Available online at www.sciencedirect.com
ScienceDirect
Discrete Applied Mathematics 154 (2006) 2418–2429
DISCRETE APPLIED MATHEMATICS
www.elsevier.com/locate/dam

Generalized Fibonacci broadcasting: An efficient VOD scheme with user bandwidth limit☆

Mingjun Edward Yan, Tsunehiko Kameda*

School of Computing Science, Simon Fraser University, Burnaby, BC, Canada V5A 1S6

Received 30 May 2004; received in revised form 29 October 2004; accepted 21 September 2005
Available online 27 June 2006

Abstract

Broadcasting is attractive in delivering popular videos in video-on-demand service, because the server broadcast bandwidth is independent of the number of users. However, the required server bandwidth does depend on how much bandwidth each user can use, as well as on the user's initial waiting time. This paper addresses the issue of limiting the user bandwidth, and proposes a new broadcasting scheme, named *Generalized Fibonacci Broadcasting* (GFB). In terms of many performance graphs, we show that, for any given combination of the server bandwidth and user bandwidth, GFB can achieve the least waiting time among all the currently known *fixed-delay* broadcasting schemes. Furthermore, it is very easy to implement GFB. We also demonstrate that there is a trade-off between the user waiting time and the buffer requirement at the user.

Keywords: Multimedia; Video-on-demand; Video broadcasting; Bandwidth minimization

1. Introduction

Video-on-demand (VOD) services aim to deliver videos "instantly" to a large population of users over a high-speed network with broadcast capability. We use the term "user" to mean either a set-top box (STB) on a TV receiver or a computer that can receive a digital video, store it in some storage and concurrently play it from storage at the predefined display rate. The traditional client–server paradigm (so-called "pull technology") is not suitable for delivering "hot" or popular videos to a massive number of users, since it does not scale well. To address the scalability issue, many broadcasting schemes (based on "push technology") have been proposed since 1995, when the pioneering work [17] was published. (A little known work [3] with an almost identical idea was first published as a Philips Research Lab technical report in 1991.) The main idea is to broadcast a set of hot videos periodically, so that a viewer can tune in onto the particular video that s/he wants to watch. Such a scheme typically divides each video into a sequence of segments and broadcasts all the segments periodically on separate (logical) channels. While a viewer is watching the current video segment, it must be guaranteed that the next segment will be downloaded in time for continuous display (continuity condition). Most of the research literature focuses on minimizing the maximum user waiting time for given server bandwidth (or equivalently, minimizing the total server broadcast bandwidth for given maximum waiting time).

☆ This work was supported by a Discovery Grant from the Natural Sciences and Engineering Research Council of Canada.
* Corresponding author. Fax: +1 604 291 3045.
E-mail address: tiko@cs.sfu.ca (T. Kameda)
URL: http://www.cs.sfu.ca/~tiko/ (T. Kameda).

0166-218X/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.dam.2006.04.019

The bandwidth here means the sum of the transfer rates (Mbits/s) of the communications channels used concurrently. Clearly, the STB must be able to receive and process data arriving on several concurrent logical channels. The most bandwidth-efficient broadcasting schemes that are currently known require the same bandwidth on the user side as that on server side [5,7,11]. With the current technology, however, the user-side bandwidth (the disk speed, in particular) is likely to be relatively narrow [9], since the STB must be inexpensive. Of course, the server must transmit a number of popular videos concurrently, while a given user receives only one of them at a time. Many broadcasting schemes, such as Pyramid Broadcasting (PB) [17], Permutation-based Pyramid Broadcasting (PPB) [1], Harmonic Broadcasting [7], and Skyscraper Broadcasting (SB) [6], are based on the *fixed start points* policy. With this policy, the STB needs to wait until the beginning of the first segment appears on a channel, before it starts to download the video. More recent results on the performance limits of the broadcasting schemes based on the fixed start points policy can be found in [2,16].

The rest of the broadcasting schemes adopts the *fixed-delay* policy in contrast to the fixed start points policy. In such schemes, the user starts downloading immediately, but waits for a fixed amount of time before s/he starts viewing the video. We propose a new fixed-delay broadcasting scheme, named *Generalized Fibonacci Broadcasting* (GFB), to address the issue of limiting the user-side bandwidth. This scheme is a generalization of a scheme described in a paper by Hu [4]. For any given combination of server bandwidth and user bandwidth, GFB can always achieve the least user waiting time among all the currently known fixed-delay broadcasting schemes. Stated in another way, for any given combination of user bandwidth and waiting time, GFB requires the least server bandwidth.

Another great advantage of GFB is that it uses much fewer segments (hence logical channels) than most other schemes for the same combination of the server bandwidth, user bandwidth and waiting time. Moreover, the mapping from the segments to the logical channels is the simplest possible, i.e., one-to-one. All this implies that it would be very straightforward to implement GFB.

The rest of the paper is organized as follows. Section 2 reviews the currently known broadcasting schemes that take user bandwidth limit into consideration. Section 3 introduces GFB and computes its waiting time as a function of several parameters, such as the server bandwidth, the user bandwidth and the bandwidth of each channel. In Section 4, we will compare GFB with all other similar schemes currently known, and also discuss how its performance is affected as its parameter values are changed. We then derive in Section 5 a formula for computing the buffer size required at the STB. Finally, Section 6 summarizes the contributions of the paper with some remarks. A preliminary version of this paper containing some of the materials presented here appeared in [19,20].

1.1. Basic notation

For simplicity, we concentrate on broadcasting one video. Throughout the paper we use the following notation:

b: video display (or consumption) rate in Mbits/s;
D: total display duration of each video in seconds;
n: number of segments each video is divided into, which equals the number of channels;
S_i: ith segment, $i = 1, 2, \ldots, n$;
D_i: duration of segment S_i in seconds. $\sum_{j=1}^{n} D_j = D$;
d_i: $= D_i/D$;
w: initial waiting time (or latency) in seconds from the time a request is made until display starts;
$w^* = w/D$: normalized wait time; $0 < w^* \leqslant 1$.
B: total server bandwidth required to broadcast one video, expressed as a multiple of b (The server bandwidth will be expressed as either Bb or simply B.);
B_j: bandwidth of the jth broadcast channel as a multiple of b, $j = 1, 2, \ldots, n$; $(B = \sum_{j=1}^{n} B_j)$;
U: user bandwidth, expressed as a multiple of b.

2. Previous work

In many existing schemes, the user bandwidth is the same as that the server uses to broadcast a video. Thus, they are not interesting from our perspective (of limiting the user bandwidth) in this paper. Here we review some of the schemes in which the user bandwidth is smaller than the server bandwidth.