A Novel Video Layout Strategy for Near-Video-on-Demand Servers

Shenze Chen & Manu Thapar

Hewlett-Packard Labs
1501 Page Mill Rd.
Palo Alto, CA 94304
email: {szchen, thapar}@hpl.hp.com

Abstract

Near-Video-on-Demand (NVOD) provides customers with a service model completely different from true Video-on-Demand (VOD). In the NVOD service model, customers' requests are not serviced immediately. NVOD servers typically support limited VCR functions. Since the NVOD service needs to be much cheaper than the VOD service, it is important to minimize the server's cost.

In this paper, we present a novel video layout strategy for NVOD servers that enforce sequential disk access. Thus, the disk bandwidth is optimally utilized. We define a model that analyzes the storage subsystem behavior, buffer requirements, and usage. Strategies that match the actual disk bandwidth to the application bandwidth requirements are developed. Using this layout strategy, each disk can deliver 50% more streams than can a VOD system, and memory buffers are reduced by almost half. Since disks and memory account for a significant portion of the total system cost in a video server, using these strategies significantly reduces server costs.

From this layout strategy we further derive two other layout strategies, *Segment-Group-Pairing (SGP)* and *Disk-Pairing (DP)*, both of which are designed to optimize disk bandwidth usage. The circumstances and conditions under which these two strategies work best are described. Finally, an experimental NVOD server prototype was built that supports both "broadcast" and "on-demand" service models. Using this prototype, we achieved our design and performance objectives and have demonstrated that our proposed layout strategies are practical.

Keywords: Near-Video-on-Demand, VOD, Multimedia Server, Video Layout

1. Introduction

Near-Video-on-Demand (NVOD) provides customers with a service model completely different from true Video-on-Demand (VOD). The VOD service model needs to immediately service customers' requests, such as: play a movie, fast forward, and fast backward. Since customers' requests arrive at a server randomly, and since each request may select a different movie, the I/O workload to the server's movie storage subsystem is quite unpredictable. In the NVOD service model, customers' requests are not necessarily serviced immediately. Instead, customers know a priori that their service requests may incur a bounded delay of maximum *t* minutes before the service is delivered. Typically, there are two models that provide NVOD services, the "broadcast" model and the "on-demand" model:

- in the "broadcast" model, the server broadcasts stored movies periodically, for example, every *t* minutes. Thus, if a customer misses the start of a broadcast, the customer must either continue with the current broadcast without seeing the beginning or wait *t* minutes for the next broadcast. In this broadcast model, NVOD servers provide only limited fast forward and fast backward capabilities. Customers can switch between broadcast streams that are staggered by *t* minutes.
- in the "on-demand" model, the server waits for customers' requests to arrive. Upon receiving a request, the server waits *t* minutes before it delivers the requested movie. If any new requests for the same movie arrive during the *t*-minute period, the server services them together by multicasting the stream to all requesters. In this model, fast forward and fast backward capabilities are not supported, but server and network resources may be saved.

Since NVOD servers do not provide full VCR functions and instant services to customers, the NVOD service needs to be much cheaper than the VOD service. It is therefore important to minimize the server's cost. (Notice that there is a significant difference between the NVOD service and the current *pay-per-view (PPV)* service provided by cable companies. In the *PPV* service model, a particular movie is broadcast at a pre-specified time. If a customer misses that time, the customer has no way to view the movie again.)

In this paper, we address issues related to the storage subsystem design and the RAM buffer requirements for the NVOD server. Specifically, we present a novel video data layout mechanism that enforces sequential disk accesses. Thus, the disk bandwidth can be optimally utilized. We then provide a model that analyzes the storage subsystem behavior. This model also covers buffer requirements and usage. Strategies that match the actual disk bandwidth to the application bandwidth requirements are developed. By using this layout strategy, the resultant sequential disk access allows each disk to support 50% more streams than can a VOD system. For example, if a disk supports 8 streams in a VOD system, then it can support 12 streams in an NVOD system. In addition, by using this layout strategy, the buffer memory required for each stream is reduced almost by half. Since disks and memory account for a significant portion of the total system cost, using these strategies can substantially reduce server costs.

From this layout strategy we derive two other layout strategies, *Segment-Group-Pairing (SGP)* and *Disk-Pairing (DP),* which are designed to optimize the disk bandwidth usage. The circumstances and conditions under which these two strategies work best are described.

Finally we discuss the issues related to using tape or CD as the primary storage medium for NVOD servers.

In the past few years, a number of research papers on VOD have been published [1,2,3,4,5,6,7,8,9,10,11]. Of course, any VOD server can be used to provide NVOD services with only minor modifications. However, the complexity and the high cost of VOD servers do not justify their use as NVOD servers. Very few papers on NVOD servers can be found in the literature. Cleary [12] discussed the terms of pure-video-on-demand (PVOD) and near-video-on-demand (NVOD), and reviewed the competing technologies and trials under both PVOD and NVOD. Kusayanagi et al [13] evaluated ATM multicast-based access networks using NVOD and CATV simulations. In a similar study, Sharobim [14] discussed a network architecture and a routing algorithm that support multicasting of NVOD services for large-scale ATM networks. While the previous works on NVOD addressed only concepts and the distribution networks that provide NVOD services, our study is more focused on the NVOD server design and optimizations.

The remaining of the paper is organized as follows: Section 2 describes a video layout strategy, which is used as the basic strategy in the following discussions; Section 3 presents a simple analysis of the system model; Section 4 discusses the techniques that match the disk bandwidth to the application bandwidth requirements; Section 5 presents further disk optimization strategies; Section 6 discusses the feasibility of using tape and CD as the primary storage media for NVOD servers; Section 7 is devoted to the "on-demand" service model; Section 8 describes our experimental NVOD server prototype; and Section 9 summarizes this paper.

2. A Data Layout Strategy

In this section, we present a novel video data layout strategy for NVOD servers. We start by assuming the "broadcast" model for NVOD services, as described in Section 1 above. Later in Section 7, we will see that the layout strategies developed are equivalently applied to the "on-demand" model also.

The video layout strategy places a movie on a single disk, if the disk's bandwidth meets the application bandwidth requirement for broadcasting the movie. Typically, in a VOD server, movies are striped across multiple disks for the purpose of load balancing. This prevents a popular movie from making any single disk a bottleneck. For NVOD, however, the I/O workload of a movie is determined solely by the broadcast interval, not by viewers' demands. Therefore, once the broadcast repeat interval for the movie is determined and if the bandwidth requirement can be satisfied by a single disk, placing the movie on a single disk will not cause bottlenecks, as is the case with VOD. Choosing the broadcast repeat interval depends on many factors, especially the movie's popularity, since *hot* movies need to be broadcast more frequently than *cold* movies.

When the application bandwidth requirements cannot be satisfied with a single disk, this strategy can be easily extended to store a movie on two or more disks, using the aggregate bandwidth of the multiple disks to meet the bandwidth requirements. When the bandwidth requirements are less than the disk's bandwidth, this strategy can be extended to interleave multiple movies on a single disk to fully utilize that disk's bandwidth. We will discuss both approaches in Section 4.

Normally, a disk can be considered as a series of logical blocks, with typical block size of 512 bytes. Contiguous logical blocks are usually mapped into contiguous physical blocks (except for defective blocks or cylinder and track skews). We define a *segment* to be the size of the data transfer for one I/O request. A segment may consist of multiple logical blocks. The proposed layout strategy breaks a movie into segments, *0, 1, 2, ..., n, ...*, and places these segments on a disk according to the order shown in Figure 1.

Using this strategy to layout segments on a disk, segment *0* is first placed on the disk (occupying logical blocks *0* to *x)*. Segment *n* is placed next to segment *0* (starting from logical block *x+1*). Segment *n* is in turn followed by segment *2n*, segment *3n,...,*segment *(k-1)n,* segment *1*, ... segment *kn-1,* i.e., the first segment *0* is placed in the outermost zone and the last segment *kn-1* is placed in the innermost zone of the disk. Parameter *n* is the number of segments to be played back during a repeat interval for each stream, and parameter *k* is the number of simultaneous streams of a movie, which is determined by the movie length and its repeat interval. These parameters will be discussed later in Section 3.

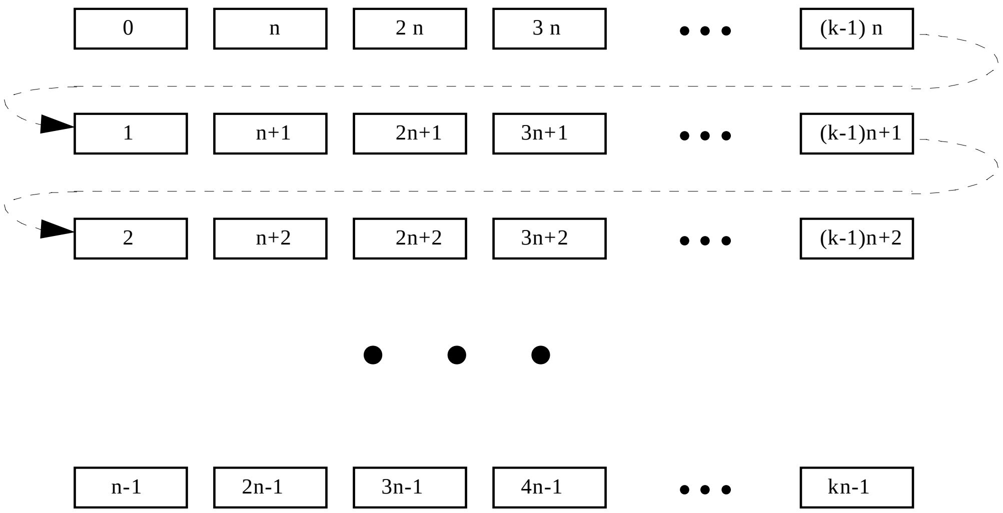

Figure 1: Movie Layout Strategy.

When the movie is broadcast, the disk is read sequentially from its beginning (logic block 0), segment by segment in the order of segment *0*, segment *n*, segment *2n*, ..., until the last segment (*kn-1* in Figure 1), and then the arm moves back to the beginning of the disk and the access pattern is repeated. Thus, the disk arm repeatedly sweeps across its surfaces from outer zones to inner zones, sequentially reading the data. These operations are detailed and analyzed in the next section. Disk seeks are eliminated using this technique, except for the seeks from the innermost to the outermost track between sweeps.

One of the advantages of storing a movie on a single disk is ease of management. If we decide to broadcast a new movie, we only need to reload one disk; if a disk fails, only one movie is off the air (assuming there are no redundant disk copies); other movies are not affected.

3. The System Model

3.1. Notations

We first define the notation used in the analysis:

L: movie length, in minutes;

t: broadcast repeat interval, in minutes;

r: movie stream bit rate, in Mbits/sec;

S: segment size (i.e., I/O transfer size), in Kbytes;

n: number of segments to be played back in a repeat interval of *t* minutes for each stream;

k: number of concurrent streams for broadcasting a movie at *t*-minute intervals;

B: disk bandwidth requirement for broadcasting a movie;

C: disk capacity requirement for a movie;

T: playback time of one segment at the bit rate *r*.

3.2. The Model

At the system level, the NVOD server broadcasts many movies simultaneously. Each movie can have its own bit-rate and repeat interval. Given a movie's length *L* and its repeat interval *t*, which are typical parameters known by the system administrator, the number of streams for the movie can be calculated by:

$$k = \left\lceil L/t \right\rceil \qquad (1)$$

For example, a two-hour movie with a broadcast interval of 10 minutes needs 12 streams. For each stream, a certain amount of buffer space is allocated in system memory. Minimally, a "Ping-Pong" buffer (two segments) is needed for each stream, as shown in Figure 2. While one buffer is transmitting data to the distribution network, the other buffer is being filled from disk.

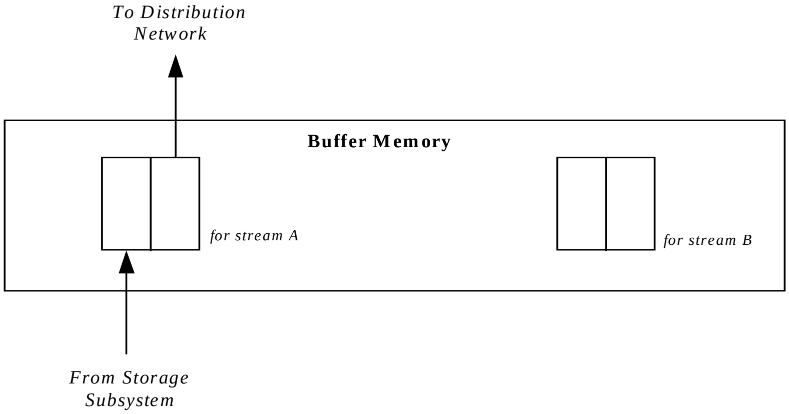

Figure 2: The Buffer Structure.

The transmit time of a segment, *T*, depends on the segment size *S* and the transmit rate *r*:

$$T = 7.8125\, S / r \qquad (2)$$

(The coefficient 7.8125 in equation (2) is resulted from unit translations of milliseconds for *T*, Mbits/sec for *r*, and Kbytes for *S*. Coefficients in the following equations exist for the same reason.)

The number of segments to be played back for a stream during a repeat interval is given by:

$$n = \left\lceil \frac{7{,}680tr}{S} \right\rceil \tag{3}$$

Now it is clearer that, in Figure 1, segment *0* and *n* are *t* minutes apart in a stream. We consider a sequential read of *k* segments of data to be a *"service round"*, such as reading segments *0, n, ..., (k-1)n*. Within a service round, each segment read belongs to a different stream of the same movie. Because of the ping-pong buffer characteristics, each service round must finish within a buffer playback time of *T* milliseconds. Each service round starts (i.e., I/O requests are issued) exactly at the *T*-millisecond period boundary (Figure 3). In this way, the disk arm sweeps from logic block *0* (in the outer zone) to the last logic block used by the movie (in the inner zone) in *t (= nT)* minutes, and then jumps back to logic block *0* and sweeps again. Thus, the parameter *n* also represents the number of service rounds in one sweep.

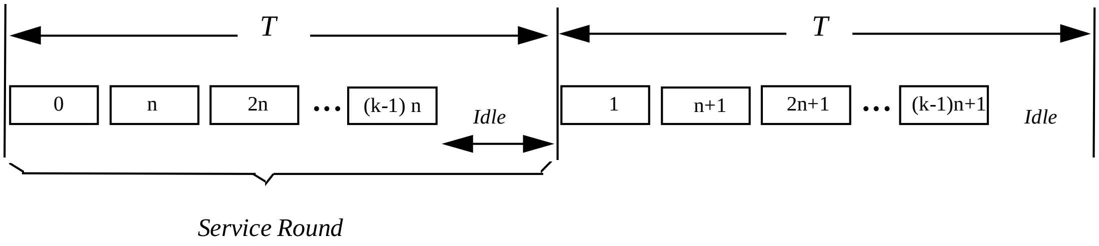

Figure 3: A Service Round and Its Period T.

Notice that during a service round the retrieval of *k* segments can be finished in less than *T* milliseconds. In this case, the disk becomes idle since no buffer is available for the next data transfer. An optimal design would minimize this idle period in order to fully utilize the disk bandwidth. Fortunately, for sequential disk accesses the time for retrieval of *k* segments is basically determined by the disk data transfer time and is therefore more predictable than for the random accesses found in most VOD systems. This issue will be addressed further in later sections.

3.3. Bandwidth Requirement

In the model above, the disk is required to sequentially read *k* segments of data every *T* milliseconds. This requires the disk to provide a minimum sustained sequential read bandwidth of

$$\begin{aligned} B &= k\,S\,1000\,/\,(T\,1024) \\ &= k\,r\,/\,8 \qquad \text{Mbytes/sec} \end{aligned} \tag{4}$$

Today's modern disks use the zone-bit-recording technique to achieve high capacity. This results in higher bandwidths in outer zones than inner zones. In order to use this mechanism, we must make sure that the disk inner zone (not the average) bandwidth meets the requirement:

$$B = k\, r / 8 \quad <= \quad Disk\_Bandwidth \qquad (5)$$

Example 1: The measured sequential read *Disk_Bandwidth* for IBM DFHS drives [15] ranges from 5.1 to 6.9 MB/sec, depending on whether the disk is accessing inner or outer zones. From Equation (5), a single such drive can support the broadcast of a two-hour movie coded at a rate of 3 Mbits/sec with a maximum of 13 streams. From Equation (1), the minimum repeat interval is 9.3 minutes. If the movie is to be broadcast at 5-minute intervals, then two disks are needed (see Section 4.1).

3.4. Capacity Requirement

Calculating the capacity requirement for storing an *L*-minute movie on a disk is straightforward:

$$C = L\, r\, 60 / 8 \qquad Mbytes \qquad (6)$$

For example, a 4-Gbyte IBM DFHS drive can store a three-hour movie coded at a 3-Mbits/sec rate, or a 2.2-hour movie coded at a 4-Mbits/sec rate.

Given the above bandwidth and capacity requirements for storing and broadcasting a movie, a disk could be selected that "just" satisfied these requirements, since any extra disk bandwidth or capacity would be wasted. For example, if the disk had a much higher bandwidth, it would generate a longer idle period, as shown in Figure 3.

We also point out that with this model disks do not need to be homogeneous. For example, fast disks could be used to store movies with higher-bit-rates or with shorter repeat intervals, and slow disks could be used to store lower-bit-rate movies, etc. This would give the system administrator more flexibility in leveraging the existing disk drives.

3.5. Buffer Requirement

In the above discussions, we assumed a ping-pong buffer for each stream (not each movie), with the size of each buffer equal to the disk I/O transfer size *S*. This size *S* immediately determines the service round period *T* for a given movie bit rate (Equation (2)). A smaller value of *S* results in lower buffer costs but may increase system overhead, since more I/O requests, interrupts, and context switches are generated. These may result in a higher CPU and I/O channel overhead. This trade-off needs to be carefully evaluated. Obviously, this model can be extended easily using three or more buffer segments for each stream.

3.6. Changing the Broadcast Repeat Interval

There are circumstances in which people want to change the broadcast repeat interval for a movie. For example, a movie may be broadcast at a repeat interval that is longer in the afternoon than in the evening. With the video layout strategy described in Section 2, it is easy to change the broadcast repeat interval *t*. Specifically, by using this strategy, if a disk supports broadcasting a movie at a repeat interval *t*, then it can also broadcast at intervals *2t, 3t, ..., (k-1)t* by skipping some segments in each service round. Continuing with Example 1 shown in Section 3.3, an IBM DFHS drive can support broadcasting a two-hour movie coded at a 3-Mb/sec rate at 10-minute intervals. This implies that *k*=12 segments (for 12 streams of this movie) need to be read in a service round, as shown in Figure 3. Now, to broadcast the movie at 30-minute intervals, the arm still performs one sweep every 10 minutes, but during the first sweep, it reads segments *0, 3n, 6n,* and *9n* for the first service round, segments *1, 3n+1, 6n+1, and 9n+1* for the second service round, etc. During the second sweep, it reads segments *n, 4n, 7n,* and *10n* for the first service round, segments *n+1, 4n+1, 7n+1,* and *10n+1* for the second service round, etc. During the third sweep, it reads segments *2n, 5n, 8n,* and *11n* for the first service round, etc. Thus, in each service round, only 4 segments are read and each segment is read once every three sweeps in the 30 minutes. This assures broadcasting the movie at 30 minutes intervals. Thus, a program maker can easily select a repeat interval to fit a schedule without reloading the movie.

3.7. Fault Tolerance

When fault tolerance is desirable, the above layout strategy can also be applied to a RAID device, such as a RAID 3. Normally, RAID 3 can be viewed as a single “large” disk with a very high sequential access bandwidth. Disk spindles in a RAID 3 are typically synchronized to provide a high sequential bandwidth. Since our layout strategy enforces sequential disk accesses, it matches perfectly with the RAID 3 sequential access performance advantages. RAID 3 also has a nice feature that when a disk fails, RAID 3 continues to deliver data at the same rate as in the normal mode without performance degradation. Therefore, with only a single disk failure, no movie streams will be interrupted. Since a RAID 3 disk array consists of multiple disk drives, it might provide more bandwidth and capacity than required for broadcasting a movie. In this case, the technique described in Section 4.2 could be used to broadcast multiple movies from a single RAID 3.

An alternative for providing fault tolerance is the disk mirroring. To reduce costs ($$/MB), we can use a large disk to store redundant copies of multiple movies. When the cost of the mirrored disk is lower than that of a RAID controller, disk mirroring is justified. Since there is no intelligent RAID controller to detect disk failures, when a disk fails, typically the host will time out and resends the command to the redundant disk. Hence, one should be careful about the delay caused by the time out (and resend) and its impact on the disk sweeping process described above.

4. Matching Disk Bandwidth with Application Requirement

Using the basic layout strategy described in Section 2, we place a movie on a single disk. We may, however, need to place one movie on multiple disks or multiple movies on a single disk in order to match the available disk bandwidth with the application bandwidth requirements for the broadcast.

4.1. Placing a Movie on Multiple Disks

As mentioned in the previous sections, if a single disk does not meet either the bandwidth or the capacity requirements for a movie, then two or more disks are needed to store the movie.

Capacity: This is straightforward. If a single disk's capacity is not large enough to hold a movie, then we need to add another disk. Typically, MPEG-2 movie streams are coded at rates in the range of 1.5 - 8 Mbits/sec. By today's disk technology, 9-GB disks are already available that can hold 2.5-hour movies coded at an 8-Mb/sec rate. Therefore, capacity may not be a real problem.

Bandwidth*:* As mentioned in Example 1 in Section 3.3, using the basic layout strategy, today's state-of-the-art disk drive can only support the broadcast of a movie coded at 3 Mb/sec at a minimum interval of 9.3 minutes. For any broadcast interval less than that, two or more disks are needed. Consider a scenario in which a 4-GB movie must be broadcast at repeat intervals of 5 minutes. We have two ways to achieve this:

- Use two 4-GB disks, with each one configured independently to support the broadcast at a 10-minute interval. The two disks are "synchronized" in such a way that they start to play with an offset of 5 minutes.

- Use two 2-GB disks, and stripe the movie on the two disks by placing segments *0, 2n, 4n, ...* on disk 1 and segments *n, 3n, 5n, ...* on disk 2 (see Figure 1). The two disks start to sweep simultaneously so that the bandwidths of the two disks are aggregated to meet the requirement.

Obviously, the first strategy requires a double storage capacity, but it provides fault tolerance in that, when one disk fails, the system still broadcasts the movie, but at a longer repeat interval. The second strategy lowers system costs, but if any one of the disks fails, the movie stops.

4.2. Placing Multiple Movies on a Single Disk

In contrast to the case in Section 4.1, if a disk's available bandwidth is far greater than required for broadcasting a movie, then we can either place multiple movies on a single disk to utilize the bandwidth, or select a slower (and cheaper) disk with a lower bandwidth. In the following, we will focus on the former, and present a strategy to layout multiple movies on a single disk.

From Example 1 in Section 3.3, we know that an IBM DFHS type drive can support broadcasting a two-hour movie coded at a 3 Mbits/sec rate at a repeat interval of 10 minutes, which implies that one disk can support 12 movie streams using the proposed basic layout strategy. Now, if movies are to be broadcast at a repeat interval of 30 minutes at four streams per movie, we can use the same disk that is laid out for broadcast at 10-minute intervals, and use the strategy described in Section 3.6 to broadcast the movie at 30-minute intervals. This, however, wastes 1/3 of the available disk bandwidth. If we know the movie is not to be broadcast at higher frequencies, e.g., every 10 or 20 minutes, then we can place three movies on a single disk to fully utilize its bandwidth. If the average movie length is two hours (that requires 2.7 GB per movie), then a 9-GB drive can be used to hold the three movies. We can still use the basic layout strategy, but the three movies are interleaved, as shown in Figure 4.

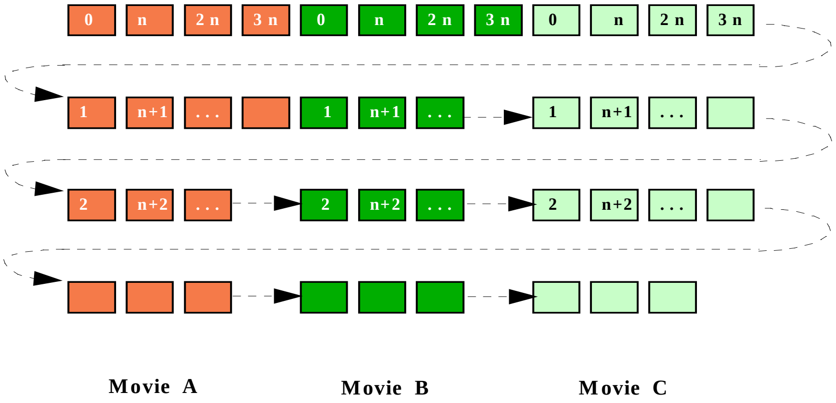

Figure 4: Interleaving Multiple Movies on a Single Disk.

During each service round, 12 segments are read from the disk, 4 segments for each movie. The disk arm takes 30 minutes for each sweep from the outer zones to the inner zones. As shown in the figure, the movie lengths are not necessarily the same. When one stream reads and displays the last segment of a movie, it stops and waits until the next start time (at 30 minutes boundaries). In Figure 4, the second service round reads only 11 segments, the third round 10 segments, and the fourth round 9 segments. By using large capacity drives, we can lower the system cost, since using 9 GB drives is always cheaper than using 2 GB or 4 GB drives in terms of $$/MB.

5. Disk Optimization Strategies

In Section 2, we presented a basic movie layout strategy for NVOD servers. In this section, we discuss some schemes for optimizing disk bandwidth usage. These optimizations are based on the above basic layout strategy. As mentioned in Section 3, modern disks use the zone-bit-recording technique. This results in higher bandwidths in the outer zones than in the inner zones. For the NVOD layout strategy described above, we must base our calculations on the disk inner-zone bandwidths. This implies that the higher bandwidths in outer zones are wasted, i.e., the idle periods for accessing outer zones will be longer than those for inner zones. Table 1 shows the (measured) sustained sequential bandwidths for IBM DFHS drives. From this table, we see that the bandwidths for the outermost zone could be 35% higher than that for the innermost zone.

Zones	**Number of Cylinders**	**Sustained Sequential Bandwidth (MB/s)**
Zone 1 (outer)	1877	6.91
Zone 2	955	6.91
Zone 3	48	6.65
Zone 4	309	6.47
Zone 5	348	6.14
Zone 6	115	5.90
Zone 7	213	5.73
Zone 8	189	5.53
Zone 9	130	5.30
Zone 10 (inner)	206	5.10

Table 1: Zone Bandwidth for IBM DFHS Drives.

In the following sections, we describe two layout strategies designed to utilize the higher outer-zone bandwidths.

5.1. The Segment-Group-Pairing (SGP) Strategy

This strategy is based on a concept similar to “track-pairing” [16], but instead of pairing a track in an outer zone with a track in an inner zone, it pairs a segment group in an outer zone with a segment group in an inner zone. The segment group can be defined in different ways. For instance, we can either divide the first row of *k* segments in Figure 1 into two groups and make them a *segment group pair*, or make the first row of *k* segments one group, the second row another group, and then pair these two groups.

For ease of discussion, we use the first method to group segments below. The analysis is also immediately applicable to the second method, since only the group data size matters.

Now the system works in the following way. During each service round, the disk arm reads the first group in the outer zone; then it seeks all the way to its peer group in the inner zone; after rotational latency, it reads the second group; then it seeks back to the outer zone and waits for the next service round. The purpose here is to trade the two seeks and rotational latencies occurring in each service round for the higher bandwidth (or lower transfer times) of outer zones. If the overall effective sustained bandwidth is higher than the inner zone bandwidth, then this Segment-Group-Pairing layout strategy is preferable to the basic strategy described in Section 2. Otherwise, the basic strategy should be used.

Let *Bmax*, *Bmin* be the disk outermost and innermost bandwidths, and *MAX_SEEK*, *MAX_ROT* be the maximum seek time and rotational latency (we must do the worst-case analysis here as we did in Section 3 for the basic layout strategy). Then the effective bandwidth achieved using the SGP strategy is calculated by dividing the size of the group pairs by the time used to read the group pairs. This read time includes the data transfer times for the two groups plus the times for two seeks and two rotational latencies:

$$Beff = S\,k / [0.5\,S\,k\,(1/Bmax + 1/Bmin) + 2\,(MAX\_SEEK + MAX\_ROT)\,1024 / 1000\,]$$

$$= 1 / [0.5\,(1/Bmax + 1/Bmin) + 2048\,(MAX\_SEEK + MAX\_ROT) / (1000\,S\,k)] \qquad (7)$$

where *S k* is the data size transferred for the group pair. From the above equation, it is obvious that the bigger the *S k*, the greater the *Beff*. For a given *k* (which is determined by the movie length *L* and the broadcast repeat interval *t*, as shown in equation (1)), this implies a larger buffer size *S*.

Example 2: For IBM DFHS drives, if we have *Bmax* = 6.9 MB/sec, *Bmin* = 5.1 MB/sec, *MAX_SEEK*=16.5 ms, and *MAX_ROT*=8.34 ms, then for a two-hour movie broadcast at *t*=10 minutes, the achievable effective bandwidths using the SGP strategy are as follows:

Buffer Size per Stream (*S*)	**Effective Bandwidth** (*Beff*)
64 K (x 2)	4.24 MB/sec
128 K (x 2)	4.90 MB/sec
256 K (x 2)	5.36 MB/sec

Table 2: Effective Bandwidth of SGP Strategy.

From Section 3, we know that the basic layout strategy can utilize a maximum of *Bmin* = 5.1 MB/sec of disk bandwidth with a segment size of *S*=128 K. From Table 2, we see that the SGP strategy can do a better job in terms of disk bandwidth utilization than the basic strategy (5% improvement), but at the cost of doubling the buffer requirements. So there is a memory-to-disk-bandwidth trade-off.

The above discussions assume that the two groups in each pair have the same size, implying different access times for the two groups in outer and inner zones. We can also make the two groups have different sizes, but the same access time. This makes the layout process and the host issuing requests more difficult, since the data sizes to be laid out and read vary according to zones. This “equal-time” mechanism, however, can provide a 7% improvement over the basic strategy, as compared to the 5% improvement achievable with the “equal-size” mechanism above.

5.2. The Disk Pairing (DP) Strategy

The Disk Pairing (DP) strategy is also derived from the basic strategy introduced in Section 2. The DP strategy places a movie A with a higher bit rate in the outer zones of two homogeneous disks, and a second movie B with a lower bit rate in the inner zones of these two disks, given that both movies are broadcast at the same repeat interval *t*. Specifically, it places the first half of movie A (i.e., the first half of the segment sequence in Figure 1) on disk 1, starting from outer zone 1, followed by the first half of movie B. Then it places the second half of movie A on disk 2, starting from outer zone 1, followed by the second half of movie B. Now both disks repeat the sweeping pattern from beginning to end as before. From previous discussions, we know that the disk sweep time is equal to the broadcast repeat interval *t*, regardless of the movie length and bit rate. Thus, the system will work properly if the sweep start times of the two disks are offset by $t / 2$ minutes. Now, while the first disk is transferring data for movie A from the outer zones, the second disk broadcasts movie B from the inner zones. When the first disk sweeps to movie B $t / 2$ minutes later, the second disk jumps back to the outer zones and continues to broadcast movie A. The maximum allowable bit rate for movies A and B is determined by the disk outer zone and inner zone bandwidths, as shown in equation (5).

Example 3: By using this layout strategy, a pair of IBM DFHS disks can store and broadcast two 2-hour movies at 10-minute repeat intervals, and these two movies can be coded at a maximum of 4.6 Mbits/sec and 3.4 Mbits/sec, respectively. On the other hand, if the basic layout strategy is used, then both movies must be coded at a maximum of 3.4 Mbits/sec. Thus, using the DP strategy the overall effective disk bandwidth is *Beff* = 6 MB/sec, which is 17% higher than that achievable using the basic layout strategy.

The same paradigm can also be used on a long movie and a short movie, if both are broadcast at *t*-minute intervals. In this case, more streams are needed for the long movie (see equation (1)), which implies that more data segments need to be transferred within a service round period of *T*. Therefore, a higher bandwidth is needed for the long movie. To continue with the previous Example 3, if both movies were coded at a 4-Mbits/sec rate, then the bandwidth of a pair of these disks could support the broadcast of a 138-minute movie and a 102-minute movie (assuming the capacity requirement could be met). If each movie is stored on a single disk, then the disk bandwidth allows the broadcast of only two 102-minute movies.

Finally, while this paradigm can be extended to three or more disks for even finer tuning, the achievable benefits may be limited. Storing movies on multiple disks decreases movie availability, since any disk failure stops the broadcast. Therefore, only one disk pair is recommended for this paradigm.

6. Tape or CD as a Primary Storage Medium

While disks were used as the primary storage medium for NVOD servers in the previous sections, in this section we consider the feasibility of using tape or CD as the primary storage medium. The motivation for using tape or CD is simple: low cost.

With today’s tape technology, tapes with a bandwidth of 5 MB/sec, 9 MB/sec, and even 15 MB/sec will be available in the market this year. All of them have a fast/wide SCSI interface that allows them to transfer data at a 20-MB/sec burst rate. Using the data layout strategy introduced in Section 2, the access pattern to the storage medium is strictly sequential. This makes it possible to use tape as the primary storage medium and read video data directly from tape. In the case of disk, after sweeping the disk from the outer zone to the inner zone in *t* minutes, the disk arm jumps back to the outer zone and then continues the sweep. For tape, this “jump back” corresponds to a rewind to the beginning of the tape. If the tape has extra bandwidth and is fast enough, the rewind delay may not be a problem. For instance, if the broadcast of a movie requires only 4.5-MB/sec of bandwidth (corresponding to a two-hour movie coded at a 3-Mbits/sec rate and broadcast at 10-minute intervals), and the tape has 9 MB/sec of bandwidth, then the idle periods of the service round at the end and at the beginning of the tape may be long enough to cover the rewind time, depending on the rewind speed. If the rewind is not fast enough, then two tape drives will be needed to support the broadcast of one movie. While one drive is rewinding, the other drive takes over and continues the broadcast. This requires a doubling of the tape drive cost. Currently, the cost ratio for tape media compared with disk is about 1:300. Therefore, the major cost of using tape as the primary storage medium is the cost for the tape drives. If the cost of two tape drives is less than the cost of a magnetic disk, then reading directly from tape can be justified. In addition, another advantage of using tape is that, when we want to broadcast a new movie, the correct tape simply needs to be inserted, while with disk, the new movie has to be pre-loaded onto the disk.

In a traditional hierarchical storage management (HSM) system, data stored on tapes is typically staged onto disk and then retrieved from disk. This paradigm may not be appropriate for NVOD servers, if we try to save money on disks and use tape as the primary storage medium. The main reason is the limited disk bandwidth. When data is staged onto disks on-line, half of the disk bandwidth is used for the staging operation, and only the remaining half bandwidth is used for broadcasting. This makes little sense and is therefore not recommended. Of course, an NVOD server can always has a tertiary storage and stage a movie from the tertiary storage to a disk off-line before the movie broadcast begins.

Finally, the same arguments on tape also apply to CD’s. We can use the same layout strategies to support movies broadcast from CD’s directly. There is no rewind delay problem with CD’s. Unfortunately, the currently available CD’s have very limited bandwidth, and multiple CD’s are needed to broadcast a movie periodically. Therefore, it makes sense to use CD as a primary storage medium only if the cost of multiple CD drives is less than that of a magnetic disk.

7. The NVOD On-Demand Service Model

In the previous sections, we used the “broadcast” NVOD service model that assumed an NVOD server always broadcasts movies at regular intervals regardless if there is any viewing requests. In this section, we consider the alternate “on-demand” model for NVOD services. In this model, after a viewing request has been received, the NVOD server starts a waiting period of *t* minutes during which it accumulates all requests for the same movie. Thus, the maximum waiting time for a viewer is *t* minutes. After the movie starts, if the server receives more requests for the same movie, then after a maximum of *t* minutes, a new stream is started. In this model, since a new stream is started only when there is demand, server and network bandwidth can be saved. On the other hand, since a stream is multicast to many viewers, this model does not support fast forward or fast backward functions.

Although this “on-demand” model differs from the previous “broadcast” model, the video layout strategy proposed in Section 2 is immediately applicable to this model. Specifically, assume a disk can deliver a movie at *t*-minute intervals (see Example 1). After having received the first request, the server starts a timer of *t* minutes. When this timer expires, the disk arm starts to sweep from the outer zones to the inner zones, which takes *t* minutes for one sweep. But within each service round, only one segment is read, i.e., during the first sweep, it reads segments *1, 2, ..., n-1,* and during the second sweep, it reads segments *n, n+1,...,* (Figure 1). If a new request arrives at any time, then at the beginning of the next sweep, a new stream is started by reading the segments *1, 2, ...,* along with the segments for the first stream, in their corresponding service rounds. In this way, the system can have a maximum of *k* simultaneous streams for this movie (Figure 1), and the maximum waiting time for each viewing request is *t* minutes.

Finally, we point out that this “on-demand” model can use all of the techniques described in Section 4 and 5: placing a movie on multiple disks, placing multiple movies on a single disk, and their respective disk optimization strategies.

8. An Experimental NVOD Server Prototype

In the system model described in Section 3, we focused on the disk bandwidth requirement, but ignored the server and system software overhead, such as context switching, interrupt processing, etc. Therefore, the calculated values for the maximum number of streams or the minimum repeat interval supported by disk only provides a performance upper bound. In order to study the dynamic behavior of an NVOD server and the actual performance achieved by using the layout strategies presented above, we built an experimental server prototype. This prototype uses a Pentium-based system with 32MB of memory running a real-time kernel. The storage interface is a PCI-based fast/wide SCSI channel with two IBM DFHS drives attached to the bus. The network interface is a ZeitNet PCI-based ATM/OC-3 adapter card, which is connected to a client with an experimental hardware MPEG decoder via a FORE System ATM switch. For simplifying the experiment, only the basic video layout strategy was implemented in the NVOD server prototype. The movies stored in the two disks are MPEG transport streams encoded at a constant bit-rate of 3 Mbit per second.

In a previous study [2], we found that in a VOD system, each IBM DFHS disk could support a maximum of 8 streams, given that each stream is allocated a 512K memory buffer (with a 128K I/O transfer size). On the other hand, our analysis in Section 3 shows that, by using our proposed video layout strategy, each such disk can support a maximum of 13 streams in an NVOD system. Based on these observations, we set our goal for the experiment to use 256K buffer memory for each stream (with a 128K I/O transfer size) and to achieve 12 streams from each disk. However, with the 128K segment size, the ZeitNet ATM host adapter card sends a partly filled ATM cell at every segment boundary. This caused some confusion to the simple hardware MPEG decoder used in the experiment. Therefore, we set the segment size to 96K, which is an integer multiple of the ATM cell payload size (48 bytes).

As a result, when we allocated 192K buffer to each stream (the ping-pong buffer), each disk delivered 9 streams. In this case, the time used to send a 96K segment through the ATM interface was *T=250 ms*, which was not enough to cover the software overhead plus the disk access time for retrieving 10 or more segments. Next we increased the buffer per stream to 288K (three segments) to allow more time for the software overhead and the disk accesses. This caused each disk to deliver 12 streams as we expected.

Thus, with this configuration, we can deliver two movies and a total of 24 simultaneous streams from two disks to the client. At the client side, a viewer can randomly pick one of the two movies and switch back and forth between the 12 staggered streams for that movie. The movie quality is quite satisfactory. In addition, the prototype also implemented the capability of broadcast at intervals of *2t*, *3t*, ..., and the option of specifying the number of times a movie needs to be repeated. All of these selections can differ from movie to movie. Finally, the prototype supports both the “broadcast” and “on-demand” NVOD service models.

In summary, from the prototype, we have experimentally shown that, by using our proposed video layout strategy, each disk in an NVOD server can support 50% more streams than can a VOD system. At the same time, the buffer memory required by each stream is reduced almost by half. Because of the sequential access pattern enforced by the layout strategy, the effective disk utilization can almost achieve its theoretical upper bound. All these results meet our original design goals and expectations.

9. Summary

In this paper, we presented several novel movie layout strategies for NVOD servers and analyzed for each of these strategies, as summarized below:

- The basic layout strategy places a movie on a single disk with data interleaved in such a way that it supports a movie broadcast at a repeat interval of *t* minutes. This strategy achieves high disk utilization by enforcing sequential disk accesses. One drawback of this strategy is that the maximum achievable bandwidth is determined by the disk inner-zone bandwidths, while the higher outer-zone bandwidths are wasted.

- The Segment-Group-Pairing (SGP) strategy utilizes the higher bandwidths in the disk outer zones, but doubles the size of the memory buffers. When the buffer size for each stream increases from 256 K to 512 K, the disk bandwidth utilization is improved over the basic layout strategy by 7%.

- The Disk-Pairing (DP) strategy places a movie with a high bandwidth requirement (either a long movie or one coded at a higher bit rate) and a movie with a lower bandwidth requirement together on a pair of disks. This strategy utilizes the available disk bandwidth best, but requires that the two movies be broadcast at the same repeat interval. The improvement on disk bandwidth utilization over the basic layout strategy can be as much as 17% for some disk drives.

- When a single disk's bandwidth does not meet the bandwidth requirements for broadcasting a movie, the basic strategy can be extended to stripe the movie across two or more disks, and the aggregate bandwidths of multiple disks can be used to meet the bandwidth requirements. When a single disk's bandwidth is much larger than required for broadcasting a movie, the basic strategy can be extended to interleave multiple movies on a single disk to fully utilize that disk's bandwidth.

- It is also possible to broadcast a movie directly from tape by using the basic layout strategy (the SGP and DP strategies don't make sense here). Because of the rewind delay, two tape drives may be needed for broadcasting each movie. Therefore, using tape as the primary storage medium for NVOD servers makes sense only if the cost of the two tape drives is lower than the cost of a magnetic disk. This same argument can be applied to CD's. Because of the currently limited CD bandwidth, multiple CD's are needed to broadcast a movie. Thus, using CD as the primary storage media can be justified only when the multiple CD drives are cheaper than one disk drive.

- NVOD services can also be provided by using an "on-demand" model. In this model, the server delivers a stream only when there is a request. Each request may subject to a maximum waiting period of t minutes. This model may save some network bandwidth, but does not support fast forward and fast backward functions. All of the video layout strategies proposed in this paper apply to both the "broadcast" and "on-demand" models for providing NVOD services.

Finally, through an experimental NVOD server prototype, we have demonstrated that our designs are practical. Using the proposed layout strategies, each disk delivers 50% more streams than can VOD systems, and the buffer memory required is reduced by almost half. This may significantly decrease the server storage and memory costs.

Acknowledgment:

The authors wish to thank John Wilkes, Yitzhak Birk, and Glenna Mayo for their helpful comments on this work, especially for the discussions on the "equal-time" vs. "equal-size" mechanisms for the Segment-Group-Pairing layout strategy.

References:

[1]. Gemmell, D.J. and Han, J., "*Multimedia Network File Servers: Multi-Channel Delay Sensitive Data Retrieval,*" Multimedia Systems, 1(6):240-252, 1994.

[2]. Chen, Shenze and Thapar, Manu, "*Fibre Channel Storage Interface for Video-on-Demand Servers,*" Proc. of Multimedia Computing and Networking'96, San Jose, CA, Jan. 1996.

[3]. Shenoy, Prashant J. and Vin, Harrick M., "*Efficient Support for Scan Operation in Video Servers,*" Proc. of ACM Multimedia'95, San Francisco, CA, Nov. 1995, pp.131-140.

[4]. Rangan, P.Venkat and Vin, Harrick M., "*Efficient Storage Techniques for Digital Continuous Multimedia,*" IEEE Trans. on Knowledge and Data Engineering, Special Issue on Multimedia Information Systems, Aug. 1993.

[5]. Berson, S., Muntz, R., Ghandeharizadeh, S., and Ju, X., "*Staggered Striping in Multimedia Information Systems,*" in Proc. of SIGMOD'94, 1994.

[6]. Chen, Shenze and Thapar, Manu, "*Zone Bit Recording Enhanced Video Data Layout Strategies,*" Proc. of MASCOTS'96, San Jose, CA, Feb. 1996, pp.29-35.

[7]. Chen, Shenze and Thapar, Manu, "*I/O Channel and Real-Time Disk Scheduling for Video-on-Demand Servers,*" Proc. of 6th Int'l Workshop on Network and Operating System Support for Digital Audio and Video, Zushi, Japan, Apr. 1996, pp.113-120.

[8]. Buddhikot, M.M., Parulkar, G.M., and Cox, J.R. Jr., "*Design of a Large Scale Multimedia Storage Server,*" Computer Networks and ISDN Systems, 27, pp.503-517, 1994.

[9]. Yu, P., Chen, M.S., and Kandlur, D.D., "*Grouped Sweeping Scheduling for DASD-based Multimedia Storage Management*", Multimedia System Journal, 1:99-109, 1993.

[10]. Little, D.C. and Venkatesh, D., "*Prospects for Interactive Video-on-Demand,*" IEEE Multimedia, Fall 1994, pp.14-24.

[11]. Dan, Asit and Sitaram, Dinkar, "*Scheduling Policies for an On-Demand Video Server with Batching,*" Proc. ACM Multimedia'94, San Francisco, CA, Oct. 1994, pp.15-24.

[12]. Cleary, K. "*Video on Demand - Competing Technologies and Services,*" Proc. International Broadcasting Convention, Amsterdam, Netherlands, Sept. 1995, pp.432-437.

[13]. Kusayanagi, M. et al, "*ATM-based access network with multicast function for multimedia services,*" Proc. of the SPIE / Broadband Networks: Strategies and Technologies, Vol. 2450, Amsterdam, Netherlands, Mar. 1995, pp.46-56.

[14]. Sharobim, H.R., *“Dedicated server multicast routing for large scale ATM networks,”* Proc. of 3rd Int’l Conf. on Intelligence in Broadband Services and Networks, Heraklion, Greece, Oct. 1995.

[15]. IBM Disk Manual.

[16]. Birk, Yitzhak, *“Track-Pairing: a Novel Data Layout for VOD Servers with Multi-Zone-Recording Disks,”* Proc. of 2nd IEEE Int’l Conf. on Multimedia Computing and Systems, Washington D.C., May 1995.