Maximizing Performance in a Striped Disk Array[1]

Peter M. Chen
David A. Patterson

Abstract

Improvements in disk speeds have not kept up with improvements in processor and memory speeds. One way to correct the resulting speed mismatch is by striping data across many disks. In this paper, we address how to stripe data to get maximum performance from the disks. Specifically, we examine how to choose the striping unit, i.e. the amount of logically contiguous data on each disk. We synthesize rules for determining the best striping unit for a given range of workloads. We show how the choice of striping unit depends on only two parameters: 1) the number of outstanding requests in the disk system at any given time, and 2) the *average positioning time*×*data transfer rate* of the disks. We derive an equation for the optimal striping unit as a function of these two parameters; we also show how to choose the striping unit without prior knowledge about the workload.

1. Introduction

In recent years, computer technology has advanced at an astonishing rate: processor speed, memory speed, and memory size have grown exponentially over the past few years [Bell84, Joy85, Moore75, Myers86]. However, disk speeds have improved at a far slower rate. As a result, many applications are now limited by the speed of their disks rather than the power of their CPUs [Agrawal84, Johnson84]. As improvements in processor and memory speeds continue to outstrip improvements in disk speeds, more and more applications will become I/O limited.

One way to increase the data rate (bytes transferred per second) and the I/O rate (I/O requests per second) from a file system is by distributing, or striping, the file system over multiple disks. In this paper, we examine how to choose the striping unit, i.e. the amount of logically contiguous data to store on each disk. If this choice is made incorrectly, 80% or more of the potential disk throughput can be lost. Our goal is to synthesize rules for determining the optimal striping unit under a variety of loads, request sizes, and disk hardware parameters. We show how the choice of striping unit depends on only two parameters: 1) the number of outstanding requests in the disk system at any given time, and 2) the *average positioning time*×*data transfer rate* of the disks. We derive an equation for the optimal striping unit as a function of these two parameters; we also show how to choose the striping unit without prior knowledge about the workload.

---

[1]This paper will appear in the 17th Annual International Symposium on Computer Architecture (SIGARCH 1990).

2. Definitions

We define the *striping unit* as the maximum amount of logically contiguous data that is stored on a single disk (see Figure 1). A large striping unit will tend to keep a file clustered together on a few disks (possibly one); a small striping unit tends to spread each file across many disks. Unlike [Patterson88, Chen89], we do not include any redundant data into our data striping scheme; data from each file is simply distributed round-robin over the disks.

We use parallelism to describe the number of disks that service a user request of data. A higher degree of parallelism increases the transfer rate that each request sees. However, as more disks cooperate in servicing each request, fewer independent requests can be serviced simultaneously. We define the degree of *concurrency* of a workload as the average number of outstanding user requests in the system at one time. A small striping unit causes higher parallelism but supports less concurrency in the workload; a large striping unit causes little parallelism but supports more concurrency in the workload.

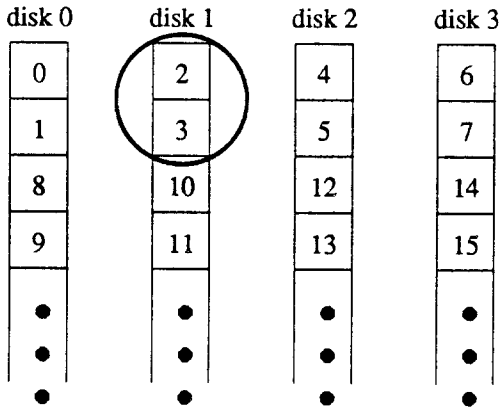


Figure 1: Definition of a Striping Unit. This figure shows the mapping of logical data to the disks for a striping unit of two sectors. The numbers in the figure are logical sectors; the circled two sectors constitute one stripe unit.

3. Previous Work

Disk striping is not a new concept—Cray Research has been striping files over multiple disks for many years to increase data rate [Johnson84]. However, with the proliferation of smaller diameter disk drives, striping over many disk drives could provide order of magnitude benefits in performance/cost, capacity/cost, power, and volume [Patterson88]. As a result, disk striping research has increased dramatically over the past few years.

Kim [Kim86] proposes a striping unit of one byte (byte-interleaving). Using queuing models, she finds that, under light loads, byte-interleaving yields higher throughput than a collection of non-cooperating disks (disks with an infinitely large striping unit). She also notes that byte-interleaved disks reach saturation under much lighter loads than non-cooperating disks.

Livny, et al. [Livny87] propose a scheme called declustering where the striping unit is 1 track (26 KB in their study) and compare its performance to a scheme with an infinitely large striping unit, called clustering. They conclude that declustering consistently yields higher throughput than clustering. They attribute this difference to two factors: 1) declustering allows increased parallelism, and 2) declustering load-balances the disks by spreading each file across multiple disks.

Patterson, et al. [Patterson88] investigate five ways to introduce redundancy into disk arrays to increase data availability. One of the redundancy schemes, RAID Level 3, has a striping unit of one byte. Two of the redundancy schemes, RAID Levels 4 and 5, have a striping unit of one block (block-interleaving), where a block remains unspecified.

Chen [Chen89] conducts hardware experiments on an Amdahl mainframe to further investigate two of the redundancy schemes in [Patterson88]. As part of his evaluation, he compares disk arrays with a striping unit of one sector (4 KB) and one track (40 KB). The workloads that we use in this experiment are essentially the same as in [Chen89].

Reddy, et al. [Reddy 89] evaluate a range of disk striping schemes ranging from byte-interleaving to block-interleaving, with a typical block size of 4 KB. In his evaluation, Reddy, et al. assume that byte-interleaved disks are rotationally synchronized with each other, but that block-interleaved disks are not synchronized. Reddy, et al. also propose several hybrid striping schemes where blocks are interleaved

across units which are themselves made of several byte-interleaved disks.

4. Experimental Introduction

4.1. Simulator

Using a disk simulator, we evaluate the performance of disk arrays with various striping units and disk parameters under several workloads. Because we are primarily interested in the performance of the disk subsystem, we that assume the CPU and the data path to memory are infinitely fast. Our system is thus completely disk limited.

At the start of a run with workload concurrency N, N user requests for data are issued. Depending on the striping unit, each user request is mapped into one or more disk requests. When one user request finishes, another user request is generated, maintaining a degree of concurrency of N. This process continues until 1000-N requests have been issued, after which the last N user requests are allowed to complete. Thus, a total of 1000 user requests are issued per run. This number of user requests was found to be sufficient to render the start-up and ending overhead of each run insignificant. Five independent runs are averaged together in order to produce a tight confidence interval. Each data value in this paper has a 90% confidence interval whose width is less than 5% of that data value (typically 1%-3%).

4.2. Disk Parameters

Several types of disks were modeled. The default disks, approximately the same as an Amdahl 6380A [Thisquen88], are characterized by parameters given in Table 1 and Figure 2. Note that the default disks are rotationally synchronized, where we define being rotationally synchronized as rotating in unison. I.e., disks that are synchronized rotate at the same rate and have sector 0 on each track pass underneath the read/write head at the same time. When multiple disks cooperate on a single user request, the user must wait until all disks have transferred their data. In a rotationally unsynchronized disk system, the rotational latency which a multi-disk request sees is approximately a full rotation; in a rotationally synchronized disk system, this rotational latency is one-half of a full rotation. In past research, disk arrays with a striping unit of byte are usually synchronized [Kim86, Reddy89], whereas disk arrays with a striping unit larger than one byte are not [Reddy89, Livny87]. This practice stems from the common assumption that 1) a group of

Default Disk Parameters	
bytes/sector	512
sectors/track	60
tracks/cylinder	15
cylinders/disk	885
average seek	14.7 ms
full rotation	16.7 ms
average rotational latency	8.35 ms
rotationally synchronized	yes
number of disks	16

Table 1: Specifications of Default Disk System.

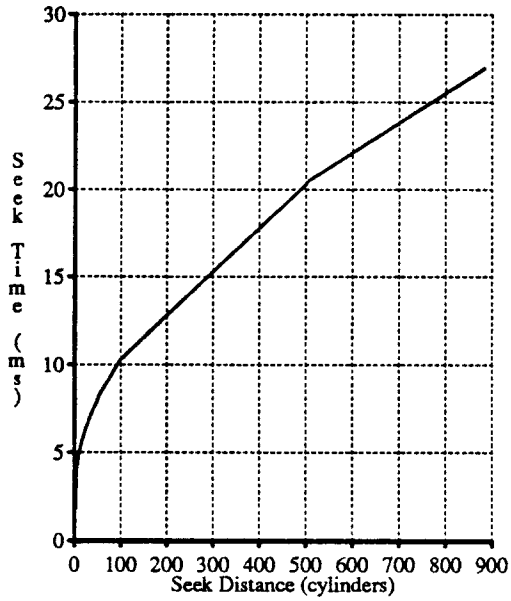


Figure 2: Seek Time Model. Graphed above is seek time in ms as a function of seek distance in cylinders [Thisquen88]. We model this as

$$seektime(x) = \begin{cases} 0 & \text{if } x=0 \\ 1.9-x/50+\sqrt{x} & \text{if } 0<x\leq 50 \\ 8.1+.044*(x-50) & \text{if } 50<x\leq 100 \\ 10.3+.025*(x-100) & \text{if } 100<x\leq 500 \\ 20.4+.017*(x-500) & \text{if } 500<x\leq 885 \end{cases}$$

byte-interleaved disks is viewed as a single unit and can only service one request at a time and 2) block-interleaved disks allow independent requests but do not cooperate together on one request. However, synchronizing block-interleaved disks does not inherently prevent them from operating independently and servicing different requests to different addresses. And, synchronized block-interleaved disks still benefit from synchronization when cooperating on a single request. Throughout this paper, we use rotationally synchronized disks.

For all disks modeled, we assume sector gaps are zero-length and head switch time is instantaneous. We do, however, model the cylinder switch time and skew the sector layout of consecutive cylinders to

maximize the performance of sequential requests.

To explore how the striping unit affects performance for a variety of disk parameters, we experiment with several modifications on the above default disks. Variations explored were: disks that seeked twice as quickly (all seek times were halved), disks that rotated twice as quickly, and disks that had twice as many sectors per track.

4.3. Workload

The workload supplied to the disks is characterized by three parameters: degree of concurrency, request size, and request starting location. The degree of concurrency is varied between 1 and 20. At a concurrency of 1, each newly issued request sees an idle system; by concurrency 20, the 16-disk system is saturated.

Four distributions of request sizes were used:

(1) *exp4k*: An exponential distribution with a mean of 4 KB.

(2) *exp16k*: An exponential distribution with a mean of 16 KB.

(3) *norm400k*: A normal distribution with a mean of 400 KB and a standard deviation of 400 KB.

(4) *norm1.5m*: A normal distribution with a mean of 1.5 MB and a standard deviation of 1.5 MB.

The starting location for each request consists of two components: starting disk and starting sector on that disk. The starting disk is chosen uniformly out of all the disks in the system; the starting sector on the disk is chosen uniformly out of all sectors on that disk. This location distribution does not favor any disk over any other; i.e. over time, independent of the striping unit and request size distribution, each disk will see approximately the same number of requests. Some past research (for example, [Chen89]), assumed the presence of *hot* disks, i.e. disks that received more accesses than the others. However, we believe that hot disks are becoming less of a problem for two reasons: first, the increasing file cache size of today's systems will buffer hot data from small files. Second, a striped disk system will spread large files (files much larger than the striping unit) across all disks in a round-robin fashion. This round-robin distribution will result in each disk containing file data which is separated by N striping units. So, unless a user accesses striping units 0, N, 2N, etc. of a file more frequently than 1, N+1, 2N+1, etc., there is no reason to expect any disk to see more accesses to that file than any other disk.

5. Metrics

Common disk system performance metrics are throughput and response time. With a fixed level of concurrency, higher throughput generally leads to faster response time. In this paper, we use throughput as the main performance metric. Most throughput values will be given as a percentage of the maximum throughput over all striping units. For example, if the maximum throughput over all striping units for a particular workload is 10 megabytes per second, and a striping unit S, yields a throughput of 3 megabytes per second, then the throughput for striping unit S will be given as 30% of maximum throughput.

6. General Performance Trends

Most importantly, disk striping impacts the amount of data that each disk transfers before re-positioning (seeking and rotating to the next request). This amount of data has a drastic influence on disk throughput. For our default disks, if a disk transfers one sector per request, throughput will be .02 MB/s; if it transfers one track per request, throughput will be .8 MB/s; if it transfers one cylinder per request, throughput will be 1.6 MB/s. In choosing the striping unit, we strive to maximize the amount of useful data each disk transfers per request and still make use of all disks. Large striping units maximize the amount of data a disk transfers per access but require higher concurrency in the workload to make use of all disks. Small striping units can make use of all disks even with low workload concurrency, but cause the disks to transfer less data per access.

Figure 3 shows the throughput versus the striping unit for a range of sizes and concurrencies. We vary the striping unit from .5 KB (1 sector) to 450 KB (1 cylinder). At any fixed striping unit, the throughput increases with larger request sizes and higher degrees of concurrency. Increasing request sizes result in each disk accessing more data per request; higher degrees of concurrency are able to make use of more disks. In order to compare trends from different workloads more easily, we scale the throughput of each workload, expressing it as a percentage of the maximum throughput (Figure 4).

In Figure 4, there are three categories of workloads: workloads whose maximum throughput is at the smallest striping unit (e.g. Figure 4b concurrency 1), workloads whose maximum throughput is at the largest striping unit (e.g. Figure 4b concurrency 20), and workloads whose maximum throughput is between the smallest and largest striping unit (e.g. Figure 4c concurrency 6).

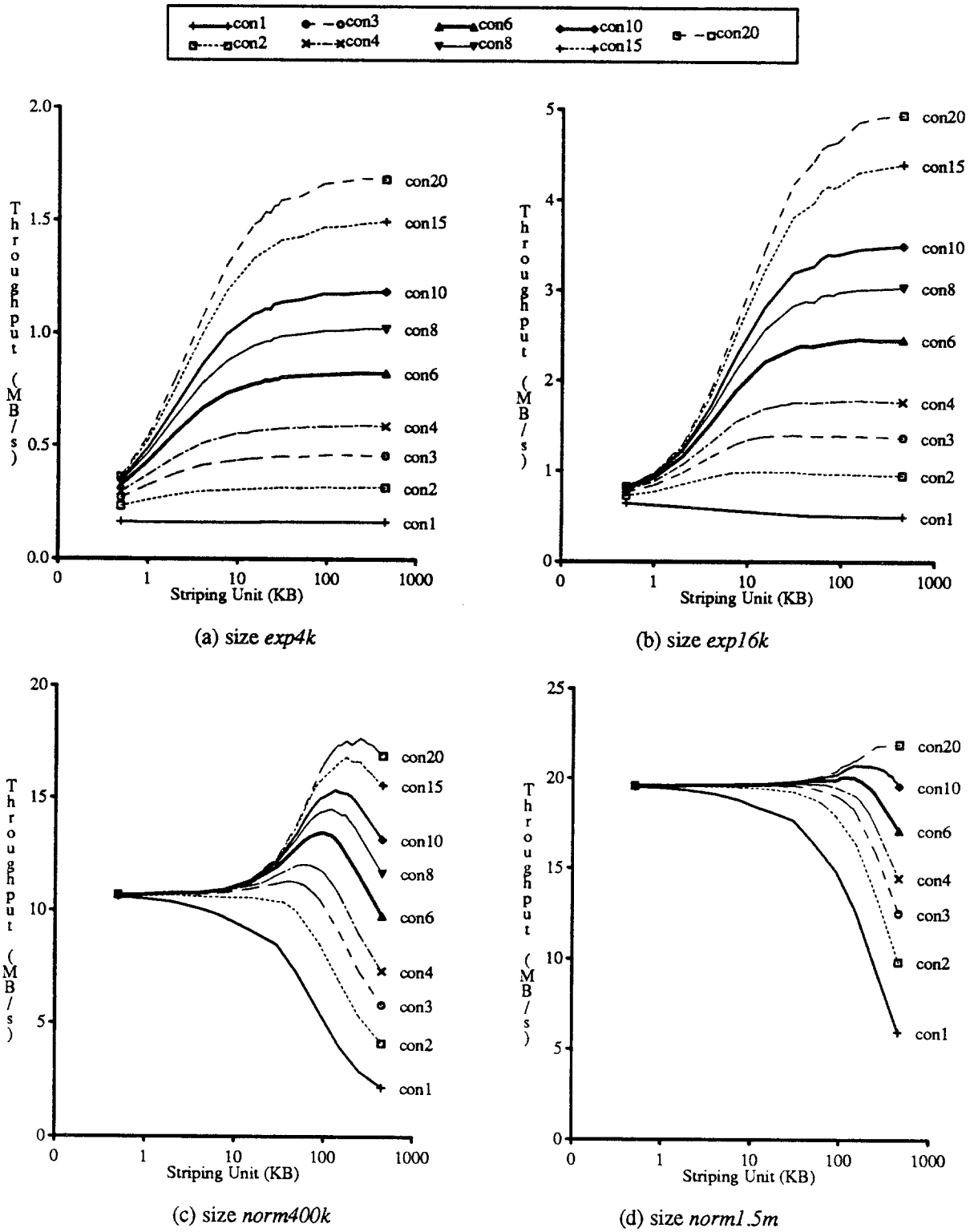


Figure 3: Throughput for a Range of Sizes and Concurrencies. Throughput is shown as a function of striping unit. The throughput increases with larger request sizes and higher degrees of concurrency, as expected.

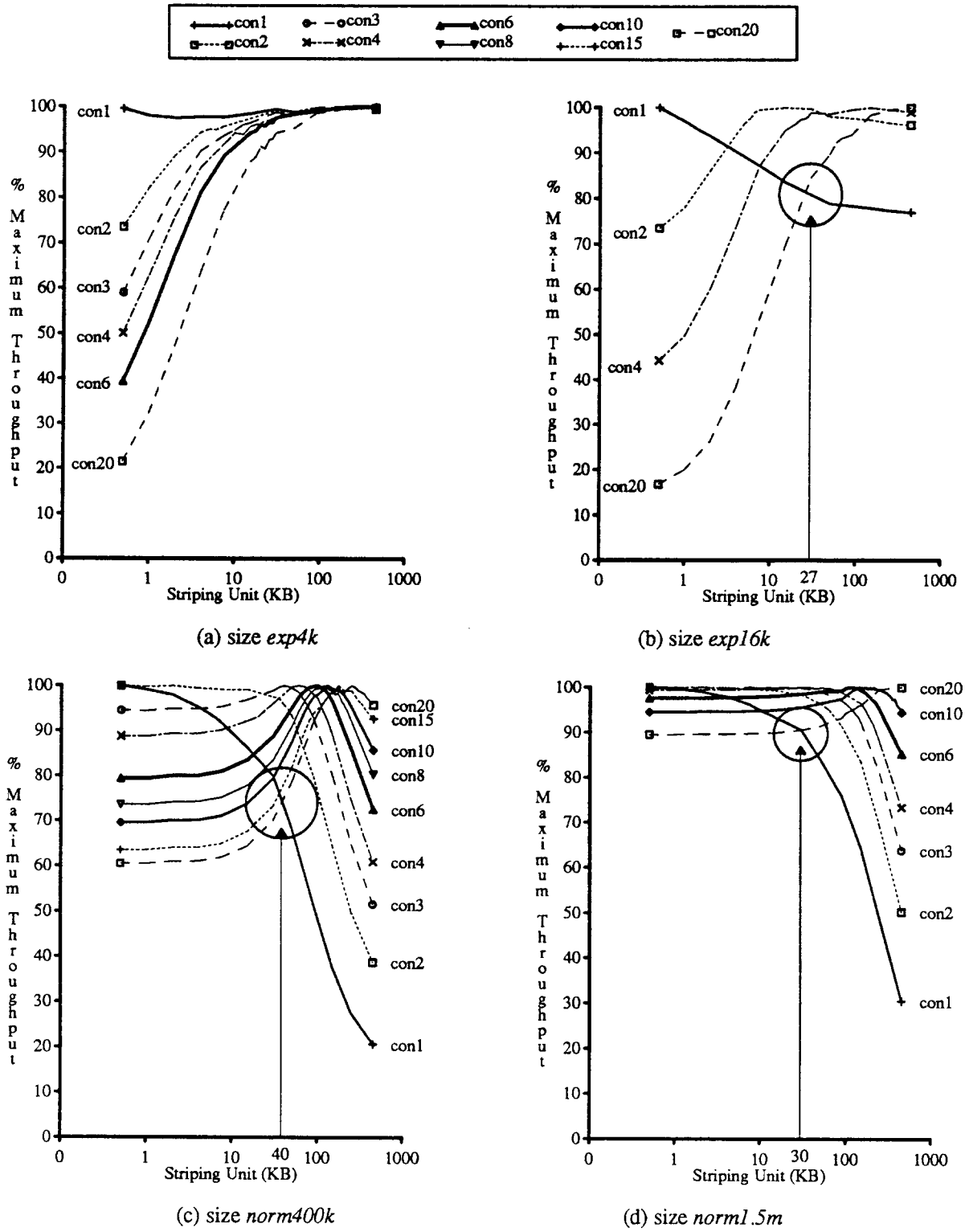


Figure 4: Percentage of Maximum Throughput for a Range of Sizes and Concurrencies. Percentage of maximum throughput is shown as a function of striping unit. The circled point on each graph indicates the striping unit which guarantees the highest percentage of maximum throughput to all workloads shown on that graph.

When the maximum throughput of a workload is at the smallest possible striping unit, the workload has low enough concurrency that it makes best use of the disks by having the maximum possible parallelism. In particular, at a concurrency of one, the striping unit which yields the highest throughput is 1 sector (.5 KB)—all other striping units at a concurrency of 1 will yield far less throughput. An exception is the *exp4k* request size distribution. Because striping over multiple disks decreases only the data transfer time, the response time of 4 KB requests can at most be decreased by 2 ms,[2] which is less than 10% of the total response time. In addition, even in an idle system, involving multiple disks in a request can sometimes lead to worse performance. For example, when disks take different amounts of positioning time, the request must wait for the slowest-positioning disk to transfer its data. When disks are rotationally synchronized, the rotational latency among multiple disks is usually equal. However, even with rotationally synchronized disks, if the involved disks do not start at the same cylinder, the positioning time will vary, causing slower response time. For large request sizes, the advantage gained in decreased data transfer time far offsets this small penalty in positioning synchronization. For *exp4k*, however, the amount of time saved in data transfer is approximately equal to the amount of time lost to positioning synchronization.

When the maximum throughput of a workload is at the largest possible striping unit (450 KB), the workload has enough concurrency that we should maximize the amount of data each disk transfers per request—the workload concurrency will inherently use all the disks. For example, for a concurrency of 20, each disk can service a different user request, and throughput is maximized by having each request access one disk.

When the maximum throughput of a workload is between the smallest and largest striping units, the workload has enough concurrency that each request should not occupy all the disks. However, concurrency in these workloads is low enough that having each request access only one disk would not use all the disks.

---

[2] 4 KB is 8 sectors (.133 of track) and takes .133 * 16.7 ms to transfer.

7. Choosing the Striping Unit

If one knows the parameters of a workload, i.e. the request size distribution and the concurrency, one can use Figure 4 to choose the striping unit which maximizes throughput. However, in most systems, the exact workload is not known. One or possibly both the request size and the concurrency will be unspecified. Thus, it is desirable to be able to choose a good striping unit with as little knowledge about the workload as possible. In this paper, we will strive to maximize the *minimum percentage throughput over a range of considered workloads.* In other words, we wish to guarantee the highest percentage of maximum throughput to all workloads in consideration. For example, if the request size is known to be *norm400k* but the concurrency is unknown, we can use Figure 4c to choose a striping unit. In Figure 4c, over the range of concurrencies between 1 and 20, the striping unit which maximizes the minimum percentage throughput is 40 KB. At that striping unit, all workloads considered yield at least 74% of their maximum possible throughput. Note that when the request size distribution is known, only the maximum and minimum concurrency workloads need to be graphed to calculate the desired striping unit.

Figure 5 graphs the percentage of maximum throughput for systems where the workload concurrency is known, but the request size distribution is unknown. Using the same "maximize the minimum percentage throughput" criterion as above, we can choose a desirable striping unit for a range of request sizes. Note that even if only the concurrency is known, *it is possible to choose a striping unit which yields over 95% of the maximum throughput for all request sizes.* On the other hand, if only the request size is known, then the best striping unit choice can guarantee only 70%-90% of the maximum throughput for all workload concurrencies (Figure 4). Thus, *concurrency is the important workload parameter in choosing the striping unit.*

To further examine how concurrency affects the choice of a striping unit when the average request size is unknown, we graph a range of possible striping unit choices at each concurrency (Figure 6). We display the range of striping units at each degree of concurrency which guarantees at least 95% of maximum throughput for all request sizes (*exp4k, exp16k, norm400k, norm1.5m*). We can express our choice of striping unit as a linear function of workload concurrency by 1) fixing the striping unit choice for a concurrency of one at 1 sector (.5 KB), and 2) measuring the minimum slope of any striping unit vs. con-

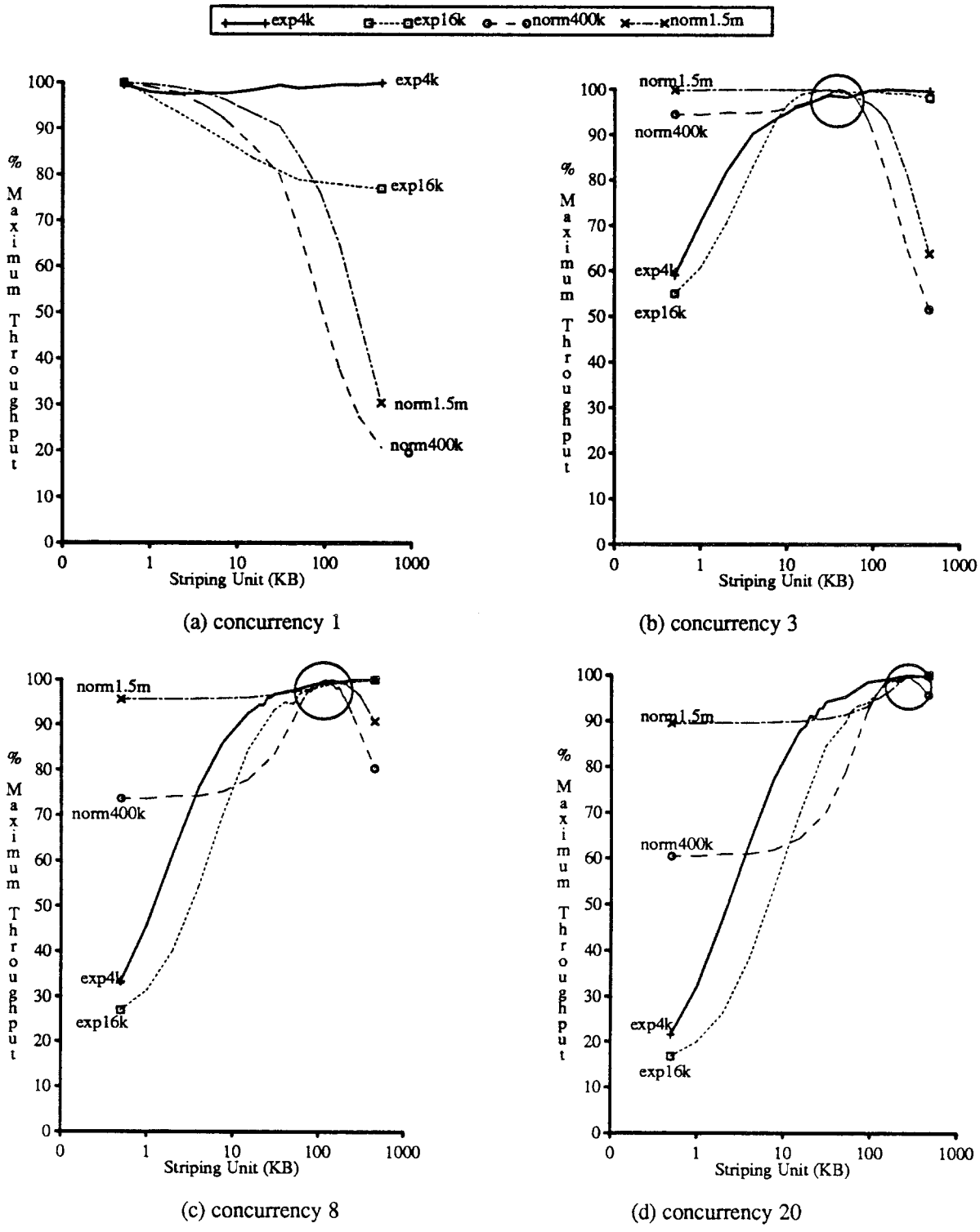


Figure 5:Percentage of Maximum Throughput for a Fixed Concurrency. When the concurrency is known, it is possible to choose a striping unit which yields over 95% of maximum throughput for all request sizes. The circled area on each graph is the range of striping units that guarantees 95% of maximum throughput for all workloads.

currency line that lies entirely in the displayed range. Our choice of striping unit in Figure 6 can then be expressed as

$$\textit{Striping Unit in KB} = 9.8\ \textit{KB} \times (\textit{Degree of Concurrency} - 1) + .5\ \textit{KB} \quad (1)$$

The slope of this line is 9.8 KB/Degree of Concurrency. This means that for every additional simultaneously outstanding request in the system, the striping unit should be increased by 9.8 KB. We shall see later how to express this slope in terms of disk parameters.

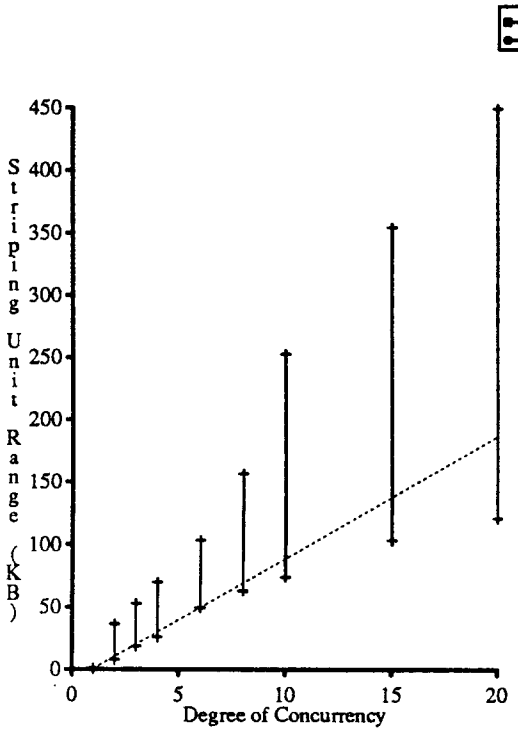


Figure 6: Striping Unit Chosen versus Concurrency—Default Disks. Shown here is the range of striping units which yield at least 95% of the maximum throughput for all request sizes (*exp4k, exp16k, norm400k, norm1.5m*). Also shown is the line with smallest slope which lies entirely in the striping unit range. The largest striping unit simulated was 450 KB.

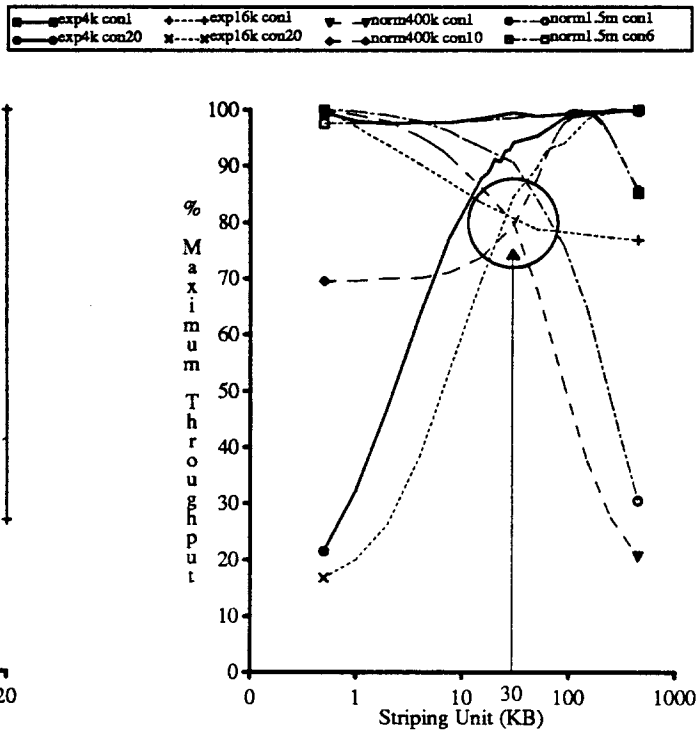


Figure 7: Percentage of Maximum Throughput for a Wide Range of Workloads. Shown here is the percentage of maximum throughput for a wide range of concurrencies and request sizes. The striping unit which guarantees the highest percentage of maximum throughput to each workload is 30 KB.

If little or no workload information is given, we can choose a good compromise **striping** unit by graphing the maximum and minimum concurrency for a range of request sizes (Figure 7). In Figure 7 we consider a lower range of concurrencies for workloads with higher average request sizes. This was done because systems with users who issue large requests (such as supercomputers) typically have fewer simultaneous users than systems with users who issue small requests (such as networks of workstations).

The striping unit which guarantees the highest percentage of maximum throughput to all workloads in Figure 7 is 30 KB. At this striping unit, all workloads considered yield at least 80% of maximum throughput. In determining *a priori* the best compromise striping unit, one which suits a wide range of request sizes and concurrencies, imagine decreasing the striping unit from infinity toward zero. At infinity, each request would be serviced by one disk. Decreasing the striping unit from this minimum-parallelism/maximum-concurrency point would cause some requests to be serviced by multiple disks. This would increase the data transfer rate of each request; however, the positioning done by these additional disks would cause a penalty in disk utilization. If changing disk technology were to cause this positioning penalty to decrease, we would expect the best compromise striping unit to also decrease, causing more parallelism. If changing disk technology were to cause the data transfer rate to increase, we would expect the best compromise striping unit to also increase. We hypothesize that, for all disk technologies, the best compromise striping unit should be such that the time to transfer a stripe unit will be comparable to and proportional to an average positioning time. Or, stated slightly differently,

$$\textit{Compromise Striping Unit} = Z \times \textit{average positioning time} \times \textit{data transfer rate} \qquad (2)$$

where we expect Z, the data transfer time for the best compromise striping unit over the positioning time, to be roughly equal to 1. We call Z the *zero-knowledge coefficient*, for it applies when no workload information is given. For the default disk, the best compromise striping unit is 30 KB. A request which accesses exactly one 30 KB striping unit would, on average, see a 23.5 ms positioning delay (14.7 ms seek plus 8.35 ms rotation), and a 16.7 ms data transfer. Thus Z in this case is .72.

Similarly, we wish to express the slope of the striping unit vs. concurrency line (Equation 1) in terms of disk parameters. As in Equation 2, we hypothesize that this slope will be proportional to the average positioning time multiplied by the data transfer rate.

$$slope = S \times average\ positioning\ time \times data\ transfer\ rate \quad (3)$$

For the default disk parameters, S, which we call the *concurrency-slope coefficient*, is .24. Substituting into Equation 1, we can express our striping unit choice at each concurrency *con* as

$$Striping\ Unit = S \times average\ positioning\ time \times data\ transfer\ rate \times (con-1) + .5\ KB \quad (4)$$

To verify these hypotheses, we repeat the simulation study above with different disk parameters. The disk parameters that we vary are seek speed, rotational speed, and sectors per track. These parameters all impact the *average positioning time*×*data transfer rate* factor in Equations 2 and 4.

disk type	(1) average positioning time	(2) data transfer rate	(1)x(2)	concurrency-slope coefficient	zero-knowledge coefficient
normal	23.1	1.8	41.4	.24	.72
2X fast seek	15.7	1.8	28.2	.22	.67
2X fast rotate	18.9	3.59	67.8	.23	.65
2X KB/track	23.1	3.59	82.8	.23	.63

For each disk technology, we calculate *S*, the *concurrency-slope coefficient*, as for the default disks (data is shown in Appendix Figure 8). First, at each concurrency, we measure throughput for a range of request sizes and determine the range of striping units which guarantee 95% of the maximum throughput to all request sizes. We then plot the line with the minimum slope which lies entirely in the striping unit range for all concurrencies. Lastly, we solve { $slope = S \times average\ positioning\ time \times data\ transfer\ rate$ } for *S*.

For each disk technology, we calculate *Z*, the *zero-knowledge coefficient*, by 1) graphing the percentage of maximum throughput for the same range of workloads as in Figure 7 (Appendix Figure 9), 2) determining the best compromise striping unit by the "maximizing the minimum percentage throughput" criterion, and 3) dividing the transfer time of the best compromise striping unit by the average positioning time.

Over a technology range where the *average positioning time*×*data transfer rate* varies by a factor of 3, from 28.2 to 82.8, both *S* and *Z* vary by only a small amount.[3] This verifies our model in which both the slope of the striping unit vs. concurrency line and the striping unit choice made without any workload information are proportional to the disks' *average positioning time*×*data transfer rate*.

[3]Note to the reviewer: we are in the process of collecting data for varying number of disks in the disk system. This will allow us to verify or extend the equations above for different numbers of disks.

8. Conclusions

We have seen that the striping unit choice is primarily dependent on the concurrency of the applied workload and is relatively insensitive to the request size distribution of the workload. Knowing the concurrency of the applied workload allows one to choose a striping unit which yields close to optimal performance for all request sizes. This choice can be expressed as

$$Striping\ Unit = S \times average\ positioning\ time \times data\ transfer\ rate \times (con-1) + .5\ KB$$

where S, the concurrency-slope coefficient, is approximately ¼. This relationship, and the specific value of S, was shown to hold over a wide range of disk technologies.

Also, without knowing the concurrency of the workload, the best compromise striping unit for a wide range of workloads and concurrencies can be chosen by

$$Compromise\ Striping\ Unit = Z \times average\ positioning\ time \times data\ transfer\ rate$$

where Z is roughly $\frac{2}{3}$.

Both the slope of the striping unit vs. concurrency line and the best compromise striping unit are dependent on only one disk parameter: the *average positioning time*×*data transfer rate*.

9. Future research

We are continuing to evaluate disk striping, with and without redundancy. Issues include varying the number of disks to stripe over, the redundancy scheme used, the effects of disk synchronization, and the possible use of zero-latency disk accesses for single track transfers.

10. References

[Agrawal84] R. Agrawal, D.J. DeWitt, *"Whither hundreds of processors in a database machine?,"* Proceedings of the International Workshop on High-Level Architectures, 1984.

[Bell84] C.G. Bell, *"The Mini and Micro Industries,"* IEEE Computer, Vol. 17, No. 10 (October 1984), pp. 14-30.

[Chen89] Chen, *"An Evaluation of Redundant Arrays of Disks Using an Amdahl 5890,"* to appear in Proceedings of the 1990 ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems, May 1989.

[Johnson84] O.G. Johnson, *"Three-dimensional wave equation computations on vector computers,"* Proc. IEEE, vol. 72, January 1984.

[Joy85] B. Joy, presentation at ISSCC 1985 panel session, Feb. 1985.

[Kim86] M.Y. Kim, *"Synchronized Disk Interleaving,"* IEEE Transactions on Computers, Vol. C-35, No. 11, November 1986, pp. 978-988.

[Lineback85] J.R. Lineback, *"New features tune unix for high-end machines,"* Electronics, August 1985.

[Livny87] M. Livny, S. Khoshafian, H. Boral, *"Multi-Disk Management Algorithms,"* Proceedings of the 1987 ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems, pp. 69-77.

[Moore75] G. Moore, *"Progress in Digital Integrated Electronics,"* Proc. IEEE Digital Integrated Electronic Device Meeting, 1975, p. 11.

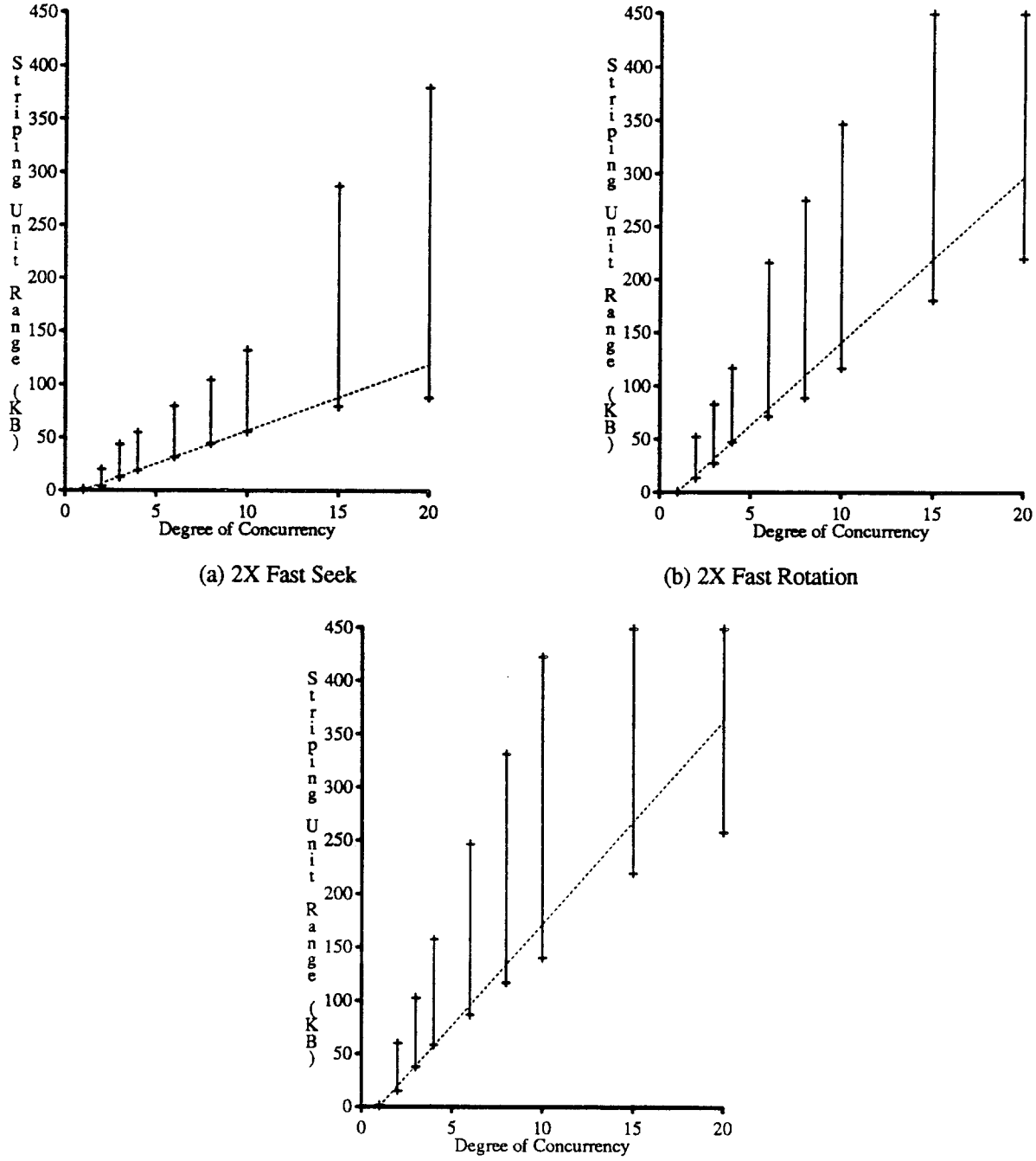
[Myers86] W. Myers, *"The Competitiveness of the United States Disk Industry,"* IEEE Computer, Vol. 19, No. 11, January 1986, pp. 85-90.

[Patterson88] D. Patterson, G. Gibson, R. Katz, *"A Case for Redundant Arrays of Inexpensive Disks (RAID),"* Proceedings of the 1988 ACM SIGMOD Conference on the Management of Data, pp. 109-116.

[Reddy89] A.L.N. Reddy, P. Banerjee, *"Performance Evaluation of Multiple-Disk I/O Systems,"* to appear in IEEE Transactions on Computers, December 1989.

[Salem86] K. Salem, H. Garcia-Molina, *"Disk Striping,"* Proceedings of the Second Data Engineering Conference, 1986, pp. 336-342.

[Thisquen88] J. Thisquen, *"Seek Time Measurements"*, Amdahl Peripheral Products Division Technical Report, May 9, 1988.



(a) 2X Fast Seek

(b) 2X Fast Rotation

(c) 2X Sectors per Track

Appendix Figure 8: Striping Unit Chosen versus Concurrency. Shown here is the range of striping units which yield at least 95% of the maximum throughput for all request sizes (*exp4k, exp16k, norm400k, norm1_5m*). The largest striping unit simulated was 450 KB.