Multimedia Systems
DOI 10.1007/s00530-008-0147-8

REGULAR PAPER

Storage technique for real-time streaming of layered video

Sooyong Kang · Sungwoo Hong · Youjip Won

Received: 20 September 2006 / Accepted: 30 September 2008

Abstract Scalable streaming technology has been proposed to effectively support heterogeneous devices with dynamically varying bandwidth. From the file system's point of view, scalable streaming introduces another dimension of complexity in disk scheduling. Most of the existing efforts on multimedia file systems are dedicated to I/O scheduling algorithm and data placement scheme that efficiently guarantee I/O bandwidth. The important underlying assumption in these efforts is that most of the multimedia file accesses are simple playback operations and therefore are sequential. However, this workload characteristic is not valid in scalable streaming environment. In a scalable streaming environment, i.e., when only a subset of imagery is retrieved, the playback does not necessarily coincide with the sequential access on the file. The current file structure and the file system organization leaves much to be desired for supporting scalable streaming service. In this work, we propose a file system scheme, *Harmonic Placement* to efficiently support scalable streaming. The basic idea of Harmonic placement is to cluster the frequently accessed layers together to avoid unnecessary disk seeks. The data blocks are partitioned into two sets with respect to the layers: *lower* layers and *upper* layers. In Harmonic placement, the data blocks in the lower layers are placed with respect to their frame sequence and the data blocks in the upper layers are clustered according to the layers they belong to. We develop elaborate performance models for three different file system schemes: *Progressive placement*, *Interleaved Placement* and *Harmonic Placement*. We investigate the performance of the file server with different file system schemes. It was found that file system performance is very sensitive to the file organization scheme. When most of the service requests are for low-quality video (e.g., 128 Kbits/s ISDN), Progressive placement scheme supports twice as many sessions as the Interleaved placement scheme. When most of the service requests are for high-quality video (e.g., 1.5 Mbits/s MPEG-2 DVD quality), Interleaved placement can support twice as many requests as Progressive placement. In both cases, Harmonic placement scheme yields the most promising performance.

Keywords Scalable streaming · File system · Storage · Multimedia · Layered encoding

Communicated by P. Shenoy.

Primitive version of this work has appeared on Proceedings of NOSSDAV '06, Providence, Rhode Island, USA.
This work is in part funded by KOSEF throught National Research Lab (ROA-2007-000-200114-0) and by HY-SDR center at Hanyang University.

S. Kang
Division of Information and Communications,
Hanyang University, Seoul, South Korea
e-mail: sykang@hanyang.ac.kr

S. Hong · Y. Won (✉)
Department of Electronics and Computer Engineering,
Hanyang University, Seoul, South Korea
e-mail: yjwon@ece.hanyang.ac.kr

S. Hong
e-mail: toggiya@ece.hanyang.ac.kr

1 Introduction

1.1 Motivation

Due to the advances in computer network and communication technology, it is possible to disseminate bandwidth-demanding high quality multimedia information in ubiquitous fashion. Advances in network technology not only brings the increase in network bandwidth but also the emergence

Published online: 10 December 2008

of more *diverse* communication mediums. The bandwidths of different communication mediums, e.g., fibre-to-the-home (FTTH), xDSL, Cable Modem, WiFi, Wi-PAN etc., vary by orders of magnitude. Further, increases in CPU clock speed and storage density as well as advances in low power technology for mobile-embedded devices enables the creation of a wide variety of information devices. These include not only legacy computers (e.g., desktop computer or notebook), but also various types of information appliances(e.g., Set-Top Box, personalized video recorder, PDA, Smartphone etc.). Key implication of the above-mentioned technology trend is the *variability* in the client's access speed and it is affected by the congestion state of the subnet, moving speed of the mobile station, and channel interference in the mobile cellular network and so on.

For real-time multimedia playback, variability in the client's access speed means that the same contents can be accessed with different playback rates. Layered Encoding, a state-of-art compression technology, has been proposed to effectively cope with the variability in available bandwidth. When contents are created using the layered encoding scheme, the streaming server can retrieve a certain subset of information and reconstruct meaningful imagery with this partial information. Layered encoding and scalable streaming relieves the burden of maintaining multiple files of the same content each having a different playback rate. Layered encoding saves a significant amount of storage overhead and file management overhead. A benefit of these technical advancements is that various sectors in industry are trying to replace existing cable service providers by delivering TV contents over the Internet Protocol [12]. Convergence of legacy broadcast media, e.g., cable TV or satellite, and Internet Protocol will further boost the rapid proliferation of streaming service. We envision that efficient support of bandwidth variability in streaming service will become more and more important.

Storage, delivery, and compression technology are the three major technologies which make real-time multimedia streaming possible. Among these, advanced in compression technology and network transport technology have increased the efficiency of providing multimedia service over dynamically changing user bandwidth availability. We feel that not enough attention has been given to the storage technology to efficiently support bandwidth variability in scalable streaming. The dominant workload in real-time multimedia streaming is *playback* operation. Storage and file system technology for real-time multimedia application has put great emphasis on timely retrieval of the data. Numerous algorithms in disk scheduling, file allocation, file system layout, and buffer cache management have been proposed for this issue. However, when the file is encoded with layers and playback rate dynamically changes, the notion of sequential playback does not necessarily coincide with sequential access on the file data blocks. This is because accessing a fraction of each frame may yield random-access-like behavior even though the application accesses the frame in sequential fashion. Thus, the layered encoding scheme introduces another dimension of complexity from file system's point of view. In this work, we aim to fill in the chasm between file system technology and scalable streaming of layer encoded content. When the file is placed contiguously, playback of a single layer multimedia file yields simple sequential scan. However, playback with partial retrieval may no longer yield sequential scan. This is because only a subset of individual frame information may be accessed. There has been a 40% annual improvement on disk transfer rate, which is mostly due to the increase in linear bit density on the magnetic surface. On the other hand, disk seek time has been increasing at a rate of only 8% [9]. Given the difference between the growth rate of the two important factors in disk service time, it becomes more important to organize the data *properly* with respect to workload so that disk seek time is minimized.

We propose a novel file system technique, *Harmonic Interleaving* to efficiently support scalable streaming. The basic idea of Harmonic Interleaving is to properly cluster the frequently accessed layer information together. By doing this, we aim to avoid unnecessary disk seek overhead and subsequently improve the file system efficiency.

1.2 Related works

File systems support for real-time multimedia playback has been the subject of intense research for nearly a decade and it has now reached a sufficient maturity. The objective of these works has been to examine the relationship between real-time requirements of individual playbacks and file system level data retrieval scheduling [5,15,19,22]. A number of works proposed disk striping techniques to efficiently support scalable streaming. Chieh et al. [7] proposed a file placement scheme where the video file is coded in a Laplacian or Gaussian pyramid. Keeton et al. [14] examined the issue of scalable video organization strategy in parallel disk arrays in a standard file server environment. There have been a number of proposals for efficiently servicing scalable streaming from the file system's point of view. To reduce unnecessary disk seek operations, Chang et al. [4] proposed to organize a file into frame groups. Within each frame group, data blocks in the same layer are clustered together. The basic idea in this work is similar to the Progressive placement in our work. Anastasiadis et al. [1] evaluated various disk striping schemes for layer encoded video. Song et al. [30] proposed Constant Time Length(CTL)-based block allocation in a single disk and to use it as a striping unit for disk array. Both of these works focused on determining the striping unit for disk array in scalable streaming. They did not deal with the file organization in a disk.

A user may access the file in non-sequential fashion. The possibility of VCR-like operations, e.g., fast-forward, fast-backward, etc., requires careful treatment in terms of the file placement and retrieval scheduling. A number of works exploited multi-resolution file organization to support interactive operations [2,6,16,27]. Ng et al. [20] developed a scheme to support multiple rate playback in large scale video servers. Their scheme allows the user to specify the playback rate and the server adjusts the playback rate via transmitting subset of frames. A layered encoding scheme is also used to enhance the fault tolerant retrieval of multimedia data. By distributing layers among different physical disks, the streaming server can survive disk failure with graceful degradation on quality of retrieved stream [28].

The encoding scheme is one of the most important constituents in real-time streaming service. To effectively handle the heterogeneity in network connection bandwidths and hardware capability of client terminals, a number of encoding schemes have been proposed. The simplest approach is to prepare several versions of the same video contents for several playback rates [11]. While this approach is conceptually simple, it entails significant storage overhead. To reduce this storage overhead, scalable encoding scheme exploits the scalability in temporal, spatial axis or both. Fine grain scalable coding [17,21] enables more sophisticated control over QoS with respect to the change of playback rate.

Bandwidth variation in small time scale requires changes in the set of selected layers. This layer selection process has to be tightly coupled with the encoding mechanism. Saparilla et al. [26] proposed an optimal layer adding/dropping policy. Cuetos et al. [8] proposed an layer selection algorithm for *stored* content. Kang et al. proposed to export the frame type information to router level so that the router can maintain multiple queues for each frame type. They developed the Active Queue Management algorithm which exploits the unequal importance of the frame type [25]. Rejaie et al. [24] developed a peer-to-peer networking technique for scalable video streaming. In their framework, the receiver dynamically selects the subset of sending peers to maximize the overall throughput. The receiver coordinates the delivery of the layer encoded stream. To survive short-term glitch due to quality adaptation, they proposed to maintain a certain amount of buffer at the receiver.

Modern journaling file systems, such as Sun ZFS, SGI XFS and Linux Ext3, equip advanced disk caching schemes at operating system kernels. These caching schemes can unburden disk bandwidth by using cached data instead of accessing disks. However, the effect of caching in streaming server rapidly degrades when multiple popular objects whose aggregated size exceeds the cache capacity. Hence, careful design of the file organization scheme is inevitable regardless of the existence of caching scheme.

This paper proposes a file organization scheme for scalable streaming of layered encoded contents. Unlike the other works in storage for scalable streaming, our work addresses the fundamental issue of how the layer-encoded file needs to be structured for scalable streaming. We developed an elaborate performance model which can provide rigorous ground for disk admission control, bandwidth allocation, or server capacity planning. The primitive version of this paper [13] only considered that the multimedia server has multiple number of video objects. In this paper, we extended the scope of our work that it also deal with the case when the multimedia server has only one hot video object.

The rest of the paper is organized as follows. Section 2 introduces three file organization techniques for multimedia content. Sections 3 and 4 develop a performance model for file system in single object multiple session and multiple object multiple session environment, respectively. In Sect. 5, we perform an experiment to verify the accuracy of the proposed model and compare the performance of different file organization techniques. Section 6 concludes the paper.

2 File organization schemes

We view a media file as a collection of logical storage units called segments. A segment can be a frame or group of pictures. Each storage unit consists of layers each of which has constant and pre-determined size. While the size of each layer can be different each other,[1] the same layers in all storage units have identical size. Streaming application dynamically selects the proper subset of layers for transmission for each logical storage unit. To satisfy real-time requirement of multimedia data retrieval, the disk scheduler uses a round-based disk scheduling scheme. In the round based disk scheduling scheme, the time line is partitioned into a fixed time interval called round. The disk retrieves a certain amount of data blocks in each round to satisfy the soft real-time constraints of the playback [31]. The amount of data blocks retrieved in a round can be constant or variable. When the disk retrieves constant amount of data blocks, it is called constant data length (CDL). When the disk retrieves a certain playback length's amount of data blocks, it is called constant time length (CTL) [3]. In CTL, the amount of data blocks read in a round may vary. In this work, we assume that the amount of data read in a round (or period) is determined based on the CTL. We model three different file organization schemes: Progressive placement, Interleaved placement, and Harmonic placement. Among them, Progressive and Interleaved placement schemes are two extreme and straightforward ways of organizing a file and are introduced for comparison with

[1] In this paper, we assume that the size of each layer increases as the layer index increases.

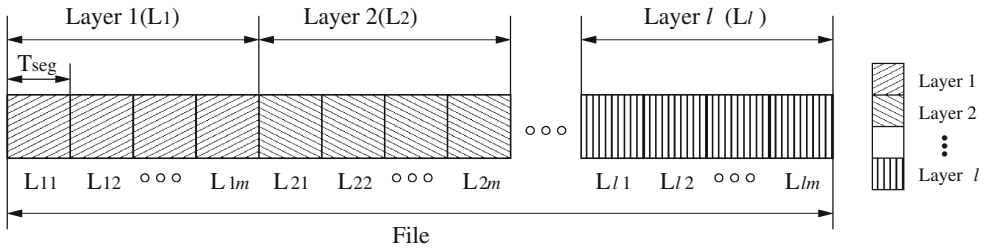


Fig. 1 Progressive placement scheme: in the figure, l and m mean the number of layers and the number of segments, respectively, in the object. And T_{seg} and L_{ij} mean the playback time of a segment and the data unit for j-th segment of i-th layer ($1 \leq i \leq l$, $1 \leq j \leq m$)

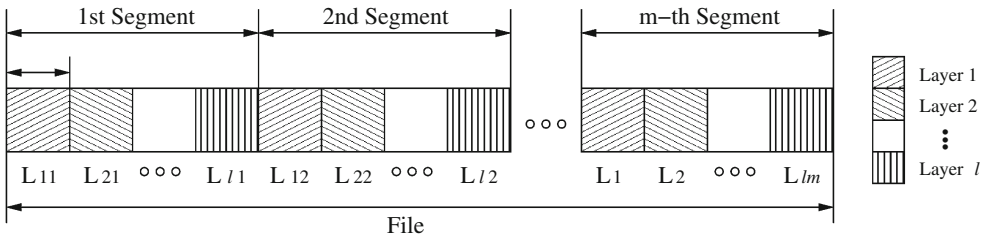


Fig. 2 Interleaved placement

the Harmonic placement scheme. The Progressive placement scheme clusters the data blocks in a layer together (Fig. 1). In this scheme, a file is physically organized as a sequence of layers and each layer consists of the data blocks of the respective layer for each logical unit.

This allocation scheme manifests itself when network bandwidth availability is very limited and when the streaming server can transport the lowest layers most of the time. The Progressive placement scheme entails significant disk head movement overhead when the server transports larger numbers of layers. We call the frame-major placement scheme Interleaved placement. This is plain sequential placement. As in legacy file organization, the file is physically organized as a sequence of logical units (frames) and each logical unit is organized as a collection of layers. When the streaming server retrieves data blocks in all layers, disk access yields a sequential access pattern not only from a logical aspect but also from a physical aspect. When the server transports only the proper subset of layers, either file access entails undesirable seek operation or server needs to discard some of the retrieved information. The Interleaved placement scheme manifests itself when the server transports most of the layers (Fig. 2).

In practice, the network bandwidth availability varies widely. Also, the speed of the client connection varies from a few hundreds Kbits/s (e.g., WCDMA) to tens of Mbits/s (e.g., VDSL). File organization should be carefully designed so that it can efficiently support a wide variety of QoS requirements. We believe that neither Progressive placement nor Interleaved placement scheme is desirable from the perspective of file system efficiency.

In this study, we propose a novel file organization scheme, *Harmonic Placement*. Figure 3 illustrates the underlying server organization for the proposed scheme.

Figure 3 illustrates the underlying server organization for the proposed scheme. User can access the content via wide variety of different network medium whose bandwidth capability varies widely, e.g., cable modem, xDSL, T1, 56 kbps modem etc. To effectively cope with the variety in connection medium, content providing system consists of a number of servers and each server is dedicated to harbor the contents for a given bandwidth connection. Once the connection speed of the incoming request is determined, the incoming request is directed to the appropriate server and is serviced from the respective server. Figure 4 illustrates the file organization under the Harmonic placement scheme.

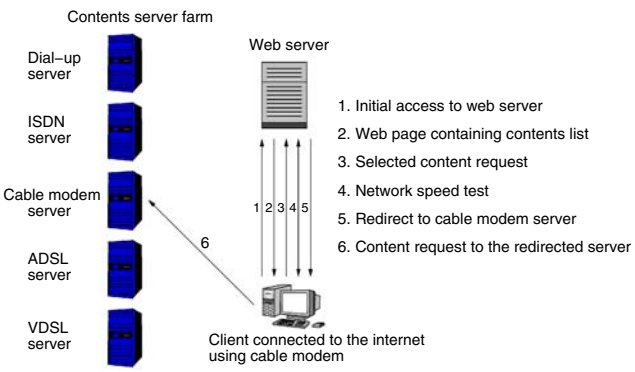


Fig. 3 Underlying service model

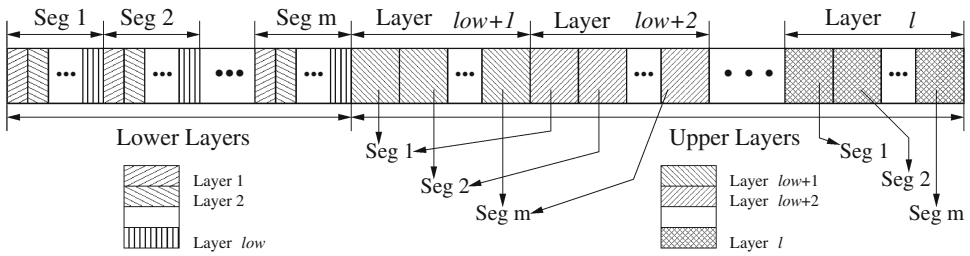


Fig. 4 Harmonic placement: in the figure, *low* means the number of lower layers in the object

In Harmonic placement, the layers are partitioned into two groups: a set of lower layers and a set of upper layers. We call these sets L_{lower} and L_{upper} in this paper. For example, with five layers, the layers can be partitioned as lower layers {L1, L2, L3} and upper layers {L4, L5}. Harmonic Placement adopts Progressive Placement for the upper layers and Interleaved Placement for lower layers. Using this scheme, we can reduce disk seek time by clustering the frequently accessed layers together. Hence, when only lower layers are accessed most of the time and information in the upper layers is rarely used, this scheme outperforms the other schemes. However, if the upper layers are accessed frequently, the disk seek overhead can result in lower performance. Therefore, the layers need to be carefully partitioned by considering both network bandwidth and the client device. The effectiveness of Harmonic placement is subject to the layer partitioning policy and the variability in network bandwidth availability.

3 Modeling the file system: single object multiple sessions

3.1 Background

We first develop a file system performance model where multiple sessions access a single object. We call this *single object multiple sessions (SOMS)* case. Later in this work, we present a generic performance model for multiple object multiple sessions case. We use *disk operation efficiency* as a performance metric for the file system. Disk operation efficiency is defined as $\frac{T_{\text{read}}}{T_{\text{read}}+T_{\text{overhead}}}$, where T_{read} and T_{overhead} correspond to data transfer time and the disk overhead. Disk overhead, in this paper, is defined as an aggregated seek time in a round. There are a number of algorithms to determine the next position of the disk arm and we adopt the C-LOOK disk scheduling algorithm in our modeling [29]. Table 1 illustrates the notations.

There exist a single multimedia object in the storage and multiple sessions accessing it. At any given time instance, each session accesses a different offset of the object. Disk head movement from one service offset to another induces seek overhead.

Let N denote the number of active sessions in a round and $N_i (0 \leq N_i \leq N)$ denote the number of sessions accessing data blocks in layers up to layer i, i.e., $L_1, \ldots, L_i$ in a round. The total amount of data read in the round corresponds to $\sum_{i=1}^{l} r_i \cdot R \cdot N_i$. The time to read data blocks excluding disk overhead can be formulated as in Eq. 1.

$$T_{\text{read}} = \frac{\sum_{i=1}^{l} r_i \cdot R \cdot N_i}{B} \tag{1}$$

After retrieving all data blocks for a round, disk head returns to the first cylindrical position for the next round (C-Look). We call this operation as *return seek*. As the number of sessions increases, the actual head position after *return seek* approaches to the beginning of the object regardless of the placement scheme assuming every new session starts from the beginning of the object. Hence, for simplicity, we assume that the *return seek* always positions the disk head to the beginning of the object. Then the *return seek* overhead only depends on the position where the last data block read in the round is located.

3.2 Progressive placement scheme

In the Progressive placement scheme, data blocks in the same layer are clustered together in the disk platter. The total seek overhead in the Progressive placement scheme consists of three components: seek within a layer (intra-layer seek), inter-layer seek, and *return seek*. Contiguously, disk seek for each session occurs in each layer range. Within a layer, the number of seeks and the average seek distance corresponds to $\min\{N_i, L_i\}$ and $\lfloor \frac{L_i}{\min\{N_i+1, L_i\}} \rfloor$. Inter-layer seek denotes the seek operation from the last access position of a layer to the first access position of the next layer. As can be seen in Fig. 5, inter-layer seek consists of cylinders in two consecutive layers ($d = d_1 + d_2$).

Assume that the number of sessions accessing layer k and $k+1$ is approximately the same (case 1) and the number of sessions accessing layer $k+2$ is much less than the number of sessions accessing layer $k+1$ (case 2). In case 1, the average seek distance in the layer k and $k+1$ are relatively shorter than the average seek distance in layer $k+2$. Also, since the size of layer k is smaller than that of layer $k+1$, the average seek distance in layer k is shorter than that of

Table 1 Notations

Notation	Description
Common	
B	Disk bandwidth
R	round length in seconds
l	Number of layers
$T(x)$	Seek time where x is the cylindrical distance
r_i	Data rate of layer i
N	Total number of active sessions
SOMS environment	
L_o	Size of the object in # of cylinders
L_i	data amount of layer i in # of cylinders
N_i	Number of sessions that reads layer i in a round
MOMS environment	
L_o^i	Size of the object i in # of cylinders
L_j^i	Data amount of layer j of object i in # of cylinders
N_o	Number of objects stored in disk
N^i	Number of sessions that serves object i
N_j^i	Number of sessions that reads layer j of object i in a round
α	Index of the first object that is served by some session
ω	Index of the last object that is served by some session

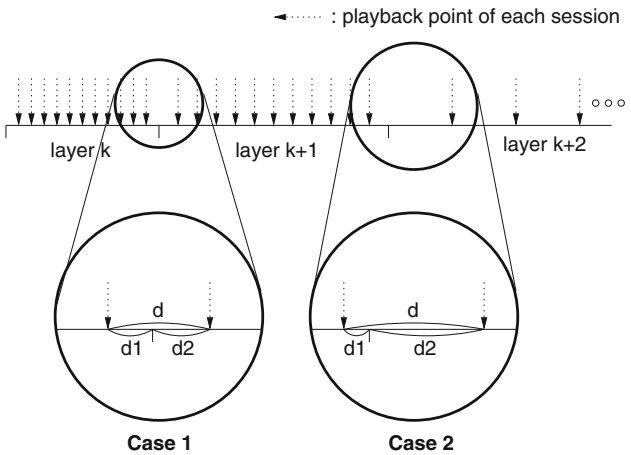


Fig. 5 Disk head movement from the last read point of a layer to the first read point of the next layer in Progressive placement scheme. Case 1 shows the case when every session reads data blocks from both layers and Case 2 shows when much less sessions read data blocks from the next layer

layer $k+1$. Hence, it is feasible to ignore the seek distance in layer k, d_1. In case 2, i.e., when disk head moves from layer $k+1$ to layer $k+2$, not only the size of total data in layer $k+1$ is much smaller than that of layer $k+2$ but also the number of sessions in layer $k+1$ is much larger than that of layer $k+2$. The average seek distance in layer $k+1$ is sufficiently shorter than that of layer $k+2$ and therefore we ignore it in our model. We can approximate the distance of head movement in both cases as $d = d_1 + d_2 \approx d_2$ and the seek time becomes $T(d) = T(d_1 + d_2) \approx T(d_2)$, which can be seen as the seek time to read the data block that should be read first in the next layer and is already included in the intra-layer seek model. As a result, we do not need to consider the head movement between layers separately without loss of generality. Therefore, the total seek overhead in layer i, $T_{body,i}$, can be modeled as in Eq. 2.

$$T_{\text{body},i} = \begin{cases} T(\lfloor \frac{L_i}{N_i+1} \rfloor) \cdot N_i & \text{if } N_i < L_i \\ T(1) \cdot L_i & \text{if } N_i \geq L_i \end{cases} \quad (2)$$

To calculate the distance of *return seek*, we first obtain the position of the disk head after reading all data blocks for a round. Let P_i be the probability that the last read block in a round is in layer i. Then $P_i = \frac{N_i - N_{i+1}}{N}$, for $i < l$ and $P_i = \frac{N_i}{N}$, for $i = l$. If the last data block belongs to layer i, the average distance of the *return seek* is $\lfloor \sum_{j=1}^{i-1} L_j + \frac{L_i}{N_i+1} N_i \rfloor$ (Fig. 6). The average time for *return seek* corresponds to $T\left(\left\lceil \sum_{i=1}^{l} P_i \cdot \left(\left\lfloor \frac{L_i}{N_i+1} N_i + \sum_{j=1}^{i-1} L_j \right\rfloor \right) \right\rceil\right)$.

Hence, the total seek overhead in a round can be calculated, in average, as in Eq. 3.

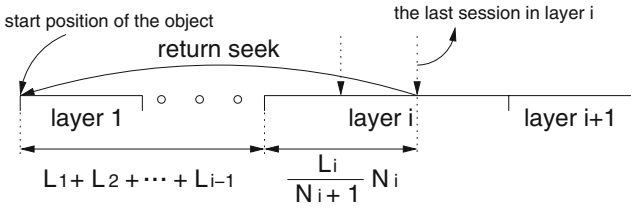


Fig. 6 The cylindrical distance of the *return seek* in the Progressive placement scheme

$$T_{\text{overhead}} = \sum_{i=1}^{l} T_{\text{body},i} + T\left(\left\lceil \sum_{i=1}^{l} P_i \cdot \left(\lfloor \frac{L_i}{N_i+1} N_i + \sum_{j=1}^{i-1} L_j \rfloor \right) \right\rceil\right) \quad (3)$$

3.3 Interleaved placement scheme

In the Interleaved placement scheme, data blocks are laid-out with respect to the segment sequence which is in most cases a frame sequence. Since the size of a segment is usually much smaller than the capacity of a cylinder, data blocks in the same segment are likely to be stored in the same cylinder.[2] Unlike Progressive Placement, the seek distance mainly depends on the interval between two adjacent sessions. The interval between two adjacent sessions depends both on the size of the object, L_o, and the number of concurrent sessions, N. To obtain the worst case performance bound, we assume that the playback offsets of the sessions are evenly distributed. Since the seek-time profile is a convex function, given a total seek distance, total seek time is maximized when all seeks are of the same distance. The number of seeks in a round in the Interleaved scheme is $\min\{N, L_o\}$ and the average seek distance can be calculated as $\left\lfloor \frac{L_o}{\min\{N+1, L_o\}} \right\rfloor$. Therefore, the average seek overhead except the *return seek* overhead in a round, T_{body}, is formulated as in Eq. 4.

$$T_{\text{body}} = \begin{cases} T\left(\lfloor \frac{L_o}{N+1} \rfloor\right) \cdot N & \text{if } N < L_o \\ T(1) \cdot L_o & \text{if } N \geq L_o \end{cases} \quad (4)$$

Total cylindrical distance from the start to the end of the data retrieval in a round is the same as the *return seek* distance (Fig. 7). The *return seek* overhead can be calculated as $T\left(\lfloor \frac{L_o}{N+1} \cdot N \rfloor\right)$.

Therefore, the total seek overhead in a round can be formulated as in Eq. 5.

$$T_{\text{overhead}} = T_{\text{body}} + T\left(\left\lfloor \frac{L_o}{N+1} \right\rfloor \cdot N\right) \quad (5)$$

3.4 Harmonic placement scheme

The Harmonic placement scheme divides data blocks into lower layer data blocks and upper layer data blocks. It applies the Interleaved placement scheme for lower layer data blocks and Progressive placement scheme for upper layer data blocks. Let low and L_{lower} be the number of lower layers and the number of cylinders needed to store lower layer data blocks, respectively. Then, $L_{\text{lower}} = \sum_{i=1}^{\text{low}} L_i$ and the seek overhead in the lower layer, $T_{\text{body,low}}$, can be calculated in a similar fashion as in the Interleaved scheme as in Eq. 6.

$$T_{\text{body,low}} = \begin{cases} T\left(\lfloor \frac{L_{\text{lower}}}{N+1} \rfloor\right) \cdot N & \text{if } N < L_{\text{lower}} \\ T(1) \cdot L_{\text{lower}} & \text{if } N \geq L_{\text{lower}} \end{cases} \quad (6)$$

The seek overhead in the upper layer i, $T_{\text{body},i}$, can be calculated in a similar fashion as in the Progressive scheme as in Eq. 7.

$$T_{\text{body},i} = \begin{cases} T\left(\lfloor \frac{L_i}{N_i+1} \rfloor\right) \cdot N_i & \text{if } N_i < L_i \\ T(1) \cdot L_i & \text{if } N_i \geq L_i \end{cases} \quad (7)$$

The sum of all seek overheads in each upper layer is $\sum_{i=low+1}^{l} T_{\text{body},i}$. We can verify from the formula that if there is no session access to the upper layers ($N_i = 0$, low+1 $\leq i \leq l$) the seek overhead in upper layers becomes 0.

Based upon whether the last block belongs to lower layer or upper layer, we can calculate the *return seek* distance as $\lfloor \frac{L_{\text{lower}}}{N+1} \cdot N \rfloor$ or $\lfloor \frac{L_i}{N_i+1} N_i + \sum_{j=1}^{i-1} L_j \rfloor$, respectively. The average distance of the *return seek* in Harmonic scheme becomes $\sum_{i=1}^{\text{low}} P_i \cdot \lfloor \frac{L_{\text{lower}}}{N+1} \cdot N \rfloor + \sum_{i=\text{low}+1}^{l} P_i \cdot \lfloor \frac{L_i}{N_i+1} N_i + \sum_{j=1}^{i-1} L_j \rfloor$. Therefore, the total seek overhead in a round can be calculated, on average, as in Eq. 8.

$$T_{\text{overhead}} = T_{\text{body,low}} + \sum_{i=\text{low}+1}^{l} T_{\text{body},i} + T\left(\left\lceil \sum_{i=1}^{\text{low}} P_i \cdot \lfloor \frac{L_{\text{lower}}}{N+1} \cdot N \rfloor + \sum_{i=\text{low}+1}^{l} \left(P_i \cdot \lfloor \frac{L_i}{N_i+1} \cdot N_i + \sum_{j=1}^{i-1} L_j \rfloor \right) \right\rceil\right) \quad (8)$$

4 Modeling the file system: multiple objects multiple sessions

In practice, there are a number of multimedia contents in the storage and a number of clients access different objects. We call this *multiple objects multiple sessions (MOMS)* case. In

[2] For example, for DVD quality MPEG-2 (1.5 Mbits/s) video content, I-frame size is around 150 kB and Seagate Barracuda (ST35641AS) has 500 GByte storage capacity with 16,353 cylinders. On the average, each cylinder stores approximately 30 MByte [10].

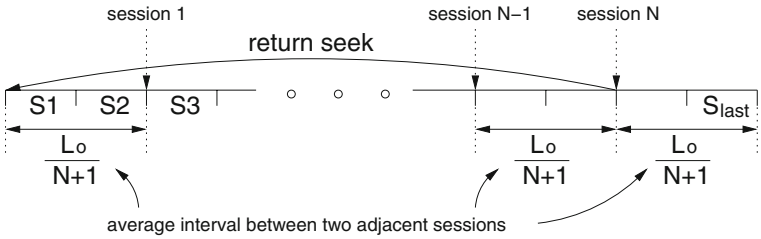


Fig. 7 The cylindrical distance of the *return seek* in the interleaved placement scheme

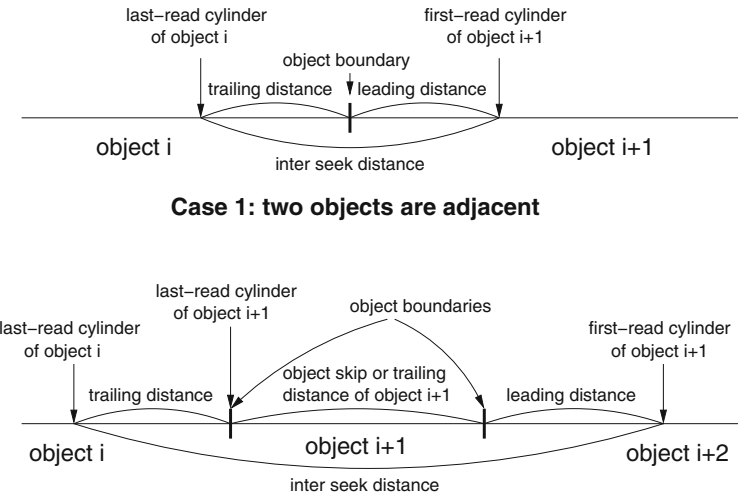


Fig. 8 *Inter-seek* distance in the MOMS environment. In Case 2, the object $i+1$ is just skipped because no session refers it

this section, we develop a performance model for MOMS case.

When there are multiple objects, we can categorize the disk seeks into three categories: *intra-object seek*, *inter-object seek* and *return seek*. Intra-object seek occurs while reading data blocks in different segments or blocks in an object. Inter-object seek occurs after reading the last necessary data block in an object and going forth to the first necessary data block in the next object. In the rest of this paper, we call the intra-object seek as *intra-seek* and inter-object seek as *inter-seek*. The *intra-seek* in multiple objects case is not any different from the *intra-seek* in a single object case and we can apply the same model used in the single object case to model the *intra-seek* in multiple objects case. Hence, in this section, we focus on modeling the *inter-seek* and the *return seek*.

Let object i and object j be the two objects accessed in consecutive fashion. The *inter-seek* distance between object i and object j consists of three components: *trailing distance* (d_t), *object skip* (d_s) and *leading distance* (d_l). Trailing distance, d_t, is the cylindrical distance from the last accessed block (*last-read block*) to the end (*last cylinder*) of object i. Object skip, d_s, is the seek distance when skipping objects between object i and j. Leading distance, d_l, is the seek distance from the beginning of object j to the first accessed cylinder in the object. Figure 8 illustrates the *inter-seek* distance.

The *inter-seek* overhead can be represented as $T(d_t + d_s + d_l)$. However, since d_l is modeled in the *intra-seek* model, it can be omitted in the *inter-seek* model. Object skip, d_s, in case 2 in Fig. 8 can be thought of as the trailing seek for object $i+1$ where the *last-read cylinder* corresponds to the start of the file. Hence, we can model the *inter-seek* overhead on a per object basis with only trailing seek, d_t. Let Δ_i denote the distance between the first cylinder and the *last-read cylinder* of object i (Fig. 9).[3] Then, d_t = object size $- \Delta_i$.

Before defining the seek overhead model for each scheme, we first formulate the data transfer time. Let N^i be the number of sessions accessing object i and $N^i_j (0 \leq N^i_j \leq N)$ be the number of sessions accessing layer $j (1 \leq j \leq l)$ of object i. Then, the total amount of data retrieved in a round corresponds to $\sum_{i=1}^{N_o} \sum_{j=1}^{l} r_j \cdot R \cdot N^i_j$. Given a disk transfer rate B, we can model the time to transfer data blocks read from the disk as in Eq. 9.

$$T_{\text{read}} = \frac{\sum_{i=1}^{N_o} \sum_{j=1}^{l} r_j \cdot R \cdot N^i_j}{B} \tag{9}$$

[3] Actually, Δ_i corresponds to the *return seek* distance in the SOMS environment.

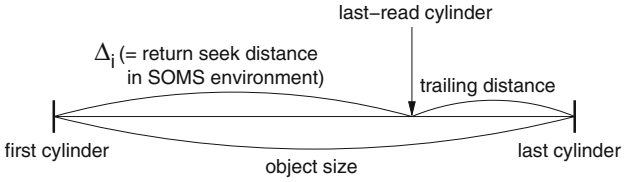


Fig. 9 The actual distance of the *trailing distance* in object i

Assume that the popularity of each object is the same. Let α and ω be the indices of the first and last object being accessed, respectively. The average values of α and ω are $\lceil \frac{N_o}{N+1} \rceil$ and $\lceil \frac{N_o}{N+1} N \rceil$, respectively.[4] As the number of sessions increases, the head position after *return seek* approaches to the beginning of the object α regardless of the placement scheme and therefore, we assume that the *return seek* always positions the disk head to the beginning of the object α. The *return seek* distance can be computed as the sum of the distance from the first cylinder of object α to the last cylinder of object $\omega - 1$, i.e., $\sum_{i=\alpha}^{\omega-1} L_o^i$, and Δ_ω (Fig. 10).

4.1 Progressive placement scheme

Since the *intra-seek* in the MOMS environment occurs in exactly the same manner as in SOMS environment, the overhead of *intra-seek* to read data blocks in layer j of object i, $T^i_{\text{intra},j}$, can be modeled as in Eq. 10.

$$T^i_{\text{intra},j} = \begin{cases} T(\lfloor \frac{L^i_j}{N^i_j+1} \rfloor) \cdot N^i_j & \text{if } N^i_j < L^i_j \\ T(1) \cdot L^i_j & \text{if } N^i_j \geq L^i_j \end{cases} \tag{10}$$

The total *intra-seek* overhead in object i becomes $\sum_{j=1}^{l} T^i_{\text{intra},j}$.

Let us calculate the trailing seek, d_t, for object i. The key ingredient is to determine the last-read cylinder in the object, from which we can derive the trailing seek. The last-read cylinder can be found using probability in a similar manner to the SOMS environment. Assume that $N^i > 0$. Let P^i_j be the probability that the last data block that is read in object i belongs to layer j. Then $P^i_j = \frac{N^i_j - N^i_{j+1}}{N^i}$, for $j < l$ and $P^i_j = \frac{N^i_j}{N^i}$, for $j = l$. When the last data block that is read in object i belongs to the layer j, the value of Δ_i becomes $\sum_{k=1}^{j-1} L^i_k + \lfloor \frac{L^i_j}{N^i_j+1} N^i_j \rfloor$ (Fig. 11), on average.

Hence, the expected value of Δ_i becomes $E[\Delta_i] = \sum_{j=1}^{l} \left(P^i_j \cdot \left(\lfloor \frac{L^i_j}{N^i_j+1} N^i_j + \sum_{k=1}^{j-1} L^i_k \rfloor \right) \right)$. Let $d_t^{i,j}$ denote the trailing distance in object i when the last-read block belongs to layer j. Then, $d_t^{i,j} = \sum_{k=1}^{l} L^i_k - \left(\sum_{k=1}^{j-1} L^i_k + \lfloor \frac{L^i_j}{N^i_j+1} N^i_j \rfloor \right) = \sum_{k=j+1}^{l} L^i_k + \left(L^i_j - \lfloor \frac{L^i_j}{N^i_j+1} N^i_j \rfloor \right) \approx \sum_{k=j+1}^{l} L^i_k + \lfloor \frac{L^i_j}{N^i_j+1} \rfloor$, on average. An expected value of d_t in object i, say $E[d^i_t]$, can be computed as $\sum_{j=1}^{l} P^i_j \cdot \left(\lfloor \frac{L^i_j}{N^i_j+1} \rfloor + \sum_{k=j+1}^{l} L^i_k \right)$. Hence, the expected *inter-seek* overhead in object i can be represented as $T\left(\lceil E[d^i_t] \rceil\right)$.

When $N^i = 0$ where object i is not accessed at all, $\Delta_i = 0$ and we can regard the first cylinder of the object as the last-read cylinder. Then, when the object i is not accessed, the trailing distance of object i becomes the number of cylinders occupied by the object i, L^i_o. And the *inter-seek* overhead becomes $T(L^i_o)$. Combining those two cases, we can represent the *inter-seek* overhead in object i, T^i_{inter}, as in Eq. 11.

$$T^i_{\text{inter}} = \begin{cases} T(\lceil E[d^i_t] \rceil) & \text{if } N^i > 0 \\ T(L^i_o) & \text{if } N^i = 0 \end{cases} \tag{11}$$

Since the sum of *intra-seek* overhead and *inter-seek* overhead in object i is $(\sum_{j=1}^{l} T^i_{\text{intra},j}) + T^i_{inter}$, the sum of all *intra-seek* overhead and *inter-seek* overhead in each object that occurs in a round can be represented as (*inter-seek* does not occur in the last object) $\sum_{i=\alpha}^{\omega} \sum_{j=1}^{l} T^i_{\text{intra},j} + \sum_{i=\alpha}^{\omega-1} T^i_{\text{inter}}$. The distance of the *return seek* is approximated as the sum of the object size from object α to object $\omega - 1$ and Δ_ω. The average *return seek* overhead of Progressive scheme can be calculated as $T\left(\left\lceil \sum_{i=\alpha}^{\omega-1} L^i_o + E[\Delta_\omega] \right\rceil\right)$. Finally, the overall seek overhead of the Progressive scheme in a round can be modeled as in Eq. 12.

$$T_{\text{overhead}} = \sum_{i=\alpha}^{\omega} \sum_{j=1}^{l} T^i_{\text{intra},j} + \sum_{i=\alpha}^{\omega-1} T^i_{\text{inter}} + T\left(\left\lceil \sum_{i=\alpha}^{\omega-1} L^i_o + E[\Delta_\omega] \right\rceil \right) \tag{12}$$

4.2 Interleaved placement scheme

The total *intra-seek* overhead during reading data blocks of object i in MOMS environment, T^i_{intra}, can be calculated, similar to in SOMS environment, as in Eq. 13.

$$T^i_{\text{intra}} = \begin{cases} T(\lfloor \frac{L^i_o}{N^i+1} \rfloor) \cdot N^i & \text{if } N^i < L^i_o \\ T(1) \cdot L^i_o & \text{if } N^i \geq L^i_o \end{cases} \tag{13}$$

The distance from the start of object i to the last-read block in the object, Δ_i, corresponds to $\lfloor \frac{L^i_o}{N^i+1} \cdot N^i \rfloor$, which is

[4] Actually, if the number of active sessions(N) are larger than or equal to the number of objects(N_o) those values become 1 and N_o, respectively, which means that every objects are served by at least one session.

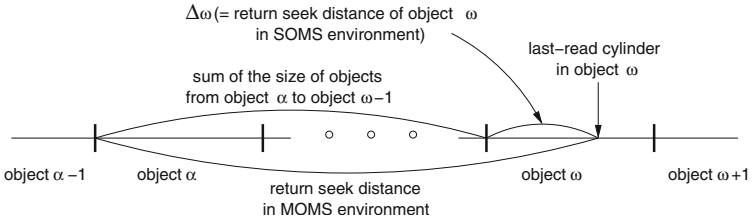


Fig. 10 The *return seek* distance in the MOMS environment

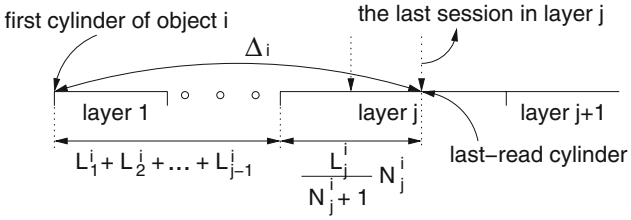


Fig. 11 The value of Δ_i in the Progressive placement scheme

derived from the *return seek* model in the SOMS environment (Fig. 12).

Subsequently, the trailing distance in object i becomes $\lfloor L_o^i - \frac{L_o^i}{N^i+1} \cdot N^i \rfloor$ and the *inter-seek* overhead, T_{inter}^i, is $T_{\text{inter}}^i = T\left(\lfloor L_o^i - \frac{L_o^i}{N^i+1} \cdot N^i \rfloor\right)$. The *return seek* distance of Interleaved scheme in MOMS environment is the sum of the value of Δ_ω and the sum of object sizes from object α to object $\omega - 1$. Hence, the return seek overhead becomes $T\left(\sum_{i=\alpha}^{\omega-1} L_o^i + \lfloor \frac{L_o^\omega}{N^\omega+1} \cdot N^\omega \rfloor\right)$. Finally, the total seek overhead of the Interleaved placement scheme in MOMS environment can be modeled as in Eq. 14.

$$T_{\text{overhead}} = \sum_{i=\alpha}^{\omega} T_{\text{intra}}^i + \sum_{i=\alpha}^{\omega-1} T_{\text{inter}}^i + T\left(\sum_{i=\alpha}^{\omega-1} L_o^i + \lfloor \frac{L_o^\omega}{N^\omega+1} \cdot N^\omega \rfloor\right) \quad (14)$$

4.3 Harmonic placement scheme

Let L_{lower}^i be the number of cylinders where data blocks belonging to the lower layers in object i are stored. Then $L_{\text{lower}}^i = \sum_{j=1}^{\text{low}} L_j^i$. The *intra-seek* overhead in lower layer of object i, $T_{\text{intra,low}}^i$, can be formulated, in a similar fashion to that of the Interleaved placement scheme, as in Eq. 15.

$$T_{\text{intra,low}}^i = \begin{cases} T(\lfloor \frac{L_{\text{lower}}^i}{N^i+1} \rfloor) \cdot N^i & \text{if } N^i < L_{\text{lower}}^i \\ T(1) \cdot L_{\text{lower}}^i & \text{if } N^i \geq L_{\text{lower}}^i \end{cases} \quad (15)$$

Intra-seek overhead in upper layer j of object i, $T_{\text{intra},j}$, can be obtained in a similar fashion to the Progressive placement scheme, as in Eq. 16.

$$T_{\text{intra},j}^i = \begin{cases} T(\lfloor \frac{L_j^i}{N_j^i+1} \rfloor) \cdot N_j^i & \text{if } N_j^i < L_j^i \\ T(1) \cdot L_j^i & \text{if } N_j^i \geq L_j^i \end{cases} \quad (16)$$

And the total *intra-seek* overheads in all upper layers of object i corresponds to $\sum_{j=\text{low}+1}^{l} T_{\text{intra},j}^i$.

Assume that $N^i > 0$. Depending upon the category where the last-read block belongs to (lower or upper), Δ_i is obtained in a similar fashion as in the Progressive scheme or Interleaved scheme. The expected value of Δ_i in the Harmonic scheme becomes $E[\Delta_i] = \sum_{j=1}^{\text{low}} P_j^i \cdot \lfloor \frac{L_{\text{lower}}^i}{N^i+1} \cdot N^i \rfloor + \sum_{j=\text{low}+1}^{l} P_j^i \cdot \lfloor \frac{L_j^i}{N_j^i+1} N_j^i + \sum_{k=1}^{j-1} L_k^i \rfloor$. Let j be the layer to which the last-read block belongs. When j belongs to the lower layer set, the trailing distance of object i, becomes $d_t^{i,j} = L_o^i - \lfloor \frac{L_{\text{lower}}^i}{N^i+1} \cdot N^i \rfloor \approx \lfloor L_o^i - \frac{L_{\text{lower}}^i}{N^i+1} \cdot N^i \rfloor$. When j belongs to the upper layer set, $d_t^{i,j} = L_o^i - \sum_{j=\text{low}+1}^{l} \lfloor \frac{L_j^i}{N_j^i+1} N_j^i + \sum_{k=1}^{j-1} L_k^i \rfloor \approx \sum_{j=low+1}^{l} \left(\lfloor \frac{L_j^i}{N_j^i+1} + \sum_{k=j+1}^{l} L_k^i \rfloor \right)$. Hence, the expected value of tailing distance corresponds to $E[d_t^i] = \sum_{j=1}^{\text{low}} P_j^i \cdot \left(\lfloor L_o^i - \frac{L_{\text{lower}}^i}{N^i+1} \cdot N^i \rfloor \right) + \sum_{j=\text{low}+1}^{l} P_j^i \cdot \left(\lfloor \frac{L_j^i}{N_j^i+1} + \sum_{k=j+1}^{l} L_k^i \rfloor \right)$. When $N^i = 0$, $\Delta = 0$ and the trailing distance becomes L_o^i. Therefore, the *inter-seek* overhead in object i of the Harmonic scheme, T_{inter}^i, can be calculated as in Eq. 17.

$$T_{\text{inter}}^i = \begin{cases} T\left(E[d_t^i]\right) & \text{if } N^i > 0 \\ T(L_o^i) & \text{if } N^i = 0 \end{cases} \quad (17)$$

Hence, the total seek overhead in object i, which is sum of the *intra-seek* and *inter-seek* overhead, is $T_{\text{intra,low}}^i + \sum_{j=\text{low}+1}^{l} T_{\text{intra},j}^i + T_{\text{inter}}^i$. We can get the total seek overhead in a round except the *return seek* overhead as $\sum_{i=\alpha}^{\omega} \left(T_{\text{intra,low}}^i + \sum_{j=\text{low}+1}^{l} T_{\text{intra},j}^i \right) + \sum_{i=\alpha}^{\omega-1} T_{\text{inter}}^i$. The *return seek* distance of Harmonic placement scheme is the sum of Δ_ω and object size from object α to object $\omega - 1$, i.e., $\left\lceil \sum_{i=\alpha}^{\omega-1} L_o^i + E[\Delta_\omega] \right\rceil$. Therefore, the total seek overhead of the Harmonic scheme

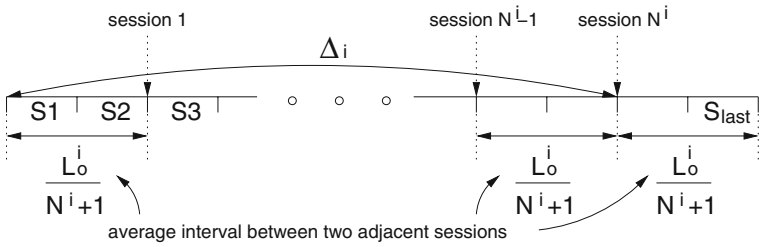


Fig. 12 The value of Δ_i in the Interleaved placement scheme: $S1, S1, \ldots, S_{last}$ are segments

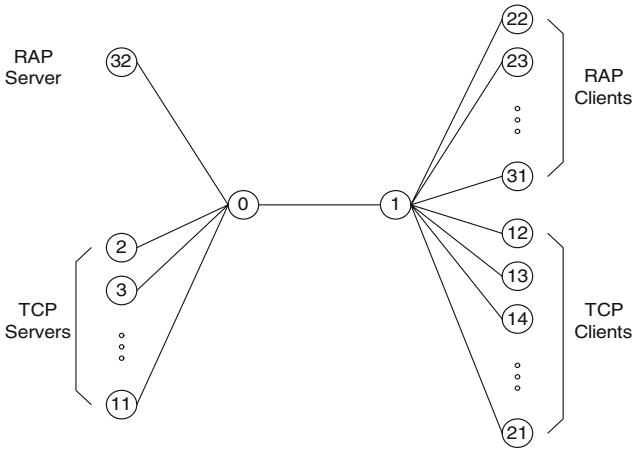


Fig. 13 Network topology: there exist 20 simultaneous sessions

in a round can be modeled as in Eq. 18.

$$T_{\text{overhead}} = \sum_{i=\alpha}^{\omega} \left(T^i_{\text{intra,low}} + \sum_{j=\text{low}+1}^{l} T^i_{\text{intra},j} \right) + \sum_{i=\alpha}^{\omega-1} T^i_{\text{inter}} + T\left(\left\lceil \sum_{i=\alpha}^{\omega-1} L^i_o + E[\Delta_\omega] \right\rceil \right) \quad (18)$$

5 Performance experiment

5.1 Experiment setup

We performed a physical experiment to examine the performance and effectiveness of the proposed scheme. Our experiment focuses on two issues. First, we verify the accuracy of the analytic models developed in this work. Second, and more importantly, we examine the performance of the individual placement schemes. We investigate the performance of the placement schemes in two terms of disk operation efficiency and service capacity.

The network bandwidth trace is obtained via a network simulator, *ns* [18]. Our network topology has a dumbbell setting (Fig. 13).

A single server (node 32) is servicing ten multimedia streaming clients using rate adaptive protocol (RAP) [23]. To simulate the actual network environment, we introduce ten HTTP sessions which share the bottleneck link. There are ten HTTP server nodes and ten HTTP client nodes. Each server and client pair forms a single session. Table 2 presents the parameters used in *ns*.

In our simulation, we use four bottleneck link (between node 0 and node 1) bandwidths: 2.6, 7.7, 15.4, and 30 Mbits/s. Each of the bottleneck link bandwidths is chosen to serve 20 subscribers of ISDN (128 Kbits/s), or different speed DSL (or Cable Modem) subscribers (384, 768 and 1.5 Mbits/s). In our experiment, encoding rate is 900 Kbits/s and there are four layers. Bandwidths of individual encoding layers are allocated with respect to [11]. Table 3 illustrates the bandwidth allocation for each layer.

Effectiveness of the Harmonic placement scheme critically relies upon the partitioning strategy of the layers. The partitioning strategy should properly reflect the access speed of the majority of the users. A server for ISDN clients may store multimedia objects with only layer 1 into the lower layer set and other layers into the upper layer set. In this case, the Harmonic placement scheme becomes exactly the same as the Progressive placement scheme. A server for 384 Kbits/s subscriber bandwidth can form the lower layer set with layer 1 and layer 2. A server for 768 Kbits/s subscriber bandwidth may put layer 1, layer 2, and layer 3 into the lower layer set. Finally, a server for high bandwidth subscribers(1.5 Mbits/s), may store multimedia objects by putting all layers into the lower layer set. In this case, the Harmonic placement scheme becomes exactly the same as the Interleaved placement scheme. In this way, using Harmonic placement scheme, four different kinds of servers that store multimedia objects in four different ways provide streaming service to four groups of clients whose subscriber line bandwidth is different from each other. Since large scale Internet Service Providers (ISPs) that provides streaming service to clients operate more than tens of homogeneous streaming servers in general, they can use Harmonic placement scheme only through reforming the homogeneous server farm into the heterogeneous server farm without additional hardware cost.

We use a SAMSUNG HM120JC disk (120 GB capacity, 75,000 cylinders and 30 MBytes/s bandwidth) in our experiment. Seek time profile is extracted by measuring seek time

Table 2 NS parameters

Parameter	value
Flows	20 (TCP × 10, RAP × 10)
Fine Grain Adaptation	True
Bottleneck Bandwidth	2,560 Kbits/s, 7,680 Kbits/s, 15,360 Kbits/s, 30,000 Kbits/s
Bottleneck delay	100 ms
Bottleneck queue type	Drop tail
TCP source type	TCP/Sack1
TCP delay Acks	False
Data packet size	1,024 byte
Acknowledge packet size	40 byte
Simulation length	3,600 s

Table 3 Bandwidth allocation for individual layers

Layer	Layer/Cumulative BDW (Kbits/s)	Subscriber line type	Partition
1	76.8/76.8	128 Kbits/s Dual ISDN	$\{L_1\}, \{L_2, L_3, L_4\}$
2	153.6/230.4	384 Kbits/s DSL or Cable Modem	$\{L_1, L_2\}, \{L_3, L_4\}$
3	230.4/460.8	768 Kbits/s DSL or Cable Modem	$\{L_1, L_2, L_3\}, \{L_4\}$
4	439.2/900	1.5 Mbits/s or over DSL	$\{L_1, L_2, L_3, L_4\}$,{}

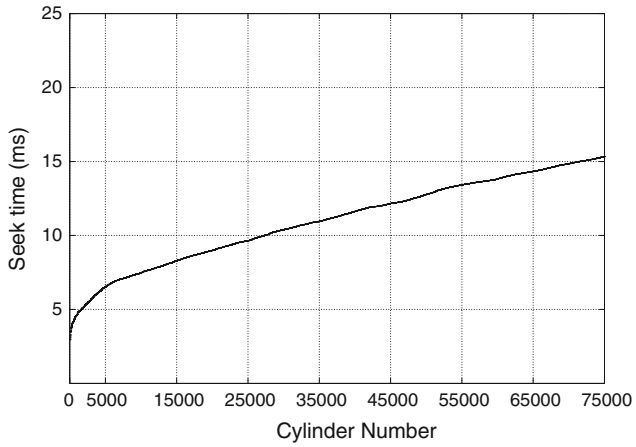


Fig. 14 Seek-time profile of the Samsung HM120JC disk

for all the logical address numbers. Figure 14 illustrates the seek-time profile of the disk.

Session arrival is modeled using the Poisson model. The playback length of the object is 1 h. Figure 15 illustrates the experiment process. The bandwidth trace is generated using *ns* and is converted into I/O trace (sequence of device block address) using a file system mapping table. Then, the sequence of I/O requests is fed to the physical disk.

5.2 Model accuracy

We first examine the relationship between the subscriber line bandwidth and QoS. Figure 16 illustrates the effect of bottleneck link bandwidth on the transmitted layers. The X-axis denotes the highest layer transmitted and the y-axis denotes the respective time fraction. With ISDN subscribers (128 Kbits/s), each session can transmit only layer 1. When the average available bandwidth is 384 Kbits/s, each session can transmit up to layer 2 in about 90% of the time and only layer 1 in about 10% of the time. When the average available bandwidth is 768 Kbits/s, each session can transmit up to layer 3 in about 80% of the time, up to layer 2 in about 10% of the time and only layer 1 in about 10% of the time. Finally, when the average available bandwidth is 1,500 Kbits/s, each session can transmit up to layer 4 in about 80% of the time, up to layer 3 in about 10% of the time and up to layer 2 in about remaining 10% of the time.

We verify the accuracy of our performance models by comparing the disk operation efficiency of individual placement schemes obtained from physical experiment and the disk operation efficiency computed from the performance model. Figure 17 illustrates disk operation efficiency of performance models and experiment results. There are 193 concurrent sessions in the single object case (Fig. 17a), and 131 concurrent sessions in the multiple objects case (Fig. 17b). The number of concurrent sessions is chosen in order to saturate the disk bandwidth with the highest subscriber line bandwidth (1.5 Mbits/s) in Progressive placement scheme. As we can see from the figure, performance model of each scheme closely approximates the experimental result. In case of single object (Fig. 17a), the difference between the model and the experiment is within 8% for Harmonic

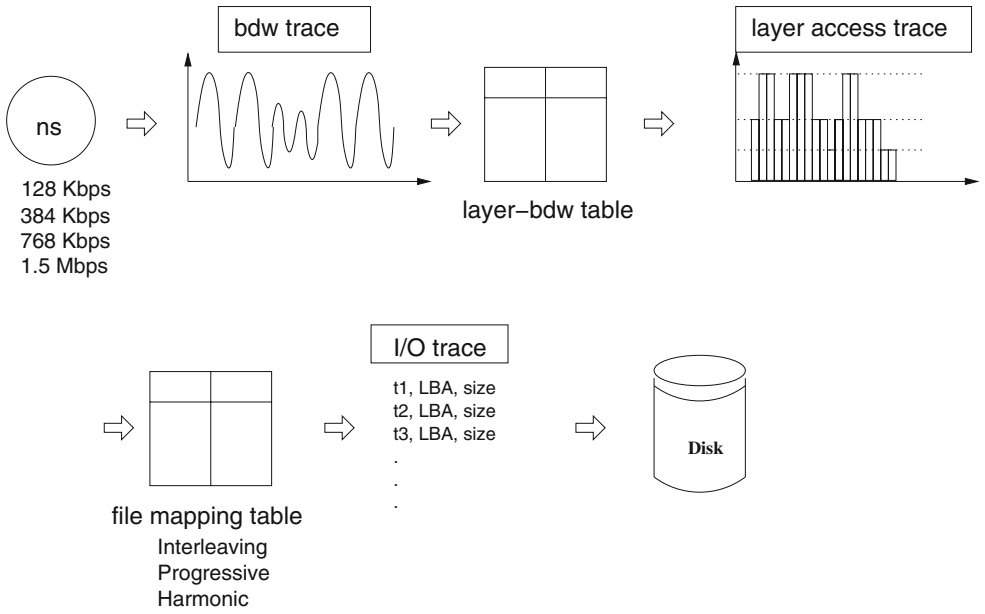


Fig. 15 Experiment methodology

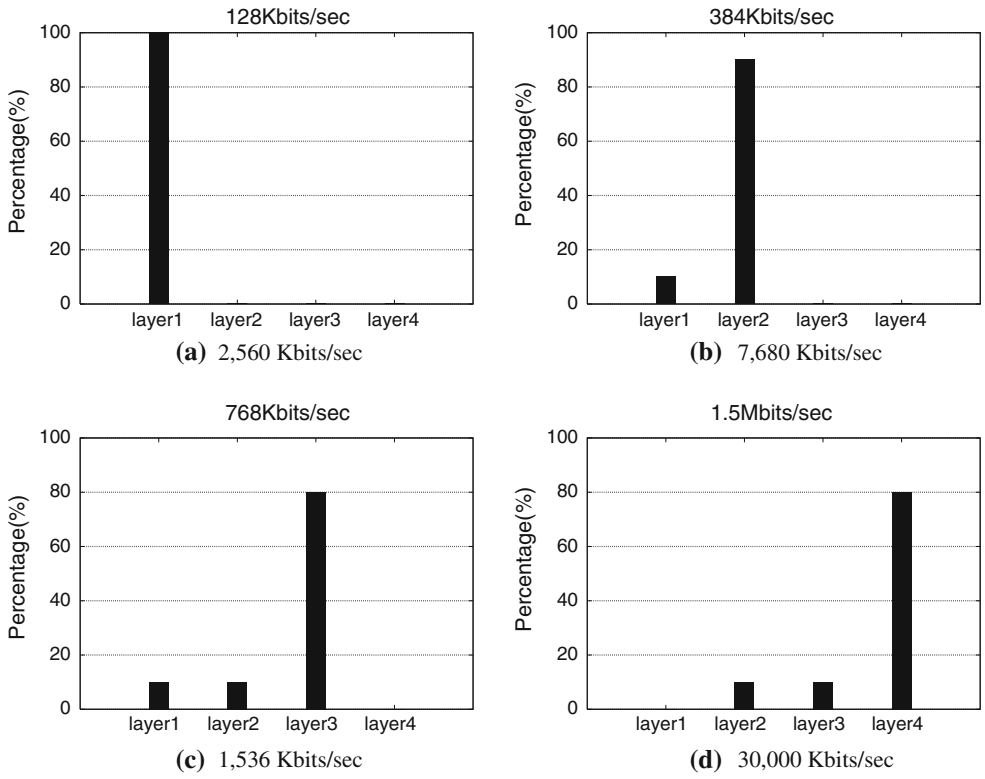
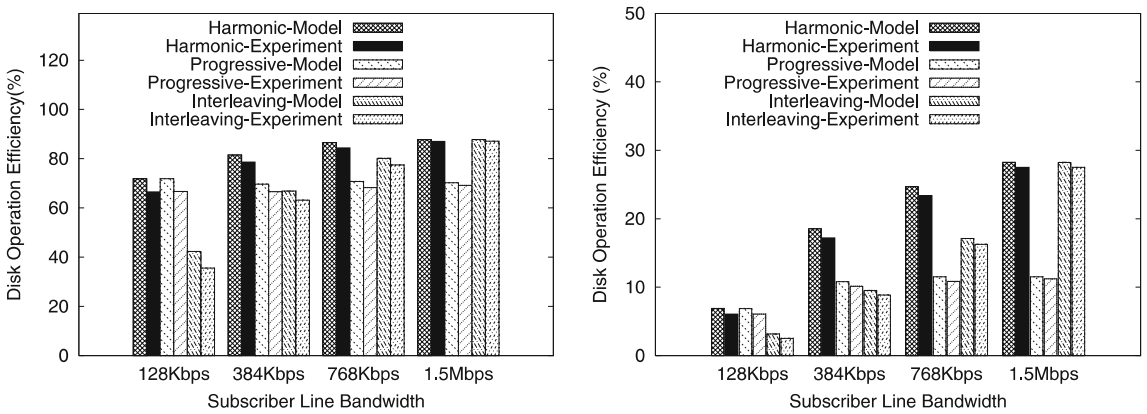


Fig. 16 Proportion of highest layers transmitted in each round under different bottleneck link bandwidth

placement scheme and Interleaved placement scheme and 4% for Progressive placement scheme. The efficiency of the model is always higher than the actual efficiency since we ignored the disk head movement from the last read point of a layer to the end of the layer in the inter-layer seek model ($T(d_1)$ in Sect. 3.2).

Let us look at Fig. 17a in more detail. The Harmonic placement scheme outperforms the other two. When the



(a) Single Object with 193 concurrent sessions **(b)** Multiple Objects with 131 concurrent sessions

Fig. 17 Disk operation efficiency: physical experiment versus analytical model

subscriber line is relatively slow, the Progressive placement scheme shows better performance than the Interleaved placement scheme. This is because Progressive placement scheme yields shorter disk seeks. On the other hand, when the subscriber line is relatively fast, the Progressive placement scheme causes not only more but also longer disk seeks than the Interleaved placement scheme. As the subscriber line becomes faster, the Interleaved placement scheme shows better performance than the Progressive placement scheme. Harmonic placement scheme yields the best performance in all cases.

Figure 17b illustrates the disk operation efficiency drawn from the analytical model and physical experiment in multiple objects case. Ten objects are stored in the disk and object access frequency is uniformly distributed. There are 131 concurrent sessions. Disk bandwidth is saturated with 131 sessions each of which the subscriber line bandwidth is 1.5 Mbps in Progressive placement scheme. The number of concurrent sessions in multiple objects case is much smaller than that in single object case. This is because accessing multiple objects entails longer disk seek operation and therefore disk operation becomes less efficient. The performance model of each scheme closely approximates the experiment result. The Harmonic placement scheme shows better disk operation efficiency than other schemes in all subscriber line bandwidths.

5.3 Disk utilization

We examine how the disk operation efficiency changes subject to the number of concurrent sessions. The graphs in Fig. 18 illustrate the disk operation efficiency under different number of sessions. They include both single object case as well as multiple objects case. Figure 18a–d show the disk operation efficiency for SOMS case from 30 to 193 sessions. The Harmonic placement scheme essentially becomes the same as the Progressive placement scheme and the Interleaved placement scheme for 128 and 1.5 Mbits/s subscribers, respectively. Due to this reason, Fig. 18a and d have only two plots. In all figures, the Harmonic placement scheme exhibits the best disk operation efficiency. When the subscriber line is slow, i.e., 128 or 384 Kbits/s, the Progressive placement scheme outperforms the Interleaved scheme and in the other cases the Interleaved placement scheme outperforms the Progressive scheme.

Figure 18e–h shows the disk operation efficiency for MOMS case. The number of sessions varies from 20 to 131. Harmonic placement scheme shows better disk operation efficiency regardless of the number of sessions. As the number of sessions increases, not only the disk operation efficiency of each scheme increases but also the difference of disk operation efficiency among each scheme increases as well.

5.4 Server capacity

From a practical point of view, the prime interest will be the effect of the placement scheme on the maximum number of sessions which the server can afford. We examine how the placement scheme affects the maximum number of concurrent sessions for each subscriber line bandwidth. Figure 19 illustrates the maximum number of concurrent sessions for SOMS and MOMS cases.

Figure 19a shows the maximum number of sessions for each placement scheme with a single object. The Harmonic placement scheme and the Progressive placement scheme can service approximately 2,600 simultaneous streams for a 128-Kbits/s subscriber line. For a 128-Kbits/s subscriber line, only data blocks in layer 1 are transmitted to clients. Since data blocks in layer 1 occupy 26 cylinders in the disk used in our experiment, the maximum number of seeks are 26, with an average of 1 cylinder of seek distance, regardless of the number of streams if the number of sessions

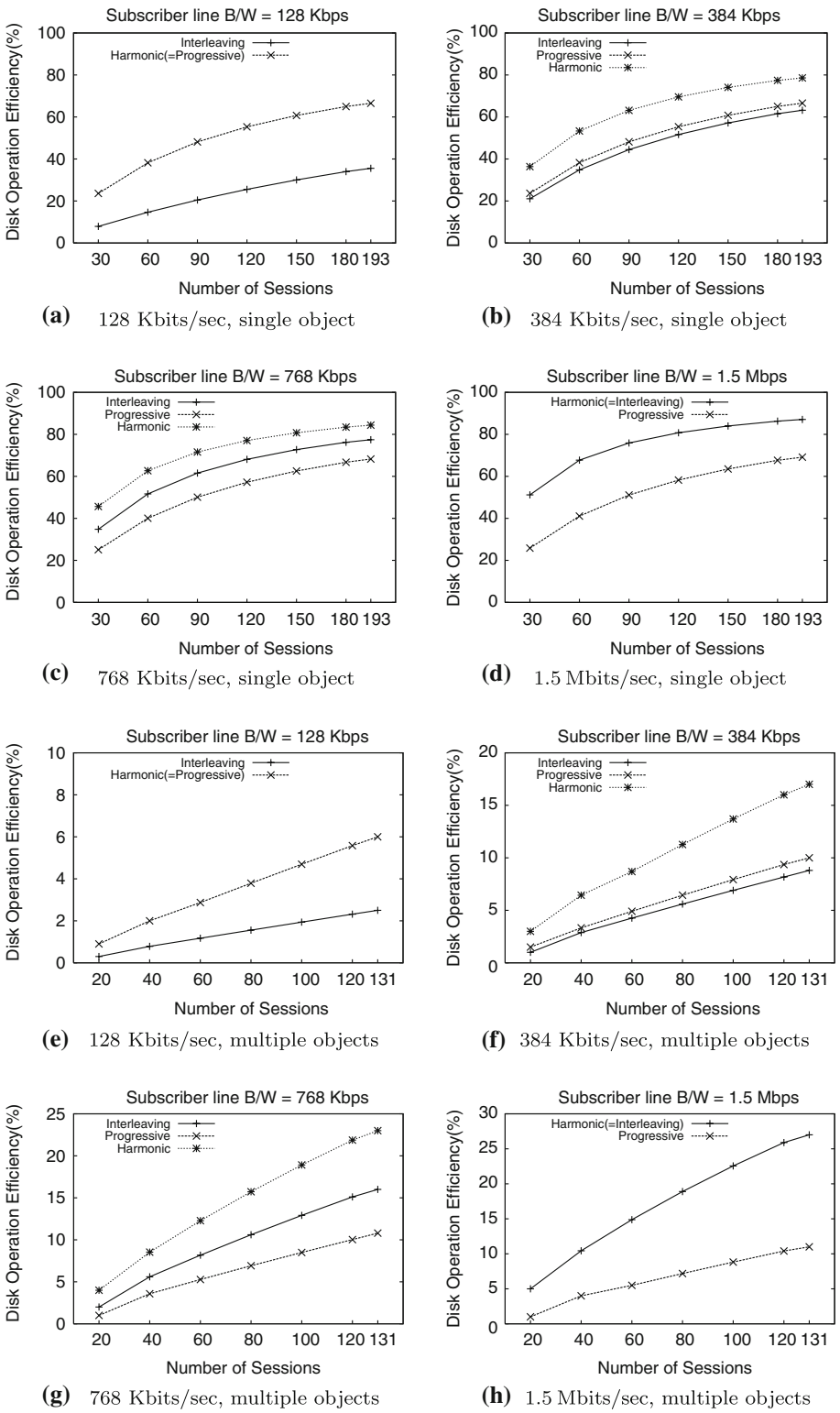


Fig. 18 Number of sessions versus disk operation efficiency

exceeds 26. Therefore, about 2,600 streams are served with at most 26 very short distance disk seeks and 1 (relatively) long distance *return seek* that occurs at the end of each round, which results in very high disk operation efficiency. Actually, in our experiment, the disk operation efficiency in this case was 96.4%. When the subscriber line bandwidth is 1.5 Mbits/s, the disk can service theoretically about 300 simultaneous sessions. However, since data blocks in layer

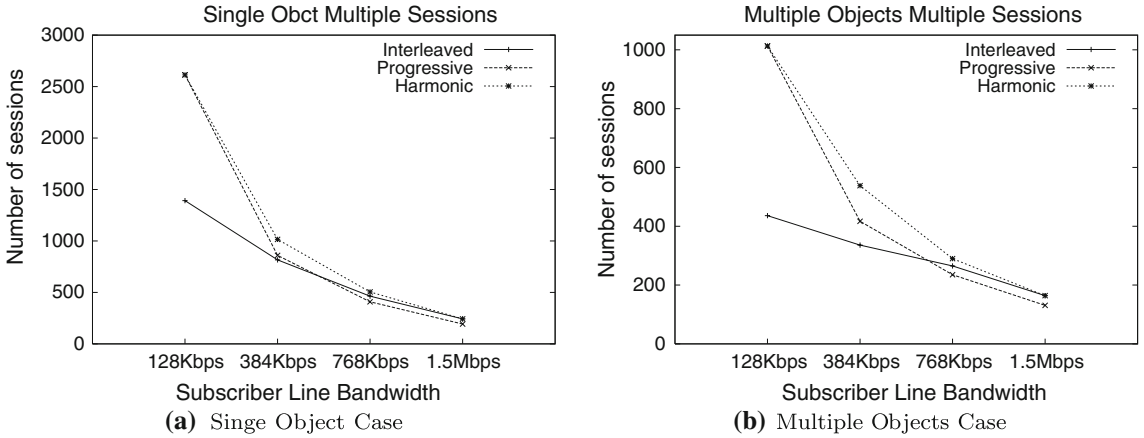


Fig. 19 Maximum number of concurrent sessions

4 are not transmitted about 20% of the time, the layer-based clustering of the Interleaved placement scheme cannot be fully exploited. The Interleaved placement scheme and Harmonic placement scheme can only service 243 simultaneous sessions. This corresponds to a disk operation efficiency of 87.72%. As the amount of data to be read in a round becomes smaller, disk overhead constitutes of dominant fraction of disk operation and, as a result, the effect of data placement scheme becomes more significant. Therefore, as the subscriber line bandwidth becomes smaller, the difference of the maximum number of sessions in each scheme becomes larger.

Figure 19b shows the maximum number of sessions for MOMS case. As for MOMS case, the Harmonic placement scheme can serve more sessions than the other schemes. The Harmonic placement scheme and the Progressive placement scheme can serve approximately 1,013 simultaneous streams, while the Interleaved placement scheme can serve only about 436 streams when the subscriber line bandwidth is 128 Kbits/s. This is due to the fact that the intra-object seek overhead in the Interleaved placement scheme is much larger than that of the others. However, because of the inter-object seek, the disk operates less efficiently for MOMS case. Consequently, the maximum number of sessions becomes much smaller when there are more objects. With 128 Kbits/s subscribers, the disk operation efficiency of the Harmonic placement scheme and the Interleaved placement scheme was about 33.2 and 7.9%, respectively. For a 1.5 Mbits/s subscriber line, the Harmonic placement scheme (also equivalent to Interleaved placement) and Progressive placement scheme can serve 164 and 131 sessions, respectively. The disk operation efficiency of the Harmonic (= Interleaved) placement scheme and the Progressive placement scheme correspond to 32.19 and 11.22%, respectively. As we have observed, the Interleaved placement scheme needs to be avoided especially when providing low-quality video streaming service to clients.

5.5 Caching effect

Modern operating system adopts buffer cache to maintain frequently accessed I/O objects in memory. The prime objective of using buffer cache is to reduce I/O latency via servicing incoming I/O request from main memory. Due to buffer cache effect, major fraction of the I/O requests are serviced from main memory. Since the buffer cache greatly enhances the efficiency of the I/O subsystem, it carries an important implication from our perspective. In this work, we propose new file placement technique to improve the overall system performance. It is possible that the effect of the proposed file placement technique can become less significant when major fraction of the I/O requests are serviced from buffer cache. To verify the caching effect, we conduct a set of experiment with disk cache enabled in the multiple objects environment.

Figure 20 shows the disk operation efficiency under varying number of sessions. In these experiments, buffer cache size is set to 512 MByte and 1 GByte, respectively. We use LRU algorithm for the cache replacement policy, and use Zipf distribution (with parameter set so that 80% of total sessions access top 20% of popular objects) to model the object popularity distribution. Different from the previous experiments, we use much larger number of objects. This is to avoid the situation where entire files are cached into main memory. There exist 259 objects in the disk (120 GB capacity). Given the number of sessions, the disk operation efficiency decreases as the number of objects in disk increases, since objects occupy the wider range of cylinders. As we can see from the figures, disk operation efficiencies in these experiments are smaller than those in Fig. 18. As the cache

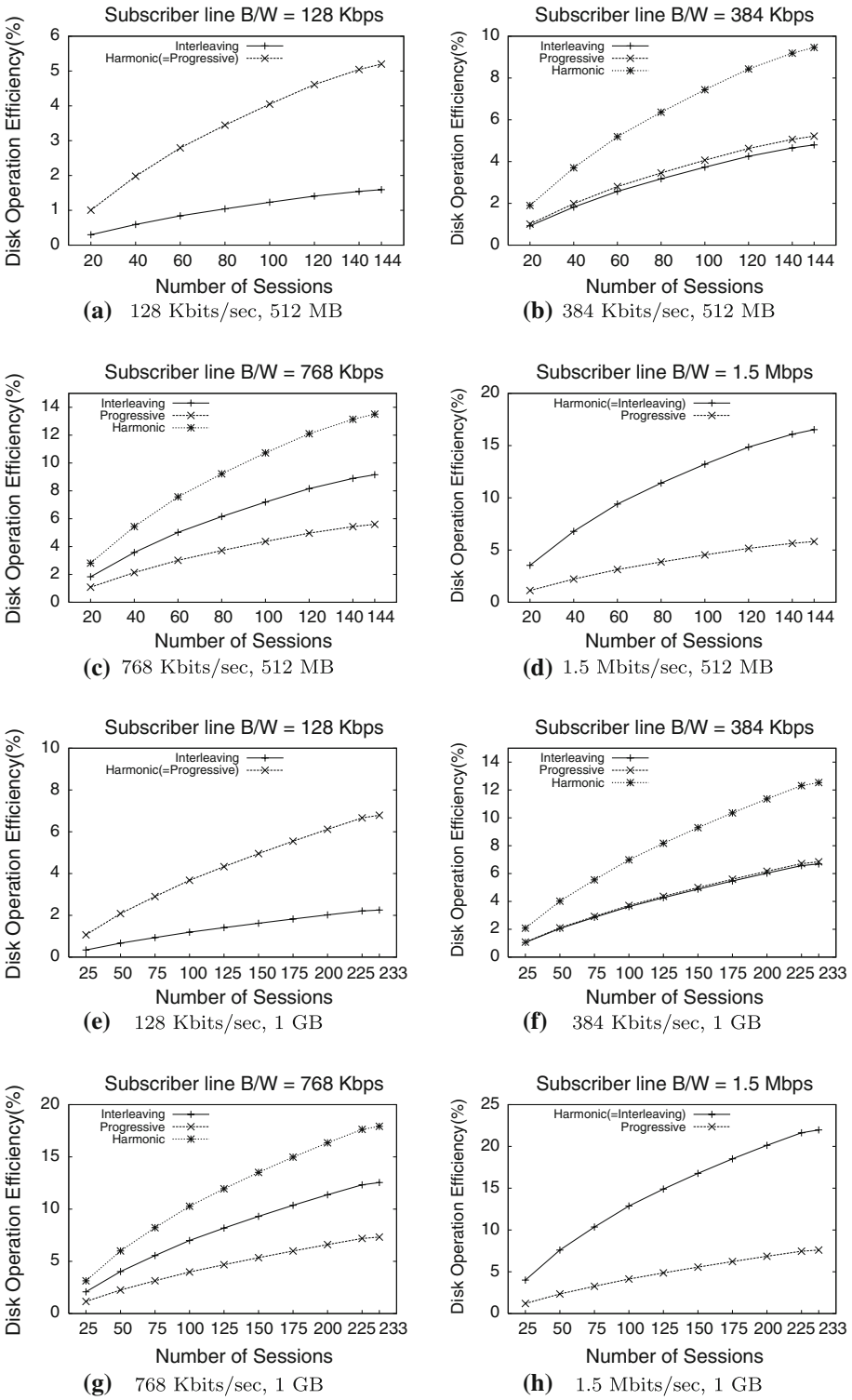


Fig. 20 Number of sessions versus disk operation efficiency with caching

size increases, the disk operation efficiency increases since the number of disk accesses decreases.

Figure 21 shows the number of sessions that can be served simultaneously. Not surprisingly, the performance difference among individual placement schemes decreases due to the caching effect. However, we can see that the Harmonic placement scheme outperforms others by up to 17% even with 2 GB of cache capacity. Hence, we can conclude that even

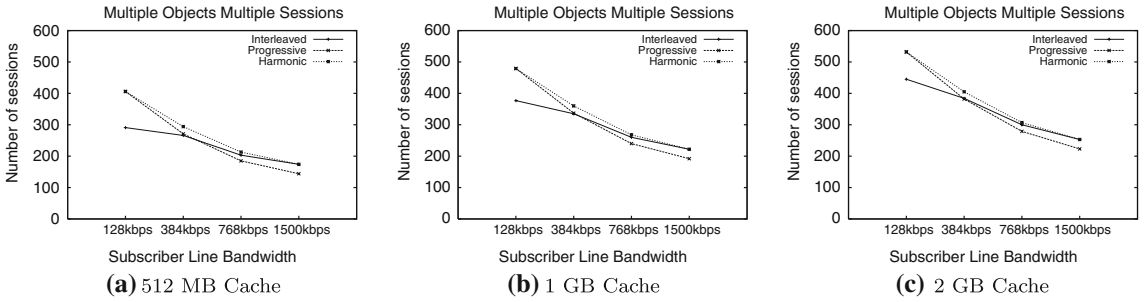


Fig. 21 Maximum number of concurrent sessions with caching

with the large-sized cache, the Harmonic placement scheme still brings significant performance improvement.

6 Conclusion

Scalable streaming technology effectively copes with the heterogeneous receivers and dynamically varying network bandwidth. Scalable streaming technology is the result of synergetic advancement in compression and network technology. Networking and compression technology has made collaborative advancement effectively reflecting each other. On the other hand, the file system technology which constitutes another important axis for multimedia streaming did not reflect the compression and networking technology in its direction of advancement. In this work, we sought to fill in the chasm between the file system technology and scalable streaming technology. Our goal was to determine if it is possible to support scalable streaming in a more efficient fashion from file system's point of view. Legacy file system for multimedia streaming put great emphasis on efficiently supporting sequential workload with real-time guarantee. However, in scalable video streaming, the playback of the content does not necessarily yield sequential access because only a subset of the file content is accessed in stride fashion. We examined three file organization schemes for multimedia content: Interleaved, Progressive, and Harmonic placement. The challenge in this research is to accurately predict the file system performance with a given file organization schemes. We successfully develop an elaborate model which effectively captures the file system behavior. The accuracy of the model is confirmed via physical experiment. The performance model of file organization scheme in this work provides rigorous ground for admission control, resource allocation etc. We found that Harmonic placement scheme is the most efficient way of organizing the video content, especially when the file is accessed via scalable streaming. Through this research, we conclude that the current file organization schemes which sequentially places the frames with respect to temporal order leaves much to be desired to be effectively used in scalable video streaming application. The Harmonic placement scheme yields the most promising performance in scalable streaming environment.

References

1. Anastasiadis, S.V., Sevcik, K.C., Stumm, M.: Scalable and fault-tolerant support for variable bit-rate data in the Exedra streaming server. ACM Trans. Storage **1**(4), 419–456 (2005)
2. Chang, E., Zakhor, A.: Scalable video data placement on parallel disk arrays. In: Proceedings of IS&T/SPIE International Symposium on Electronic Imaging: Science and Technology, vol. 2185. Image and Video Databaseds II, pp. 208–221, San Jose, CA, USA (1994)
3. Chang, E., Zakhor, A.: Admission control and data placement for vbr video servers. In: Proceedings of 1st International Conference on Image Processing, pp. 316–329 (1994)
4. Chang, E., Zakhor, A.: Disk-based storage for scalable video. IEEE Trans. Circuits Syst. Video Technol. **7**(5), 758–770 (1997)
5. Chen, M.-S., Kandlur, D.D., Yu, P.S.: Optimization of the grouped sweeping scheduling (gss) with heterogeneous multimedia streams. In: Proceedings of the first ACM international conference on Multimedia, pp. 235–242. ACM Press, New York (1993)
6. Chen, MS, Kandlur, D., Yu, P.S.: Storage and retrieval methods to support fully interactive playout in a disk-array-based video server. Multimedia Syst. **3**(3), 126–135 (1995)
7. Chiueh, T.C., Katz, R.: multi-resolution video representation for parallel disk arrays. In: Proceedings of ACM Multimedia '93, pp. 401–409, New York, USA, August (1993)
8. de Cuetos, P., Ross, K.W.: Adaptive rate control for streaming stored fine-grained scalable video. In: Proceedings of the 12th international workshop on Network and operating systems support for digital audio and video, pp. 3–12. ACM Press, New York (2002)
9. Hsu, W.W., Smith, A.J.: The performance impact of i/o optimizations and disk improvements. IBM J. Res. Dev. **48**(2), 255–289 (2004)
10. http://www.seagate.com
11. Real Networks. Inc. Helix producer user's guide, June (2002)
12. ITU-T. http://www.itu.int/itu-t/iptv/events/072006/docs/od/fgiptv-od-0001e.doc
13. Kang, S., Won, Y., Rho, S.: Harmonic interleaving: File system support for scalable streaming of layer encoded object. In: Proceedings of the 16th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'06) Newport, Rhode Island, May (2006)

14. Keeton, K., Katz, R.: Evaluating video layout strategies for a high-performeance storage server. Multimedia Syst. J. **3**(2), 43–52 (1995)
15. Kenchammana-Hosekote, D.R., Srivastava, J.: Scheduling Continuous Media on a Video-On-Demand Server. In: Proceedings of International Conference on Multi-media Computing and Systems pp. 19–28, Boston, MA, USA. IEEE, New York (1994)
16. Law, K.K.W., Lui, J.C.S., Golubchik, L.: Efficient support for interactive service in multi-resolution vod systems. Multimedia Syst. J. **8**(2), 133–153 (1999)
17. Li, W.: Overview of Fine Granularity Scalability in MPEG-4 Video Standard. IEEE Trans. Circuits. Syst. Video Technol. **11**(3), 301–317 (2001)
18. McCanne, S., Floyd, S.: Network simulator. www-mash.cs.berkeley.edu/ns (2002)
19. Mokbel, M.F., Aref, W.G., Elbassioni, K., Kamel, I.: Scalable multimedia disk scheduling. In: Proceedings of 20th International Conference on Data Engineering, 2004, pp. 498–509 (2004)
20. Ng, J.K.-Y., Xiong, S., Shen, H.: A multi-server video-on-demand system with arbitrary-rate playback support. J. Syst. Softw. **51**(3), 217–227 (2000)
21. Radha, H.M., van der Schaar, M., Chen, Y.: The mpeg-4 fine-grained scalable video coding method for multimedia streaming over ip. IEEE Trans. Multimedia **3**(1), 53–58 (2001)
22. Rangan, P., Vin, H., Ramanathan, S.: Designing an on-demand multimedia service. IEEE Commun. Magaz. **30**(7), 56–65 (1992)
23. Rejaie, R., Handely, M., Estrin, D.: Rap: an end-to-end rate-based congestion control mechanism for realtime streams in the internet. In: Proceedings of IEEE Infocom, pp. 1337–1345, New York, USA, March (1999)
24. Rejaie, R., Ortega, A.: Pals: peer-to-peer adaptive layered streaming. In: Proceedings of Network and Operating System Support for Digital Audio and Video Monterey, CA, USA, June (2003)
25. Kang, S.R., Zhang, Y., Dai, M., Loguinov, D.: Multi-layer active queue management and congestion control for scalable video streaming. In: Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS'04) (2004)
26. Saparilla, K.W., Despina, R.: Optimal streaming of layered video. In: Proceedings of IEEE INFOCOM (2000)
27. Shenoy, P.J., Vin, H.M.: Efficient support for interactive operations in multi-resolution video servers. Multimedia Syst. **7**(3), 241–253 (1999)
28. Shenoy, P.J., Vin, H.M.: Failure recovery algorithms for multimedia servers. Multmedia Syst. **8**(1), 1–19 (2000)
29. Silberschatz, A., Galvin, P.B.: Operating System Concepts. Wiley, New York (1998)
30. Song, M., Shin, H.: Replication and retrieval strategies for resource-effective admission control in multi-resolution video servers. Multimedia Tools Appl. **28**(3), 347–372 (2006)
31. Won, Y., Srivastava, J.: Smdp: minimizing buffer requirements for continuous media servers. ACM Multimedia Syst. J. **8**(2), 105–117 (2000)