots, C_N$. The bandwidth of C_i is r/i. Here, r is the consumption rate of the data. Therefore, bandwidths for channels are not always equivalent to the data consumption rate. For example, when a server broadcasts 5 Mbps data encoded by MPEG2, $r = 5$ Mbps. The server broadcasts $S_{i,1}, \ldots, S_{i,i}$ via C_i repetitively. Since S_1 is frequently broadcast in C_1, clients' waiting time can be

Manuscript received October 28, 2005; revised February 6, 2007. This work was supported in part by Special Coordination Funds for Promoting Science and Technology: "Yuragi Project" and in part by Grant-in-Aid for Young Scientists (B) numbered 18700085 from the Ministry of Education, Culture, Sports, Science and Technology of Japan.

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

M. Tsukamoto is with the Department of Electrical and Electronics Engineering, Faculty of Engineering, Kobe University, Kobe 657-8501, Japan (e-mail: tuka@kobe-u.ac.jp).

S. Nishio is with the Department of Multimedia Engineering, Graduate School of Information Science and Technology, Osaka University, Osaka 565-0871, Japan (e-mail: nishio@ist.osaka-u.ac.jp).

Digital Object Identifier 10.1109/TBC.2007.903639

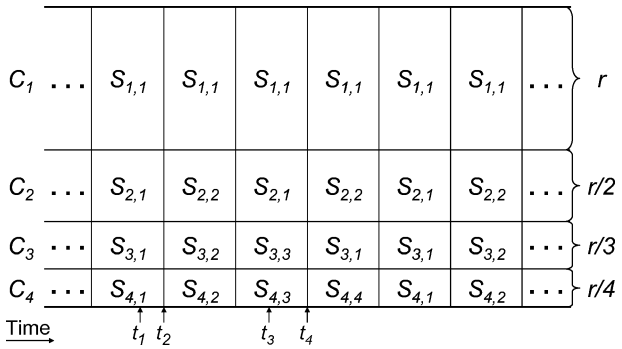

Fig. 1. A broadcasting schedule under the HB scheme ($N = 4$).

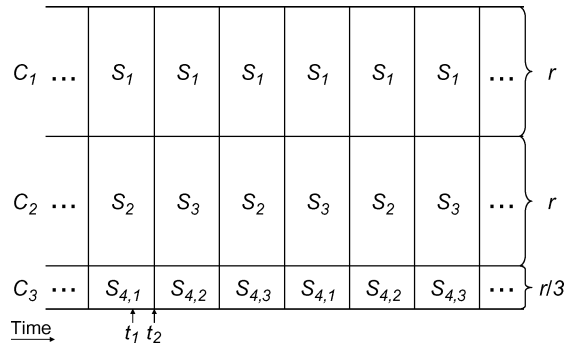

Fig. 2. A broadcasting schedule under the CHB scheme ($N = 4$).

reduced. Fig. 1 shows an example of the broadcasting schedule under the HB scheme when $N = 4$.

PB (Pagoda Broadcasting) [9], [10], [12], QHB (Quasi Harmonic Broadcasting) [11], and PHB (PolyHarmonic Broadcasting) [13] schemes are based on the HB scheme. That is, these schemes divide the data into segments of equal sizes. The FB (Fast Broadcasting) [7] scheme, which is another HB-based scheme, uses fixed channel bandwidths. The AB (Alternative Broadcasting) [16] scheme focuses on a single channel broadcasting. However, their performances are less than the HB scheme except under specific conditions.

The HB scheme has a problem. When a client starts playing S_1 as soon as it starts receiving S_1, the data cannot always be played without interruption. Suppose that a user who demands data at t_1 (Fig. 1) starts playing S_1 at t_2 as soon as the client starts receiving S_1. While the client will finish playing $S_{2,1}$ at t_3, it finishes receiving $S_{2,1}$ at t_4. Since the client cannot finish receiving $S_{2,1}$ until the time to play $S_{2,1}$, the client cannot play the data without interruption.

The CHB (Cautious Harmonic Broadcasting) scheme [11] solves this problem. The CHB scheme divides a segment S_j $(j = 4, \ldots, N)$ into $j-1$ sub-segments. The server uses $N-1$ channels. The bandwidth of C_1 or C_2 is r and that of C_k $(k = 3, \ldots, N-1)$ is r/k. The server broadcasts S_1 in C_1, S_2 and S_3 in C_2, and $S_{k+1,1}, \ldots, S_{k+1,k}$ in C_k repetitively. Since clients finish receiving S_k before the time to play S_k, a client that plays S_1 as soon as it starts receiving S_1 can play the data all the way through without interruption. The necessary bandwidth for the CHB scheme is larger than that for the HB scheme. The broadcasting schedule under the CHB scheme when $N = 4$ is shown in Fig. 2.

In DSB (Dynamic Skyscraper Broadcasting) [1], SB (Skyscraper Broadcasting) [4], Fuzzycast [5], and PB (Pyramid Broadcasting) [15] schemes, bandwidths for channels are fixed. Therefore, these schemes are easily implemented. However, these performances are less than those of HB or CHB scheme.

In this paper, we do not assume that a client can play the data as soon as it starts receiving the data. Many continuous media data have units for playing such as GOP for MPEG encoding. By considering such units, our proposed scheme reduces the waiting time more than conventional schemes (Section V-C). Main differences between the paper and our previous paper [16] are consideration of such units and the number of channels. The paper deals with multiple channels while our previous paper deals with a single channel.

III. Proposed Scheme

We propose the *Asynchronous Harmonic Broadcasting* (AHB) scheme to reduces clients' waiting time when units to be played are given. The AHB scheme adopts the HB scheme's idea that the bandwidth of each channel is adjusted so that a client finishes receiving a segment before it plays the segment. A difference between the AHB scheme and conventional schemes such as the HB or the CHB is that the AHB scheme divides data into separate units to be played. Therefore, in the AHB scheme, units and segments have the same meaning. Due to the recent popularization of digital broadcasting systems, the number of users of those systems is increasing. To offer a good service to those users, a shorter waiting time is still required. Hence, the AHB scheme is useful for current broadcasting systems.

A. Assumed Environment

- Continuous media data cannot be played as soon as a client starts receiving the data. The data can be played after the client finishes receiving a unit to be played.
- A server broadcasts divided data repetitively via multiple channels.
- A client can concurrently receive the data via multiple channels.
- A client has a buffer large enough to store the data.
- A client starts receiving the data after the client's user demands the data.

An example of our assumed environment is broadcasting movie data encoded by MPEG2 via a satellite digital broadcasting system. As we mentioned in Section I, digital broadcasting systems can use multiple channels. Here, channels mean logical channels in a broadcasting system. In addition, MPEG2 data cannot be played as soon as the client starts receiving a GOP because of restrictions of its compression technology. MPEG2 data can be played after the client finishes receiving the GOP.

B. The Scheduling Process

We denote D as the playing time of a continuous media data, and r as the data consumption rate. Note that r is constant while clients play the data (constant bit rate). Furthermore, $a_1, \ldots, a_N$ denote the data sizes for units to be played, i.e., the data size needed to play some portions of a continuous media data such as a GOP or a frame. According to $a_1, \ldots, a_N$, the data are divided

into N segments $S_1, \ldots, S_N$. That is, the data size for S_i $(i = 1, \ldots, N)$ is a_i. The playing time of S_i, which is denoted by p_i, becomes a_i/r and $D = p_1 + \cdots + p_N = (a_1 + \cdots + a_N)/r$. The scheduling process is shown below:

1) Divide the data into $S_1, \ldots, S_N$ so that the data size for S_i is a_i $(i = 1, \ldots, N)$.
2) Determine the arbitrary value of b_1.
3) Use N channels $C_1, \ldots, C_N$ and set the bandwidth of C_j $(j = 2, \ldots, N)$, which is denoted by b_j, by the following equation:

$$b_j = \frac{a_j}{\frac{a_1}{b_1} + p_1 + \cdots + p_{j-1}}. \quad (1)$$

4) If $b_1 + \cdots + b_N$ is larger than the available bandwidth, decrease b_1 and repeat the previous process. Otherwise, increase b_1 and repeat the previous process. If the summation is almost the same as the available bandwidth, go to the next process.
5) Broadcast S_i via C_i repetitively.

b_1 is adjusted so that the total bandwidth $B = b_1 + \cdots + b_N$ is equal to the available bandwidth by repeating calculation. By setting the bandwidth by (1), a client can finish receiving S_j before it starts playing S_j. The detail reason follows.

C. Playing Without Interruptions

When a server broadcasts a continuous media data according to the broadcasting schedule produced by the AHB scheme, a client that starts playing S_1 as soon as it finishes receiving S_1 can play the data without any interruptions. The reason is given below.

A client finishes receiving S_1 after a_1/b_1 from the time that it starts receiving S_1. Shortly after that, the client starts playing S_1. The client starts playing S_i $(i = 2, \ldots, N)$ after $p_1 + \cdots + p_{i-1}$ from the time that it starts playing S_1. Therefore, it takes $a_1/b_1 + p_1 + \cdots + p_{i-1}$ for the client to start playing S_i after it starts receiving S_1. On the other hand, since the bandwidth of C_i is given by (1), the broadcasting interval of S_i is $a_1/b_1 + p_1 + \cdots + p_{i-1}$. Hence, it takes $a_1/b_1 + p_1 + \cdots + p_{i-1}$ for the client to finish receiving S_i after it starts receiving S_1. In this way, the client can finish receiving S_i before it starts playing S_i.

The name AHB is derived from the fact that the broadcasting intervals of S_i are not synchronized with that of S_1.

For example, Fig. 3 shows a broadcasting schedule under the AHB scheme where $B = 2.64r$, $a_1 = 3r$, $a_2 = r$, $a_3 = 3r$, and $a_4 = 4r$. In this case, b_1 becomes $1.5r$.

D. Implementation

A server divides a continuous media data into each unit to be played. The server adjusts b_1 according to the available bandwidth and broadcasts segments repetitively according to the schedule produced by the AHB scheme. When the server broadcasts a segment, it adds the data number and the segment number to the front of the segment, enabling clients to receive the data number and the segment number of the received segment. Since this additional information is written in front of segments, clients cannot receive segments from midstream. Since the data size for additional information is small compared with that for the whole data, we ignore the time needed to broadcast it. Note that, since segments are not received midstream, each segment is divided into several sub-segments so that the interval of sub-segments is equal to the interval of S_1.

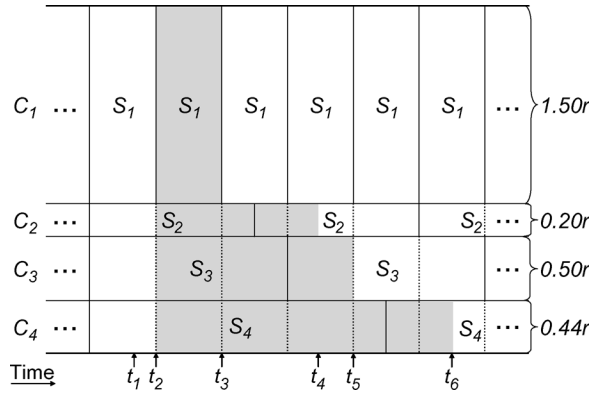

Fig. 3. An example of a broadcasting schedule under the AHB scheme.

When a user demands to play a continuous media data, the user's client starts receiving the desired data from the broadcast data. Then, the client plays the data as soon as it finishes receiving S_1. While the client plays the data, it receives the broadcast data and stores the data into its buffer. After all sub-segments of S_i $(i = 2, \ldots, N)$ are received, the client quickly combines them to play S_i. The client plays the combined S_i as soon as the client finishes playing S_{i-1}. In this way, the client can play the data right through without any interruptions. For example, suppose that a user demands the data at t_1 (Fig. 3). A dashed line indicates a separator of sub-segments and the shaded area indicates the data received by a client that demands the data at t_1. The user's client starts receiving S_1 from t_2 and starts playing S_1 at t_3 as soon as the client finishes receiving S_1. t_3 is within the S_2 reception period and the client starts playing S_1 while it receives S_2. Since $b_1 = 1.5r$, the playing time of S_1 is longer than the time needed to broadcast S_1. Accordingly, the client finishes playing S_1 at t_4. The time needed to receive S_2 is the interval of C_2, $t_4 - t_2$. Since the client starts receiving S_2 from t_2, the client finishes receiving S_2 at t_4. Therefore, the client can play S_2 without interruption. The client finishes receiving S_3 before the client finishes playing S_2 (t_5), and finishes receiving S_4 before it finishes playing S_3 (t_6). Accordingly, the client can play the data right through without any interruptions.

IV. Analysis

A. The Necessary Bandwidth

The necessary bandwidth for broadcasting a continuous media data with the AHB scheme B is:

$$\begin{aligned} B &= \textstyle\sum_{i=1}^{N} b_i \\ &= b_1 + \sum_{i=2}^{N} \frac{a_i}{\frac{a_1}{b_1} + p_1 + \cdots + p_{i-1}} \\ &= b_1 + \sum_{i=2}^{N} \frac{a_i b_1 r}{a_1 r + (a_1 + \cdots + a_{i-1}) b_1}. \end{aligned} \quad (2)$$

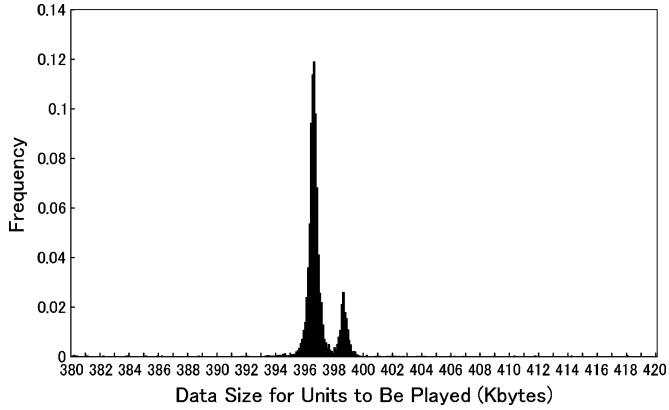

Fig. 4. The histogram of data sizes needed to play the data: GOP sizes of MPEG2 movie data. We can see that data sizes needed to play the data have different sizes.

TABLE I
STATISTICS OF DATA FOR EVALUATION: THESE DATA ARE BASED ON ACTUAL MPEG2 MOVIE DATA. WE USE THE MOVIE DATA FOR EVALUATIONS

Minimum (Kbytes)	70
Maximum (Kbytes)	458
Average (Kbytes)	392
Variance (Kbytes)^2	1,163
Total (Mbytes)	2,349
Segments	5,994

B. The Average Waiting Time

The maximum waiting time W_{max} is given by the client that demands data shortly after S_1 is broadcast. Since the interval of S_1 is a_1/b_1,

$$W_{max} = \frac{2a_1}{b_1}. \tag{3}$$

The minimum waiting time W_{min} is given by the client that demands data shortly before S_1 is broadcast.

$$W_{min} = \frac{a_1}{b_1}. \tag{4}$$

Since the waiting time is supposed to be uniformly distributed, the average waiting time W_{ave} is

$$W_{ave} = \frac{3a_1}{2b_1}. \tag{5}$$

V. PERFORMANCE EVALUATION

In this section, we evaluate the necessary bandwidth and the average waiting time under the AHB scheme. Moreover, we compare the average waiting time with that under the HB scheme and the CHB scheme in the case where a client waits so that it can play the data without any interruption. In this section, we use data for 60 minutes encoded by MPEG2 ($r = 5$ Mbps). Since units for playing a data encoded by MPEG2 are GOP, $a_1, \ldots, a_N$ are given by the data sizes for the GOP. Fig. 4 shows the histogram of units to be played, i.e., the GOP. Table I shows the relevant statistics. Although the distribution of units to be played does not always resemble the histogram, it is clear that data sizes for units to played are not constant. These relatively small differences between the MPEG-2 GOP sizes have a strong influence on the system performance because the times to broadcast them are long.

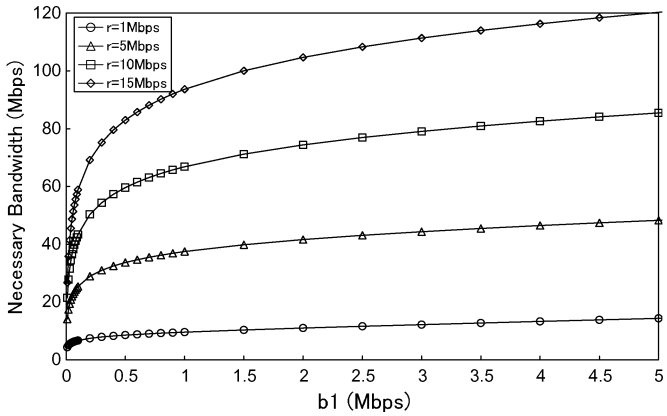

Fig. 5. Necessary bandwidth under the AHB scheme.

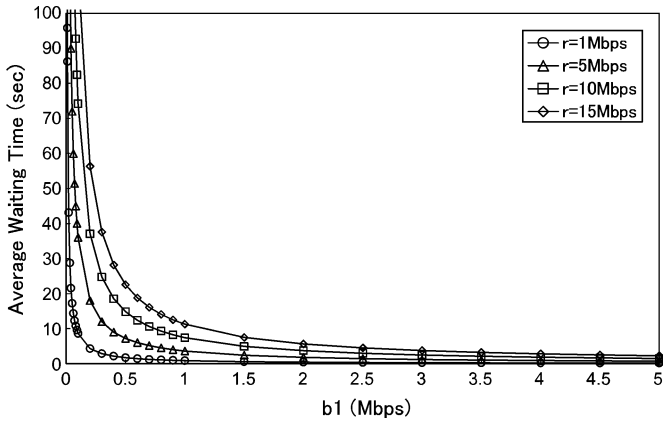

Fig. 6. Average waiting time under the AHB scheme.

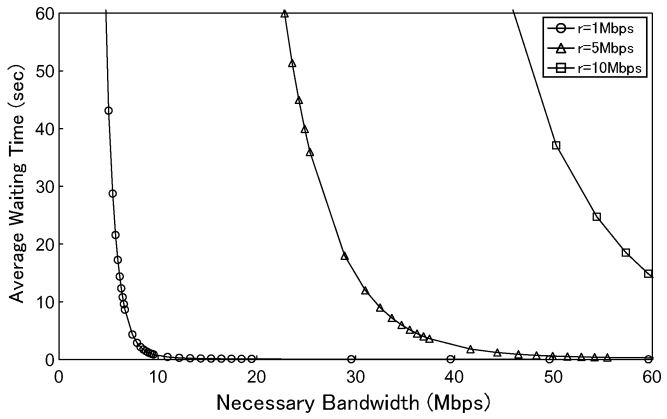

Fig. 7. The relationship between bandwidth and average waiting time under the AHB scheme.

The summary for evaluation part follows. The average waiting time under the AHB scheme is reduced further as the available bandwidth increases (Figs. 5–7). In addition, the average waiting time is shorter than the HB or CHB scheme.

This is because the AHB scheme considers units to play the data and produces an effective broadcast schedule (Fig. 9).

A. The Necessary Bandwidth Under the AHB Scheme

The necessary bandwidth for broadcasting continuous media data under the AHB scheme is shown in Fig. 5. The horizontal axis represents b_1 and the vertical axis the necessary bandwidth. We can see that a larger b_1 requires a larger bandwidth and the increasing rate becomes lower as b_1 increases. For example, in the case where $b_1 = 0.1$ Mbps, the necessary bandwidth is 25 Mbps, and where $b_1 = 0.2$ Mbps, the necessary bandwidth is 28 Mbps. Where $b_1 = 0.9$ Mbps, the necessary bandwidth is 36 Mbps, and where $b_1 = 1.0$ Mbps, the necessary bandwidth is 37 Mbps. Although b_1 is increased by the equivalent 0.1 Mbps, the necessary bandwidths increase by $28 - 25 = 3$ Mbps or $37 - 36 = 1$ Mbps. This is because the interval of S_1 $(= a_1/b_1)$ decreases in inverse proportion to b_1. Hence, a larger b_1 further decreases the interval of S_1. Since the interval of S_1 is included in the denominator of (1), a larger b_1 increases the necessary bandwidth at the lower rate.

For example, in the case where $b_1 = 80$ Kbps, the necessary bandwidth is 24 Mbps. Since the bandwidth of a general satellite digital broadcasting system is approximately 24 Mbps, this is practical.

B. The Average Waiting Time

The average waiting time under the AHB scheme is shown in Fig. 6. The horizontal axis represents b_1 and the vertical axis the average waiting time. It is clear that a larger b_1 reduces the average waiting time more. This is because clients can receive S_1 more frequently since a larger b_1 further decreases the interval of S_1.

The relationship between the bandwidth and the average waiting time under the AHB scheme, i.e., a combination of Figs. 5 and 6, is shown in Fig. 7. The horizontal axis represents the bandwidth and the vertical axis the average waiting time. In the case of broadcasting MPEG2 (5 Mbps) data for 60 minutes via a 24 Mbps satellite digital broadcasting system, the average waiting time is 47.3 seconds.

C. Comparison With the HB Scheme or the CHB Scheme

By extending conventional schemes to delay the start of playback by the largest size of units, clients can play the data without interruption. Therefore, we compare the AHB scheme with these extended conventional schemes such as the HB scheme or the CHB schemes. In both the HB scheme and the CHB scheme, the data are assumed to be played as soon as clients start receiving it. Hence, in cases where the data are played after clients finish receiving a unit to be played, clients cannot always finish receiving the unit before the time to play it. Accordingly, clients have to wait to finish receiving the unit before playing it. Therefore, the waiting time is different from that in the case where the data are played as soon as the client starts receiving it. An example follows: In the case where $a_1 = 3r$, $a_2 = r$, $a_3 = 3r$, and $a_4 = 4r$, the AHB scheme divides the data into $S_1, \ldots, S_4$ according to the data sizes for the units to be played. However, the HB scheme divides

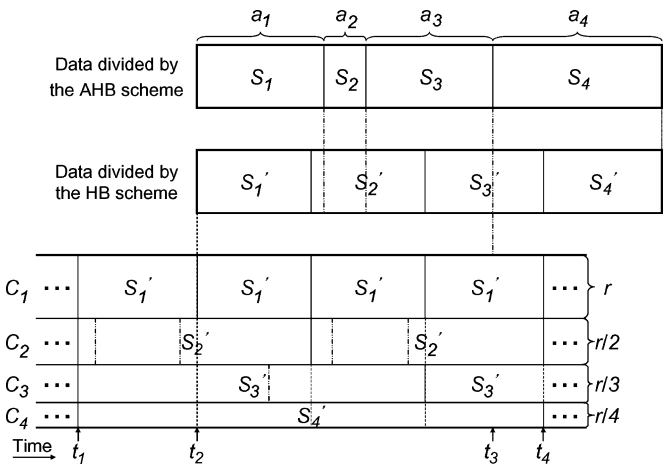

Fig. 8. Data division under the AHB and HB schemes: In the HB scheme, clients that demand the data at t_1 have to wait $t_4 - t_3$ from t_2 to play the data without interruption.

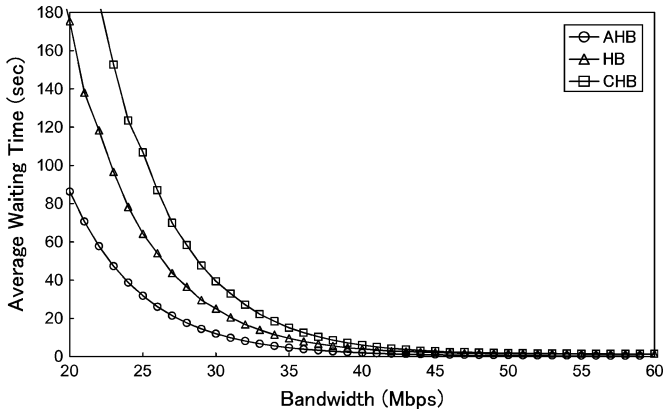

Fig. 9. Average waiting time under the AHB scheme, the HB scheme, and the CHB scheme.

the data into $S_1', \ldots, S_4'$ without concerning $a_1, \ldots, a_4$, and broadcasts them as shown in Fig. 8 (a single segment includes some GOPs). Suppose that a client that demands the data at t_1 starts playing the data at t_2 as soon as it finishes receiving S_1 (some portion of S_1 included in S_2' is received in C_2). To play the data without interruption, the client has to finish receiving S_4 before t_3, i.e., the time to play S_4. However, since the client finishes receiving S_4', which is included in S_4, at t_4, the client cannot finish receiving S_4 until t_3. In this case, by starting to play S_1 after $t_4 - t_3$ from the time it finishes receiving S_1', the client can play the data without interruption until the end of the data. In this case, the waiting time increases by $t_4 - t_3$. In this way, we calculated the average waiting time under the HB scheme and the CHB scheme. The result is shown in Fig. 9. The horizontal axis represents the bandwidth and the vertical axis the average waiting time. The value of b_1 under the AHB scheme and the number of segments under the HB scheme and the CHB scheme are adjusted according to the available bandwidth. In the comparison, the frame rate is 30 fps, the number of frames in a GOP is 15, and the encoding is constant bit rate with non-interlace. Other parameters are the same as previous evaluations.

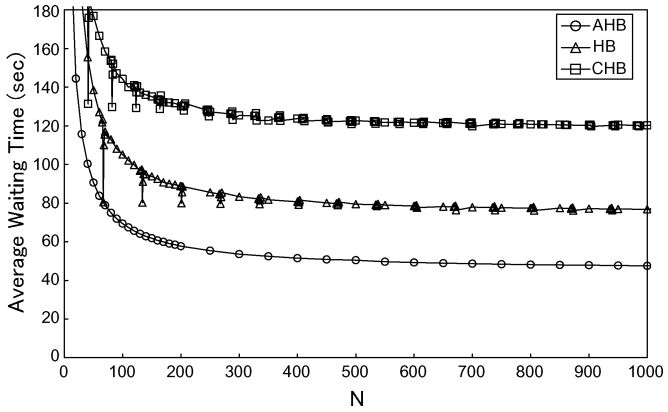

Fig. 10. The relationship between the number of segments and average waiting time.

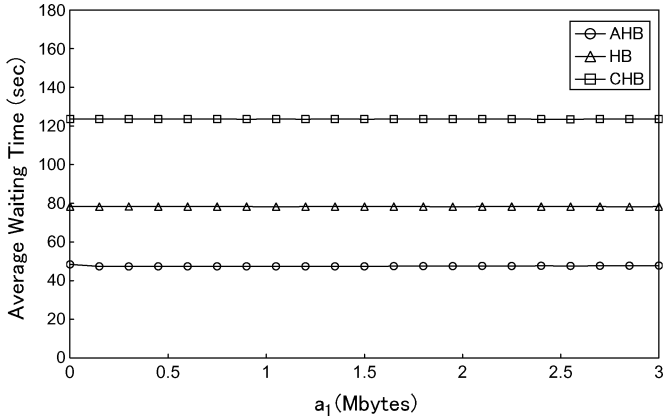

Fig. 11. The relationship between the data size for a_1 and average waiting time.

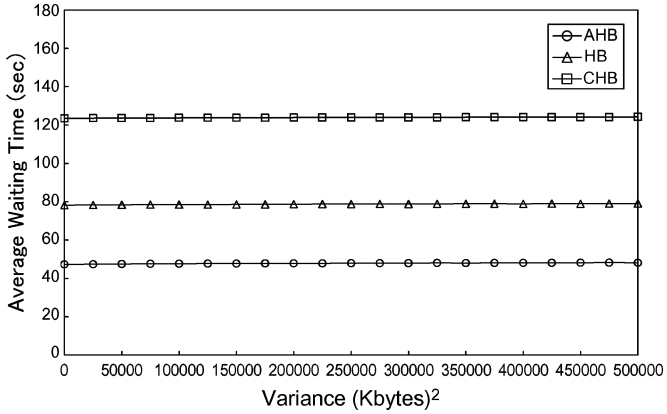

Fig. 12. The relationship between the variance of $a_1, \ldots, a_N$ and average waiting time.

The average waiting time under the HB scheme and the CHB scheme are longer than that under the AHB scheme. Moreover, the average waiting time under the HB scheme is shorter than that under the CHB scheme. This is because the number of segments under the HB scheme is larger than that under the CHB scheme. Since the interval of S_1 under the HB scheme is shorter than that under the CHB scheme, the average waiting time is shorter. However, since the average waiting time depends on the data sizes for units to be played, the average waiting time under the HB scheme is not always shorter than that under the CHB scheme. For example, in the case where $a_1 = a_2 = a_3 = a_4 = r$, and the bandwidth is $2.34r$, the average waiting time under the CHB scheme is shorter than that under the HB scheme.

When broadcasting MPEG2 data (5 Mbps) for 60 minutes via a 24 Mbps satellite digital broadcasting system, the average waiting time under the AHB scheme is 47.3 seconds, whereas that under the HB scheme is 78.3 seconds and that under the CHB scheme stretches out to 123 seconds.

VI. Discussion

The AHB scheme reduces the average waiting time where units to be played are given beforehand. By changing b_1, the necessary bandwidth is adjusted to the available bandwidth. The number of channels depends on the server's request for waiting time reduction. The server may determine a suitable number of channels so that the waiting time is practical.

The summary for discussion part follows. The average waiting time under the AHB scheme is reduced further as the number of segments increases (Fig. 10). In addition, the data size of the first segment and variance of data sizes for segments do not influence the average waiting time (Figs. 11 and 12). Therefore, we can say that, as the number of segments increases, our proposed scheme gives shorter waiting times.

A. Comparison With Previous Studies

As mentioned above, the average waiting time under the AHB scheme is shorter than that under both the HB scheme and the CHB scheme. In this way, the average waiting time under the AHB scheme is shorter than that under schemes that do not consider units to be played.

B. The Data Sizes to be Played and the Average Waiting Time

In Section V, the GOP was regarded as a unit for playing portions of a data encoded by MPEG2. Accordingly, $a_1, \ldots, a_N$ were given by the data sizes for the GOP. However, the average waiting time under the AHB scheme depends on $a_1, \ldots, a_N$. In this section, we discuss the effect of data sizes for units to be played on the average waiting time.

1) The Number of Segments and the Average Waiting Time: Since the average waiting time under the AHB scheme depends on the number of segments N, we discuss the effect of N on the average waiting time.

We assume that the data sizes for units to be played are equal $(a_1 = \cdots = a_N = a)$. We adjust a so that the total amount of units, $a_1 + \cdots + a_N$, is $5\ \mathrm{M} \times 60 \times 60/8 = 2250$ Mbytes. Fig. 10 shows the average waiting time when broadcasting such continuous media data ($r = 5$ Mbps) for 60 minutes via a 24 Mbps bandwidth. The horizontal axis represents the number of segments N and the vertical axis the average waiting time. It is clear that a larger N has a greater effect at reducing the average waiting time. However, a larger N further decreases the reduction rate because a becomes smaller in inverse proportion to N. For example, in the case where $N = 50$, a is 45 Mbytes. To achieve the necessary bandwidth of 24 Mbps, b_1 becomes 5.96 Mbps. In this case, the average waiting time is $(3 \times 45\ \mathrm{M} \times 8)/(2 \times 5.96\ \mathrm{M}) = 90.6$ seconds. Where

$N = 100$, a is 22.5 Mbytes and the average waiting time is $(3 \times 22.5\ \mathrm{M} \times 8)/(2 \times 3.89\ \mathrm{M}) = 69.4$ seconds. Whereas, in the case where $N = 950$, a is 2.37 Mbytes and the average waiting time is $(3 \times 2.37\ \mathrm{M} \times 8)/(2 \times 596\ \mathrm{K}) = 47.7$ seconds. Where $N = 1000$, a is 2.25 Mbytes and the average waiting time is $(3 \times 2.25\ \mathrm{M} \times 8)/(2 \times 568\ \mathrm{K}) = 47.5$ seconds. In this way, although N is equivalently increased by 50, since the decreasing rate of a is different, the rate of the average waiting time reduction decreases.

Under both the HB scheme and the CHB scheme, which divide data into segments in equal size, the average waiting time is sharply reduced repetitively. This occurs because the boundaries that divide the data match the boundaries of units to be played. Hence, a client that starts playing the data as soon as it finishes receiving the first unit to be played (S_1 in Fig. 8) can play the data without any interruptions. When using 24 Mbps, the HB scheme divides the data into 67 segments, and the CHB scheme divides the data into 41 segments (These numbers of segments are calculated by the given equations in [11] and [6]). Hence, if N is some integral times of these values, the average waiting time is reduced sharply. For example, when $N = 67, 134, 201, \ldots$, the average waiting time under the HB scheme is reduced sharply. Particularly, when $N = 67$, the boundaries that divide the data under the AHB scheme are the same as those under the HB scheme. Hence, the average waiting time under the HB scheme is equal to that under the AHB scheme.

2) The Effect of a_1 on the Average Waiting Time: Since a_1 is included in (5), a_1 can be a fatal factor in the average waiting time. Therefore, we discuss the effect of a_1 on the average waiting time.

To maintain consistency with Section V, only a_1 is changed. $N = 5994$, $r = 5$ Mbps, and $a_2 = \cdots = a_N = a$. a is adjusted so that the total amount of units to be played, $a_1 + \cdots + a_N$, becomes 2349 Mbytes. Fig. 11 shows the average waiting time for the case of broadcasting MPEG2 data (5 Mbps) for 60 minutes via a 24 Mbps network.

In the case where $a_1 = 3$ Mbytes, the playing time of S_1 is 4.8 seconds. Since the playing time of a GOP is generally 0.5 seconds, such a large S_1 is not practical. However, to accurately evaluate the performance, we allow a sufficiently wide range of a_1. From Fig. 11, we can see that a_1 does not affect the average waiting time. While a larger a_1 further decreases $a_2, \ldots, a_N$, the decreasing rate is smaller than the increasing rate of a_1. Hence, the bandwidths of $C_2, \ldots$, and C_N decrease. Accordingly, since b_1 increases, the average waiting time does not change. For example, in the case where $a_1 = 100$ Kbytes, by giving b_1 of 25.4 Kbps, the necessary bandwidth becomes 24 Mbps and the average waiting time is 47.3 seconds. In the case where $a_1 = 1$ Mbyte, by giving b_1 a value of 253 Kbps, the necessary bandwidth becomes 24 Mbps and the average waiting time is 47.4 seconds. In this case, since a 10-times larger a_1 causes b_1 to become also 10-times larger, the average waiting time does not change. However, in the case where the increasing rate of b_1 becomes smaller, a larger a_1 has a greater effect on the average waiting time. For example, in the case where $a_1 = 100$ Mbytes, by giving b_1 of 10.1 Mbps, the necessary bandwidth becomes 24 Mbps and the average waiting time is 119 seconds. However, where $a_1 = 1000$ Mbytes, by giving b_1 of 20.3 Mbps, the necessary bandwidth becomes 24 Mbps and the average waiting time extends to 591 seconds. Since an a_1 10 times as large makes b_1 only twice as large, the average waiting time increases. One reason that the average waiting time under the HB scheme and the CHB scheme changes little is that N is sufficiently large. Accordingly, because of the difference between the boundaries that divide the data and that of the units to be played ($t_4 - t_3$ in Fig. 8), the average waiting time changes only a little. Hence, the average waiting time does not change.

TABLE II
STATISTICAL VALUES OF $a_1, \ldots, a_N$ FOR FIG. 12: TO EVALUATE THE EFFECT OF DATA SIZES, WE GENERATED $a_1, \ldots, a_N$ BY NORMAL DISTRIBUTION. THE TOTAL SIZE IS CONSTANT

Variance $(\mathrm{Kbytes})^2$	0	5×10^5
Minimum (Kbytes)	392	1.1
Maximum (Kbytes)	392	3,065
Average (Kbytes)	392	733
Total (Mbytes)	2,349	2,349
Segments	5,994	3,202

As a result, you do not have to care the size of a_1, even when you can control the size of a_1 in practical use.

3) The Effect of Variance on the Average Waiting Time: Since $a_1, \ldots, a_N$ depend on the data broadcast, we discuss the effect of the variance of $a_1, \ldots, a_N$ on the average waiting time.

In this section, we generate $a_1, \ldots, a_N$ by normal distribution, of which the average is 392 Kbytes. We adjust N so that the total data size for the units to be played is 2349 Mbytes. The negative value is ignored. The statistic values, when the variance is 0 Kbytes^2 and 5×10^5 Kbytes^2, are shown in Table II. Fig. 12 shows the average waiting time for broadcasting continuous media data ($r = 5$ Mbps) for 60 minutes via a 24 Mbps bandwidth. The horizontal axis represents the variance and the vertical axis the average waiting time.

It is clear that the variance does not affect the average waiting time. This is because, when the data size for a segment changes, the bandwidth of the channel that broadcasts the segment also changes. As a result, the average waiting time changes only a little. For example, in the case where the variance is 0 Kbytes^2, $a_1 = 392$ Kbytes, $b_1 = 99$ Kbps, and the average waiting time is $(3 \times 392\ \mathrm{K} \times 8)/(2 \times 99\ \mathrm{K}) = 47.5$ seconds. In the case where the variance is 5×10^5 Kbytes^2, $a_1 = 724$ Kbytes, $b_1 = 180$ Kbps, and the average waiting time is $(3 \times 724\ \mathrm{K} \times 8)/(2 \times 180\ \mathrm{K}) = 48.3$ seconds. In this way, since a larger a_1 increases b_1 more, the average waiting time changes a little.

Therefore, where N is sufficiently large and the data size for units to be played is sufficiently small, the distribution of $a_1, \ldots, a_N$ does not affect the average waiting time of the AHB scheme. Hence, Fig. 12 reveals that you do not have to care the distribution, even when you can control the variance of $a_1, \ldots, a_N$ in practical use. For example, in the case of broadcasting 5 Mbps of continuous media data for 60 minutes via a 24 Mbps bandwidth, if N is larger than approximately 500, $a_1, \ldots, a_N$ does not affect the average waiting time (Fig. 10).

C. A Modification for VBR

MPEG has two types of bit rate. One is CBR, in which the consumption rate is constant while the data are played. The other is VBR (Variable Bit Rate), in which the consumption rate varies

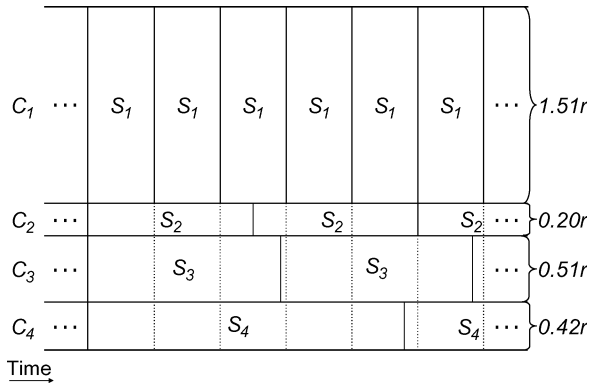

Fig. 13. Broadcasting schedule under the AHB scheme for VBR: the AHB scheme can be easily implemented for VBR by modifying Formula(2).

while the data are played. In this paper, although the data was assumed to be CBR, here, we introduce an application of the AHB scheme to VBR. In VBR, since the consumption rate varies with every unit to be played, let r_i $(i = 1, \ldots, N)$ indicate the consumption rate of S_i. In the case of VBR, since the same arguments in Section III-C are established $(p_i = a_i/r_i)$, a client that starts playing the data as soon as it finishes receiving S_1 can play the data completely without any interruptions by giving the bandwidth of C_j $(j = 2, \ldots, N)$ by (1). Hence, the AHB scheme can be easily implemented also in the case of VBR. The necessary bandwidth is not given by (2), but:

$$B = b_1 + \sum_{i=2}^{N} \frac{a_i b_1}{a_1 + \left(\frac{a_1}{r_1} + \cdots + \frac{a_{i-1}}{r_{i-1}}\right) b_1}. \tag{6}$$

Fig. 13 shows the broadcasting schedule in the case where B, $a_1, \ldots, a_4$ are the same as in Fig. 3, and the consumption rates of segments are $r_1 = r$, $r_2 = 1.2r$, $r_3 = 0.8r$, and $r_4 = 1.1r$.

VII. Conclusion

In many cases, since continuous media data have units for playing portions of the data, the data divided into given data sizes are not played as soon as clients start receiving it. Accordingly, clients have to wait until a unit is received to play the data. In this paper, we proposed a scheme to reduce the waiting time considering such units.

Our proposed scheme, the AHB scheme, reduces clients' waiting time where units to be played are given beforehand. By producing a broadcasting schedule that enables clients to finish receiving a segment before starting to play the segment, the AHB scheme reduces the waiting time for clients.

Our evaluations show that the average waiting time under the AHB scheme is shorter than that under both the HB scheme and the CHB scheme in cases where the data are assumed to be played after clients finish receiving a unit to be played.

Our future work includes development of a scheme to reduce the number of channels in order to simplify implementation.

References

[1] 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.

[2] 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.

[3] 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.

[4] 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.

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

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

[7] L.-S. Juhn and L. M. Tseng, "Fast data broadcasting and receiving scheme for popular video service," *IEEE Trans. Broadcasting*, vol. 44, no. 1, pp. 100–105, Mar. 1998.

[8] "Moving Picture Experts Group," [Online]. Available: http://www.chiariglione.org/mpeg/

[9] J.-F. Paris, "A simple low-bandwidth broadcasting protocol for video-on-demand," in *Proc. of the Int. Conf. on Computer Communications and Networks} (IC3N'99)*, Oct. 1999, pp. 118–123.

[10] 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.

[11] 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.

[12] J.-F. Paris, S. W. Carter, and D. D. E. Long, "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.

[13] 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.

[14] P. Kallender, "Internet-by-Satellite Flexes Muscles," *Computing Japan*, vol. 6, no. 6, June 1999.

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

[16] T. Yoshihisa, M. Tsukamoto, and S. Nishio, "A scheduling scheme for continuous media data broadcasting with a single channel," *IEEE Trans. Broadcasting*, vol. 52, no. 1, pp. 1–10, March 2006.

Tomoki Yoshihisa received the Bachelor's, Master's, and Doctor's degrees from Osaka University, Osaka, Japan, in 2002, 2003, 2005, respectively. Since 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.

Masahiko Tsukamoto received the B.E., M.S., and Dr.E. degrees from Kyoto University, Kyoto, Japan, in 1987, 1989, and 1994, respectively. Since 1989 to 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 1996, he became an associate professor at the same department. Since 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 2004. His research interests include wearable computing and ubiquitous computing.

Shojiro Nishio received his B.E., M.E., and Dr.E. from Kyoto University, Japan, in 1975, 1977, and 1980. He was with the Department of Applied Mathematics and Physics of Kyoto University from 1980 to 1988. In October 1988, he joined the faculty of the Department of Information and Computer Sciences of Osaka University. He became a full professor at the Department of Information Systems Engineering of Osaka University in August 1992. He has been a full professor at the Department of Multimedia Engineering at the same university since April 2002. From April 2000 to July 2003, he served as the founding director of the Cybermedia Center of Osaka University. He has been serving as the dean of the Graduate School of Information Science and Technology of this university since August 2003. 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 *ACM Transactions on Internet Technology*, and is currently involved with the editorial board of *Data and Knowledge Engineering*. He is a fellow of IEICE and IPSJ, and he is a member of eight learned societies, including ACM and IEEE.