Cost-Based Media Server Design

Edward Chang and Hector Garcia-Molina
Department of Computer Science
Stanford University
{echang, hector}@cs.stanford.edu

Abstract

Conventional wisdom holds that reducing disk latency leads to higher disk utilization, maximizing disk utilization leads to higher throughput, and employing a faster disk leads to better performance. All of this is true when building a conventional file or database system. In this paper we show that these principles can be misleading when applied to a media server. To design such a server, we propose a cost-based approach that focuses on the per-stream costs. We give various examples to illustrate the design process.

Keywords: multimedia, disk latency, memory utilization, per-stream cost.

1 Introduction

Maximizing throughput is a common design objective for a media server. To improve throughput, two approaches have been used: reducing disk latency (i.e., seek overhead and rotational delay) and minimizing the required memory. To reduce disk latency we can either employ efficient disk scheduling [13, 14, 15] or enact intelligent data placement policies [9, 11]. Both methods effectively cut down the worst-case seek distance for IOs. However, the improvement in throughput that a disk reduction scheme can achieve is limited, since memory, rather than disk, soon becomes the resource bottleneck [6, 7, 8].

Effective memory use can also improve throughput. The memory required by a server can be reduced through fixed-order IO scheduling and through buffer sharing schemes [3, 10]. However, fixed-order IO scheduling worsens the disk latency [6, 15] and buffer sharing can be ineffective if disk latency is reduced [5, 6]. In general, a good media server design thus must take both disk and memory use into consideration.

In this paper we consider such a combined disk/memory design, taking into account a third critical parameter: hardware costs. We start by deriving formulas that show that, contrary to conventional wisdom, reducing disk latencies does *not* lead to improved performance of a media server. We also show that maximizing disk utilization is undesirable due to the rapid growth of the memory required to support the higher disk utilization.

Given these insights, we present a model for computing per-stream, combined memory and disk costs, and use it to find the number of streams a server should support in order to minimize costs. By examining a set of real disks, we discuss how to select a good server configuration. We also observe that selection of disks is delicate: often, a faster or larger disk does not give us a better configuration.

Section 2 describes our design parameters and analytical model, while Section 3 shows the relationship between disk utilization and disk latency, and between disk utilization and required memory. In Section 4 we propose our cost model and derive formulas that lead to the minimum per-stream cost. Finally, Section 5 considers the the selection of disks for a server, using specifications and prices or real disks.

1.1 Related Work

The work we present here is an extension of our earlier work on disk scheduling and memory management for multimedia servers [4, 5, 6]. Our proof that reducing latency does not improve disk utilization, and our formula for the optimum number of streams are new. As far as we know, this is the first time the selection of disks is addressed, in the context of reducing overall costs. In the full version of this paper we will briefly survey papers on disk scheduling algorithms and data placement policies for multimedia servers (some of these references were given at the beginning of this section). However, most of these papers, except for [12], do not consider hardware costs as we do here. Reference [12] uses costs to compare data placement policies, but it does not address the cost minimization issue.

2 Analytical Model

To analyze the performance of a media server, we typically are given the following parameters:

- TR: the disk's data transfer rate.
- DR: the display rate of the media. Each stream requires a display rate of DR ($DR < TR$). For simplicity we assume that the display rates are equal.
- $\gamma(d)$: a concave function that computes the disk latency given a seek distance d. For convenience, we will refer to the combined seek and rotational overhead as the disk latency.

The media server has the following tunable parameters, which can be adjusted, within certain bounds, to optimize system throughput:

- T: the period for servicing a round of requests. As discussed below, T must be made large enough to accommodate the maximum number streams we expect to handle.
- S: the segment size, i.e., the number of bytes read for a stream with a contiguous disk IO.
- N: the maximum number of concurrent requests the media server allows.

We assume that the media server services the requests in rounds. During a round of service (time T), the media server reads one *segment* of data (of size S) for each of the N requested *streams*. Note that the amount of memory needed for supporting a stream may be larger or smaller than the segment size S, depending on the disk scheduling and memory management policy used [6]. To simplify our discussion, we assume for now that each stream requires a full segment worth of main memory. That is, the total memory required by the server is $N \times S$. This situation occurs, for example, if streams do not share memory and the IOs can be scheduled in a fixed order from one service round T to another. (Memory sharing can reduce the required memory, but variability in the service order can increase it.) In Section 5 we consider a different memory use scenario, and we argue that even with memory sharing, the general conclusions of our study hold.

The media server can employ many possible values for T, S, and N. However, some values will make it impossible to deliver data for each stream at the appropriate rate due to the violation of certain constraints. Other values will lead to suboptimal performance. For optimal feasible performance, the parameters T, S, and N need to satisfy the equations we derive next.

In a feasible system, the amount of data retrieved in a period, S, must be at least as large as the amount of data displayed. That is, $S \geq DR \times T$. However, if we want a stable system, the input rate should equal the output rate, else in every period we would accumulate more and more data in memory. Thus, we have the equation

$$S = DR \times T. \quad (1)$$

In a feasible system, the period T must be large enough so that all necessary IOs can be performed. Since T is fixed and cannot vary depending on the number of concurrent requests in the system at a particular moment, we must make T large enough to accommodate N seeks and to transfer N segments. Furthermore, in computing these seek times, we have to assume a worst-case situation, so that no matter where the segments are located on disk, we will have enough time to read them.

The total seek overhead for N requests is $\sum_{i=1}^{N} \gamma(cyl_i)$, where γ gives the seek delay for the i^{th} request. The total transfer time for N requests, each of size S, at a transfer rate TR, is

$$T_{Transfer} = N \times S/TR. \quad (2)$$

As we stated above, the period T must be larger than or equal to the worst-case seek and transfer times, i.e., $T \geq T_{Seek} + T_{Transfer}$. For optimal performance, however, we take the smallest feasible T value, since otherwise we would be wasting both disk bandwidth and memory resources. That is,

$$T = T_{Seek} + T_{Transfer} = N \times (\gamma(cyl_i) + S/TR). \quad (3)$$

Substituting $T = S/DR$ (Eq. 1) into Equation 3, we can solve for S, the segment size needed to support the N requests:

$$S = \frac{N \times \gamma(cyl_i) \times TR \times DR}{TR - DR \times N}. \quad (4)$$

This equation tells us how much data (per stream) needs to be read in each cycle, and is hence useful for computing the disk utilization.

3 Disk Utilization

This section shows the relationship between disk latency and disk utilization. We also show why pushing disk utilization to the maximum may not be desirable.

3.1 Disk Latency vs. Disk Utilization

One definition of disk utilization is the percentage of time the disk is busy (not idle). However, this definition does not reflect how efficiently a disk is used. For example, a disk can spend all its time seeking without transferring data to achieve 100% utilization. To measure the effectiveness of disk scheduling or data placement, we define instead disk utilization as the fraction of time the disk performs useful work, i.e., transfers data. Let D_{util} denote disk utilization. Given that the media server services N requests, disk utilization can be written as

$$D_{util} = \frac{N \times S/TR}{T}.$$

Replacing T with S/DR (Eq. 1) yields

$$D_{util} = \frac{N \times DR}{TR}. \quad (5)$$

This tells us that, given N and the DR/TR ratio, disk utilization is constant, regardless of the disk latency $\gamma(cyl_i)$! Thus, reducing disk latency does not lead to better disk utilization. That is, since a disk scheduling policy cannot change DR nor TR of a given disk, a smart disk scheduling policy that reduces disk latency does not help improve disk utilization. In other words, smart scheduling policies do not increase the amount of useful work done by the disk.

The reason why reduced latency does not imply better utilization is that media servers must deliver data to meet deadlines. To prepare data in time for the clients, the server must read the proper amount of data to compensate for disk latency. If disk latency is high, the server must read more data for a request to sustain a longer data consumption period before the request's next read can be scheduled. Conversely, if the latency is low, the server reads less data. Reading more data than necessary wastes memory, and reading less data than required violates the real-time constraint. Thus, the media server always adjust the amount of data it must transfer according to the latency of the disk, and disk utilization remains constant.

3.2 Disk Utilization vs. Memory Requirement

Equation 5 shows that the disk utilization, D_{util}, is directly proportional to N. In a traditional file system, increasing disk utilization is good, as long as response time does not suffer too much. Since in our multimedia server after the initial delay to start the playback the continuous data supply is "guaranteed," one might be tempted to make utilization very high, as this leads to a higher throughput N. Unfortunately, memory use is adversely affected by high throughput.

To illustrate the impact of memory, Figure 1 graphs the segment size S as a function of N (Equation 4). This graph is for the parameter values that will be shown in in Table 1 in Section 5. The figure shows that S grows slowly when N is small, but as $N \times DR$ approaches TR, the denominator approaches 0 and the value of S grows rapidly.

The total memory needed to sustain the N streams, with no memory sharing among streams, is $N \times S$, so the total memory needed is even larger than what the figure shows. Memory sharing can reduce the total memory required by a constant factor (typically it cuts it by a factor of 2), so even with memory sharing among streams the requirements grow dramatically.

Since memory use grows rapidly, one must take

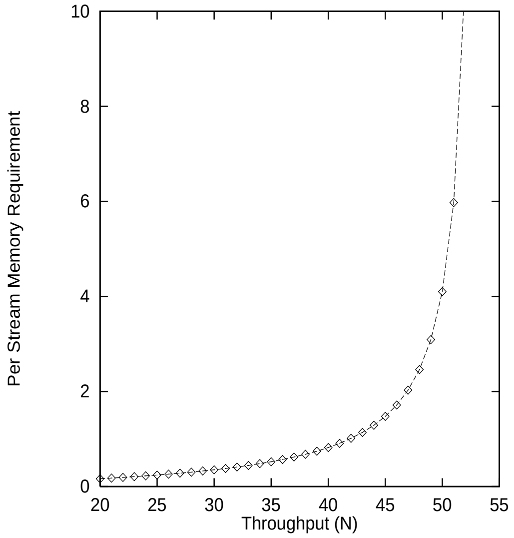


Figure 1: S vs. N

into account the *cost* of memory in selecting a good throughput value. To illustrate, suppose that the price of a disk is \$500 and the memory cost is \$5 per MByte (using other numbers for the cost gives similar results). Figure 1 shows that the amount of memory required for each stream is 1.14 MByte when $N = 43$ and 4 MBytes when $N = 50$. If we buy one disk to support 50 users, we need one disk and 200 MBytes of memory, which cost a total of \$1,500. However, if we use the disk to support only $N = 43$ users, the total cost is only \$750, half of the cost of increasing disk bandwidth to support 50 users. Given \$1,500, we are better off buying two disks and the needed memory to support a total of $2 \times 43 = 86$ users. This is a significant improvement over using one disk with higher disk utilization. In the next section where we derive formulas that give us the value of N that minimizes the per-stream cost.

4 Per-Stream Cost

The per-stream hardware cost includes the cost of memory, disk, CPUs, and peripherals. For simplicity, we assume that the CPU and peripheral costs are fixed, independent of the number of streams, so they can be treated as a constant term. In the remainder of this section we focus only on memory and disk cost.

Let C_d denote the cost of a disk, and C_m the cost of one MByte of memory. Suppose each stream requires S amount of memory. The per-stream cost function, denoted as $C_s(N)$, can be written as

$$C_s(N) = C_d/N + C_m \times S. \tag{6}$$

Substituting for S the right hand side of Equation 4 we get

$$C_s(N) = C_d/N + C_m \times \frac{N \times \gamma(cyl_i) \times TR \times DR}{TR - (DR \times N)}. \tag{7}$$

The disk cost is amortized by N, i.e., the larger N is, the lower the per-stream disk cost. On the other hand, the per-stream memory cost grows rapidly with N as shown in Figure 1.

4.1 Minimum Per Stream Cost

To minimize $C_s(N)$, we take the first derivative of Equation 6 and set it to zero; we get

$$-C_d N^{-2} + C_m \times DR \times TR \times \gamma(cyl_i) \times (TR - DR \times N)^{-1}$$
$$+ C_m \times DR^2 \times TR \times \gamma(cyl_i) \times N \times (TR - DR \times N)^{-2} = 0.$$

Multiplying both sides by $N^2 \times (TR - DR \times N)^2$ we have

$$-C_d \times (TR - DR \times N)^2$$
$$+ C_m \times DR \times TR \times \gamma(cyl_i) \times N^2 \times (TR - DR \times N)$$
$$+ C_m \times DR^2 \times TR \times \gamma(cyl_i) \times N^3 = 0.$$

Rearranging terms we obtain

$$(C_m \times DR \times TR^2 \times \gamma(cyl_i) - C_d \times DR^2) \times N^2$$
$$+ 2 \times C_d \times DR \times TR \times N - C_d \times TR^2 = 0.$$

Since $C_s(N)$ is convex, by solving for N and retaining the positive root, we obtain the optimal N^* that minimizes $C_s(N^*)$ as

$$N^* = \frac{TR}{DR} \times \frac{TR \times \sqrt{C_d \times C_m \times DR \times \gamma(cyl_i)} - C_d \times DR}{C_m \times TR^2 \times \gamma(cyl_i) - C_d \times DR}. \tag{8}$$

Substituting N^* back into Equation 6, we get the minimum per-stream cost.

4.2 Storage Cost

Given a required throughput level N_R, we can now determine how many disks d and how much memory to buy. That is, we select d so that N_R/d is as close as possible to N^* on each disk. For instance, if N_R is 38 and N^* is 10, we buy 4 disks, and enough memory to support the resulting segment size S. This gives us the lowest per-stream costs and hence the lowest overall cost. However, we still need to consider the storage of the media, and the associated costs.

The storage space required depends on the number, the length, and the average bitrates of the presentations stored by the media server. For example, a movie-on-demand server may require much larger storage space than a news-on-demand one since a typical movie plays longer and may require higher display quality (and hence more bits to encode a frame) than a typical news clip. Therefore, the number of disks that leads to minimum per-stream cost may be insufficient (e.g., for a movie-on-demand server) or more than enough (e.g., for a news-on-demand server) for storing data. We discuss these two cases separately.

1. *Less disk space is needed than what the minimum per-stream cost suggests.* In this case, we have space left unused on the disks. One alternative is to buy fewer disks, i.e., just enough to satisfy the storage requirement. This, as we have shown in Section 3.2, increases disk utilization on each disk and actually increases the per-stream cost. It is hence undesirable. Therefore, one is better off maintaining the number of disks at the minimum per-stream cost level by "wasting" some disk space.

2. *More disk space is needed than what the minimum per-stream cost suggests.* If the optimal number of disks is not adequate for storing all presentations, some of the "colder" presentation can be stored on tertiary storage (i.e., optical disks and tapes). Most likely we will anyway have backup copies on tertiary storage (for failure recovery), so the storage space does not cost us more. Of course, cold presentations would suffer much grater initial latencies.

   An alternative is adding more disks. This makes the server operate at a higher per-stream cost. As we will see in the next section, under-utilized disks are not as expensive as over-utilized disks, so the cost of storing the presentations may not increase cost too much beyond what is required for simply playing them back.

Parameter Name	*Value*
Disk Capacity	*9.0 GBytes*
Min. Transfer Rate TR	*80 Mbps*
Max. Transfer Rate TR	*124 Mbps*
Max. Rotational Latency Time	*8.33 milliseconds*
Min. Seek Time	*0.6 milliseconds*
Max. Seek Time	*19.0 milliseconds*

Table 1: Seagate Barracuda 9 ST19171 Disk Parameters

5 Evaluation

In this section we consider a variety of real disks, to illustrate the disk and memory costs, and the process of selecting a good configuration. We start by considering a Seagate Barracuda 9 ST-19171N disk [2]; its parameters are listed in Table 1. We use $TR = 80$ Mbps as the disk transfer rate and use the worst disk latency, including the maximum seek distance and a full rotational delay ($19 + 8.33 = 27.33$ ms). We assume a display rate DR of 1.5 Mbps, which is sufficient to sustain typical video playback.

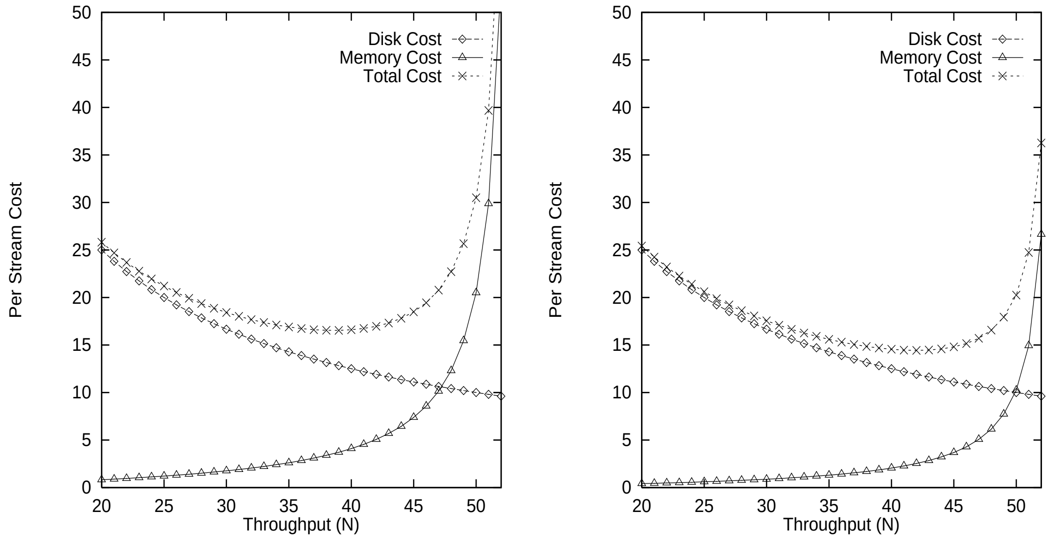
5.1 Example 1

Suppose the price of a disk is \$500 and the cost of memory is \$5 per MByte. Substituting these numbers into Equation 8, we obtain $N^* = 38$, much lower than what the the maximum disk bandwidth would support, which is $\frac{TR}{DR} = 53$.

Figure 2(a) plots the per-stream disk cost, memory cost, and total cost. The figure shows that indeed when N^* is around 38, the per-stream cost is the lowest: \$16.5. If we push the disk utilization to, say, past 48, we have to pay a higher premium ($>$ \$22.7). The server can indeed support 48 streams, but the cost per stream is 37% higher than necessary.

Figure 2(b) repeats the same experiment, under the assumption that memory sharing cuts the total memory use by half. Notice that the per-stream cost curves are quite similar, except that N^* shifts slightly to the right. Thus, memory sharing does not change the optimal throughput significantly. Pushing disk bandwidth beyond 49 or 50 continues to be undesirable.

Of course, the actual numbers we give here are just examples for our current scenario. If we use a different cost factor, then the values will be different. However, the shape of the curves and the overall conclusions will



(a) Memory Not Shared

(b) Memory Shared

Figure 2: Per Stream Cost vs. N

be similar.

5.2 Example 2

We extend the example (no memory sharing) to show how the server should be configured given a target throughput level to support. Suppose that the server is built to support up to 380 simultaneous users, and suppose that the server must provide storage for 108 GBytes of data. Since $N^* = 38$ minimizes the per-stream cost, using $\frac{380}{38} = 10$ disks minimizes the total cost if storage space is not considered. Since each stream costs \$16.5, supporting 380 streams costs a total of \$6,270.

To store 108 GBytes of data we need two more of these disks. Assume then that we distribute our 380 users onto 12 disks, as opposed to 10 disks. Then each disk must service up to 32 requests simultaneously. The per-stream cost based on $N = 32$ is \$17.7, and hence a total cost of \$6,726 is incurred: a \$456 cost increase. Note that this is less that the cost of the two extra disks ($2 \times 500 = \$1,000$), since in going to 12 disks we reduced the memory costs somewhat.

5.3 Choosing Disks

In this section we show how to select a disk model when a variety of choices are available. Table 2 lists the specifications and prices of some disks that we found on the Web [1]. Although there are many more models available, and prices for the same disk often vary a lot, we believe that the ones in our table form a representative sample. Table 2 divides the disks into three groups based on their storage capacity: 2 GBytes, 4 GBytes, and 9 GBytes. In each group, the table sorts the disks by vendors and then by their prices. We use a two character symbol to represent each disk model: the first character indicates the storage capacity (e.g., 2 means 2 GBytes) and the second character the sequence of the disk in the table. For instance, the 4 GByte Seagate ST-14209 disk in the 6^{th} row of the table is represented by symbol $4F$. We continue to assume that the memory cost is \$5 per MByte. Figure 3 plots the minimum per-stream cost versus N^* for all disks listed in the table, using the minimum data transfer rate and the worst disk latency.

Figure 3 can guide the selection of the best disk, depending on whether total storage capacity is an issue:

- *Minimizing cost given a throughput requirement only.* If the storage capacity is not a concern, the best disk to buy is simply the one that yields the minimum per-stream cost, i.e., disk $4H$. Disks $2A$, $4E$, and $4G$ are close runners-up. All these disks share one common characteristic: they have both lower latency/price and higher TR/price ratios than their peers (shown in Table 3).

- *Minimizing cost given both throughput and capacity requirements.* In this case, we first find the best candidates in each capacity group (i.e., 2, 4, and 9 GByte groups), then we find the globally

Symbol	*Disk Model*	*Avg/Max Seek*	*RPM*	*Worst Latency*	*Min/Max TR*	*Price*
2A	*Seagate ST-32107N*	*8.5/18*	*7200*	*26.33*	*47/88*	$349
2B	*Seagate ST-31051N*	*9/22*	*5400*	*33.11*	*44/66*	$349
4C	*IBM DCAS 34300*	*8.5/15*	*5400*	*26.11*	*40/NA*	$379
4D	*Micropolis 3242AV*	*8.9/16*	*7200*	*24.33*	*46/80*	$469
4E	*Micropolis 4345NS*	*7.9/15*	*7200*	*23.33*	*76/125*	$529
4F	*Seagate ST-14209N*	*9/18*	*7200*	*26.33*	*47/88*	$499
4G	*Seagate ST-34371N*	*9/17.5*	*7200*	*25.83*	*80/122*	$559
4H	*Seagate ST-23501N*	*7.7/18.3*	*10000*	*24.3*	*122/177*	$625
9I	*Seagate ST-410800*	*11/23*	*5400*	*24.11*	*44/65*	$659
9J	*Seagate ST-19171N*	*9.7/19.0*	*7200*	*27.32*	*80/124*	$959

Table 2: Disk Parameters and Prices

Disk	*Latency/Price* (10^{-6})	*TR/Price* (10^{-2})
2A	*75*	*13.5*
2B	*95*	*12.6*
4C	*69*	*10.5*
4D	*52*	*9.8*
4E	*44*	*14.3*
4F	*53*	*9.4*
4G	*46*	*14.3*
4H	*39*	*19.5*
9I	*52*	*6.6*
9J	*29*	*8.3*

Table 3: Latency/Price and TR/Price Ratios

optimal disk.

Given a group of disks with the same storage capacity, we want to select a disk that not only has small per-stream cost, but also has a small N^*. The reason why a small N^* is preferable is because, if the per-stream cost of two disks are the same, the disk that has a smaller N^* can be bought in a larger quantity, and subsequently provides more storage space. Conversely, if two disks have the same N^*, the one with smaller per-stream cost is the choice. Using these two criteria, we can first eliminate some bad choices from each group. That is, we can eliminate a given disk if another disk in its category appears in the area below it or to its left in Figure 3. This is because these other disks either have a lower per-stream cost (with the same N^*), or a smaller N^* (with the same per-stream cost), or both. In Figure 3 we can eliminate from the 4 MByte category disks 4D, 4F, and 4G.

To select from among the remaining candidates in the same capacity group, we need to know the total storage requirement. For example, comparing between disks 4C and 4H, we should select 4H because of its lower per-stream cost, as long as when bought in the optimal quantity, it yields sufficient storage space for our presentations. If it does not, disk 4C may be preferable. It may have a higher cost per stream, but since we get more of them at the optimal configuration (smaller N^*), we will have more total storage space.

Finally, we need to select from which group we want to buy disks. In addition to the per-stream cost and N^*, we have to also consider the capacity of the disks. In Figure 3 we can eliminate disks 2A and 2B since they are close to disk 4C in both the per-stream cost and N^* but provide only a half of storage capacity. Also, disk 9J may be eliminated since 4C has a much smaller N^* and smaller per-stream cost. The choice between 4C and 9I, however, is not as obvious and requires knowledge of the exact disk space requirement (as was the case with the 4C, 4H choice).

In summary, a faster (e.g., disk 4G) or a larger disk (e.g., disk 9J) may not be the best choice for a server. To make a decision, one must take into account both the cost of the improving the storage or speed of a disk, and any additional cost for the memory needed to support the higher performance disk. The overall storage capacity of the disks, relative to the size of the presentations, is also an important factor. In the full version of the paper we will provide additional examples and a more detailed analysis of the disk selection process.

6 Conclusion

In this paper we have shown that (1) given a disk model, reducing disk latency does not improve disk

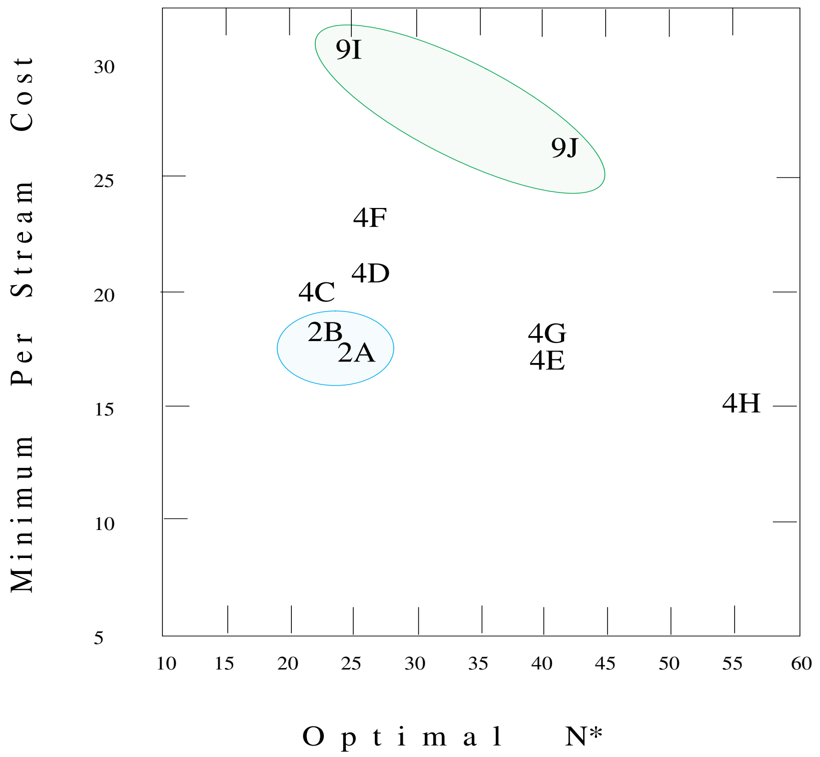


Figure 3: Per Stream Cost vs. N^* Samples

utilization for a media server, (2) maximizing disk utilization (the N or throughput for a disk) may be counter productive because of increasing memory costs, and (3) using a faster disk may not lead to higher throughput. We proposed a cost framework that takes both disk and memory into consideration, in order to minimize the per stream cost. We believe that our cost-based approach leads to a balanced and optimal media server design.

References

[1] Bason computer, inc. *URL: http://www.baconcomputer.com*, November 1997.

[2] Seagate barracuda 9lp family product specification. *URL: http://www.seagate.com*, 1997.

[3] E. Chang and Y.-Y. Chen. Minimizing memory requirements in a multimedia storage system. *Stanford Technical Report SIDL-WP-1996-0045*, 1996.

[4] E. Chang and H. Garcia-Molina. Effective memory use in a media server (extended version). *Stanford Technical Report SIDL-WP-1996-0050 URL: http://www-diglib.stanford.edu*, 1996.

[5] E. Chang and H. Garcia-Molina. Bubbleup - low latency fast-scan for media servers. *Proceedings of the 5th ACM Multimedia Conference*, pages 87–98, November 1997.

[6] E. Chang and H. Garcia-Molina. Effective memory use in a media server. *Proceedings of the 23rd VLDB Conference*, pages 496–505, August 1997.

[7] E. Chang and H. Garcia-Molina. Medic: Memory and disk cache for media servers. *Stanford Technical Report SIDL-WP-1997-0076 URL: http://www-diglib.stanford.edu*, November 1997.

[8] E. Chang and H. Garcia-Molina. Reducing initial latency in media servers. *IEEE Multimedia*, 4(3):50–61, July-September 1997.

[9] S. Ghandeharizadeh, S. Kim, and C. Shahabi. On configuring a single disk continuous media server. *Sigmetrics Performance Evaluation*, 23(1):37–46, May 1995.

[10] R. Ng and J. Yang. Maximizing buffer and disk utilizations for news on-demand. *Proceedings of the 20th VLDB Conference*, pages 451–462, 1994.

[11] Y.-J. Oyang and C.-H. Wen. A multimedia storage system for on-demand playback. *IEEE Transaction on Consumer Electronics*, 41(1):53–64, 1995.

[12] B. Ozden, R. Rastogi, and A. Silberschatz. Disk striping in video server environment. *Proc. IEEE Multimedia Computing and Systems*, pages 17–23, June 1996.

[13] A. Reddy and J. Wyllie. I/o issues in a multimedia system. *Computer*, 2:69–74, March 1994.

[14] F. Tobagi, J. Pang, R. Baird, and M. Gang. Streaming raid-a disk array management system for video files. *First ACM Conference on Multimedia*, August 1993.

[15] P. Yu, M.-S. Chen, and D. Kandlur. Grouped sweeping scheduling for DASD-based multimedia storage management. *Multimedia Systems*, 1(1):99–109, January 1993.