A Scheduling Method for Bandwidth Reduction in Selective Contents Broadcasting

Tomoki Yoshihisa

Academic Center for Computing and Media Studies, Kyoto University
Honmachi, Yoshida, Sakyo-ku, Kyoto, 606-8501, Japan, yoshihisa@media.kyoto-u.ac.jp

ABSTRACT

Due to the recent popularization of digital broadcasting systems, selective contents, i.e., users watch their selected contents, have attracted great attention. For example, in a quiz program, a user selects his/her answer and watches the video content corresponding to the answer. Although a server can deliver content according to users' preferences, the necessary bandwidth to play the content continuously becomes large because the server has to broadcast several streams. However, by scheduling those streams considering the sequence in which to play them, the necessary bandwidth can be reduced. In this paper, we propose a scheduling method for bandwidth reduction in selective contents broadcasting. In our proposed method, by producing a broadcast schedule using a state transition graph, the necessary bandwidth is effectively reduced.

Keywords: Broadcasting, Selective Contents, Video on Demand, Continuous Media Data, Streaming

1 Introduction

Due to the recent popularization of wireless digital broadcasting systems such as DVB (Digital Video Broadcasting) and DMB (Digital Multimedia Broadcasing), selective contents, i.e., users watch their selected contents, have attracted great attention[10]. Examples for selective contents are listed below.

- In a quiz program, after watching several potential answers to the quiz, the user selects his/her answer. If the answer is correct, the user watches the video content for the correct answer. Otherwise, the user watches the content for the incorrect answer.
- In a news program, after watching an overview for each news story, the user selects the news story that he/she is interested in and watches it.
- In a TV drama, the user selects the main character's behavior. According to the selection, the story of the drama interactively changes.

By providing selective contents, users can watch their preferred contents. However, the server has to deliver several contents to provide selectivity.

On the other hand, when delivering continuous media data such as music or movies, playing the data without interruption is important for clients. In selective contents broadcasting, the necessary bandwidth to play the data without interruption becomes large compared to that of non-selective contents because the server broadcasts more streams. However, by scheduling those streams considering the sequence in which to play them, the necessary bandwidth can be reduced. By reducing the necessary bandwidth, the server can broadcast contents with much selectivity. Therefore, a merit of bandwidth reduction is that the server can provide enjoyable contents that can meet clients' preferences.

In this paper, we propose a scheduling method for bandwidth reduction in selective contents broadcasting. In our proposed method, the necessary bandwidth is effectively reduced by producing an effective broadcast schedule using state transition graphs.

The remainder of the paper is organized as follows. In Section 2, we explain selective contents. We explain our proposed method in Section 3 and evaluate it in Section 4. In Section 5, we discuss our proposed method, and finally, we conclude the paper in Section 6.

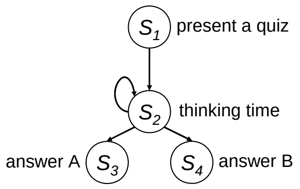

Figure 1: A play-sequence graph for a quiz program.

2 Selective Contents

In selective contents broadcasting, since users select their preferred contents and watch them, selective contents have sequences to play contents. Generally, since contents are played sequentially, state transition graphs are suitable to describe the sequence. In this paper, we call the state transition graph for describing the sequence of contents the *play-sequence graph*.

2.1 Play-Sequence Graph

In a play-sequence graph, each node represents a state in which the client plays content. When the client finishes playing the content, the state transits to the next node. The depth represents the elapsed time to transit from the root to the node. In some cases, the state can transit to an upper node. For example, a play-sequence graph for a quiz program is shown in Figure 1. In the quiz program, the user selects his/her answer from two potential answers, A or B. Node S_1 is the state in which the client plays the video presenting the quiz. The playing time of the video is 2 min. The state transits to the next node, S_2, 2 min. after starting to play S_1. S_2 is the state in which the client plays the video that explains answers A and B. The playing time of the video is 1 min. The user selects his/her answer from A or B while the video is playing. If the user selects answer A, the state transits to S_3. If the user selects answer B, the state transits to S_4. In this way, the state transits to the next node according to the user's choice. In the case where the user does not select his/her answer while S_2 is playing, the state transits to the beginning of S_2 again or automatically transits to the next node, S_3 or S_4. Since the original playing time of S_2 is shorter than S_1, the vertical distance between S_2 and S_3 or S_4 is shorter than that between S_1 and S_2. S_3 is the state in which the user selects answer A. The video for the correct answer is played in S_3. S_4 is the state in which the user selects answer B, and the video for the incorrect answer is played. By producing a broadcast schedule using play-sequence graphs, we effectively reduce the necessary bandwidth for playing data without interruption.

Play-sequence graphs have various structures. Therefore, finding a scheduling method that effectively reduces the necessary bandwidth consistently is difficult. Our proposed method reduces bandwidth by transforming play-sequence graphs to multiway trees and using those structures. Play-sequence graphs can be transformed to multiway trees by applying the following three operations.

2.1.1 Abbreviation

Clients can play received contents whenever they want by storing them in a buffer. Accordingly, since clients can always transit to previous states, state transitions that transit backward in time can be abbreviated. For example, in Figure 1, if the user does not select his/her answer, the state transits to the beginning of S_2 again. This state transition can be abbreviated. Not only state transitions to the same state, but also state transitions to the previous state, e.g., from S_2 to S_1, can be abbreviated. By abbreviating state transitions, we can ignore state transitions that do not affect system performance.

2.1.2 Merge

Nodes that do not have multiple branches can be merged with the next node. For example, in Figure 1, since S_1 does not have multiple branches, it can be merged with S_2. By merging nodes, we can simplify the play-sequence graph.

2.1.3 Split

By splitting a node into two nodes, the state transition can be described as a state transition without branches. By splitting a node, we can synchronize the time to start playing contents for each branch. For example, the play-sequence graph described in Figure 2-A can be transformed to Figure 2-B.

By applying the above operations, we can generalize play-sequence graphs into a multiway trees. For example, in the case of the quiz program, the play-sequence graph is transformed, as shown in Figure 3.

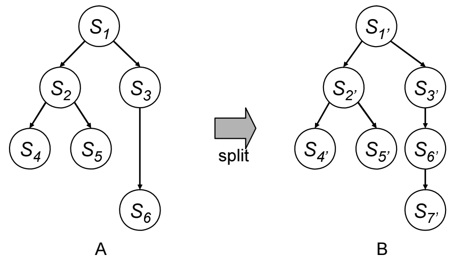

Figure 2: A split of a state in a play-sequence graph.

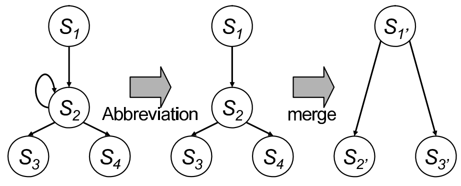

Figure 3: A simplification of a play-sequence graph.

2.2 Broadcast Schedule

In this section, we explain the effectiveness of our proposed method compared with that of a simple method.

In a simple schedule (the *Simple Method*), to play data without interruption, each datum is broadcast via several broadcast channels at the same time with starting playing the data. The broadcast schedule for the quiz program under the simple method is shown in Figrure4. The play-sequence graph is shown in Figure 3. The data consumption rate is 5 Mbps (MPEG2)[3]. C_1 and C_2 are broadcast channels, and each bandwidth is the same as the consumption rate. $D_{1'}$, $D_{2'}$, and $D_{3'}$ are data played in states $S_{1'}$, $S_{2'}$, and $S_{3'}$, respectively. Time proceeds from left to right. When broadcasting starts, $D_{1'}$ and $D_{2'}$ are broadcast via C_1. $D_{3'}$ is broadcast via C_2 3 min. after the broadcasting start time. When the client selects A, $D_{2'}$ starts playing after $D_{1'}$ has played. When the client selects B, $D_{3'}$ starts playing after $D_{1'}$ has played. In this case, the necessary bandwidth becomes 10 Mbps. However, by using a client's buffer, the necessary bandwidth can be reduced compared with that under the simple method.

2.3 Formulation

In this subsection, we formulate the problem of bandwidth reduction in selective contents broadcasting. Variables for the formulation are shown in Table 1. The following equations are established.

$$b(i)d_b(i) = rd_p(i) \tag{1}$$

$$t_{bf}(i) = t_{bs}(i) + d_b(i) \tag{2}$$

$$t_{pf}(i) = t_{ps}(i) + d_p(i) \tag{3}$$

This is an optimization problem to minimize the bandwidth used to broadcast data between times t_s and t_f. To play the data without interruption, clients have to finish receiving the data before it is finished playing. This is the constraint of the problem. As a result, The problem is formulated as follows:

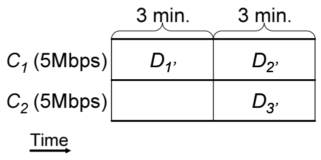

Figure 4: A broadcast schedule under the simple method.

Table 1: Variables for formulation

Symbol	Explanation
n	The number of states
r	The data consumption rate
$d_p(i)$	The time needed to play D_i
$b(i)$	The bandwidth for broadcasting D_i
$d_b(i)$	The time needed to broadcast D_i
$t_{bs}(i)$	The time to start broadcasting D_i
$t_{bf}(i)$	The time to finish broadcasting D_i
$t_{ps}(i)$	The time to start playing D_i
$t_{pf}(i)$	The time to finish playing D_i
$t_s(i)$	$\min_{i=1,\cdots,n}(t_{bs}(i))$, The time to start broadcasting
$t_f(i)$	$\min_{i=1,\cdots,n}(t_{bf}(i))$, The time to finish broadcasting
$I(t)$	$\{i \mid t_{bs}(i) \leq t < t_{bf}(i)\}$, The set of data number i that is broadcast at time t

$$\text{minimize} \quad \max_{t_s \leq t < t_f} \left(\sum_{i \in I(t)} b(i) \right)$$
$$\text{subject to} \quad t_{bf}(i) \leq t_{pf}(i).$$

When broadcasting a quiz program whose play-sequence graph is shown in Figure 3, $n = 3$, $r = 5$ Mbps, $d_p(1) = d_p(2) = d_p(3) = 3$ min. In the case of the simple method (Figure 4), $b(1) = b(2) = b(3) = 5$ Mbps, $t_{bf}(2) = t_{bf}(3) = t_{bf}(1)+3$, $t_{pf}(2) = t_{pf}(3) = t_{pf}(1)+3$, $t_{bf} = t_{pf}$. Although this satisfies the constraint, the necessary bandwidth, $\max(\sum b(i)) = 10$ Mbps, can be reduced.

To solve the problem, we have to produce a broadcast schedule and determine $b(i)$ and $t_{bs}(i)$ in order that the necessary bandwidth is effectively reduced. Since the combination of these variables is infinity, finding the optimal schedule is difficult. However, when $t_{bf} = t_{pf}$ is established, the bandwidth is effectively reduced. This is derived from Equations 1, 2 and 3. From these equations, $b(i) \propto t_{pf}$ is derived. Hence, $b(i)$ is reduced by reducing t_{pf}. Since $t_{bf}(i) \leq t_{pf}(i)$, the minimum value of t_{pf} is t_{bf}. Therefore, when $t_{bf} = t_{pf}$, the bandwidth is effectively reduced.

In this paper, we propose a scheduling method based on the above idea.

2.4 Basic Idea

A broadcast schedule under our proposed method is shown in Figure 5. The bandwidth for C_1 is 5 Mbps and that for C_2 is 2.5 Mbps. $D_{3'}$ is broadcast via C_2. Since the playing time of $D_{3'}$ is 3 min. and the consumption rate is 5 Mbps, $D_{3'}$ takes 6 min. to be broadcast. The client starts receiving $D_{3'}$ via C_2 when $D_{1'}$ starts playing. After $D_{1'}$ has played, if the client selects answer B, $D_{3'}$, which is stored in the client's buffer, starts playing. In this way, clients can play $D_{3'}$ without interruption after $D_{1'}$ has played. The necessary bandwidth is 7.5 Mbps. This is 25% smaller than that under the simple method. In this way, our proposed method reduces the necessary bandwidth.

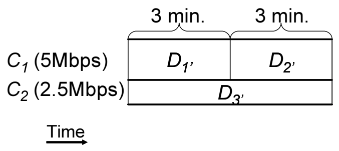

Figure 5: A broadcast schedules under our proposed method.

3 Proposed Method

We propose a scheduling method in selective contents broadcasting called the "CCB (Cumulated Contents Broadcasting)" method. The CCB method reduces the necessary bandwidth to play data without interruption by using play-sequence graphs, as explained in Subsection 2.1. The name derives from broadcast schedules under the CCB method that look like cumulated contents.

3.1 Assumed System Environment

Our assumed system environment is listed below.

- The broadcast data is a selective content.
- The server can concurrently broadcast data via multiple channels.
- Clients can concurrently receive data from multiple channels.
- Once clients start playing the data, clients can play the data without interruption.
- Clients have sufficient buffer capacity to store the received data.

Due to the recent popularization of digital broadcasting, interactive contents such as selective contents have attracted great attention. In the case of physical channels, the number of available channels has an upper limit. However, in this paper, we assume that the channels are logical channels, and we do not set a limit on the number of channels. In digital broadcasting, clients can receive broadcast data from multiple logical channels almost concurrently. Practical examples are broadcasting quiz or news programs via ground wave or satellite digital broadcasting. Such general broadcasting systems have approximately 20 Mbps bandwidth.

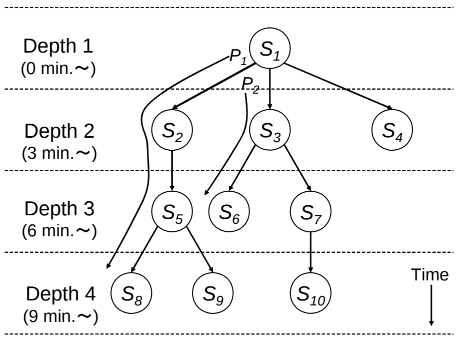

Figure 6: A play-sequence graph transformed to a multiway tree.

3.2 Scheduling Process

The scheduling process under the CCB method is divided into two stages: transformation of the play-sequence graph and production of the broadcast schedule.

3.2.1 Transformation of Play-Sequence Graph

In many cases, play-sequence graphs have network structures. However, we transform play-sequence graphs to multiway tree structures. By the transformation, we can produce a broadcast schedule that can reduce the necessary bandwidth consistently.

By applying operations explained in Subsection 2.1, all play-sequence graphs can be transformed to multiway trees. An example of a transformation to a multiway tree is shown in Figure 6. The playing times for nodes are equivalent. For example, if the playing time of the content for S_1 is 3 min., contents in the second depth, S_2, S_3, and S_4 are played between 3 and 6 min. Contents in the third depth are played between 6 and 9 min. In this way, by splitting nodes by the minimum playing time in all nodes, we can synchronize the time to start playing each content datum.

3.2.2 Production of Broadcast Schedule

Let n be the number of states, the maximum depth is b, the playing time of each content is d, and the consumption rate is r. D_i $(i = 1, \cdots, n)$ is the content data played in state S_i. B_j $(j = 1, \cdots, b)$ is the set of content data that are played in the j^{th} depth. For example, in the case of Figure 6, $n = 10$, $b = 4$, $d = 3$ min. Routes from the root to a leaf are indicated by P_k $(k = 1, \cdots, p)$, and the number of routes is p. p_k is the number of nodes included in P_k. For example, in Figure 6, $p_1 = 4$ when $k = 1$, and $p_2 = 3$ when $k = 2$. For ease of viewing the figure, some P_k are not shown.

First, the server selects the main route, P_M, from P_k. Contents included in P_M are broadcast via the first channel. When p_M is larger, the bandwidth can be reduced further. However, clients that select P_k $(p_k < p_M)$ may need to wait until the next program is broadcast. The server uses $m = \lceil n/p_M \rceil$ channels, $C_1, \cdots, C_m$, and sets the bandwidth for $C_1, \cdots, C_{m-1}$ to r. The bandwidth for C_m is explained later.

Since broadcast channels are logical channels, the server can control bandwidth for each channel. The scheduling process continues as follows. Here, i is the target channel number for scheduling, j is the time slot number from the time to start broadcasting, k is the number of scheduled content data in the target channel. R_j is a set of content data that are included in B_j, and they have not been placed in the broadcast schedule. r_j is the number of content data included in R_j. The scheduling process continues as follows.

1. Schedule $\{D_l | S_l \in P_M\}$ from the root to the leaf for C_1.
2. $i = 2$
3. $j = 2, k = 0$
4. Schedule $\min(p_M - k, j - k, r_j)$ contents included in R_j for C_i. A smaller content data number in R_j is scheduled earlier. Add the number of scheduled content data to k.
5. If $j < p_M$, increment j and go to process 4. Otherwise, increment i.
6. If $i \leq m$, go to process 3. Otherwise, finish scheduling.

In process 4, although a smaller content data number in R_j is scheduled earlier, the order does not influence system performance.

The bandwidth of C_m, b_m, is calculated as follows. Here, i, j, and k are determined as follows. The number of content data scheduled in C_m is $k = n - (m-1)p_M$. j is the depth of i th content data scheduled in C_m.

$$b_m = \max_{i=1,\cdots,k} \left(\frac{i}{j} r \right) \tag{4}$$

To play the data without interruption, clients have to finish receiving the data before the data is finishing playing. In the proposed method, since the number of scheduled content data is given by $\min(p_M - k, j - k, r_j)$ in process 4, clients can play the data without interruption.

For example, in the case of shown in Figure 6, first, R_1=$\{D_1\}$, R_2=$\{D_2, D_3, D_4\}$, R_3=$\{D_5, D_6, D_7\}$, and R_4=$\{D_8, D_9, D_{10}\}$. When the main route is P_1, since $p_1 = 4$, the number of necessary channels is $m = 3$. In process 1, D_1, D_2, D_5, and D_8 are scheduled for C_1 sequentially. Here, R_1 becomes empty, R_2=$\{D_3, D_4\}$, R_3=$\{D_6, D_7\}$, and R_4=$\{D_9, D_{10}\}$. Next, in process 4, $\min(4-0, 2-0, 2) = 2$ content data (D_3 and D_4) are scheduled for C_2. R_2 becomes empty and $k = 2$. In process 5, since $j < p_M$, increment j, and j becomes 3. In process 4, $\min(4-2, 3-2, 2) = 1$ content data (D_6) is scheduled for C_2. Here, R_3=$\{D_7\}$, $k = 3$. In process 5, since $j < p_M$, increment j, and j becomes 4. In process 4, $\min(4-3, 4-3, 2) = 1$ content datum (D_9) is scheduled for C_2. In process 5, since $j < p_M$ is not satisfied, increment i, and i becomes 3. In process 6, since $i \leq m$, repeat process 3. In the repetition of process 4, since the number of content data in R_2 is 0, no content data

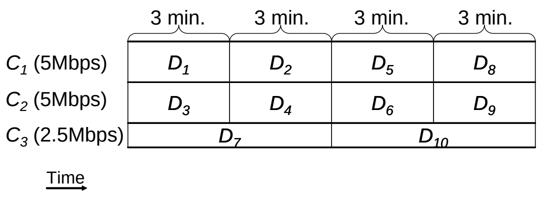

Figure 7: A broadcast schedule under the CCB method ($M = 1$).

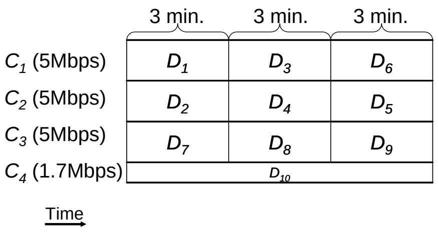

Figure 8: A broadcast schedule under the CCB method ($M = 2$).

is scheduled. In process 5, since $j < p_M$, increment j, and j becomes 3. In process 4, $\min(4-0, 3-0, 1) = 1$ content data (D_7) is schedule for C_3. In process 5, since $j < p_M$, increment j, and j becomes 4. In process 4, $\min(4-1, 4-1, 1) = 1$ content data (D_{10}) is scheduled for C_3. In process 5, since $j < p_M$ is not satisfied, increment i, and i becomes 4. In process 6, since $i \leq m$ is not satisfied, the scheduling process is finished. The bandwidth for C_m, $b_m = \max(\frac{1}{3} \times 5, \frac{2}{4} \times 5) = 2.5$ Mbps. Finally, the broadcast schedule is produced, as shown in Figure 7. In addition, when the main sequence is P_2, the broadcast schedule is produced, as shown in Figure 8.

3.3 Analysis

Here, we analyze our proposed method. The necessary bandwidth is given by the following equation.

$$(m-1)r + b_m \tag{5}$$

The time needed to broadcast the program is $d \times p_M$. When all probabilities of selecting a route P_k are equivalent, the waiting time until the next program is

$$\frac{1}{p} \sum_{p_k < p_M} (p_M - p_k) \,. \tag{6}$$

Note that, we assume that the waiting time is zero if the next program is broadcast before the previous program is finished playing.

4 Evaluations

In this section, we describe an evaluation of our proposed method. For evaluations, We use the play-sequence graph that is a multiway tree in which node has e branches. That is, $B_i = e^{i-1}$. Since conventional methods such as PHB-PP

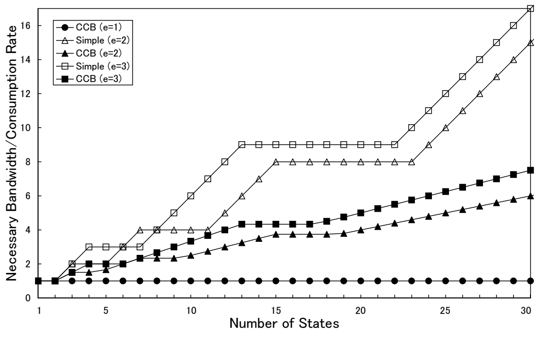

Figure 9: The necessary bandwidth under the CCB method and the simple method.

and MTB (described in Subsection 5.2) do not consider selective contents, they cannot be originally applied for selective contents broadcasting. Therefore, we compare our proposed method with the simple method.

4.1 Necessary Bandwidth

When a server broadcasts selective contents, the play-sequence graph is supposed to be decided considering the necessary bandwidth. Hence, we calculated the necessary bandwidth under the CCB method to play the data without interruption. We also calculated the necessary bandwidth under the simple method. The results are shown in Figure 9. The horizontal axis indicates the number of states. The vertical axis indicates the necessary bandwidth divided by the consumption rate since the necessary bandwidth is proportional to the consumption rate. “CCB (e=i)” ($i = 1, 2, 3$) is the CCB method in the case where the number of outgoing branches from a node is i. “Simple (e=i)” is the simple method. Since the necessary bandwidth in the case of $i = 1$ is always 1.0, and that is the same as that for CCB (e=1), the graph of that case is not shown in the figure. In the CCB method, P_M is selected so that $p_M = b$. In Figure 9, we can see that our proposed CCB method reduces the necessary bandwidth compared with that under the simple method. This is because the CCB method produces an effective broadcast schedule considering the sequence to play contents. For example, in the case where the program presents three quizzes ($e = 2$, $n = 15$) and the consumption rate is 5 Mbps, the necessary bandwidth under the CCB method is 18.75 Mbps. Since the necessary bandwidth under the simple method is 40 Mbps, we can say that our proposed method reduces the necessary bandwidth 53% shorter than the simple method. This is practical because general digital broadcasting systems have at least 20 Mbps.

4.2 Maximum Buffer Size

In the CCB method, by storing the received contents into clients' buffers, clients can play the data without interruption. To investigate the necessary buffer size, we simulated the maximum buffer size that clients use. The results are shown in Figure 10. The horizontal axis indicates the number

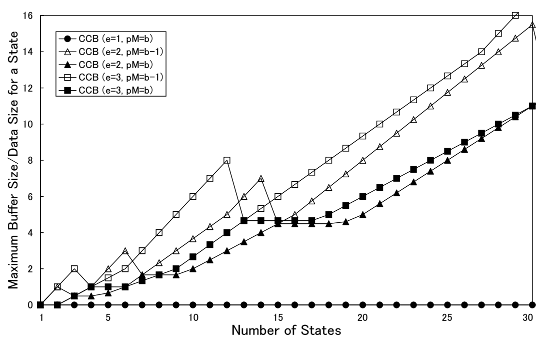

Figure 10: The maximum buffer size under the CCB method.

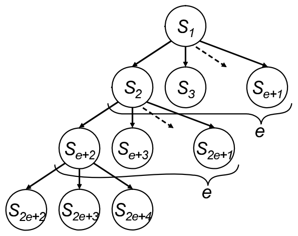

Figure 11: A play-sequence graph for evaluation.

of states and the vertical axis indicates the maximum buffer size divided by the data size. "CCB (e=i, pM=j)" ($j = b, b-1$) is the CCB method in the case of $p_M = j$. Since the play-sequence graph is a multiway tree, j becomes b or $b-1$. In the simple method, since clients do not have to store data into their buffers, the maximum buffer size is not shown in the figure. In Figure 10, we can see that the maximum buffer size under $p_M = b$ is shorter than that under $p_M = b-1$. This is because a larger p_M needs a larger bandwidth and the speed at which broadcast data is received becomes fast. For example, in the aforesaid case ($e = 2$, $n = 15$), if the consumption rate is 5 Mbps, the necessary buffer size under the CCB method becomes 506 Mbytes. However, this is practical because recent general set-top-boxes, which are clients for digital broadcasting, have at least a 60 Gbytes hard disk.

4.3 Influence of Main Route

To investigate the influence of the main route P_M, we evaluate our proposed method by setting p_M, the number of states in P_M, as a parameter. We use the play-sequence graph which is $B_1 = 1, B_j = e\ (j \neq 1, b), B_b = n-(b-1)e$ to set p_M as a parameter. That is, the graph has e nodes in each depth and one of them has e outgoing branches, as shown in Figure 11. The number of states, n, is 30 for evaluations.

4.3.1 Necessary Bandwidth and Main Route

The necessary bandwidth and p_M are shown in Figure 12. The horizontal axis indicates p_M and the vertical axis indicates the necessary bandwidth divided by the consumption rate. Since the maximum depth of $e = 4$ is $\lceil(30-1)/4\rceil+1 = 9$, the graph is depicted until $p_M = 9$ in that case. In Figure

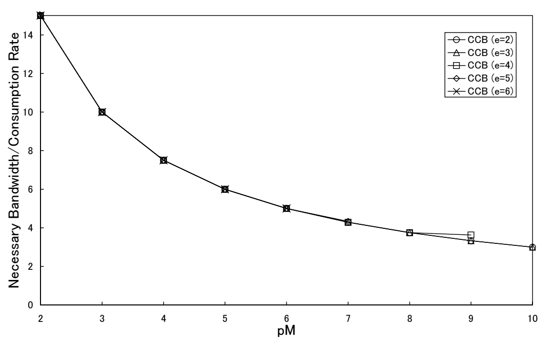

Figure 12: The play-sequence graph and the necessary bandwidth.

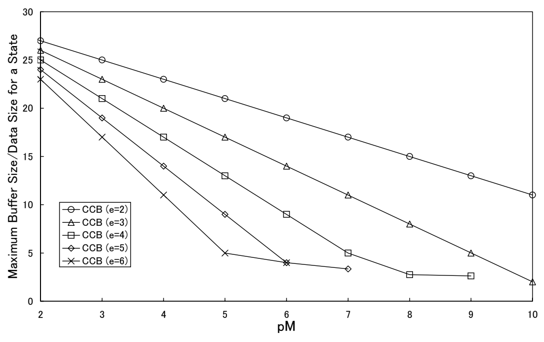

Figure 13: The play-sequence graph and the maximum buffer size.

12, we can see that a larger p_M reduces the necessary bandwidth further. This is because the time needed to broadcast the program becomes long as p_M increases. Accordingly, the server can use a long time to broadcast the data and the necessary bandwidth becomes smaller.

In addition, e does not influence the necessary bandwidth. This is because the data size for broadcasting data does not change, even if e changes, because n is fixed. However, in the case where p_M contents are not scheduled for a channel, the necessary bandwidth increases. For example, when $p_M = 9$ under CCB (e=4), since only eight content data are scheduled for C_2, the necessary bandwidth increases compared with other cases. If the consumption rate is 5 Mbps, the necessary bandwidth is 18.1 Mbps when $e = 4$. The necessary bandwidth when $e = 3$ is 16.7 Mbps. This is 8% larger than that when $e = 4$.

4.3.2 Maximum Buffer Size and Main Route

The influence of the main route on the maximum buffer size is shown in Figure 13. The horizontal axis indicates p_M, the vertical axis indicates the maximum buffer size divided by the data size. Since the time to broadcast the data becomes long as p_M increases, clients do not have to store the data in their buffers for a long time. Therefore, a large p_M gives a smaller maximum buffer size. When p_M becomes larger than the depth of the play-sequence graph, since the produced broadcast schedule does not change significanly, the maxi-

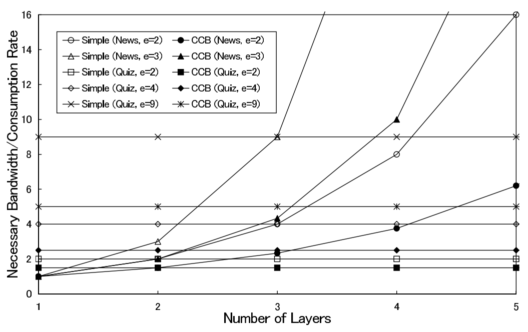

Figure 14: The necessary bandwidth for a news and a quiz program.

mum buffer size decreases only a little.

4.4 Application

As application examples, suppose the case of broadcasting a quiz program or a news program. In the quiz program, first, clients watch the quiz (S_1 in Figure 3). After that, they select the answer from e potential answers and watch the corresponding content (in the case where $e = 2$, S_2 or S_3 in Figure 3). The program repeats the above process according to the number of quizzes. This is a general pattern for a quiz program, and we suppose that the model is practical. In the news program, first, clients watch the overviews for each category such as sports, news, and business news. Overviews include overviews for e categories. Clients select their preferred category from them. Each category's overview includes overviews for more specific categories such as baseball news and basketball news. In these play-sequence graphs, since the depths of all leaves are equivalent, the waiting times are always 0.

The necessary bandwidth in above cases is shown in Figure 14. The horizontal axis indicates the number of layers, that is, the number of quizzes for a quiz program and the number of overviews for a news program. "Simple (News, e=i)" ($i = 2, 3$) is the necessary bandwidth for a news program under the simple method when $e = i$. "Simple (Quiz, e=i)" ($i = 2, 4, 9$) is that for a quiz program when $e = i$. "CCB (News/Quiz, e=i)" is that under the proposed method. In this figure, we can see that the necessary bandwidth increases as the number of overviews in a news program increases. This is because, as the number of overviews increases, since the branches of the play-sequence graph increases, the server has to broadcast those overviews within the broadcast time. On the other hand, in the quiz program, the necessary bandwidth does not change even when the number of quizzes increases. This is because the time to broadcast the data is lengthened as the number of quizzes increases. By using this lengthened time, the server can broadcast the increased quiz. For example, in the case of broadcasting the quiz program (5 Mbps) using 23 Mbps (digital broadcasting system), the server can broadcast a quiz program that has four potential answers under the simple method. In our proposed method, the server can broadcast a quiz program that has eight potential answers.

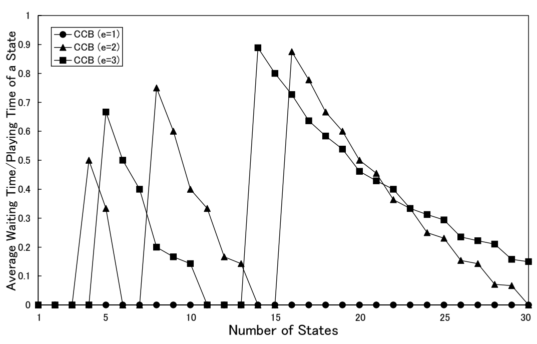

Figure 15: The average waiting time under the CCB method.

5 Discussion

5.1 Effectiveness of Our Proposed Method

In Figure 9, clearly, the necessary bandwidth under the CCB method is smaller than that under the simple method. On the other hand, our proposed CCB method needs clients' buffers to store the data, as shown in Figure 10. In this way, our proposed method reduces the necessary bandwidth by using clients' buffers.

5.1.1 Waiting Time to Next Program

When p_M is larger, the bandwidth can be reduced further. However, clients that select P_k ($p_k < p_M$) may wait until the next program is broadcast. We investigated the waiting time. Since the waiting time is not a main scope of the paper, we discuss it here. The results are shown in Figure 15. For the evaluation, we used the play-sequence graph shown in Figure 11. The horizontal axis indicates the number of states and the vertical axis indicates the waiting time. The waiting time is divided by the playing time of the content because the waiting time is proportional to the playing time. Since P_M is selected so that $p_M = b$, the time needed to broadcast the program is equivalent to that of the simple method. Therefore, since the waiting time under the simple method is the same as that under the CCB method, the waiting time under the simple method is not shown in the figure. In Figure 15, we can see that the waiting time changes cyclically. This is because the waiting time does not occur when the play-sequence graph is a balanced tree. For example, in the case of CCB (e=2), the waiting time does not occur when the number of states is 6, 7, 14, or 15. In the case where the playing time of a quiz is 3 min., clients who select incorrect answers have to wait 2.6 min. on average ($e = 2, n = 16$).

5.1.2 Waiting Time and Main Route

We evaluated the influence on p_M on the waiting time. The results are shown in Figure 16. For the evaluation, we used the play-sequence graph shown in Figure 11. The horizontal axis indicates p_M and the vertical axis indicates the average waiting time divided by the playing time. From this figure, we can see that a larger p_M gives a longer waiting time. This

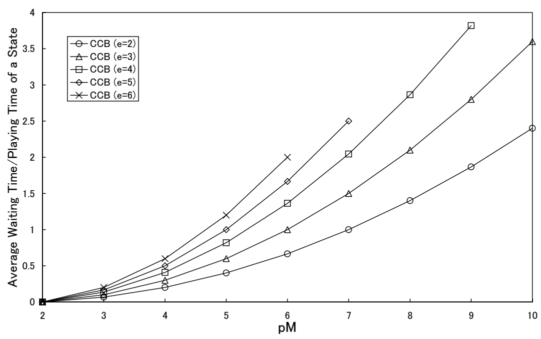

Figure 16: The play-sequence graph and the average waiting time.

is because when p_M is larger, the time needed to broadcast the data becomes longer. In addition, we can see that larger e increases the average waiting time. This is because when the play-sequence graph is given by Figure 11, a larger e gives a larger probability that $p_k < p_M$. For example, in the case where the playing time of contents is 3 min., $e = 2$, and $p_M = 6$, the average waiting time is 2 min. We think this waiting time for starting next program is tolerable. However, the system can control the waiting time by changing p_M.

5.2 Related Work

Several methods to reduce the necessary bandwidth in continuous media data broadcasting are proposed. In the Poly-Harmonic Broadcasting with Partial Preloading (PHB-PP) method[5], clients prefetch portions of the broadcast data to enable the start of the data playing without waiting. In the PHB-PP method, by dividing the data which is not prefetched, into several segments and broadcasting them, the necessary bandwidth to play the data continuously is reduced because clients can receive the subsequent data while the prefetched data is played. The Mayan Temple Broadcasting method[5], was proposed to reduce the necessary bandwidth using fewer channels than the PHB-PP method uses.

Although these methods prefetch the data, there are several methods to reduce the waiting time without prefetching. In the Harmonic Broadcasting [2] method, by dividing the data into several segments of equal sizes and broadcasting them, the waiting time is reduced. By broadcasting divided data repetitively via each channel, the waiting time can be reduced

The Pagoda Broadcasting [4] method , and the Fuzzycast [1] method also reduce the waiting time by dividing the data into equal sizes. The Pyramid Broadcasting [6] method, and the Asynchronous Harmonic Broadcasting [9] methods reduce the waiting time by dividing the data into different sizes.

We have proposed scheduling methods to reduce the waiting time for continuous media data broadcasting[7]–[9]. These methods use a near-video-on-demand technique, i.e., reducing the waiting time by broadcasting the data repetitively. In this paper, we assume that the server does not broadcast the data repetitively. The method focuses on reducing the necessary bandwidth. In addition, our assumed data is selective contents.

6 Conclusion

We proposed a method to reduce the necessary bandwidth in selective contents broadcasting. In selective contents broadcasting, users watch their desired contents. Our proposed CCB method reduces the necessary bandwidth by effectively scheduling contents considering the sequence to play the contents. In the CCB method, sequences to play contents are described using the state transition graph, which is called the play-sequence graph. By using play-sequence graphs, the CCB method reduces the necessary bandwidth effectively. In the case where the server broadcasts a quiz program that includes three questions, our proposed method reduces the necessary bandwidth 53% smaller than a simple method.

In the future, we will propose a method considering available channels and a method to reduce the waiting time.

REFERENCES

[1] R. Janakiraman and M. Waldvogel: “Fuzzycast: Efficient Video-on-Demand over Multicast,” *Proc. IEEE INFOCOM ’02*, pp. 920–929, 2002.

[2] L.-S. Juhn and L.M. Tseng: “Harmonic broadcasting for video-on-demand service,” *IEEE Trans. Broadcasting*, Vol. 43, No. 3, pp. 268–271, 1997.

[3] “MPEG Home Page,” http://www.chiariglione.org/mpeg/.

[4] J.-F. Paris: “A Simple Low-Bandwidth Broadcasting Protocol for Video-on-Demand,” *Proc. Int. Conf. on Computer Communications and Networks* (*IC3N ’99*), pp. 118–123, 1999.

[5] J.-F. Paris, D. D. E. Long, and P. E. Mantey: “Zero-delay broadcasting protocols for video-on-demand,” *Proc. ACM Multimedia’99*, pp. 189–197, 1999.

[6] S. Viswanathan and T. Imielinski: “Pyramid broadcasting for video on demand service,” *Proc. SPIE Multimedia Computing and Networking Conf.* (*MMCN ’95*), pp. 66–77, 1995.

[7] T. Yoshihisa, M. Tsukamoto, and S. Nishio: “A Scheduling Scheme for Continuous Media Data Broadcasting with a Single Channel,” *IEEE Transactions on Broadcasting*, Vol. 52, Issue 1, pp. 1-10, 2006.

[8] T. Yoshihisa, M. Tsukamoto, and S. Nishio: “A Scheduling Scheme for Continuous Media Data Broadcasting with Prefetching,” *Proc. IEEE Pacific Rim Conference Communications, Computers and Signal Processing* (*PACRIM’05*), pp. 145-148, 2005.

[9] T. Yoshihisa, M. Tsukamoto, and S. Nishio: “A Broadcasting Scheme for Continuous Media Data with Restrictions in Data Division,” *Proc. IPSJ International Conference on Mobile Computing and Ubiquitous Networking* (*ICMU’05*), pp. 90-95, 2005.

[10] Y. Zhao, D. L. Eager, and M. K. Vernon: “Scalable On-Demand Streaming of Non-Linear Media,” *Proc. IEEE INFOCOM* (2004).