Disk Striping Scalability in the *Exedra* Media Server

Stergios V. Anastasiadis* Kenneth C. Sevcik* Michael Stumm†

*Department of Computer Science
†Department of Electrical and Computer Engineering
University of Toronto

{stergios@cs,kcs@cs,stumm@eecg}.toronto.edu

ABSTRACT

Scalable network servers are increasingly identified as a critical component in the exponential growth of the Internet. We focus on media servers for variable bit rate streams and study the scalability of alternative disk striping policies, previously known and new. In contrast to results of previous studies, we show that the highest sustained number of streams that can be supported increases almost linearly with the number of disks. We believe that important role for our conclusion play the performance evaluation method that we use and a new disk space allocation technique that we introduce. Also, with reasonable technological projections, our arguments remain valid into the foreseeable future.

Keywords: media servers, storage systems, disk striping, performance evaluation, variable bit rate streams

1. INTRODUCTION

With the installed network bandwidth tripling every year,[1] scalable network servers are becoming the dominating bottleneck in the wide deployment of broadband services over the Internet.[2] In the present study, we address the fundamental question of potential scalability in the context of network servers for variable bit rate (video) streams.

Spatial and temporal compression have made practical the storage and transfer of digital video streams with acceptable quality. Standardization through the MPEG-2 specification has facilitated widespread distribution and use of compressed video content in a range of applications from studio post-production editing to home entertainment (e.g. Digital Versatile Disks). Although MPEG-2 streams can optionally be encoded in constant bit rate, it has recently been shown that equally acceptable quality can be achieved using variable bit rate encoding with average bit rates reduced by 40%.[3]

Vast storage and bandwidth capacity requirements of even compressed video streams make it necessary to stripe video files across multiple disks. Assuming that a media storage server serves requests for several different stream files, appropriate striping makes it possible to scale the number of supported streams to the limit of the server resources, independently of the particular stream files being requested by the clients. This becomes possible by retrieving different parts of each stream file from different disks, thus restricting the degree of imbalance in utilization among the disks.

A previous study focusing on a particular striping policy found that both load imbalance across disks and disk overhead caused stream striping to be efficient only on disk arrays of limited size.[4] In contrast, our experimental results show almost linear scalability for this policy, contradicting in part the conclusions of that work. We believe that the results differ in part due to the performance evaluation method used and a new disk space allocation technique that we introduce here. We also study other striping policies that increase system throughput and improve scalability even farther. Our present work has mainly focused on the striping problem for the common case of sequential playback. There are also other important resource management issues that remain to be addressed, some of which have already been studied to some extent by others.[5]

This paper is organized as follows. In Section 2, we discuss previous related work, and in Section 3 we describe the media server architecture that we use for our studies. In Section 4, we define several striping techniques, and in Section 5, we describe the experimental platform and the stream benchmarks that we used. In Sections 6 and 7, we study the performance and scalability of two disk striping policies, namely Fixed-Grain and Variable-Grain Striping respectively. In Section 8, we validate our arguments with more detailed disk model experiments, and in Section 9 we study the impact of projected disk technology improvements for the near future. Section 10 summarizes our findings.

2. RELATED WORK

Several media server designs either i) support only constant bit rate streams,[6–10] ii) make resource reservations assuming a fixed bit rate for each stream,[11,12] or iii) have only been demonstrated to work with constant bit rate streams.[13] However, recent quantitative measurements have shown that variable bit rate streams can achieve equally acceptable quality while requiring only 60% as high a bit rate.[3] Furthermore, detailed per round resource reservation for variable bit rate streams can increase the system throughput by more than a factor of two when compared to peak rate resource reservation.[14]

Paek et al., in a simulation study of constant rate streams, define *segmentation level* as (the base 2 logarithm of) the fixed number of disks across which stream data of a round are striped.[15] They show that disk efficiency is maximized when only one disk is accessed for a stream during each round, but at the cost of reduced system responsiveness. The above conclusion also holds for streams with multi-layer encoding, where subsets of a stream can be retrieved for lower resolution decoding. In a later paper, they describe a technique for reducing server buffer requirements with variable bit rate streams.[16]

Chang and Zakhor, in their study for multi-layer encoded streams, describe *non-periodic interleaving*, where an entire stream is split into parts equal to the number of disks and each part is stored on a separate disk.[17] They also describe *periodic interleaving*, where stream data for a round are stored on a single disk that changes round-robin every round. They find that although periodic interleaving does not improve the number of streams supported, it can lead to better system responsiveness. In later work, they show that periodic interleaving improves both the number of users and the system responsiveness when compared to a striping scheme where stream data for a round are striped across a fixed number (greater than one) of disks.[18] In a different study for variable bit rate streams, Chang and Zakhor suggest for future theoretical and experimental work the comparison of periodic interleaving to what they call hybrid data placement, where fixed-size blocks of stream data are placed round-robin across the disks.[19]

Shenoy and Vin study the fixed-size block striping of stream data, with both analytical and simulation methods.[4] They basically investigate a tradeoff between disk access overhead and load imbalance between disks. They find that as the number of disks increases, the load imbalance across the disks becomes higher. They conclude that a smaller block size should be chosen in order to compensate for the imbalance, but that leads to higher disk actuator overhead. They claim that the number of supported streams increases only sublinearly with the number of disks, and conclude that only disk arrays of limited size can operate efficiently. From their paper it is not clear how the load of the system is determined, and what process is assumed for the arrival of the client requests. Also, each individual block access is assumed to incur an extra disk arm movement, which can lead to an overestimatation of the disk overhead.

Reddy and Wijayaratne study the fixed-block striping technique called *Constant Data Length* (CDL), and the *Block-constrained Constant Time Length* (BCTL) technique, where each round of stream data are stored on a single disk (that changes round-robin every round) in multiples of a fixed (possibly large) block size.[20] They compare the throughput of the two techniques using eight disks and conclude that the improvement of BCTL (at large block sizes) is insignificant, which we believe is due to the particular block constraint they assumed. In this paper, we show that the best throughput of fixed-block striping technique can be improved by up to 50% using alternative striping policies.

In a study of single disk systems and constant bit rate streams, Triantafillou and Harizopoulos propose a technique called *Group Periodic Multi-Round Prefetching* that retrieves in a single round, multiple rounds worth of data.[21] In this paper, we introduce *Group-Grain Striping* which has a more general definition that makes it applicable to disk arrays and variable bit rate streams. It differs from the *Generalized Constant Data Length* described by Biersack et al. for single disk systems, where the retrieval of a fixed amount of data for a variable rate stream is distributed across a fixed number of rounds.[22]

3. THE *EXEDRA* MEDIA SERVER

We describe in this section the distributed media server architecture that we used in our studies.* The design that stores video streams on multiple disks is based on standard off-the-self components currently used in server systems. The streams are compressed according to the MPEG-2 specification, with constant quality quantization parameters

*exedra: an architectural term, meaning semicircular or rectangular niche (orig. Greek).

and variable bit rates. Clients with appropriate stream decoding capability send playback requests and receive stream data via a high-speed network, as shown in Fig. 1.

3.1. Server Architecture

We assume that the system operates using the server-push model.[23] When a playback session starts, the server periodically sends data to the client until either the end of the stream is reached, or the client explicitly requests suspension of the playback. The server-push model reduces the control traffic from the client to the server and facilitates resource reservation at the server side, when compared to a client-pull model. We also assume that data transfers occur in rounds of fixed duration T_{round}: in each round, an appropriate amount of data is retrieved from the disks into a set of server buffers reserved for each active client. Concurrently, data are sent from the server buffers to the client through the network interfaces. Round-based operation is typically used in media servers in order to keep the reservation of the resources and the scheduling-related bookkeeping of the data transfers manageable.

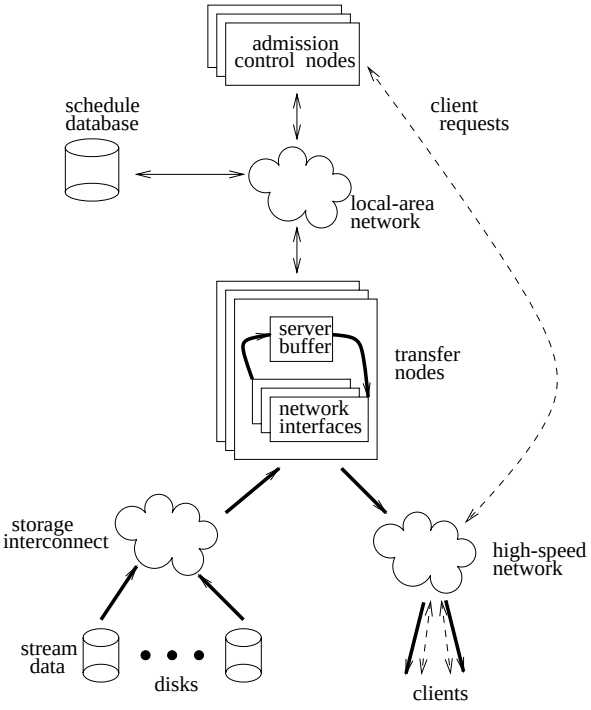


Figure 1. In the *Exedra* media server, stream data are retrieved from the disks and sent to the clients through the Transfer Nodes. Both the admission control and the data transfers make use of stream scheduling information maintained in the Schedule Database.

The large amount of network bandwidth required for this kind of service requires that the server be connected to the high-speed network through multiple network interfaces.[10] The amount of stream data periodically sent to the client is determined by the decoding frame rate of the stream and the resource management policy of the network. A reasonable policy would send to the client during each round the amount of data that will be needed for the decoding process of the next round; any other policy that does not violate the timing requirements and buffering constraints of the decoding client would be also acceptable.

The stream data are stored across multiple disks, as shown in Fig 1. Every disk is connected to a particular *Transfer Node*, through the *Storage Interconnect*, which could be either i) a standard I/O channel (e.g. Small Computer System Interface), ii) standard network storage equipment (e.g. Fibre-Channel[24]), or iii) a general purpose network (as with Network-Attached Secure Disks[25]). Recent research has demonstrated that it is possible to offload file server functionality to network-attached disks.[25] Although we believe that our design could be extended in a similar way, we leave the study of this issue for future work.

The Transfer Nodes are computers responsible for scheduling and initiating all data accesses from the attached disks. Data arriving from the disks are temporarily staged in the *Server Buffer* memory of the Transfer Node before being sent to the client through the high-speed network. We assume that the bandwidth of the system bus (such as the Peripheral Component Interconnect) is the critical resource within each Transfer Node that essentially defines the number and the capacity of the attached network or I/O channel interfaces.

Playback requests arriving from the clients are initially directed to an *Admission Control Node*, where it is decided whether enough resources exist to activate the requested playback session either immediately or within a few rounds. The computational complexity of the general stream scheduling problem is combinatorial in the number of streams considered for activation and the number of reserved resources.[26] However, we make the practical assumption that the acceptable initiation latency is limited, and use a simpler scheduling approach with complexity linear with the number of rounds of each stream and the number of reserved resources.[†] If a new playback request is accepted,

[†]Depending on the expected load and the required detail of resource reservation, the admission control process might still become a bottleneck. In that case, the admission control should be distributed across multiple nodes as shown in Fig. 1, taking into account tricky concurrency control issues that potentially arise.

commands are sent to the Transfer Nodes to begin the appropriate data accesses and transfers. We assume that standard local-area network technology is sufficient for handling the moderate control traffic between Admission Control and Transfer Nodes.

The *Schedule Database* maintains information on the amount of data that needs to be retrieved during each round for each stream and on which disks this data is stored. It also specifies the amount of buffer space required and the amount of data sent to the client by the Transfer Nodes during each round. The scheduling information is generated before the media stream is first stored and is used for both admission control and for controlling data transfers during playback. Since this information changes infrequently, it can easily be replicated to avoid potential bottlenecks.

3.2. Stride-Based Disk Space Allocation

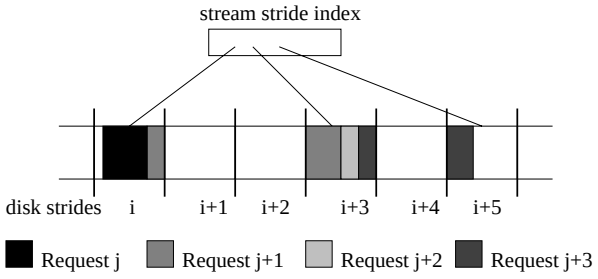


Figure 2. The stride-based allocation of disk space shown on one disk. A stream is stored in a sequence of generally non-consecutive fixed-size strides with a stride possibly containing data of more than one round. Sequential requests of one round are smaller than the stride size and thus require at most two partial stride accesses.

In our experiments, we use a new form of disk space allocation, called *stride-based allocation*. In stride-based allocation, disk space is allocated in large, fixed-sized chunks called *strides*, which are chosen larger than the maximum stream request size per disk during a round. This form of allocation makes sense in our system, because stored streams are accessed sequentially according to a predefined (albeit variable) rate; therefore, the maximum amount of data accessed from a disk during a round for a stream is known a priori. Stride-based allocation eliminates external fragmentation, while internal fragmentation remains negligible because of the large size of the streams, and because a stride may contain data of more than one round (see Fig. 2).

Although stride-based allocation seems similar to extent-based[27] and other allocation methods,[12] one basic difference is that strides have fixed size. More importantly, when a stream is retrieved, only the requested amount of data is fetched to memory, and not the entire stride (that is sequentially allocated on the disk platters). Another advantage of stride-based allocation is that it sets an upper-bound on the estimated disk access overhead during retrieval. Since the size of a stream request never exceeds the stride size during a round, at most two partial stride accesses will be required to serve the request of a round on each disk. This allows us to avoid the arbitrary number of actuator movements required by previously proposed allocation methods.[19]

It could be argued that storing the data of each disk request contiguously would reduce the disk overhead to a single seek and rotation delay (instead of two at most). However, we assume that variable disk data transfers can be specified in a granularity of a few tens of KBytes, while the total used storage space of the disks can reach hundreds of GBytes. Therefore, the cost of storage management for contiguous disk space allocation in highly utilized disks could become significant.

3.3. Reservation of Server Resources

We consider a system with D functionally equivalent disks. In the sequence definitions that follow, a zero value is assumed outside the specified range.

The stream *Network Sequence*, $\mathbf{S_n}$, of length L_n defines the amount of data, $S_n[i]$, $1 \leq i \leq L_n$, that the server sends to a particular client during round i after its playback starts. Similarly, the *Buffer Sequence* $\mathbf{S_b}$ of length $L_b = L_n + 1$ defines the server buffer space, $S_b(i)$, $0 \leq i \leq L_b$, occupied by the stream data during round i. The *Disk Sequence* $\mathbf{S_d}$ of length $L_d = L_n$ defines the total amount of data $S_d(i)$, $0 \leq i \leq L_d - 1$, retrieved from all the disks in round i for the client.

We assume that data are stored on the disks in strides. The stride size B_s is multiple of the *logical block* size B_l, which is multiple of the *physical sector* size B_p of the disk. Both disk transfer requests and memory buffer reservations are specified in multiples of the logical block size B_l. After taking into account logical block quantization issues, the disk sequence $\mathbf{S_d}$ can be derived from the network sequence $\mathbf{S_n}$ as follows: If

$$K^d(i) = \left\lceil \frac{\sum_{0 \leq j \leq i} S_n(j+1)}{B_l} \right\rceil$$

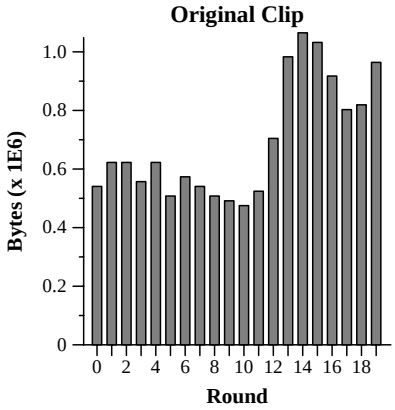


Figure 3. The data requirements of twenty consecutive rounds (one second each), in an MPEG-2 clip. Figure 4 shows how the clip is striped across two disks with alternative striping techniques.

specifies the cumulative number of blocks B_l retrieved through round i, then

$$S_d(i) = (K^d(i) - K^d(i-1)) \cdot B_l.$$

The *Disk Striping Sequence* $\mathbf{S_{md}}$ of length L_d determines the amount of data $S_{md}(i,k)$, $0 \leq i \leq L_d - 1$, that are retrieved from the disk k, $0 \leq k \leq D-1$, in round i. It can be generated from the Disk Sequence $\mathbf{S_d}$, according to the striping policy used.

We assume that each disk has edge to edge seek time $T_{fullSeek}$, single track seek time $T_{trackSeek}$, average rotation latency T_{avgRot}, and minimum internal transmission rate R_{disk}. The stride-based disk space allocation policy enforces an upper bound of at most two disk arm movements per disk for each client per round. The total seek distance can also be limited using a CSCAN disk scheduling policy. Let M_i be the number of active streams during round i of the system operation. We assume that the playback of stream j, $1 \leq j \leq M_i$, has been initiated at round l_j of system operation. Then, the total access time on disk k in round i of the system operation will have an upper-bound of:

$$T_{disk}(i,k) = 2T_{fullSeek} + 2M_i \cdot (T_{trackSeek} + T_{avgRot}) + \sum_{j=1}^{M_i} S^j_{md}(i - l_j, k)/R_{disk}$$

where $\mathbf{S^j_{md}}$ is the disk striping sequence of client j. $T_{fullSeek}$ is counted twice due to the disk arm movement from the CSCAN policy, while the factor two of the second term is due to the stride-based method. The first term should be accounted for only once in the disk time reservation structure of each disk. Then, each client j can be assumed to incur an additional maximum access time of

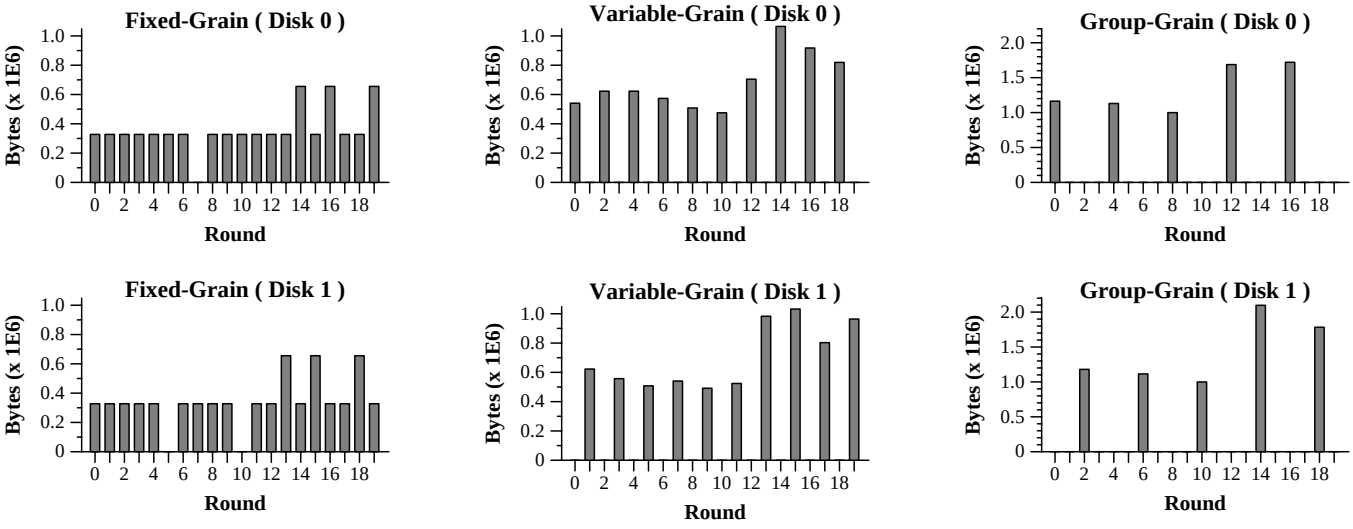
$$T^j_{disk}(i,k) = 2 \cdot (T_{trackSeek} + T_{avgRot}) + S^j_{md}(i - l_j, k)/R_{disk}$$

on disk k during round i, when $S^j_{md}(i - l_j, k) > 0$, and zero otherwise.

If R_{net} is the total high-speed network bandwidth available to the server, then the corresponding network transmission time reserved for client j in round i becomes $T^j_{net}(i) = S^j_n(i - l_j)/R_{net}$, where $\mathbf{S_n}$ is the Network Sequence of client j. The total server memory buffer reserved for client j in round i becomes $B^j(i) = S^j_b(i - l_j)$, where $\mathbf{S_b}$ is the Buffer Sequence of client j. Although, the above expressions for $T^j_{net}(i)$ and $B^j(i)$ are sufficient for the needs of the current study, accounting for available network bandwidth and buffer memory within each individual Transfer Node would require to split them into appropriate subexpressions.

4. DEFINITION OF STRIPING TECHNIQUES

In traditional storage systems, data access patterns are relatively hard to predict, making it difficult to determine optimal disk striping parameters.[28] However, with read-only sequential access being the common case for video



(a) Fixed-Grain (Bf=327,680)

(b) Variable-Grain Striping

(c) Group-Grain (G=2)

Figure 4. Data accesses for the clip of Fig. 3 using alternative striping techniques over two disks. With Fixed-Grain Striping, the needed blocks of size B_f are retrieved round-robin from the two disks every round. In Variable-Grain Striping, a different disk is accessed in each round, according to the byte requirements of the original clip. In Group-Grain Striping with G=2, stream data worth of two rounds are accessed from a different disk every two rounds.

streaming, it is possible to predict to some degree the expected system load requirements during retrieval, making it possible to determine appropriate disk striping parameters a priori for the storage and retrieval of the data.[23] In this section, we see how different striping policies exploit this characteristic of stored video streams.

4.1. Fixed-Grain Striping

With *Fixed-Grain Striping*, data are striped round-robin across the disks in blocks of a fixed size B_f, a multiple of the logical block size B_l defined previously. During each round, the required number of blocks are accessed from each disk. An example of Fixed-Grain Striping is shown in Fig. 4.(a). In the definition below, $_{modD}$ denotes the remainder of the division with D, and $_{divD}$ denotes the integer quotient of the division with D. We assume that

$$K^f(i) = \left\lceil \frac{\sum_{0 \le j \le i} S_d(j)}{B_f} \right\rceil,$$

specifies the cumulative number of blocks B_f retrieved through round i for a specific client. When $K^f_{divD}(i) - K^f_{divD}(i-1) = 0$, all blocks accessed for the client during round i lie on the same stripe of blocks. Then, the striping sequence $\mathbf{S^f_{md}}$ is equal to:

$$S^f_{md}(i,k) = D^f_0(i,k) \cdot B_f$$

where

$$D^f_0(i,k) = \begin{cases} 1, & \text{if } K^f_{modD}(i-1) < k_{modD} \le K^f_{modD}(i) \\ 0, & \text{otherwise,} \end{cases}$$

specifies the particular disks that need to be accessed at most once for the stream. When $K^f_{divD}(i) - K^f_{divD}(i-1) > 0$, the blocks accessed for the client during round i lie on more than one stripe, and the striping sequence becomes

$$S^f_{md}(i,k) = (K^f_{divD}(i) - K^f_{divD}(i-1) - 1) \cdot B_f + D^f_{>0}(i,k) \cdot B_f, \quad (1)$$

where

$$D^f_{>0}(i,k) = \begin{cases} 2, & \text{if } K^f_{modD}(i-1) < k_{modD} \leq K^f_{modD}(i) \\ 1, & \text{if } k_{modD} > max(K^f_{modD}(i-1), K^f_{modD}(i)\ ) \\ 1, & \text{if } k_{modD} \leq min(K^f_{modD}(i-1),\ K^f_{modD}(i)) \\ 0, & \text{otherwise.} \end{cases}$$

Intuitively, the first term in Eq. 1 accounts for blocks in stripes fully accessed (i.e., all D blocks, where D is the number of disks), while the second term accounts for blocks of stripes partially accessed in round i (i.e., fewer than D blocks).

4.2. Variable-Grain Striping

With *Variable-Grain Striping*, the data retrieved during a round for a client are always accessed from a single disk round-robin, as shown in Fig. 4.(b). The corresponding striping sequence becomes:

$$S^v_{md}(i,k) = (K^v(i) - K^v(i-1)) \cdot B_l,$$

when $i\ mod\ D = k$, with

$$K^v(i) = \left\lceil \frac{\sum_{0 \leq j \leq i} S_d(j)}{B_l} \right\rceil,$$

and $S^v_{md}(i,k) = 0$ when $i\ mod\ D \neq k$. Therefore, the Disk Sequence determines the particular single disk accessed and the exact amount of data retrieved during each round.

4.3. Group-Grain Striping

Variable-Grain Striping is a special case (with $G = 1$) of a method that we call *Group-Grain Striping*, where the amount of data required by a client over G rounds is retrieved every Gth round from one disk that changes round robin (see Fig. 4.(c), noting that the y-axis uses a different scale). The parameter G, $G \geq 1$, is called *Group Size*. The striping sequence for Group-Grain Striping is equal to:

$$S^g_{md}(i,k) = (K^v(i+G-1) - K^v(i-1)) \cdot B_l$$

when $i\ mod\ G = 0\ AND\ (i\ div\ G)\ mod\ D = k$, and $S^g_{md}(i,k) = 0$ otherwise.

As G increases, the fewer disk accesses lead to reduced disk overhead (although the access time per request is increased). On the other hand, the fixed round spacing between subsequent requests for a stream, basically divides the server into G virtual servers. The fixed group size G guarantees that two streams started from the same disk at rounds i and j with $i \neq j(mod\ G)$, do not have any disk transfers in a common round. This is different than increasing B_f in Fixed-Grain Striping, where accesses from different streams can randomly coincide on the same disk in the same round, resulting in the system saturating with fewer streams. We show later in more detail the potential performance effects of Group-Grain Striping. In particular, we show in section 9 that increasing G for a particular round time is advantageous with future expected changes in disk technology.

Although aggregation of disk transfers could also be achieved with an appropriate increase of round time, this could directly affect the responsiveness of the system by potentially increasing the initiation latency of each playback. Longer round time would also increase considerably the required buffer space.

5. EXPERIMENTATION ENVIRONMENT

We have designed and built a media server experimentation platform, in order to evaluate the resource requirements of the different striping techniques. The modules are implemented in about 12,000 lines of C++/ Pthreads code on AIX4.1, and currently run on a single node. The code is linked either to the University of Michigan DiskSim disk simulation package,[29] which incorporates advanced features of modern disks such as on-disk cache and zones for simulated disk access time measurements, or to hardware disks through their raw device interfaces. The indexing metadata are stored as regular Unix files, and during operation are kept in main memory. We used the MPEG-2 decoder from the MPEG Software Simulation Group for stream frame size identification.[30]

5.1. Prototype Implementation

The basic responsibilities of the media server include file naming, resource reservation, admission control, logical to physical metadata mapping, buffer management, and disk and network transfer scheduling.

With appropriate configuration parameters, the system can operate at different levels of detail. In *Admission Control* mode, the system receives playback requests, does admission control and resource reservation but no actual data transfers take place. In *Simulated Disk* mode, all the modules become functional, and disk request processing takes place using the specified DiskSim[29] disk array. There is also the *Full Operation* mode, that is not used in the current study, where the system accesses hardware disks and transfers data to fixed client network addresses.

Our experiments primarily use the system in Admission Control mode (except for our validation study, where we use the system in Simulated Disk mode). The *Admission Control* module uses circular vectors of sufficient length to represent the allocated time of each disk, the network time, and the buffer space respectively. On system startup, the disk time vectors are initialized to $2 \cdot T_{fullSeek}$, while the network time and buffer space are initially set to zero. When a new stream request arrives, the admission control is performed by checking against current available resources. In particular, the total service time of each disk in any round may not exceed the round duration, and the total network service time may also not exceed the round duration, while the total occupied buffer space may be no longer than the server buffer capacity. If the admission control test is passed, the resource sequences of the stream are added to the corresponding vectors of the module, and the stream is scheduled for playback.

5.2. Performance Evaluation Method

We expect that a fair performance evaluation method: i) demonstrates the capacity of the system for the performance parameters used, ii) is applicable to different hardware configurations, iii) is not biased towards any particular policies, iv) evaluates the expected practical operation of the system.

We assume that playback initiation requests arrive independently of one another, according to a Poisson process. The system load can be controlled through the mean arrival rate λ of playback initiation requests. Assuming that disk transfers form the bottleneck resource, in a perfectly efficient system there is no disk overhead involved in accessing disk data. Then, we choose the maximum arrival rate $\lambda = \lambda_{max}$ of playback initiation requests, that corresponds to system load 100%, to be equal to the mean service rate with which stream playbacks would complete in that perfectly efficient system. This makes it possible to show the performance benefit of arbitrarily efficient data striping policies. Subsequently, the mean service rate μ, expressed in streams per round, for streams of data size S_{tot} bytes becomes: $\mu = \frac{D \cdot R_{disk} \cdot T_{round}}{S_{tot}}$. Correspondingly, the system load ρ, is equal to: $\rho = \frac{\lambda}{\mu} \leq 1$, where $\lambda \leq \lambda_{max} = \mu$.

Another critical decision has to do with the admission control process. When a playback request arrives, it should be checked whether available resources exist for every round during playback. The test should consider the exact data transfers of the requested playback for every round and also the corresponding available disk transfer time, network transfer time and buffer space in the system. If this test fails for the next round, it has to be repeated until the first future round is found, where the requested playback can be started with guaranteed sufficiency of resources. Although we also tried alternative scheduling techniques proposed previously,[20] we did not notice significant performance improvements. We believe that this is due to the Poisson arrival process that we assume (as opposed to all requests arriving at the beginning[20]), which naturally distributes transfers from different clients across the disks of the array to different rounds.

We define as *lookahead distance* H_l the number of future rounds that are considered as candidate rounds for initiating the stream for each request before it is rejected. Playback requests not accepted are turned away rather than being kept in a queue. Practically, a large lookahead distance allows a long potential waiting time for the initiation of the playback. It cannot be unlimited in order for the service to be acceptable by the users. On the other hand, setting the lookahead distance too small can prevent the system from reaching full capacity.

We set the *basic lookahead distance* H_l^{basic} to be equal to the mean number of rounds between request arrivals $H_l^{basic} = \lceil \frac{1}{\lambda} \rceil$. Intuitively, setting $H_l = H_l^{basic}$ allows the system to consider for admission control the number of upcoming rounds that will take (on average) for another request to arrive. More generally, we define as *lookahead factor* F_l the fraction $F_l = \frac{H_l}{H_l^{basic}}$.

As the basic performance metric we choose the expected number of active playback sessions that can be supported by the server. The objective is to make this number as high as possible.

Content Type	Avg Bytes per rnd	Max Bytes per rnd	CoV per rnd
Science Fiction	624935	1201221	0.383
Music Clip	624728	1201221	0.366
Action	624194	1201221	0.245
Talk Show	624729	1201221	0.234
Adventure	624658	1201221	0.201
Documentary	625062	625786	0.028

Table 1. We used six MPEG-2 video streams of 30 minutes duration each. The coefficient of variation shown in the last column changes according to the content type.

Seagate Cheetah ST-34501	
Data Bytes per Drive	4.55 GByte
Average Sectors per Track	170
Data Cylinders	6,526
Data Surfaces	8
Zones	7
Buffer Size	0.5MByte
Track to Track Seek(read/write)	0.98/1.24 msec
Maximum Seek(read/write)	18.2/19.2 msec
Average Rotational Latency	2.99 msec
Internal Transfer Rate	
Inner Zone to Outer Zone Burst	122 to 177 Mbits
Inner Zone to Outer Zone Sustained	11.3 to 16.8 MByte

Table 2. Features of the SCSI disk assumed in our experiments.

5.3. Experimentation Setup

We used six different VBR MPEG-2 streams of 30 minutes duration each. Each stream has 54,000 frames with a resolution of 720x480 and 24 bit color depth, 30 frames per second frequency, and a $IB^2PB^2PB^2PB^2PB^2$ 15 frame Group of Pictures structure. The encoding hardware that we use allows the generated bit rate to take values between 1Mbps and 9.6Mbps.[‡] Statistical characteristics of the clips are given in Table 1, where the coefficients of variation lie between 0.028 and 0.383, depending on the content type. In our *mixed* basic benchmark, the six different streams are submitted round-robin. Where necessary, experimental results from individual stream types are also shown.

For the measurements, we assumed Seagate Cheetah ST-34501 SCSI disks, with the features shown in Table 2.[§] Except for the storage capacity, which can reach $73GB$ in the latest models, the rest of the performance numbers are typical of today's high-end drives. The logical block size B_l was set to 16,384 bytes, while the physical sector size B_p was equal to 512 bytes. The stride size B_s in the disk space allocation was set to 1,572,864 bytes. The server memory is organized in buffers of fixed size $B_l = 16,384$ bytes each, with a total space of 64MB for every extra disk. The available network bandwidth was assumed to be infinite, leaving contention for the network outside the scope of the current work.

In our experiments, the round time was set equal to one second. We used a warmup period of 3,000 rounds and calculated the average number of active streams from round 3,000 to round 9,000. The measurements were repeated until the half-length of the 95% confidence interval was within 5% of the estimated mean value of the active streams.

6. STUDY OF FIXED-GRAIN STRIPING

We first describe the results of our study of the Fixed-Grain Striping. An important feature of this method is the ability to control the disk access efficiency through the choice of block size B_f. As the block size is increased, a larger part of each access is devoted to data transfer rather than mechanical movement overhead. When a stream requests more than one block from a particular disk during a round, a maximum of two contiguous accesses are sufficient with the stride-based disk space allocation we used.

As shown in Figure 5, the number of active streams with sixteen disks and the mixed workload increases linearly as the load, ρ, increases from 10% to 50%. At loads higher than 50%, the number of streams that can be supported no longer increases. Not surprisingly, the additional load beyond 50% translates into a corresponding increase in the number of rejected streams (Fig. 6). Since increasing the lookahead factor from 1 to 30 improves the number of streams that can be supported only marginally, for the rest of our experiments we set the lookahead factor F_l to 1. This corresponds to a lookahead distance of less than 10 rounds, for a system of sixteen disks operating at load $\rho = 80\%$, and half-hour clips of 1GByte each.

[‡]Although the Main Profile Main Level MPEG-2 specification allows bitrates up to 15Mbit/sec, there is a typical point of diminishing returns (no visual difference between original and compressed video) at 9Mbit/sec. The DVD specification sets a maximum allowed MPEG-2 bitrate of 9.8Mbit/sec.

[§]From the IBM Dictionary of Computing,[31] "megabyte (megabit) : (1) for processor storage, real and virtual storage, and channel volume, 2^{20} or 1,048,576 bytes (bits), (2) for disk storage capacity and communications volume, 1,000,000 bytes (bits)."

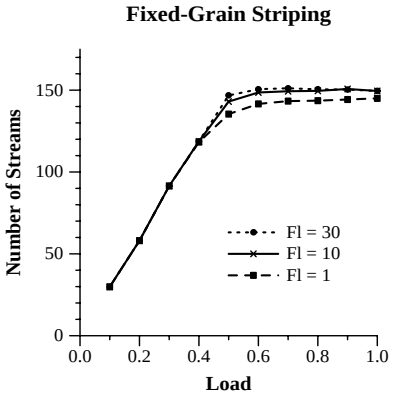


Figure 5. The sustained number of active streams with Fixed-Grain Striping flattens out at loads higher than 50% using a block size $B_f = 327,680$ with sixteen disks and the mixed workload. Changing the lookahead factor F_l from 1 to 30 increases the number of streams by less than 5%.

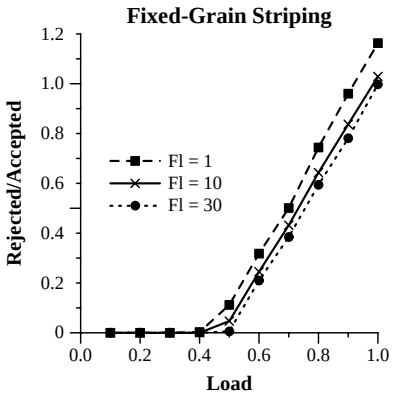


Figure 6. For Fixed-Grain Striping with $B_f = 327,680$, sixteen disks and the mixed workload, the total number of rejected streams over the total number of accepted during the measuring period. The ratio increases linearly as the load exceeds 50%, and changes by less than 20% as the lookahead factor F_l is increased from 1 to 30.

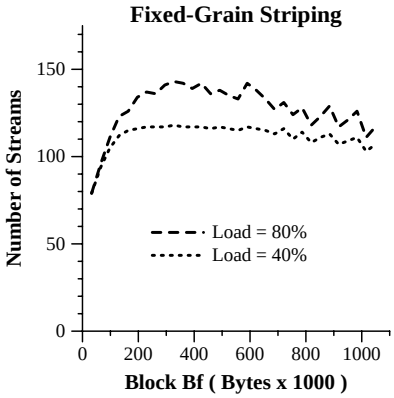


Figure 7. The number of active streams with Fixed-Grain Striping at different values of B_f. At both load values 40% and 80%, a maximum number of streams is achieved at $B_f = 327,680$. The experiments have been done over a range of B_f between $32,768$ and $1,048,576$ at steps of $32,768$. Sixteen disks have been used with the mixed workload.

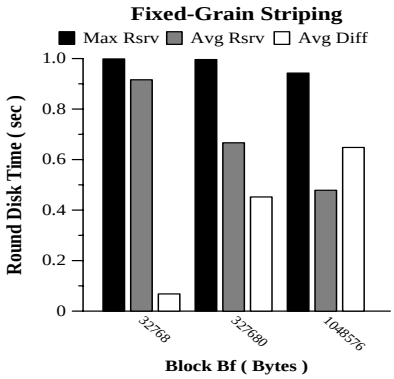


Figure 8. The maximum (Max Rsrv) and average (Avg Rsrv) disk access times reserved for a specific disk (Disk 0) in each round during the measuring period. Also, the maximum difference (Avg Diff) is shown between the reserved access times across all the disks in each round, averaged over the measuring period. Three different block sizes are tried with sixteen disks and the mixed workload at load $\rho = 80\%$.

For load values $\rho = 40\%$ and $\rho = 80\%$, we measured the number of active streams as the block size increases from $B_f = 32,768$ to $B_f = 1,048,576$ bytes at steps of 32,768. As can be seen from Fig. 7, at load 80% the number of streams initially increases until B_f becomes equal to $327,680$ and then drops. A similar behavior is noticed at 40%, although the variation in the number of streams is much smaller across different block sizes.

The Admission Control mode that was used for the above experiments allows also gathering of statistics on system resources reserved for each round during the admission control process. In particular, Fig. 8 depicts the maximum and average access time $T_{disk}(i,k)$ that was reserved during the measurement period $3,000 \leq i < 9,000$ for a particular disk ($k = 0$) in a sixteen disk configuration with load $\rho = 80\%$. While the maximum time remains close to 100% across different block sizes, the average time drops from about 90% at $B_f = 32,768$ to less than 50% at $B_f = 1,048,576$.

With the round time set to 1 sec, the average time (normalized by the round time) corresponds to the expected disk utilization and varies depending on the number of disks accessed for a stream every round. Part of it is actuator overhead and decreases as the block size becomes larger. On the other hand, the maximum difference in reserved

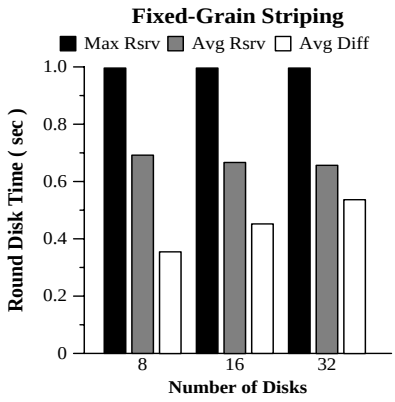


Figure 9. As the number of disks takes values 8, 16 and 32, the maximum and average reserved access time for Disk 0 (these values are respectively similar for the rest of the disks) remain almost the same. This is despite the corresponding increase in the maximum difference between reserved access times across the disks. The Fixed-Grain Striping policy is used on sixteen disks with mixed workload at load 80%.

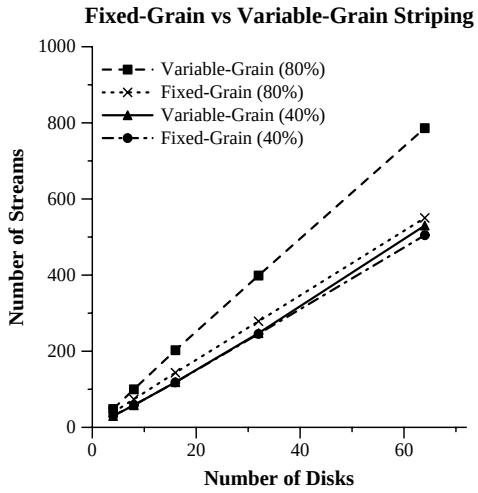


Figure 10. The sustained number of streams is measured using the Fixed-Grain Striping policy with a block size B_f that maximizes the number of streams. At load $\rho = 80\%$, the streams increase almost linearly from 39.17 to 550.23 as the number of disks increases from 4 to 64, while for $\rho = 40\%$ the corresponding increase is from 30.31 to 504.79. With Variable-Grain Striping instead, the number of streams increases from 48.11 to 786.05 at $\rho = 80\%$, and from 29.86 to 530.09 at $\rho = 40\%$.

access times in a round (Avg Diff in Fig. 8) increases on average from almost zero to above 60%, with increasing block size B_f. This could be another reason for the decrease in the average reserved time for larger block sizes.

Conclusions similar to the above affect the choice of the block size that maximizes the number of streams, and confirm results from previous studies.[4] However, we also found that the average reserved time (shown in Fig. 8 only for Disk 0) remains about the same (typically within 2%) across a disk array. Thus, the access load, on average, is equally distributed across the disks, despite variations from round to round. Furthermore, as the number of disks increases, the average time drops only slightly from 69% with 8 disks to 67% with 16 and 66% with 32 disks (Fig. 9). This implies that the capacity of the system increases almost linearly as more disks are added.

We repeated the above measurements varying the number of disks from 4 to 64 (Fig. 10). The block size B_f, that maximized the number of streams, was found to remain at $B_f = 327,680$. At 80% load, the number of streams that could be supported increased from 39.17 with 4 disks to 143.57 with 16 disks and 550.23 with 64 disks. This is within 9-14% of what perfectly linear scalability would achieve. With the load at 40%, the number of streams increased from 30.31 with 4 disks to 504.79 with 64 disk, thus reflecting the improved capacity of the system with increased number of disks at low loads.¶

7. COMPARISON WITH VARIABLE-GRAIN STRIPING

In the previous section, we studied the behavior of Fixed-Grain Striping. We found that with the mixed workload the number of streams is maximized at $B_f = 327,680$ across different number of disks and system load values. In the present section, we study Variable Grain Striping with respect to scalability, which is done for first time to the best of our knowledge.

In Fig. 11, we show the performance of Variable-Grain Striping on sixteen disks as the load increases from 10% to 100%. The number of streams grows linearly as the load increases up to 70%. This is significantly higher than the 50% load, where Fixed-Grain Striping flattened out (Fig. 5). Loads higher than 70% with Variable Grain Striping

¶At load $\rho = 40\%$, we see a slightly superlinear increase of about 4%, which is less than the relative statistical error $\gamma = 5\%$ that we allowed in our experiments. Therefore, one possible explanation for this behavior is the statistical error, which was higher at low loads and small number of disks. Another possible reason is the increased statistical benefit of multiplexing requests from a larger number of streams across a larger number of disks.

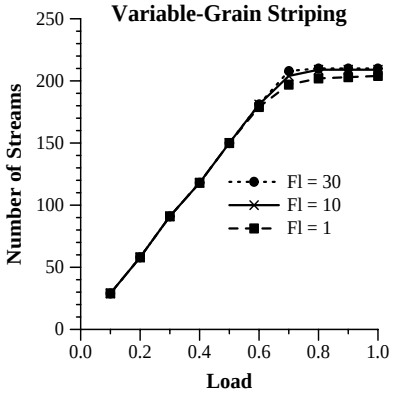


Figure 11. The sustained number of active streams with Variable-Grain Striping flattens out at loads higher than 70%. Changing the lookahead factor F_l from 1 to 30 increases the number of streams less than 5%. Sixteen disks are used with the mixed workload.

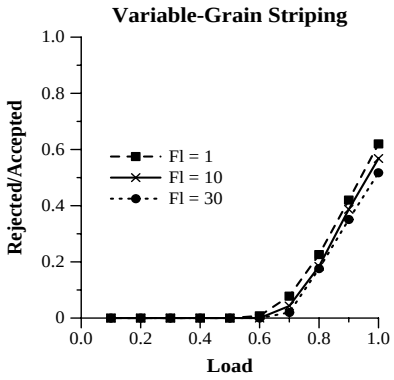


Figure 12. The total rejected streams over the total accepted during the measuring period. The ratio increases linearly as the load exceeds the 70% value, and changes by no more than 10%, as the lookahead factor F_l varies between 1 and 30. Sixteen disks are used with the mixed workload.

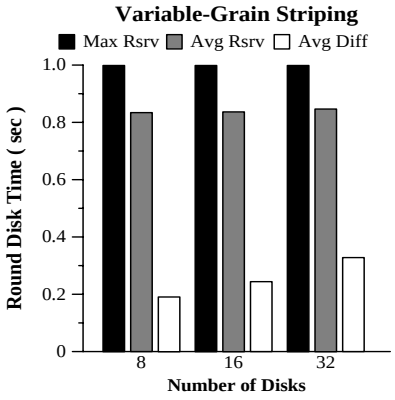


Figure 13. As the number of disks takes values 8, 16 and 32, the maximum and average reserved access time in Disk 0 (the values are respectively similar for the rest of the disks) remain almost the same. This is despite the corresponding increase in the maximum difference between reserved access times across the disks. The Variable-Grain Striping policy is used on sixteen disks with the mixed workload at load 80%.

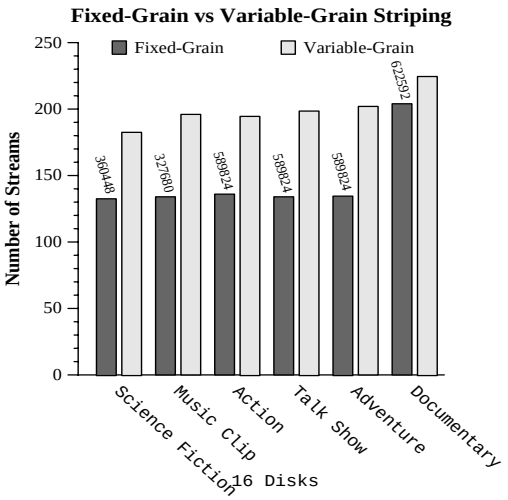


Figure 14. The advantage of Variable-Grain Striping over Fixed-Grain Striping varies among 38% in Science Fiction, 46% in Music Clip, 43% in Action, 49% in the Talk Show, 50% in the Adventure, and 11% in the Documentary. The block size shown for Fixed-Grain Striping was found to maximize the number of streams, over a range of block sizes between 32,768 and 1,048,576 with step of 32,768. The load was always set equal to 80%.

only increase the number of rejected streams as shown in Fig. 12. As before, a lookahead factor value of $F_l = 1$ attains more than 95% of the system throughput, and that is the value that we will use.

In Figure 13, we depict the reserved disk access time per round. As the number of disks increases, the average reserved time increases from 83% with 8 disks, to 84% with 16 disks, and 85% with 32 disks. We also measured the maximum number of sustained streams from 4 to 64 disks (Fig. 10). At a load of 80%, the number of streams increases from 48.11 with 4 disks, to 202.69 with 16 disks and 786.05 with 64 disks. Thus, as the number of disks increases, the number of streams remains within 3% of what perfectly linear scalability would achieve. In addition, the advantage of Variable-Grain Striping over Fixed-Grain Striping increases from 23% with 4 disks to 43% with 64 disks. To the best of our knowledge, this is the first scalability analysis of the Variable-Grain Striping policy.

In Fig. 14, we consider individual stream types in more detail. As the content type changes from Science Fiction to Documentary and the variation in data transfers correspondingly drops, the block size needs to be larger in order to maximize the performance of Fixed-Grain Striping. However, the performance remains about the same for the

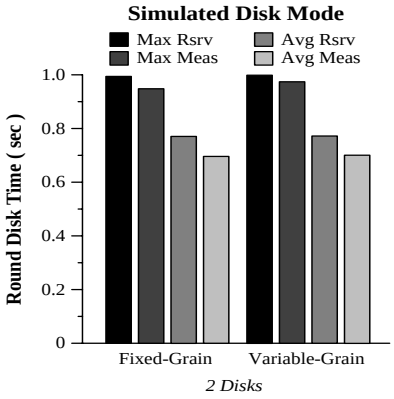


Figure 15. The reserved disk time statistics (Disk 0) are no more than 8% higher than the measured access time statistics using the detailed Seagate Cheetah ST-34501 model of Ganger et al.

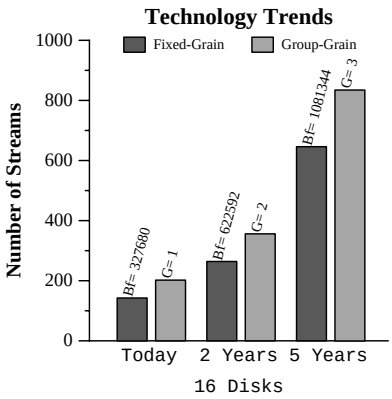


Figure 16. With reasonable technology projections, two years into the future Group-Grain Striping (generalized Variable-Grain Striping) maintains an advantage of 35% over Fixed-Grain Striping (from 41% today). The corresponding benefit in five years is no less than 29%. The shown values of B_f and G were found to maximize the throughput of the two policies respectively.

five stream types, and increases only with the Documentary stream. In contrast, Variable-Grain Striping manages to transform even minor decreases in data transfer variation into improved performance. Overall, Variable-Grain Striping maintains an advantage over Fixed-Grain Striping between 11% and 50%.

8. VALIDATION IN SIMULATED DISK MODE

In order to keep the computation time reasonable, the previous experiments were conducted with our system in Admission Control mode, where playback requests arrive leading to corresponding resource reservations, but without actual time measurement of the individual disk transfers. In the present section, we compare the statistics of the disk time resource reservations with the statistics gathered over the access times of all individual data transfers involved, using the DiskSim representation of the Seagate Cheetah ST-34501 disk.[29] A two-disk array model is used with each disk attached to a separate 20MB/sec SCSI bus, and no contention assumed on the host system bus connecting the two SCSI buses. The statistics are gathered during 6,000 rounds after a warmup period of 3,000 rounds, as before. The mixed workload is used with average number of active streams 21.23 and 23.27 for Fixed-Grain and Variable-Grain Striping, respectively, corresponding to 80% load.

As can be seen from Fig. 15, in both the average and maximum case, the reserved disk time is no more than 8% higher than the corresponding measurements using the detailed disk model of Ganger et al. The difference can be attributed to the fact that the reservation assumes a minimum disk transfer rate and ignores on-disk caching (section 3.3, Reservation of Server Resources). We believe that the achieved accuracy in the predicted disk access times during resource reservation is adequate. In fact, any extra disk geometry information that would be required to improve it is not readily available in commercial disk drives.[32]

9. EFFECT OF TECHNOLOGY TRENDS

To project disk technology improvements for the foreseeable future, we extend compound growth rates from the past linearly into the future (Table 3). In particular, we assume 30% increase in internal disk transfer rate per year, and 23% decrease in seek distance.[33] The full seek time depends linearly on seek distance, so the decrease is also 23%. However, we assumed decrease of 12% per year for the track seek time, which is dependent on the square root of the seek distance (among other factors more complex to project including settle time). Finally, we assume a rotation speed increase of 12% per year.[34] We presume that stream types and sizes will remain the same. This is a conservative projection, ignoring potential demand for higher resolution and more content rich streams.

In our experiments so far, we compared Fixed-Grain Striping to Variable-Grain Striping, which is a special case of Group-Grain Striping at $G = 1$. With current disk technology, having $G = 1$ maximizes the number of streams. But as the disk access time drops, we found it beneficial to increase G, so that G rounds worth of stream data

Disk Parameter	Today	2 Years	5 Years
Min Transfer Rate (MB/sec)	11.3	19.10	41.92
Max Seek Time (msec)	18.2	10.74	4.91
Track Seek Time (msec)	0.98	0.76	0.51
Avg Rotation Latency (msec)	2.99	2.38	1.70

Table 3. Projection of disk parameter changes in two and five years into the future.

are transferred in a single round. Specifically, when using the mixed workload, we found that two years into the future, the number of streams that could be supported with Group-Grain policy at $G = 2$ increases by 35% when compared to Fixed-Grain Striping. Five years into the future, the corresponding benefit of Group Grain Striping at $G = 3$ remains at 29%. Thus, under reasonable technological improvements, there are significant performance improvements when using Group-Grain Striping instead of Fixed-Grain Striping.

10. CONCLUSIONS AND FUTURE WORK

We presented the *Exedra* media server architecture for storage of variable-bit rate streams, and introduced the stride-based disk space allocation technique for improving efficiency and predictability in stream data accesses. We formally specified the Fixed-Grain and Variable-Grain Striping policies, and introduced the Group-Grain Striping policy. We described a method for evaluating the performance of media servers, and demonstrated an almost linear increase in the supported number of streams as the number of disks increased with Fixed-Grain Striping. The disk striping scalability is farther improved with Variable-Grain Striping, which outperforms Fixed-Grain Striping by 41% for mixed stream workload and up to 50% for individual stream types on sixteen disks. With a reasonable technological projection based on previous trends, we also show a significant advantage of Group-Grain Striping (generalized Variable-Grain Striping) over Fixed-Grain Striping two and five years into the future.

There are critical issues of distributed schedule management, replication based fault tolerance and VCR functionality, among others, that remain to be considered through the prism of variable bit rate streams. In addition, the detailed representation of stream resource requirements that was assumed, introduces the need for an extensive study related to the cost of admission control.

REFERENCES

1. G. Gilder, "Fiber Keeps Its Promise: Get ready. Bandwidth will triple each year for the next 25," *Forbes* , Apr. 1997. www.forbes.com/asap/97/0407/090.htm.
2. J. Gray, "Informal Talk, Database Group Seminar, University of Toronto." March 23, 2000.
3. S. Gringeri, K. Shuaib, R. Egorov, A. Lewis, B. Khasnabish, and B. Basch, "Traffic Shaping, Bandwidth Allocation, and Quality Assessment for MPEG Video Distribution over Broadband Networks," *IEEE Network* , 94–107, Nov/Dec 1998.
4. P. J. Shenoy and H. M. Vin, "Efficient Striping Techniques for Multimedia File Servers," in *Intl. Work. Network and Operating Systems Support for Audio and Video*, pp. 25–36, (St. Louis, MO), May 1997. An expanded version appears in Performance Evaluation Journal, vol. 38, June 1999, pg. 175-199.
5. W. J. Bolosky, R. P. Fitzgerald, and J. R. Douceur, "Distributed Schedule Management in the Tiger Video Fileserver," in *ACM Symp. Operating Systems Principles*, pp. 212–223, (Saint-Malo, France), Oct. 1997.
6. F. A. Tobagi, J. Pang, R. Baird, and M. Gang, "Streaming RAID - A Disk Array Management System for Video Files," in *ACM Multimedia Conf.*, pp. 393–400, (Anaheim,CA), Aug. 1993.
7. S. Berson, S. Ghandeharizadeh, R. Muntz, and X. Ju, "Staggered Striping in Multimedia Information Systems," in *ACM SIGMOD*, pp. 79–90, (Minneapolis, MN), May 1994.
8. M. M. Buddhikot and G. M. Parulkar, "Efficient Data Layout, Scheduling and Playout Control in MARS," in *Intl. Work. on Network and Operating System Support for Digital Audio and Video*, pp. 318–329, (Durham, NH), Apr. 1995.
9. B. Ozden, R. Rastogi, and A. Silberschatz, "Disk Striping in Video Server Environments," in *IEEE Multimedia Computing and Systems*, pp. 580–589, (Hiroshima, Japan), June 1996.

10. W. J. Bolosky, J. S. Barrera, R. P. Draves, R. P. Fitzgerald, G. A. Gibson, M. B. Jones, S. P. Levi, N. P. Myhrvold, and R. F. Rashid, "The Tiger Video Fileserver," in *Intl. Work. on Network and Operating System Support for Digital Audio and Video*, pp. 97–104, (Zushi, Japan), Apr. 1996.
11. C. Martin, P. S. Narayanan, B. Ozden, R. Rastogi, and A. Silberschatz, "The Fellini Multimedia Storage System," in *Multimedia Information Storage and Management*, S.M.Chung, ed., Kluwer Academic Publishers, (Boston, MA), 1996.
12. P. J. Shenoy, P. Goyal, S. S. Rao, and H. M. Vin, "Symphony: An Integrated Multimedia File System," in *IS&T/SPIE Multimedia Computing and Networking*, pp. 124–138, (San Jose, CA), Jan. 1998.
13. J. R. Santos, R. R. Muntz, and B. Ribeiro-Neto, "Comparing Random Data Allocation and Data Striping in Multimedia Servers," in *ACM SIGMETRICS*, pp. 44–55, (Santa Clara, CA), June 2000.
14. D. Makaroff, N. Hutchinson, and G. Neufeld, "An Evaluation of VBR Disk Admission Algorithms for Continuous Media File Servers," in *ACM Multimedia Conf.*, pp. 143–154, (Seattle, WA), May 1997.
15. S. Paek, P. Bockeck, and S.-F. Chang, "Scalable MPEG2 Video servers with Heterogeneous QoS on Parallel Disk Arrays," in *Intl. Work. Network and Operating Systems Support for Audio and Video*, pp. 342–353, (Durham, NH), Apr. 1995.
16. S. Paek and S.-F. Chang, "Video Server Retrieval Scheduling for Variable Bit Rate Scalable Video," in *IEEE Multimedia Computing and Systems*, pp. 108–112, (Hiroshima, Japan), June 1996.
17. E. Chang and A. Zakhor, "Scalable Video Data Placement on Parallel Disk Arrays," in *IS&T/SPIE Image and Video Databases*, pp. 208–221, (San Jose, CA), Feb. 1994.
18. E. Chang and A. Zakhor, "Disk-based Storage for Scalable Video," *IEEE Transactions on Circuits and Systems for Video Technology* , 758–770, Oct. 1997.
19. E. Chang and A. Zakhor, "Cost Analyses for VBR Video Servers," *IEEE Multimedia* , 56–71, Winter 1996.
20. A. L. N. Reddy and R. Wijayaratne, "Techniques for improving the throughput of VBR streams," in *IS&T/SPIE Multimedia Computing and Networking*, pp. 216–227, (San Jose, CA), Jan. 1999.
21. P. Triantafillou and S. Harizopoulos, "Prefetching into Smart-Disk Caches for High Performance Media Server," in *IEEE Multimedia Computing and Systems*, pp. 800–805, (Florence, Italy), June 1999.
22. E. Biersack, F. Thiesse, and C. Bernhardt, "Constant Data Length Retrieval for Video Servers with Variable Bit Rate Streams," in *IEEE Multimedia Computing and Systems*, pp. 151–155, (Hiroshima, Japan), June 1996.
23. P. J. Shenoy, P. Goyal, and H. M. Vin, "Issues in Multimedia Server Design," *ACM Computing Surveys* **27**, 636–639, Dec. 1995.
24. T. Clark, *Designing Storage Area Networks*, Addison-Wesley, Reading, Mass., 1999.
25. G. A. Gibson, D. F. Nagle, K. Amiri, J. Butler, F. W. Chang, H. Gobioff, C. Hardin, E. Riedel, D. Rochberg, and J. Zelenka, "A Cost-Effective, High-Bandwidth Storage Architecture," in *Conf. Architectural Support for Programming Languages and Operating Systems*, pp. 92–103, (San Jose, CA), Oct. 1998.
26. M. N. Garofalakis, Y. E. Ioannidis, and B. Ozden, "Resource Scheduling for Composite Multimedia Objects," in *Very Large Data Bases Conf.*, pp. 74–85, (New York, NY), Aug. 1998.
27. L. McVoy and S. R. Kleiman, "Extent-like Performance from a Unix File System," in *USENIX Winter Technical Conference*, pp. 33–43, (Dallas, TX), 1991.
28. E. Borosky, R. Golding, A. Merchant, L. Schreier, E. Shriver, M. Spasojevic, and J. Wilkes, "Using attribute-managed storage to achieve QOS," in *Intl. Work. Quality of Service*, pp. 203–206, (New York, NY), May 1997. Published in Building QoS into Distributed Systems, Chapman & Hall, 1997.
29. G. R. Ganger, B. L. Worthington, and Y. N. Patt, "The DiskSim Simulation Environment: Version 2.0 Reference Manual," Tech. Rep. CSE-TR-358-98, Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, Michigan, Dec. 1999.
30. MPEG Software Simulation Group, *MPEG-2 Encoder/Decoder, Version 1.2*, 1996.
31. *The IBM Dictionary of Computing.* McGraw-Hill, New York, NY, 1994.
32. B. L. Worthington, G. R. Ganger, Y. N. Patt, and J. Wilkes, "On-Line Extraction of SCSI Disk Drive Parameters," in *ACM SIGMETRICS*, pp. 146–156, (Ottawa, Canada), May 1995.
33. S. W. Ng, "Advances in Disk Technology: Performance Issues," *Computer* **31**, 75–81, May 1998.
34. C. Ruemmler and J. Wilkes, "An Introduction to Disk Drive Modeling," *Computer* **27**, 17–28, Mar. 1994.