A Broadcasting Protocol for Interactive Video-on-Demand Services

YU-WEI CHEN
Department of Computer and Information Science
Aletheia University
No. 32, Chen-Li Street, Tamsui, Taipei, 25103 Taiwan
REPUBLIC OF CHINA
http://www.cis.au.edu.tw

Abstract: This work presents a broadcasting protocol for interactive video-on-demand service. The proposed broadcasting allows the user to play the video-cassette-recorder (VCR) functions which includes the limited fast-forwarding capability. Consider a film \( S \) is divided into \( n \) equal parts \( \{S_1, S_2, \ldots, S_n\} \) and played through \( k \) channels. The user simultaneously downloads the film segments from \( k \) channels into the set-top-box (STB) and plays them in order. Every time the user wants to play VCR functions, the needed segments either saved in the user’s STB or are being continuously downloaded from \( k \) channels. The main character is that the server doesn’t need to allocate an extra stream to send the missing segments to the user. As a result, the proposed method outperforms the previous broadcasting methods without an extra stream in terms of the maximum waiting time, for the same number of channels.

Key-Words: Broadcasting, Interactive, Video-Cassette-Recorder, Video-on-Demand, Waiting Time.

1 Introduction
The so-called Video-on-Demand (VOD) system refers to the use of a combination of a television set and a set-top-box or computer, lined through the Internet to the film provider. The system allows the user to select which film to watch and use video-cassette-recorder (VCR) capabilities such as playback, forward, rewind, pause and others during viewing. One of the simplest ways to use the various VCR capabilities is for the provider to establish a channel dedicated to the user when a request to view a film is made. This VOD system is known as the true VOD system.

The main problem of the true VOD system is the need for an extremely large bandwidth for transmitting much information. Several researchers have proposed various solutions to reduce the need for bandwidth. They can be roughly categorized into three types for VOD systems – batching [1]-[3], stream tapping [4]-[6] or patching [7] and broadcasting [8]-[19].

All three VOD types sacrifice some VCR functions for saving the bandwidth of network. According to the definition of VOD service, the VCR function providing is also the important factor for consideration in design. Of course, the buffer size is also an important aspect when providing VCR capabilities; the bigger the buffer, the more segments of the film can be stored, implying the increased ability to support more VCR functions.

Of the three VOD types, broadcasting saves much more bandwidth than the other two. The broadcasting method works by first dividing the film into many segments, and then allowing the server to play them repeatedly over a few specified channels. Should a user wish to view the film, the maximum waiting time will be the length of one such segment. Different methods [13]-[19] are used to divide the film into different numbers of segments, so waiting times differs. Recently, Pâris et al. [20][21] and Chen [22] proposed “fixed-delay” broadcasting protocols. Should a user wish to view a film, the greatest waiting time will be the duration of \( m \) segments rather than one segment. The three methods [20]-[22] have shorter maximum waiting time than those in [13]-[19]. The user must also have sufficient buffer space to store the downloaded segments for continuous playback. This system is also called the near VOD system.

In recent years, an increasing number of experts and scholars have gone into this area of research [23]-[30], among which [27]-[30] use the broadcasting as a basis, proposing ways to support VCR functions more completely to achieve a more interactive VOD system. These methods make the same assumption that the capacity of harddisks will increase and prices will become more affordable, enough to support the space needed to save this type of interactive VOD services.
The broadcasting methods [27]-[30] with supporting VCR functions can be broadly categorized into two types: one with requiring an extra stream [28][29] and the other without requiring an extra stream [27][30]. References to these related research motivated the intention of our research: to propose a new broadcasting without requiring an extra stream for reducing the maximum waiting time.

This work presents a broadcasting protocol for interactive video-on-demand service. The proposed broadcasting allows the user to play the VCR functions which includes the limited fast-forwarding capability. Consider a film $S$ divided into $n$ equal parts $\{S_1, S_2, \ldots, S_n\}$ and played through $k$ channels. The user simultaneously downloads the film segments from $k$ channels into the set-top-box (STB) and plays them in order. Every time the user wants to play VCR functions, the needed segments either saved in the user’s STB or are being continuously downloaded from $k$ channels. The main character is that the server doesn’t need to allocate an extra stream to send the missing segments to the user. As a result, the proposed method outperforms the previous broadcasting methods [27][30] without requiring an extra stream in terms of the maximum waiting time, for the same number of channels.

This rest of this paper is organized as follows. Section 2 introduces preliminaries. Section 3 describes the proposed broadcasting protocol. Finally, Section 4 draws conclusions.

## 2 Preliminaries

### 2.1 Fixed-Length Segment-Scheduling Problem

Fast Broadcasting [15], Pagoda broadcasting [16][17], new Pagoda broadcasting [16] and Recursive frequency splitting broadcasting [18] are all problems of Fixed-Length Segment-Scheduling. In the fixed-length segment-scheduling problem, the film $S$ is always divided into $n$ equal parts $\{S_1, S_2, \ldots, S_n\}$ and played back through $k$ channels, the bandwidth of each of with is assumed just to suffice for the rate of use of the film during normal playback, and to allow the user can receive information from $k$ channels. Each segment transmission occupies a time slot whose duration is the period of one segment. Before segment $S_i$, the first $i-1$ segments $S_1, S_2, \ldots, S_{i-2}$ and $S_{i-1}$ must be watched, so segment $S_i$ must appear at least once every $i$ time slots to ensure proper playback without breaks. Accordingly, segment $S_i$ must use at least $1/i$ of the channel bandwidth. Under uninterrupted playback, the largest number of segments that can be divided is $n$, and the upper bound on $n$ satisfies

$$\frac{1}{1} + \frac{1}{2} + \ldots + \frac{1}{n} = k < \frac{1}{1} + \frac{1}{2} + \ldots + \frac{1}{n+1}.$$ 

In the broadcasting methods [13]-[22], if the harddisk already has the film segments downloaded on previous occasions, then of course the pause and rewind functions can work, but fast-forwarding will not occur since the remaining segments have not been completely downloaded. Next we shall introduce two types of interactive broadcasting methods: one with requiring an extra stream [28][29] and the other without requiring an extra stream [27][30].

### 2.2 Broadcasting with Requiring an Extra Stream

#### 2.2.1 Interactive Pagoda Broadcasting Scheme

This method [28] mainly uses the Pagoda broadcasting to achieve support for VCR functions. The method assumes the bulk of users have not watched the film yet, thus the frequency of commands for pause and rewind will be bigger than fast-forward. On the basis that storage mediums like harddisks are increasingly affordable, that the user already has enough temporary storage to store the whole film is also assumed. This then implies the user only has to read for their own STB instead of downloading from the server after activating pause and rewind functions. This means the server only uploads information when the user activates the fast-forward function. If the user needs to fast-forward, a request will be sent to the server; this request will let the server know which segment the user lacks and allocate an extra stream to send the missing segments to the user. Also, if the user is unable to receive the played segment in time after the fast-forward command has been sent to the server, the segment will be missed. Although this method has a shorter waiting time, extra bandwidth is still needed to support the specified VCR function like fast-forwarding. The fast-forwarding function ceases to be available when the bandwidth is fully utilized.

#### 2.2.2 Segment-Alignment Broadcasting Scheme

Chang [29] uses the Fast broadcasting to achieve support for VCR functions. On the same basis that storage mediums like harddisks are increasingly affordable, that the user already has enough temporary storage to store the whole film is also assumed. By the same way [28], the user reads their own STB after activating pause and rewind functions.
and only downloads information after activating the fast-forward function. If the user needs to fast-forward, a request will be sent to the server; this request will let the server know which segment the user lacks and allocate an extra stream to send the missing segments to the user.

2.3 Broadcasting without Requiring an Extra Stream

2.3.1 Providing VCR Functions with Active Buffer Management Scheme

Assume we want to play a film \( S \) of time length \( L \) through \( k \) channels \( \{C_0, C_1, \ldots, C_{k-1}\} \). In the initial time, stagger broadcasting protocol [8][9][27] let the whole film be repeatedly transmitted on \( C_0 \). After the time \( iL/k, \) \( 1 \leq i \leq k-1 \), the whole film will also be transmitted on \( C_i \) periodically. The maximum waiting time is \( L/k \). Fig. 1 shows an example of stagger broadcasting with four channels. We can observe from Fig. 1 that in any one time slot, all four segments of the film are playing over four separate channels.

Due to the number of segments being the same as channels in this broadcasting strategy and broadcasting in the same time slot, the buffer is not needed for skipping to and playing a certain part of the film. When user wants to skip from time \( t \) to film segment \( S_j \), the channel which the segment is playing on must be found. Only then can the jump VCR function be made. Thus we can make use of the equation provided by [27] to know segment \( S_j \) playing on channel \( C_i \) at time \( t \). The equation is as follows:

\[
C_i = \left\lfloor \left( \frac{(t - t_0)}{L/k} \right) \right\rfloor - j \mod k.
\]

2.3.2 Broadcasting Scheme with Supporting VCR Functions

Chen and Lee [30] proposed a broadcasting method for interactive VOD services. Assume dividing film \( S \) into \( n \) segments of equal length and playing through \( k \) channels. The user simultaneously downloads the film segments from \( k \) channels into STB and plays them in order. Chen and Lee’s method [30] allows the user to play the VCR functions which includes the limited fast-forwarding capability, the initial \( p \) segments cannot be fast-forwarded but the rest can support \( d \)-times fast-forwarding. They define \( n_i \) as the number of segments played in \( C_i \) as well as deduce the equation for \( n_i \) as follows:

\[
n_i = \left\lfloor p + \left( \frac{\sum_{j=0}^{i-1} n_j}{d} \right) \right\rfloor, \quad \log_2 p < i \leq k - 1.
\]

The number of all segments played in all \( k \) channels is as \( n = \sum_{i=0}^{k-1} n_i \). Fig. 2 shows an example with five channels that \( S_i \) cannot be fast-forwarded but the rest can support two-times fast-forwarding.

\[
\begin{array}{cccccccccc}
C_0 & S_0 & S_1 & S_1 & S_1 & S_1 & S_1 & S_1 & S_1 & \ldots \\
C_1 & S_2 & S_2 & S_2 & S_2 & S_2 & S_2 & S_2 & S_2 & \ldots \\
C_2 & S_3 & S_3 & S_3 & S_3 & S_3 & S_3 & S_3 & S_3 & \ldots \\
C_3 & S_4 & S_4 & S_4 & S_4 & S_4 & S_4 & S_4 & S_4 & \ldots \\
C_4 & S_5 & S_5 & S_5 & S_5 & S_5 & S_5 & S_5 & S_5 & \ldots \\
\end{array}
\]

\[
\begin{array}{cccccccccc}
t_0 & t_1 & t_2 & t_3 & t_4 & t_5 & t_6 & t_7 & t_8 & \ldots \\
\end{array}
\]

Fig. 2. \( S_1 \) cannot be fast-forwarded but the rest can support two-times fast-forwarding.

3 Proposed Broadcasting Protocol

In this section, we present a broadcasting method for interactive VOD services. Only given three parameters \( p, d \) and \( k \), our method can deduce out the “the initial \( p \) segments of film without fast forward but the rest segments supporting \( d \)-times fast forward over \( k \) channels.” The users can watch the complete film and play VCR functions.

The proposed broadcasting has the same circumstances as in [30]. The initial \( p \) segments in a film will not have fast-forwarding capabilities. This is a drawback of our method. However, with respect to films available on the market, commercials and statements of copyright will always appear in the initial parts. Of course, in business considerations, not wanting the front parts of the film to be skipped but to be watched is a reasonable assumption.
The basic concept of the proposed method is described as follows. First, we compute the frequency of appearance of each segment. Before segment $S_i$, the first $i-1$ segments $S_1, S_2, \ldots, S_{i-2}$, and $S_{i-1}$ must be watched, so segment $S_i$ must appear at least every $B(i)$, $B(i)=\lceil p+(i-p)/d \rceil$ if $p<i$, time slots to ensure proper playback without breaks. Accordingly, $S_i$ must use at least $1/B(i)$ of the channel bandwidth. Instead of each $S_i$, having to appear at least once every $i$ time slots, each $S_i$ must appear at least once every $B(i)$ time slots for $i \geq 1$. Then, the same concept with [18] is applied to schedule the segments, increasing the amount of segments. This way, the waiting time for the user correspondingly decreases.

To make our concepts clear, we shall use a small example to elaborate on the main idea of our proposed method. Consider a broadcasting protocol in which segment $S_1$ cannot be fast-forwarded but the rest can support two-times fast-forwarding. This paper uses the same definition of the slot sequence $SS(C_i, p, q)$ in [18]. The slot sequence $SS(C_i, p, q)$ is defined as an infinite sequence of time slots $[p, p+q, p+2q, \ldots]$ belonging to channel $C_i$ beginning at slot $p$, and repeating infinitely with a period of $q$ slots, where $C_i$ is one of the $k$ channels, $p \geq 0$ is an integer, and $q \geq 1$ is an integer, $0 \leq p \leq q-1$. Table 1 illustrates that each segment will be broadcast on which time slots. Fig. 3 shows the position of the film segments $\{S_1, S_2, S_3, \ldots, S_{12}\}$ in each channel when we use five channels to transmit a film.

**Table 1**

<table>
<thead>
<tr>
<th>$S_i$</th>
<th>$B(i)$</th>
<th>Selected time slots sequence</th>
<th>The rest available time slots sequences</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>$SS(C_0, 0, 1)$ for $0 \leq \ell \leq 4$</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>$SS(C_1, 0, 1)$ for $1 \leq \ell \leq 4$</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>$SS(C_0, 0, 1)$ for $2 \leq \ell \leq 4$</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>$SS(C_2, 0, 2)$</td>
<td>$SS(C_1, 1, 2), SS(C_2, 0, 1), SS(C_3, 1, 0), SS(C_4, 0, 1)$</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>$SS(C_2, 1, 2)$</td>
<td>$SS(C_3, 0, 1), SS(C_4, 0, 1)$</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>$SS(C_3, 0, 3)$</td>
<td>$SS(C_1, 1, 3), SS(C_2, 2, 3), SS(C_3, 0, 1)$</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>$SS(C_3, 1, 3)$</td>
<td>$SS(C_2, 2, 3), SS(C_4, 0, 1)$</td>
</tr>
<tr>
<td>7</td>
<td>4</td>
<td>$SS(C_4, 0, 4)$</td>
<td>$SS(C_1, 2, 3), SS(C_2, 1, 4), SS(C_3, 2, 4), SS(C_4, 3, 4)$</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>$SS(C_4, 1, 4)$</td>
<td>$SS(C_2, 3, 4), SS(C_4, 2, 4), SS(C_3, 3, 4)$</td>
</tr>
<tr>
<td>9</td>
<td>5</td>
<td>$SS(C_4, 2, 4)$</td>
<td>$SS(C_2, 2, 3), SS(C_4, 3, 4)$</td>
</tr>
<tr>
<td>10</td>
<td>5</td>
<td>SS($C_4, 3, 4$)</td>
<td>SS($C_2, 2, 3)$</td>
</tr>
</tbody>
</table>

Fig. 3. The proposed broadcasting method with five channels of where $S_1$ cannot be fast-forwarded but the rest can support two-times fast-forwarding.

The proposed method can be implemented using the following steps.

**Inputs**: three positive integers $p, d$ and $k$ where initial $p$ segments cannot be fast-forwarded, but the rest can support $d$-times fast-forwarding, over $k$ available channels.

**Outputs**: the largest number of segments that can be divided.

**Algorithm**

Step 1. Let $POOL = \{SS(C_i, 0, 1) | 0 \leq \ell \leq k-1\}$ denote the set of free channels. Let $i=1$.

Step 2. Calculate

\[ B(i) = \begin{cases} \frac{i}{d} + 1 = i & p < i \\ \frac{p+(i-p)/d}{d} & p \geq i \end{cases} \]

Step 3. Set $j = B(i)$. Segment $S_i$ must appear at least once per $j$ time slots.

Step 4. Select a slot sequence $SS(C_i, p, q)$ with the smallest value of $j \mod q$ from $POOL$ such that $q \leq j$. Let $POOL = POOL \setminus \{SS(C_i, p, q)\}$.

Step 5. Split $SS(C_i, p, q)$ into $\{SS(C_i, p+xq, aq) | 0 \leq x \leq a-1, a=\lceil j/q \rceil\}$. Segment $S_i$ is broadcasted on the slots in $SS(C_i, p, aq)$. Do the union $POOL = POOL \cup \{SS(C_i, p+xq, aq) | 1 \leq x \leq a-1\}$.

Step 6. If $POOL$ is not empty, then increase $i$ by one and go to Step 2. Otherwise, terminate this process and output the value of $i$.

As mentioned above, each segment $S_i$ must occupy at least $1/B(i)$ of bandwidth of each channel. Consequently, the biggest number of segments that
can be divided is \( n \), and the upper bound on \( n \) satisfies: \( \sum_{i=1}^{n} B(i) = k < \sum_{i=1}^{n+1} B(i+1) \) where \( B(i) = i \) if \( 1 \leq i \leq p \) and \( B(i) = \left\lfloor p + (i - p) / d \right\rfloor \) if \( p < i \).

Table 2 and 3 compare previous method [30], the proposed method and the upper bound in terms of the total segments. The proposed method clearly outperforms the previous method in terms of the longest waiting time.

**TABLE 2**
A Comparison of Total Numbers of Segments between [30] and Ours when \( S_1 \) to \( S_3 \) cannot Support Fast-Forwarding but the Rest can Support Two Times that

<table>
<thead>
<tr>
<th></th>
<th>( S_1 )</th>
<th>( S_1, S_2 )</th>
<th>( S_1, S_2 ) bound</th>
<th>( S_1, S_2, S_3, S_4 )</th>
<th>( S_1, S_2, S_3, S_4 ) bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>[30]</td>
<td>1 1 1</td>
<td>1 1 1</td>
<td>[30]</td>
<td>1 1 1</td>
<td>[30]</td>
</tr>
<tr>
<td>( S_1 )</td>
<td>2 2 2</td>
<td>2 2 2</td>
<td>3 3 3</td>
<td>3 3 3</td>
<td>3 3 3</td>
</tr>
<tr>
<td>( S_2 )</td>
<td>3 4 4</td>
<td>4 4 4</td>
<td>5 6 6</td>
<td>6 6 7</td>
<td>7 7 7</td>
</tr>
<tr>
<td>( S_3 )</td>
<td>4 7 7</td>
<td>7 7 7</td>
<td>10 11 13</td>
<td>11 13 16</td>
<td>12 12 18</td>
</tr>
<tr>
<td>( S_4 )</td>
<td>5 11 12 18</td>
<td>12 12 18 23</td>
<td>16 19 23</td>
<td>18 22 27</td>
<td>23 27 32</td>
</tr>
<tr>
<td>( S_5 )</td>
<td>6 17 20 31</td>
<td>20 20 31 40</td>
<td>25 31 40</td>
<td>30 35 50</td>
<td>40 40 60</td>
</tr>
<tr>
<td>( S_6 )</td>
<td>7 26 32 53</td>
<td>32 32 53 68</td>
<td>39 52 68</td>
<td>56 66 88</td>
<td>70 70 94</td>
</tr>
<tr>
<td>( S_7 )</td>
<td>8 40 53 89</td>
<td>53 53 89 113</td>
<td>60 89 113</td>
<td>85 107 135</td>
<td>113 113 146</td>
</tr>
<tr>
<td>( S_8 )</td>
<td>9 61 90 147</td>
<td>90 90 147 189</td>
<td>91 149 189</td>
<td>114 165 207</td>
<td>189 189 225</td>
</tr>
<tr>
<td>( S_9 )</td>
<td>10 92 150 244</td>
<td>150 150 244 313</td>
<td>138 246 313</td>
<td>171 291 347</td>
<td>313 313 375</td>
</tr>
</tbody>
</table>

**TABLE 3**
A Comparison of Total Numbers of Segments between [30] and Ours when \( S_3 \) to \( S_4 \) cannot Support Fast-Forwarding but the Rest can Support Two Times that

<table>
<thead>
<tr>
<th></th>
<th>( S_1, S_2, S_3 )</th>
<th>( S_1, S_2, S_3, S_4 )</th>
<th>( S_1, S_2, S_3, S_4 ) bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>[30]</td>
<td>1 1 1</td>
<td>[30]</td>
<td>1 1 1</td>
</tr>
<tr>
<td>( S_1 )</td>
<td>2 3 3</td>
<td>3 3 3</td>
<td>3 3 3</td>
</tr>
<tr>
<td>( S_2 )</td>
<td>3 6 7</td>
<td>7 7 9</td>
<td>7 7 9</td>
</tr>
<tr>
<td>( S_3 )</td>
<td>4 11 13</td>
<td>13 14 18</td>
<td>13 14 18</td>
</tr>
<tr>
<td>( S_4 )</td>
<td>5 18 23</td>
<td>23 28 32</td>
<td>23 28 32</td>
</tr>
<tr>
<td>( S_5 )</td>
<td>6 29 38</td>
<td>38 48 57</td>
<td>38 48 57</td>
</tr>
<tr>
<td>( S_6 )</td>
<td>7 45 68</td>
<td>68 82 97</td>
<td>68 82 97</td>
</tr>
<tr>
<td>( S_7 )</td>
<td>8 69 114</td>
<td>114 138 163</td>
<td>114 138 163</td>
</tr>
<tr>
<td>( S_8 )</td>
<td>9 105 195</td>
<td>195 230 272</td>
<td>195 230 272</td>
</tr>
<tr>
<td>( S_9 )</td>
<td>10 159 320</td>
<td>320 382 451</td>
<td>320 382 451</td>
</tr>
</tbody>
</table>

**4 Conclusion**
This work has presented a broadcasting protocol for interactive video-on-demand service. The broadcasting protocol allows the user to play the VCR functions which includes the limited fast-forwarded but the rest can support \( d \)-times fast-forwarding. The proposed method outperforms the previous broadcasting method without requiring an extra stream in terms of the maximum waiting time, for the same number of channels. In addition, the upper bound of the maximum number of cut segment was presented as well.

**Acknowledgement**
The authors would like to thank the National Science Council of the Republic of China for financially supporting this research under Contract No. NSC93-2213-E-156-003.

**References:**


