A HIGH PERFORMANCE VIDEO SERVER FOR KARAOKE SYSTEMS*

Wei-Hsin Tseng and Jau-Hsiung Huang
Communications and Multimedia Lab.
Department of Computer Science and Information Engineering
National Taiwan University, Taipei, Taiwan

Abstract

A high performance and low cost video server architecture is proposed for Karaoke systems which can store more than 200 Gbytes of video and support up to 30 concurrent users simultaneously. Moreover, a novel caching strategy, called *head caching*, is proposed which can significantly reduce the initial delay. With this strategy, the initial delay can be dramatically reduced without increasing the cache size. Besides, using *head caching* scheme, the proposed system can act as a general video server which can support fast-forward, rewind, and dynamic panning controls [1].

1. Introduction

In recent years, Karaoke TV (KTV) has become the most popular entertainment specially in Asian countries such as Japan, Taiwan, and Korea and has created a huge market value. People can order songs from a KTV server then watch the video and sing with the melody. Typically, the interval of video for a song is about 4 minutes and there is no need to support fast forward and rewind control. Besides, a KTV shop must store a large volume of videos (e.g. 5000 songs) to serve customers. The bandwidth of a compressed video using MPEG [2] is about 150 Kbytes/second such that the storage requirement for a typical KTV shop with 5000 songs (4 minutes per song) is about 180 Gbytes. It is impossible to use a large disk array to store such a volume of data using current equipments since the cost and power consumption are both too high. For example, if the capacity of a hard-disk is 2 Gbytes, 90 disk drives are needed to hold these songs.

In contrast, compact disk (CD) jukebox provides a good solution for large volume of data. For 5000 songs, only 300 CDs are needed since a CD can store up to 600 Mbytes of data. Even for 2000 movies (2 hour per movie), only 4000 CDs are needed. The cost is still reasonable and no extra power is needed. Nevertheless, such system also has two serious drawbacks as long disk swap time and low transfer rate. However, by using a smart caching strategy, the above drawbacks can be resolved as will be presented in this paper.

In this paper, the system architecture is described in section 2. The buffer size and initial delay will be discussed in section 3. Section 4 shows the improvements by using different caching strategies. Section 5 describes a general video server using the proposed system. Finally, conclusion is provided in section 6.

2. System Overview

The proposed system consists of a PC based hardware platform and control software. To achieve high performance and low cost requirements, the architecture, system bus, and peripheral channel interface must be carefully chosen.

2.1 Hardware Architecture

Figure 1 shows the architecture of the proposed system. A small disk array is used as the cache of CD players and the RAM in the host (PC-486) is the cache of the disk array [3,4]. Video data is first transferred from a CD to a hard disk by the extended controller and then from the hard disk to RAM by the host. There is an optional command called *copy* specified in SCSI-2 standard. By using *copy* command, the initiator (the host) can move data from the source device (a CD player) to the destination device (a hard disk) without handling the data stream. When the destination device gets a *copy* command from the host, it will act as an initiator to issue a *read* command to retrieve data from the source device. Without this command, the host must issue a *read* command first to retrieve data from a CD player and then issue a *write* command to store data to the hard disk. It will double the bandwidth and waste

* This work is supported by The National Science Council of R.O.C. under grants NSC-83-0408-E-002-003, NSC83-0408-E-002-005, and NSC-83-0425-002-140

Manuscript received June 27, 1994

CPU time of the host. Unfortunately, most hard disks do not support such an option.

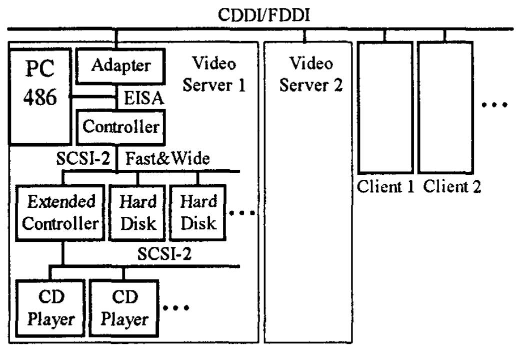


Figure 1. System Architecture of the proposed system.

In order to reduce the bandwidth and CPU load of the host, an *extended controller* is inserted between hard disks and CD players. The function of the *extended controller* is to move video data from CD player to the hard disk. In the worst case that no video is in the cache (hard-disk or host RAM), 30 CD players are needed in the system to serve 30 concurrent users. 30 CD players are too many to be implemented in the system and it must be reduced to a reasonable number by using high speed CD players.

The transfer rate of a high speed CD player can be as high as 4 to 8 times of 150 Kbytes/second such that it can serve several videos at a time. First, it pre-loads a block of video data to the hard disk then swaps the CDs to serve another video service. Meanwhile, the video data in the disk is transferred to RAM for playing. Before the data being completely consumed, the CD player swaps in the original CD and transfer another block of data to the disk.

Behind the CD players, there are mechanical arms to transport CDs between players and CD banks. The swapping time for CDs is relatively large. It consists of four steps to swap CDs. First, the CD player spins down and the arm moves the CD in the player to an empty slot in the CD bank. Then, the arm picks up the desired CD and puts it into the player. Finally, the player spins up for further reading. The spin-down time is less than one second and spin-up time is 1 to 2 seconds. The transportation time of the arm is about 2.5 seconds per direction. Therefore, the total swapping time of a CD is about 8 seconds for most of existing low-cost jukeboxes.

The same scheme is also applied between the disk array and the host RAM. The hard disk pre-loads a block of video data to the host RAM, then seeks to another track to transfer another video. Before the data being consumed, the hard disk will go back to serve it again.

2.2 Peripheral Channel Interface and System Bus

For 30 concurrent videos, the bandwidth is about 4.5 Mbytes/second. In the proposed system, video data is moved from a CD player to a hard disk first, and then moved from the hard disk to the host RAM. The bandwidth of the peripheral channel is hence doubled. That is, the channel capacity must be greater than 9 Mbytes/second. Taking protocol overheads into consideration, the bandwidth requirement is even larger than this value. Hence, high speed peripheral channel interface must be chosen. SCSI-2-Fast is a prevalent peripheral interface whose bandwidth can be up to 10 Mbytes/second. Unfortunately, the bandwidth is slightly lower than that required in the proposed system. Therefore, SCSI-2-Fast&Wide interface, whose speed is 20 Mbytes/second, is chosen in the proposed system.

The system bus in the host is another critical data path in the proposed system. The common ISA bus is not fast enough which can only transfer 1 to 2 Mbytes/second. So, EISA bus is chosen in this system to achieve higher throughput.

The throughput of a low-cost hard disk is about 1.8 to 3 Mbytes/second depending on the location of the read/write head. The seek time in the worst case is below 15 mini-seconds. The command processing time, track-to-track seek time, and head-to-head switching time are below 1 mini-second. Notice that the data must be carefully allocated to minimize the seek time because the seek time dominates the performance of a hard disk.

3. Buffer Size and Initial Delay

When multiple videos are served simultaneously, video buffer is needed for each video stream. Suppose there are N simultaneous video streams and the buffer size of each video stream is B, the total buffer size is computed according to different buffer management strategies. For the simplest method, a video stream contains two buffers, one for reading and the other for playing. The total buffer size needed by the server is $2NB$. If we divide the buffer into a couple of small blocks, the buffer can be treated as a circular queue. The block in the head is for reading and the tail for playing. No Ping-Pong buffer is needed in such a strategy and the total buffer size is reduced to NB. Furthermore, observing from the snap-shot on the buffer usage of video streams shown in Figure 2, we can find some interesting and trivial facts. The buffer which has just been served by the server is almost full and the buffer which is going to be served is almost empty. The longer time after it is served, the less data stored in the buffer. We can conclude that half of the buffers are used in any instant of time. Therefore, if the buffer is well managed,

$$S = \frac{NB}{2} \quad (1)$$

where S is the optimal total buffer size.

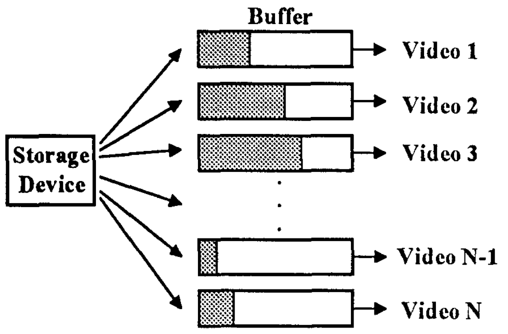


Figure 2. Buffer usage for N simultaneous videos.

The latency from a video is ordered until it begins to be played is called the *initial delay*. It may significantly degrade the service quality if it is too long. In the following, we define some notation for analysis.

R_S	≡	transfer rate of the storage device
R_V	≡	bandwidth of a video stream
D	≡	maximum seek time (it may be the seek time of a hard disk or swap time of the CD player)
T	≡	average initial delay
T_u	≡	maximum initial delay

We can see that

$$T = \frac{NB}{2R_S} + \frac{ND}{2} \quad (2)$$

$$T_u = \frac{NB}{R_S} + ND \quad (3)$$

Clearly, both the total buffer size and initial delay is proportional to buffer size B. As B gets smaller, both the total buffer size and initial delay will be smaller. However, if B is too small, videos will be discontinuous since before the server comes back to load new data to the buffer, the previous data in the buffer has already been consumed and under-flow of video occurs in the buffer. Hence, the round-trip time to serve video streams using round-robin scheme must be less than the consuming time of each buffer. Therefore,

$$\frac{B}{R_V} > N(D+\frac{B}{R_S}) \quad (4)$$

$$\text{or } B > \frac{NDR_SR_V}{R_S-NR_V} \quad (5)$$

Hence, from (1), (2), (3) and (5),

$$S = \frac{N^2DR_SR_V}{2(R_S-NR_V)} \quad (6)$$

$$T = \frac{N^2DR_V}{2(R_S-NR_V)} + \frac{ND}{2} \quad (7)$$

$$T_u = \frac{N^2DR_V}{R_S-NR_V} + ND \quad (8)$$

This result can be applied in both CD players and hard disks. From this result, we can find that as N gets larger, both the buffer size and initial delay will increase significantly. If no cache is used, the initial delay is dominated by the CD swap time of CD players. Given D = 8 seconds, Figures 3 and 4 show the average initial delays and the buffer sizes per video stream (B) respectively when different number of videos are served by a 4-time speed CD player. It shows that, by using existing low cost equipment, 3 concurrent video streams can be served by a CD player with initial delay less than 48 seconds and total buffer size 20 Mbytes. Hence, if the server has to serve 30 users concurrently, then 10 CD players are required and the initial delay remains at 48 seconds while the total buffer size equals 200 Mbytes.

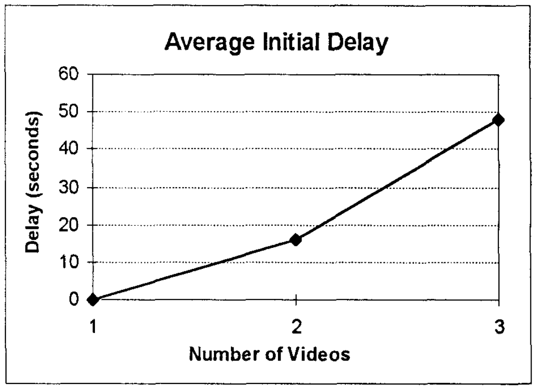


Figure 3. Initial delay caused by a CD player where D = 8 seconds and Rs = 600 Kbytes/second.

Since high speed CD player and jukebox are under developing, we may be able to find faster devices soon. Figures 5 and 6 show the average initial delays and the buffer sizes per video stream respectively when different number of videos are served by an 8-time speed CD player and a jukebox with 5-second swap time. It shows that 6 concurrent video streams can be served by a high speed CD player with initial delay 60 seconds and total buffer size 54 Mbytes. Again, if 30 users are served concurrently, the total buffer size equals 324 Mbytes. The initial delay, which is within one minute, may be still unacceptable. By caching the video data in advance, the initial delay can be reduced to few seconds as described in next section.

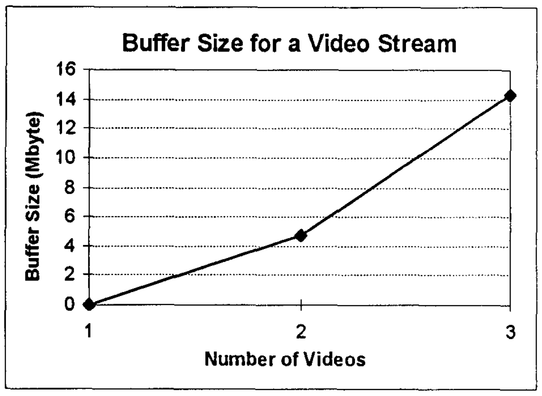


Figure 4. Buffer size required by a CD player where D = 8 seconds and Rs = 600 Kbytes/second.

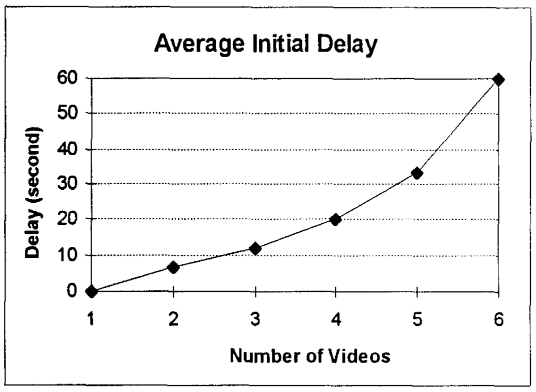


Figure 5. Initial delay caused by a CD player where D = 5 seconds and Rs = 1.2 Mbytes/second.

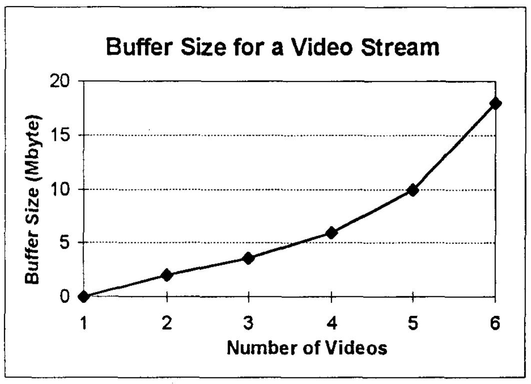


Figure 6. Buffer size required by a CD player where D = 5 seconds and Rs = 1.2 Mbytes/second.

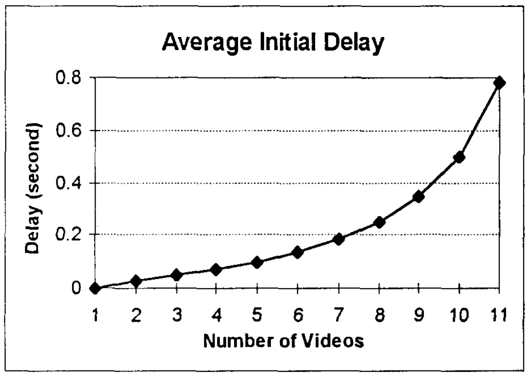


Figure 7. Initial delay caused by a hard disk where D = 25 mini-seconds and Rs = 2 Mbytes/second.

In the same way, the initial delay and total buffer size of a hard disk can be calculated. Suppose the transfer rate is 2 Mbytes/second and the maximum seek time (including protocol overhead, command processing, head movement, and rotation time) is 25 mini-seconds, Figures 7 and 8 show the initial delays and the buffer sizes per video stream respectively when a low cost hard disk is used in the disk array. We can find that, 11 simultaneous video streams can be served by a hard disk with initial delay less than one second and only 1.3 Mbytes of RAM buffer is needed. If 4 disks are used in the disk array, 32 videos can be served simultaneously. The memory size is 4 times larger and the initial delay remains the same.

From (2) and (3), we can find that the initial delay is determined by R_S and D. Practically, the initial delay must be bounded by a reasonable value such that the user feels no inconvenience in using this system. The relations among number of concurrent videos, initial delay bound, and transfer rate are important for us to learn whether the transfer rate is fast enough or not. Suppose D = 5 seconds, Figure 9 shows the numbers of concurrent videos supported by a storage device with different transfer rates. From this figure, we can find that with one minute initial delay bound, 3 concurrent videos can be supported by a 4-time speed CD player. If we double the transfer rate (e.g. 1200 Kbytes/second), 6 videos can be supported. But if we use 16-time speed CD players, only 9 instead of 12 videos can be served by a CD player. That means the seek time has become the bottleneck no matter how fast the transfer rate is.

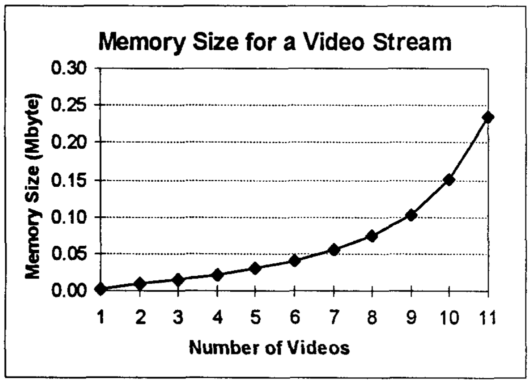


Figure 8. Buffer size required by a hard disk where D = 25 mini-seconds and Rs = 2 Mbytes/second.

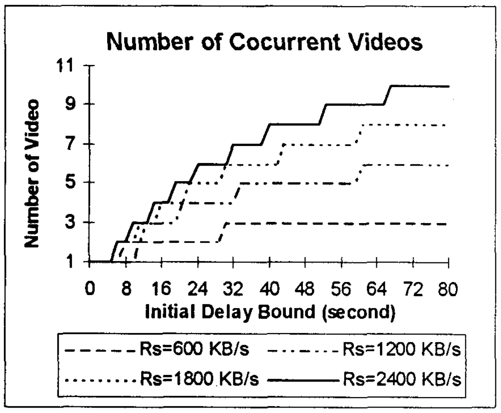


Figure 9. Number of concurrent videos bounded by initial delays with different transfer rates (D = 5 seconds).

To learn more about the impact of the seek time on the performance, we will fix the transfer rate and change the seek time to find out its impact on the initial delay. Figure 10 shows the relation between the seek time and the number of concurrent videos with R_S = 1.2 Mbytes/second. It is shown that if the initial delay bound equals 24 seconds, only three videos can be served if D = 8 while 6 videos can be served if D = 2, which doubles the number of users. However, as the initial delay bound gets larger, the difference is more obscure. Nonetheless, an initial delay within 30 seconds is strongly desired by KTV customers; hence, it is economically important to choose a jukebox with small D since much fewer CD players and jukeboxes will be required.

4. Caching

The distribution of videos requested by users is not uniformly distributed. Some hot-songs are requested more frequently than others. For these hot songs, they can be pre-stored in advance in the hard disk to reduce the initial delay. Also, the bandwidth can be reduced because that video data is retrieved directly from a hard disk and the data flow from a CD player to a hard disk is eliminated. In the following, we will examine the effects introduced by two caching mechanisms. For the rest of the paper, we choose to use a jukebox with 8-time speed CD players, D = 5 seconds, and five videos supported per CD player, which means the average initial delay from the CD player equals 33 seconds.

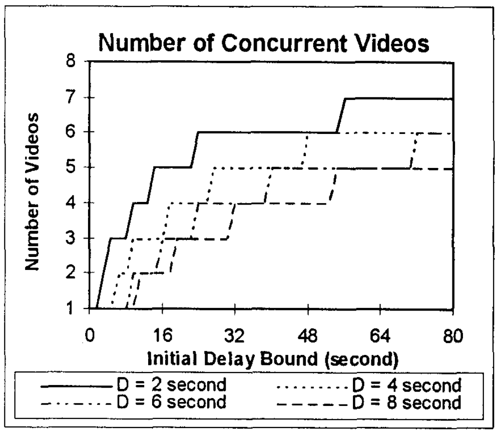


Figure 10. Number of concurrent videos bounded by initial delays with different seek times (R_S = 1200 Kbytes/second).

4.1 Whole-Song Caching

Practically, about 10 percent of songs are hot songs which are requested with probability 0.9. Hence, a natural thought is to store as many hot songs in the hard disks as possible such that the frequency of retrieving data from the slow CD players can be significantly reduced. Since the data size of a song is about 36 Mbytes, to store 500 hot songs out of 5000 in the hard disk requires 18 Gbytes of memory.

Figure 11 shows the curve of initial delay versus the hard disk capacity where the initial delay of a CD player is 33 seconds and that of a hard disk 0.5 second. From this figure, with hard disk capacity equals 10 Gbytes, the average initial delay equals 17 seconds, only half of the original 33 seconds. Clearly, if 18 Gbytes of hard disks are used, then the average initial delay equals only 3.8 seconds. This is a significant reduction in average initial delay.

Observing from the video requests, the distribution is more likely to be step-wise geometrically distributed. The reason to support this argument is that the probability to select a top-20 song is higher than that to select top-50 song. Similarly, the probability to select a top-10 song is higher than that of a top-20 song, etc.. In the following, we will find the average initial delay based on such a request distribution.

To begin with, we define the meaning of step-wise geometric distribution with decay ratio α ($\alpha < 1$) to be that by sorting all songs according to its hit ratio and assuming the probability of a song which is requested is p, then the probability of the next song is αp and so on.

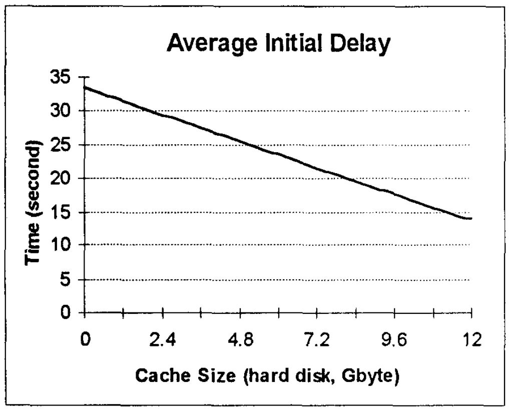


Figure 11. Initial delay vs. cache (hard disk) size, suppose 500 of 5000 songs are requested with probability 0.9, the initial delay of a CD player is 33 seconds, and the initial delay of a hard disk is 0.5 second.

Figure 12 shows the curve of initial delay versus the hard disk capacity, where the request distribution is step-wise geometric distribution with decay rate 0.998, and the initial delays and number of songs are the same as in Figure 11. From this figure, with hard disk capacity greater than 10 Gbytes, the average initial delay will be lower than 15 seconds.

4.2 Head Caching

For video applications such as KTV, there is a special feature which allows us to propose the *head caching* strategy which can achieve a very small initial delay with a smaller disk array as required by using whole-song caching. This special feature is that the video always starts playing from the beginning and no fast forward or rewind is necessary. Hence, we do not have to cache the entire song in the hard disk; instead, only the first block of video is needed to be cached in the hard disk. Such a caching scheme is called *head caching*. It can significantly reduce the cache size or store more songs in the cache.

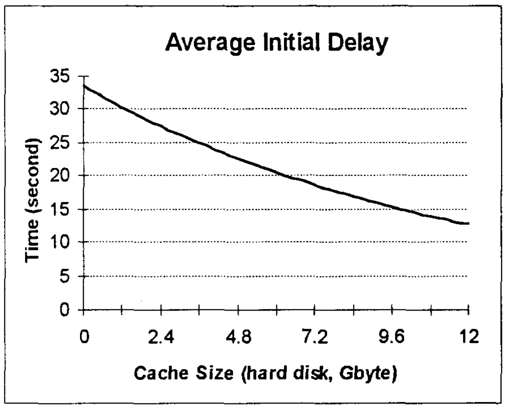


Figure 12. Initial delay vs. cache (hard disk) size, where the request distribution is step-wise geometric distribution with decay rate 0.998, the initial delay of a CD player is 33 seconds, and the initial delay of a hard disk is 0.5 second.

The scenario of this scheme is described as the following. The hard disk caches the first block of each video data and the block size is equal to the buffer size mentioned in (5). When the video is requested, the first block of data is retrieved from the cache instead of from the CD player. At the mean time, the CD is scheduled in the service table of the CD player for the following blocks. Before the data in the cache is consumed, the jukebox swaps in the CD and read the next block of video data into the buffer for further playing. Since the block size of data which is pre-stored in the cache is equal to the buffer size, the consuming time of the data in the cache is greater than the round-trip time of round-robin service of the CD player. Therefore, no under-flow may occur using *head caching* strategy.

For the first case that 10 percent of songs are requested with probability 0.9, the result using *head caching* method is shown in Figure 13 with all parameters remain the same. It shows that with only 4.8 Gbytes hard disk, all the hot-songs can be cached and the initial delay is reduced to 3 seconds. For the second case that the request distribution is step-wise geometric distribution with decay rate 0.998, the result is shown in Figure 14 which shows that with only 4 Gbytes hard disk, the initial delay is smaller than 10 seconds and with 12 Gbytes hard disk size the initial delay can be further reduced to about 1 second, which is close to that using a large disk array only[3]. These facts show the superb performance of *head caching* mechanism.

Head caching also can be applied between hard disks and host RAM. Only the first block of video data is cached in the host RAM. Before it is consumed, the next block of data will be retrieved from the hard disk. It can be shown that to cache the first block of data from the hard disk, only tens of Kbytes of RAM is needed for a song.

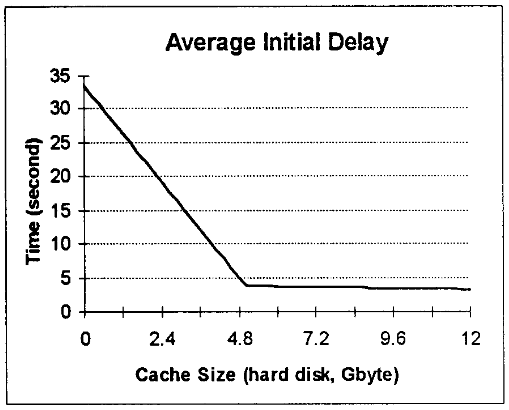


Figure 13. Initial delay vs. cache (hard disk) size when *head caching* is used. The parameters are the same as those shown in Figure 11.

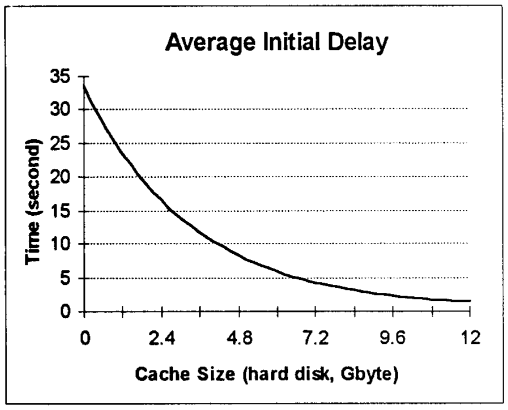


Figure 14. Initial delay vs. cache (hard disk) size when *head caching* is used. The parameters are the same as those shown in Figure 12.

5. Segmented Video

For movie-on-demand services, the video server must support fast-forward and rewind control for user interaction. Another feature, call dynamic panning, which can jump to any location of the movie using a slider must also be supported by the server. To support such features, the initial delay must be very small for user interaction. Normally, the response time after fast-forward button is pressed must be below a few seconds. It is hard to implement a movie server using CD based storage because the initial delay is too large. However, with the idea of *head caching*, a CD based jukebox plus a disk array can be used as the video server for movie-on-demand as explained below.

A movie is fragmented into many segments with equal size. Each segment contains a number of video blocks. This type of video is called segmented video in this paper. In advance, the first block of video in each segment is pre-stored in the cache. To play video, data is loaded block by block from the storage. If fast-forward is issued by the customer, the first block of the next segment of video can be played immediately because it is in the cache and no seek action from the jukebox is needed. At the mean time, the next block of the segment is requested and it is retrieved before the cached data is consumed. With such a design, user can see immediate feedback if fast-forward button is pressed.

The same scheme is also applied in rewind-control. For dynamic panning of video, the server gets the location of the slider and finds the corespondent video segment, then plays the first block of video immediately. If there are K blocks in a segment, then $\frac{1}{K}$ of the video data will be cached in the hard disk while $\frac{K-1}{K}$ of which will be stored in the jukebox. Nevertheless, since each fast forward or rewind will be K blocks away from current video position; hence, the value of K cannot be too large.

Notice that, the segment size must be carefully chosen. If the segment size is too large, the stepping interval will be too large such that users can hardly find out the relation between video segments. If the segment size is too small, the cache size will increase significantly. For example, if an 8-time speed CD player is used with D = 5 seconds and five users supported per player, then B equals 10 Mbytes, which is equivalent to 67 seconds of video. In this case, if K equals 3, then a fast forward or rewind will be 201 seconds away from current video position, which may be too large a jump. However, if a 16-time speed CD player is used with D = 2 seconds and five users supported per player, the value of B will then be reduced to 2.2 Mbytes. Hence if K equals 3, then a forward or backward jump is 44 seconds from its current position, which is more acceptable.

6. Conclusion

A high performance and low cost video server architecture is proposed which can store a large volume of videos and support up to 30 concurrent users simultaneously. With the proposed *head-caching* strategy, the initial delay can be dramatically reduced without increasing the cache size. Moreover, using *segmented video* mechanism, user can control the video playing through fast forward, rewind, and dynamic panning with small delay. Besides, the performance analysis of buffer size, initial delay, and disk size are discussed in this paper.

Lastly, it is important to note that with current products where $R_S \leq 600$ Kbytes/second and $D = 8$ seconds, it would be difficult to support many users using jukebox-based video servers. Hence, improvements in data bandwidth of CD players (R_S) and in the swap time of jukebox (D) are strongly desired to make jukebox-based video server pratical.

References

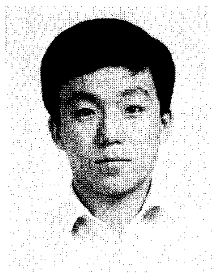
[1] James R. Allen, Blaise L. Heltai, Arthur H. Koenig, Donald F. Snow, and James R. Watson, "VCTV: A Video-On-Demand Market Test," AT&T Technical Journal, pp. 7-14, January/February 1993.

[2] Rramod Pancha and Magda El Zarki, "A look at the MPEG video coding standard for variable bit rate video transmission," INFOCOM, pp. 85-94, 1992.

[3] Fouad A. Tobagi, Joseph Pang, Randall Baird, and Mark gang, "Streaming RAID - A Disk Array Management System For Video Files," Proceedings ACM Multimedia, pp. 393-400, August 1993.

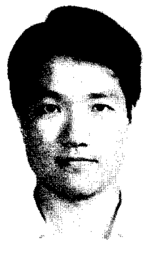
[4] A. L. Narasimha Reddy and Jim Wyllie, "Disk scheduling in a multimedia I/O system," Proceedings ACM Multimedia, pp. 225-233, August 1993.

Biographies



Wei-Hsin Tseng received his B.S. degree in computer science and information engineering (CSIE) from Tatung Institute of Technology, Taipei, Taiwan, in 1989 and M.S. degree in CSIE from National Taiwan University (NTU) in 1991. Since then, he has been a Ph.D. student in NTU.

His current research topics include forward correction coding, admission control, video modeling and video server architecture.



Jau-Hsiung Huang received the B.S. degree in electrical engineering from National Taiwan University, Taipei, Taiwan, in 1981 and the M.S. and Ph.D. degree in computer science from the University of California, Los Angeles, Los Angeles. CA. USA in 1985 and 1988 respectively.

Since 1988, he has been a member of the faculty in the Department of Computer Science and Information Engineering department, National Taiwan University, where he is currently a professor. He has published over 40 technical papers in the areas of multimedia networking, high speed networking, parallel and distributed systems and performance evaluation of computing systems.