A Queuing Analysis for Disk Array Systems

Mikito Ogata and Michael J. Flynn

Technical Report CSL-TR-90-443

August 1990

This research was conducted by Mr. Ogata while an industrial visiting scholar sponsored by Hitachi, Ltd., through the Stanford Computer Forum.

A Queuing Analysis for Disk Array Systems

by
Mikito Ogata and Michael J. Flynn

Technical Report CSL-TR-90-443
August 1990

Computer Systems Laboratory
Departments of Electrical Engineering and Computer Science
Stanford University
Stanford, California 94305-4055

Abstract

Using a queuing model of disk arrays, we study the performance and tradeoffs in disk array sub-systems and develop guidelines for designing these sub-systems in various CPU environments. Finally, we compare our model with some earlier simulation results.

Key Words and Phrases: Disk array, Striped disk, Synchronized disk

Contents

List of Figures

1 Introduction

1.1 Approaches for improving the disk I/O performance

A great deal of research has been done to increase the performance of disk systems and fill the performance gap between CPU and I/O. These efforts include:

1. Increasing the bit density
2. Increasing the data transfer rate
3. Increasing the seek speed
4. Reduction of path contention
5. Use of multi-actuator disk systems
6. Cache buffered disk systems
7. Optimization of file configuration
8. Improved seek scheduling policy
9. Multi-head disk systems
10. Semi-conductor storage disk systems
11. Disk Array systems

Some of these approaches are considered effective in improving the disk performance and are widely adapted. Others are either of limited effectiveness or are too expensive for general purpose environments. We study only the disk array system in this paper because of the current interest in this approach, its cost–performance, flexibility, and ease of integration into existing systems.

1.2 Related works

Kim [1] proposed a byte-interleaving disk array system and, using queuing models, analyzed their advantages and disadvantages. She concluded that disk arrays offers good performance only for light traffic.

Livny [2] proposed a so-called declustering, or "striped," configuration of several disk devices.

Figure 1: Recording Structure for Disk Device

Patterson, Katz, et al. [3, 4] proposed five types of disk array configurations, called RAID levels 1 to 5.

Reddy [5] studied some performance tradeoffs for disk array systems proposed above by means of simulation. He also proposed hybrid combinations of synchronized and striped disks.

Chen, et al. [6] study the effect of unit of interleaving for a RAID configuration. They deal only with highly striped configurations (16,1).

2 Modeling

2.1 Disk device

Figure 1 shows the organization of the data recording area on a typical disk device. The terms used to describe the device are:

- The disk device consists of a stack of platters (a).
- Each platter has two surfaces for recording the data (a).
- Each surface is divided into tracks (b).
- Each track is divided into sectors (b).

- The sector is the smallest addressable data unit.
- The set of tracks that have the same diameter on all data surfaces is called a cylinder (a). A cylinder contains the maximum amount of data that can be read/recorded with a single arm assembly movement.
- Each surface has its own arm, but all arms move together. Only one surface is read at a time.
- Seek time is the time to position the arm assembly at the desired track (and hence the desired cylinder).
- Latency time is the rotational delay for the disk to position the track at the correct sector starting point for a given access.
- Transfer time is the time required to transfer data stored on a track (a cylinder) to an external buffer. It does not include seek or latency time.
- Total disk access time is the sum of seek plus latency plus transfer time.

As an example, some parameters for the Hitachi DK516 disk device are presented in the following table.

Item	Value	Unit
No. of cylinders/device	1787	[Cylinders]
No. of user tracks/cylinder	15	[Tracks]
No. of servo track/cylinder	1	[Track]
No. of sectors/track	77	[Sectors]
Capacity/sector	512	[Bytes]

2.2 Disk array configuration

In this paper, we deal with various array system combinations. There are two basic methods of connection for multiple disk devices:

1. Synchronized: All disk devices within a group are synchronized for seek movement and rotation. Data is interleaved by byte and stored on all disks with (approximately) an equal amount of data. Data is transferred between data buffer and disks concurrently. Each group of synchronized disks can be regarded as a single large disk with a fast data transfer rate.

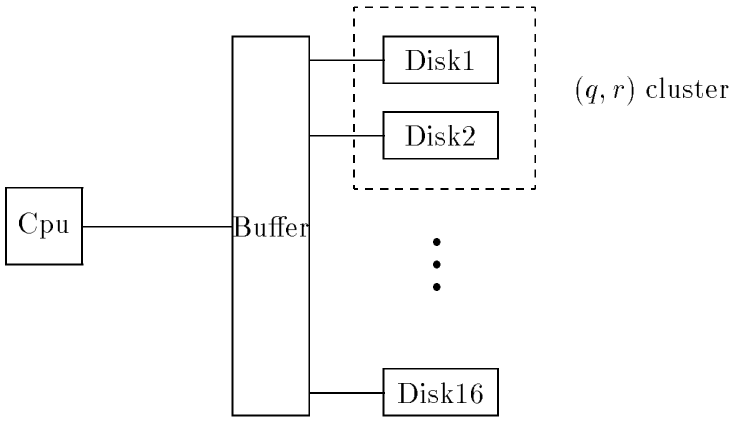


Figure 2: Open Queuing Model

2. Striped: Here disks are not synchronized for either seek movement or rotation. Data is interleaved by one or more sector units. Data is transferred to or from all disks in the group concurrently. This group is roughly regarded as a single conventional device for small (up to unit of interleaving) files and as a single large device with fast data transfer rate for large files, similar to synchronized systems.

A notation that is similar to one introduced by Reddy [5] is convenient for expressing these combinations. That is, the pair (q, r) means that

- one logical device consists of q groups of devices that are striped.
- one group of devices consists of r synchronized physical devices.

We describe the concept of a logical device later. The most significant difference between our notation and that of Reddy is that we allow concurrent data transfer for striped disks in a group.

2.3 Subsystem modeling

A queuing model for evaluating the performance of several disk array configurations is presented in this section. Figure 2 shows an open queue model. In this figure, each disk device is connected to the CPU via a data buffer. Data is accessed from or to the disk through the data buffer. All disk devices are split into groups, each a (q, r) combination. We fix the total number of disk devices at 16.

Let

(q, r)	: where q is the level of clustering
	: and r is the level of synchronizing

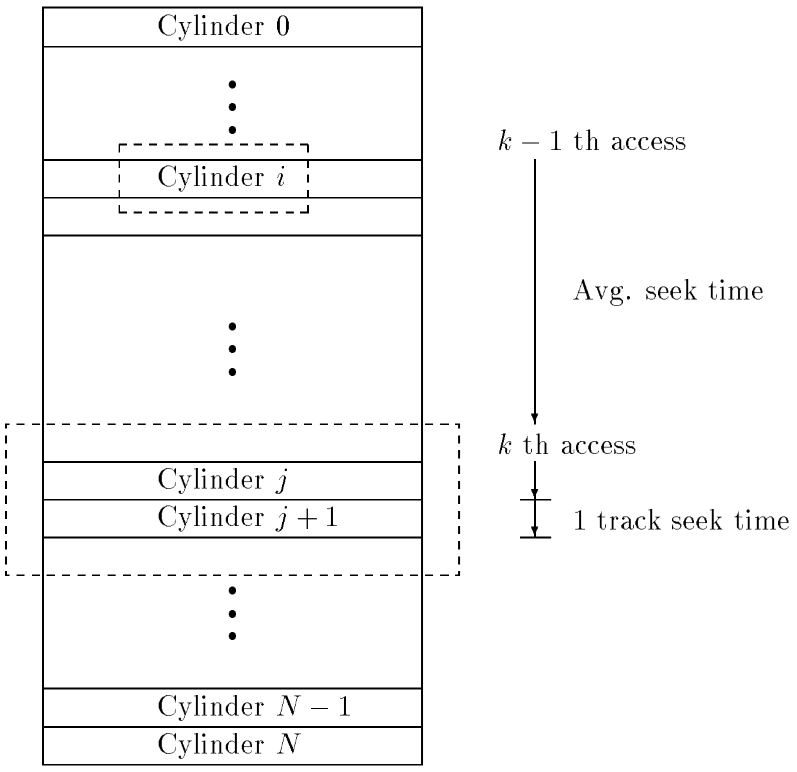


Figure 3: Seek Movement for File Access

$simul$	: number of logical devices that transfer the data concurrently
n	: the number of logical devices
$Tlat$	: latency time
$Tbuf$	: data transfer time between disk and data buffer
$Txfer$	: data transfer time between data buffer and CPU memory
$Tseek$	: seek movement time for desired track
W	: average waiting time for a logical device
c	: coefficient of variation of S
S	: average service time of device
R	: average response time (sum of R and S)

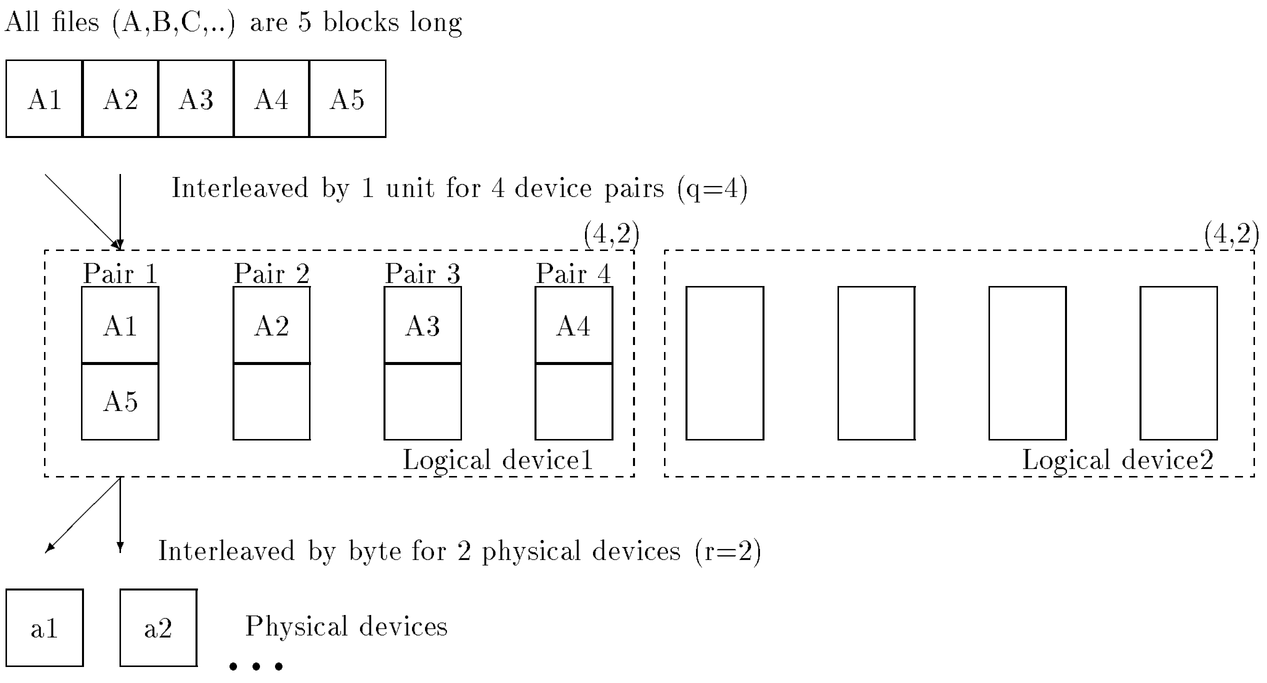

All files (A,B,C,..) are 5 blocks long
A1
A2
A3
A4
A5
Interleaved by 1 unit for 4 device pairs (q=4)
(4,2)
(4,2)
Pair 1
Pair 2
Pair 3
Pair 4
A1
A2
A3
A4
A5
Logical device1
Logical device2
Interleaved by byte for 2 physical devices (r=2)
a1
a2
Physical devices

Figure 4: Number of logical devices for 5 blocks long. Sixteen physical disks configured as a pair of (4,2). In this case n=2, since each file request accesses 8 physical disk (4×2) and there are 16 total disks ($n = \frac{16}{8} = 2$).

v	: data transfer rate between buffer and disk device in [KB/sec]
ρ	: utilization ratio of device
l	: file size in [KB]
d	: unit of data interleaving for each device in [KB]
λ	: traffic rate (the number of I/O requests per second)

A logical device is a group of physical disks that transfer data simultaneously. The group acts as a large single disk. The size of a logical device is defined as the average number of physical devices that are simultaneously engaged in a file access. Both the number of logical devices and their size depend on the unit of file distribution and the distribution itself. The number of logical devices (or NLD) is the total number of logical devices in the system. It is defined as the total number of physical disks divided by the size of the logical devices.

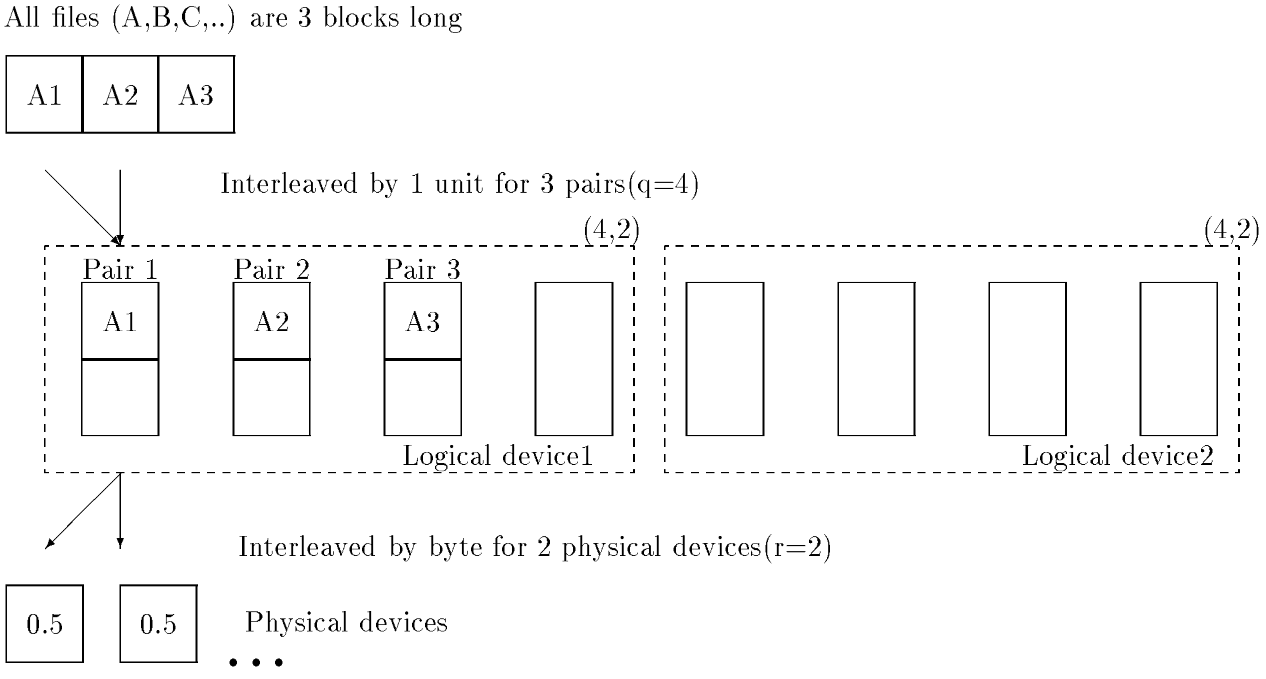

All files (A,B,C,..) are 3 blocks long
A1
A2
A3
Interleaved by 1 unit for 3 pairs(q=4)
(4,2)
(4,2)
Pair 1
Pair 2
Pair 3
A1
A2
A3
Logical device1
Logical device2
Interleaved by byte for 2 physical devices(r=2)
0.5
0.5
Physical devices

Figure 5: Number of logical devices for 3 blocks long. Sixteen physical disks configured as a pair of (4,2). In this case n=2.7, since each file request accesses 6 physical disk (3×2) and there are 16 total disks ($n = \frac{16}{6} = 2.7$).

Figure 4 shows an example of logical devices when the disk subsystem is configured (4,2). Files are 5 blocks long. Here, block is defined as a unit of file distribution. Typical block size is 4 [KB]. Because files are interleaved by 1 block for striped configurations, 5 blocks are distributed over four striped disk pairs. Also, because each striped group consists of two synchronized physical devices (r=2), data is byte-interleaved over two physical devices. The size of the logical device is $q \times r = 4 \times 2 = 8$. The number of logical devices is $\frac{16}{the\ size\ of\ logical\ device} = \frac{16}{8} = 2$.

Figure 5 shows another example using the (4,2) configuration. Here, files are 3 blocks long and the file is distributed over only three of the four pairs. For these three disk pairs, data is byte-interleaved for two physical devices as before. Each physical device has half of a block. The size of logical device = $3 \times 2 = 6$ and the number of logical devices is $\frac{16}{6} = 2.7$.

We define these values more completely as follows.

A. *simul*: the number of logical devices that transfer the data concurrently

In (q,r) configuration, the number of devices that transfer the data for the buffer concurrently is defined as

$$simul = min(q, \lceil \frac{l}{d} \rceil) \tag{1}$$

In the case of Figure 4, $simul = 4$.

B. n: number of logical devices

n is calculated from $simul$ as

$$n = \frac{16}{r \times E(simul)} \tag{2}$$

In the case of Figure 4, $n = 2$.

C. $Tlat$: latency time

We assume the starting address of a block is uniformly distributed over a track. Therefore, we assume the average latency time is half of the rotation time of disk surface, that is:

$$E(Tlat) = 16.6/2 = 8.3[ms] \tag{3}$$

D. $Tseek$: average seek time

Much work has been done on the distribution of seek time under several scheduling policies [7, 8, 9, 10, 11]. Among them, the most typical asymptotic model assumes a uniform access distribution. This specifies that any incoming I/O request is equally likely to access any cylinder.

Under this assumption we can derive (see Bitton [10]):

$$P(X = i) = \frac{2(N-i)}{N^2}$$

$$P(X \geq i) = \frac{(N-i)(N-i+1)}{N^2}$$

$$E(X_{min}) \approx \frac{N}{2simul+1}$$

$$E(X_{max}) \approx N(1 - \frac{2simul}{2simul+1}\frac{2simul-2}{2simul-1}\cdots\frac{2}{3})$$

where

X	:	random variable over $1\ to N-1$
N	:	the number of cylinders that contain the user files
i	:	seek distance
X_{min}	=	$\min(X_1, X_2, \ldots, X_k)$
X_{max}	=	$\max(X_1, X_2, \ldots, X_k)$

If we assume all of the files are distributed continuously on the disk cylinders, generally N can be calculated from file size as follows:

$$N = \frac{\textit{Total file size}}{\textit{No. of tracks per cylinder} \times \textit{No. of sectors per track} \times \textit{Sector size} \times \textit{No. of disk devices}}$$

The assumption of uniform access distribution is often used because it is easy to handle analytically. However, trace data indicate this assumption is not accurate, especially in the case of seek distance equal zero. It is reported [12, 9] that about half of all seek actions are zero-distance seeks, while this occurrence is calculated as N/N^2 under the uniform access distribution model.

On the other hand, differences among seek models or seek scheduling policies do not cause a significant difference in system performance, because

1. Seek time is not linearly proportional to the distance of the access arm travel.

2. Queue length is rarely longer than 1 in most systems.

Also, Bitton [11] and Scranton [12] indicate that the movement time of a high performance voice coil motor can be approximated as a formula:

$$Time = A + B \times \sqrt{Seek\ distance}$$

From this and our assumption that about half of the seeks have zero seek time, we assume the average seek time is approximately

$$average\ seek\ time = 0.5 \times (a + b\sqrt{N_0})$$

$$N_0 = N\left(1 - \frac{2simul}{2simul+1}\frac{2simul-2}{2simul-1}\cdots\frac{2}{3}\right)$$

where a and b are defined by the feature of mechanical motor which is implemented in the disk for seek movement. In the case of Hitachi DK-516 disk, a=3 and b=0.45082.

In addition, a long file may require an access that exceeds the cylinder boundary (see figure 3). We calculate the additional seek time caused by cylinder boundary overflow as follows. Let l be the file size of concurrent data transfer, cyl be the cylinder size, and c and d the quotient and remainder of l divided by cyl, respectively. That is,

$$l = c \times cyl + d. \qquad (0 \le d < cyl)$$

Then,

$$\begin{aligned} additional\ seek\ time &= one\ track\ seek\ time \\ &\quad \times \left(c + (1 - \frac{d}{cyl})^{simul} \times 0 + \left(1 - (1 - \frac{d}{cyl})^{simul}\right) \times 1\right) \end{aligned}$$

Where:
$Probability\ that\ no\ device\ has\ additional\ seek\ movement = (1 - \frac{d}{cyl})^{simul}$
$Probability\ that\ at\ least\ one\ device\ has\ additional\ seek\ movement = 1 - (1 - \frac{d}{cyl})^{simul}$
Then,

$$E(Tseek) = average\ seek\ time + E(additional\ seek\ time) \tag{4}$$

E. $Txfer$: data transfer time between data buffer and CPU memory

We assume the minimum size to be accessed is the sector size, which is 0.5 [KB]. If $simul$ is equal to 1, all of the file is recorded on a single logical device. Then $\lceil \frac{l}{0.5} \rceil \times 0.5$ is transferred for one I/O execution.
If $simul$ is not equal to 1, the file is distributed over $simul$ logical devices by units of d. Then, the amount of data to be transferred is $\lceil \frac{\lceil \frac{l}{d} \rceil}{simul} \rceil \times d$. Therefore,

$$Txfer = \begin{array}{ll} \frac{\lceil \frac{l}{0.5} \rceil \times 0.5}{v \cdot r} & (simul = 1) \\ \frac{\lceil \frac{\lceil \frac{l}{d} \rceil}{simul} \rceil \times d}{v \cdot r} & (simul \neq 1) \end{array} \tag{5}$$

F. $Tbuf$: data transfer time between disk and data buffer

In our model all data is stored in the data buffer, which is attached to the disk cluster before a data transfer from disk to main memory and after a data transfer from memory to disk. Therefore, $Tbuf$ is the same as $Txfer$.

G. S: service time of devices

From the components described above, we calculate S as

$$S = E(Tlat) + E(Tseek) + E(Tbuf) + E(Txfer) \tag{6}$$

H. W: waiting time for device

From the queuing theory for M/G/1 model, we get

$$W = \frac{(c^2+1)}{2} \cdot \frac{\rho}{(1-\rho)} \cdot S \tag{7}$$

Here, we assume traffic for each disk is equal over all devices. Therefore,

$$\rho = \frac{\lambda}{n} \cdot S \tag{8}$$

c is defined as the *coefficient of variation* or

$$c \overset{\text{def}}{=} \frac{\sigma_T}{\bar{T}}$$

$$\bar{T} = \int_{-\infty}^{\infty} t \cdot f(t) dt$$

$$(\sigma_T)^2 = Var(T) = \bar{T^2} - (\bar{T})^2$$

Also, we use the following:

$$\sigma_x^2 = Var(x) = Var(Tlat) + Var(Tseek) + Var(Tbuf) + Var(Txfer) \tag{9}$$

I. R: response time

$$R = W + S \tag{10}$$

2.4 System modeling

In order to calculate the system performance, we use the so-called central server model [13]. Figure 2 shows this model. In this model, the disk subsystem is connected to the CPU group via a data buffer.

Let

p_0	: probability that the disk subsystem is in idle state.
ρ_{sys}	: utilization of the disk subsystem
K	: number of logical processes
S	: service time of the disk subsystem
U	: CPU time interval between I/O requests
W	: waiting time for a device

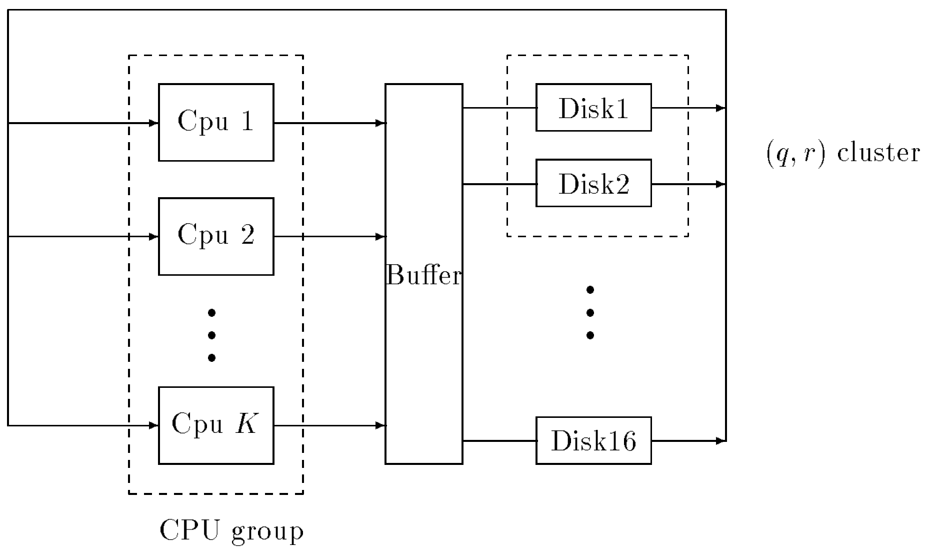


Figure 6: Central Server Model

A. K: the number of logical processes

K is a value defined as the number of independently executed logical processes. The execution of each process requires the following steps.

1. Computation in a processor for time period U
2. Issue the I/O request
3. I/O waiting time period, W
4. I/O execution time, S

B. Throughput: amount of data to be transferred per second

In this model, from Little's law,

$$(R+U)\lambda = K$$

and

$$\lambda = \frac{1-p_0}{S}$$

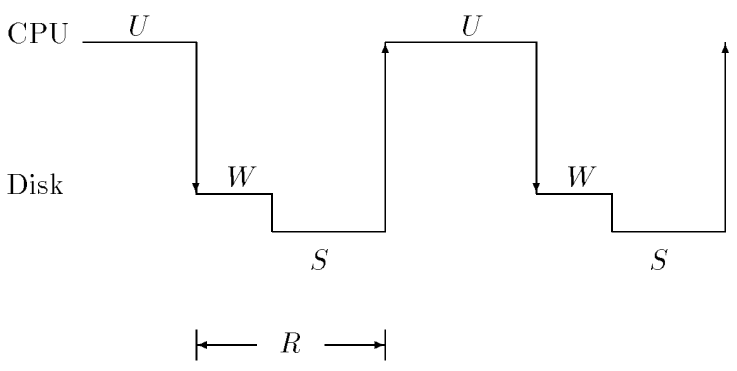


Figure 7: Timing Chart of Disk Access

Then,

$$\begin{aligned} R &= \frac{KS}{1-p_0} - U \\ &= \frac{KS}{\rho_{sys}} - U \end{aligned} \tag{11}$$

On the other hand, from equations (7) and (10) we have another expression for R:

$$R = \frac{1+c^2}{2} \cdot \frac{\rho}{1-\rho} \cdot S + S$$

$$\rho = \frac{\lambda}{n} \cdot S$$

Besides,

$$\rho_{sys} = \lambda S$$

From the above equations, we can find λ as:

$$\lambda = \frac{-B \pm \sqrt{B^2 - AC}}{A}$$

where $A = \frac{1}{n} \cdot \{(1+c^2)S^2 - 2S(S+U)\}$
$B = (S+U) + \frac{K}{n} \cdot S$
$C = -2K$

However, since $\rho \le 1$ requires $\lambda S \le n$ and for the root λ, we have

$$\begin{aligned}\lambda_1 &= \frac{-B-\sqrt{B^2-AC}}{A} \\ &> \frac{-2B}{A} \\ &= -\frac{n}{S}\frac{2U+2S+\frac{2KS}{n}}{2U+S-c^2S} \\ &\ge \frac{n}{S} \qquad\qquad (when\ (c^2-1)S \le 2U)\end{aligned}$$

Therefore, λ_1 is not a valid solution in the cases of $(c^2-1)S \le 2U$ and $c^2 \le 1$. Thus, the only feasible solution is the root:

$$\lambda = \frac{-B+\sqrt{B^2-AC}}{A} \tag{12}$$

Using this λ, we define:

$$Throughput = \lambda \times E(l) \tag{13}$$

C. Estimate of K

K is defined as the number of logical processes executed by an ensemble of processors without overlapping I/O execution. K can be estimated from the number of processors and the number of tasks in a system environment. In a real system, each processor has many tasks allocated to it and each task can be executed by the processor with CPU time (U) overlapped with I/O time ($S+W$). Let N be the number of processors and m be the maximum number of tasks. The traffic rate is estimated as

$$\lambda = \frac{number\ of\ processors}{U} = \frac{N}{U}$$

On the other hand, from Little's law

$$\lambda = \frac{K}{R+U} = \frac{K}{U+W+S}$$

Then,

$$K = \frac{U+W+S}{U} \times N \tag{14}$$

From this equation, K approaches N when the system is CPU bound (W is approximately equal to zero and U approaches infinity). K approaches m when the system is I/O bound (U approaches zero).

2.5 Other terminology

We define some additional terms as follows.

Concurrency	: The value of $q \times r$ for each (q,r) configuration.
Striped configuration	: A (q, r) configuration where q is not equal to 1, r is equal to 1.
Synchronized configuration	: A (q, r) configuration where r is not equal to 1, q is equal to 1.
Mixed configuration	: A (q, r) configuration where both q and r are not equal to 1.
Block	: Minimum unit of file transfer from disk sub-system cluster to buffer or from buffer to CPU (in bytes). The block must consist of an integral number of sectors. To avoid capacity loss, the minimum block size should be rx sector size.
Depth	: Unit size of interleaving (number of blocks or KB) for a striped or a mixed configuration.

Figure 8 shows the concept of depth. This figure shows a file size of 8 blocks and depth of 2 blocks. In this paper, the block size is 1 [KB]. We could use any unit for depth, such as 1 [sector], 1 [block], 1 [KB], 1 [page], 1 [track], etc., but choose the block or KB for simplicity. In our model, the file is interleaved for striped disks by the unit of depth and interleaved for synchronized disks by the byte.

3 Basic behavior of system

3.1 Subsystem behavior

A. Workload1

A basic workload distribution that we use to evaluate the subsystem performance is:

- File size is $4 \times M (M = 1, 2, ...8)$ [KB].
- One size of file (value of M) has 70 [%] traffic.
- The other sizes of files have equally $30 \times 1/7$ [%] traffic.

We consider other workload distributions at the end of the paper. We call the 70 [%] traffic size the peak size.

B. Number of Logical Devices (NLD)

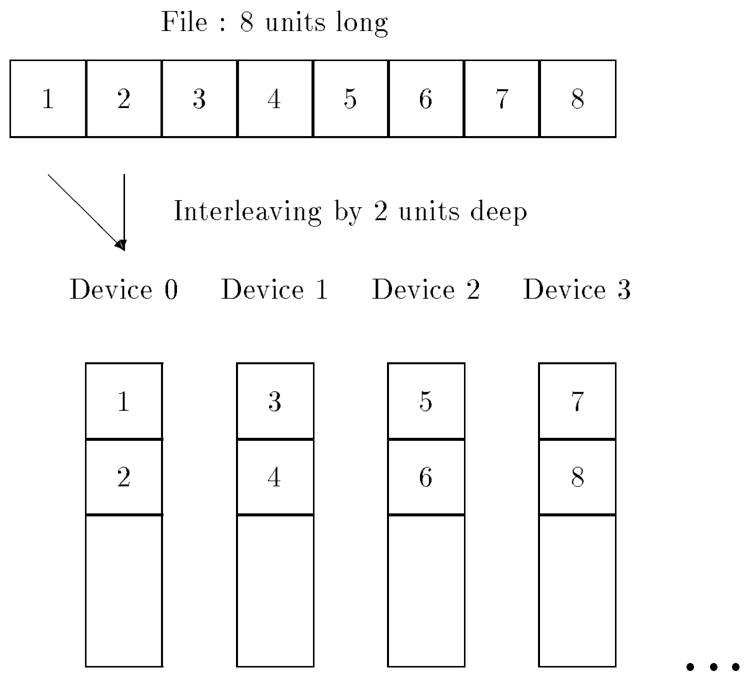


Figure 8: Concept of Depth

As reported [14], striping involves the tradeoff of increasing the data transfer rate by a factor of n while reducing the number of logical devices in the storage subsystem by the same factor of n.

The NLD or n is calculated by our model using equation (2). Figure 9 shows NLD when peak size is changed from 4[KB] to 32[KB]. From this figure, NLD is reduced as peak size is increased. Logically, the NLD has the following characteristics.

(1). If q is the same over two configurations, larger r induces smaller NLD. This is why r synchronized physical devices behave logically as only one device for all file sizes. That is,

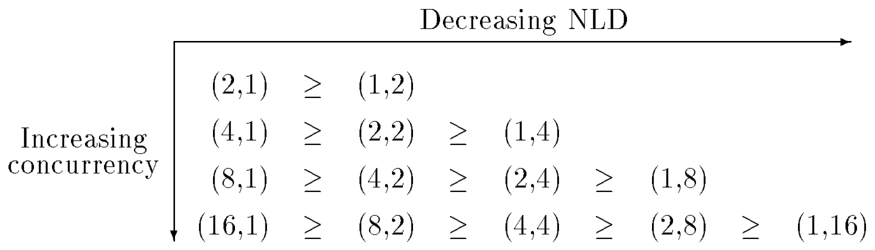
$$\begin{array}{c|ccccccccc}
 & \multicolumn{9}{c}{\text{Decreasing NLD} \longrightarrow} \\
 & (1,1) & > & (1,2) & > & (1,4) & > & (1,8) & > & (1,16) \\
\text{Increasing} & (2,1) & > & (2,2) & > & (2,4) & > & (2,8) & & \\
\text{concurrency} & (4,1) & > & (4,2) & > & (4,4) & & & & \\
\downarrow & (8,1) & > & (8,2) & & & & & &
\end{array}$$

Figure 9: Number of Logical Devices

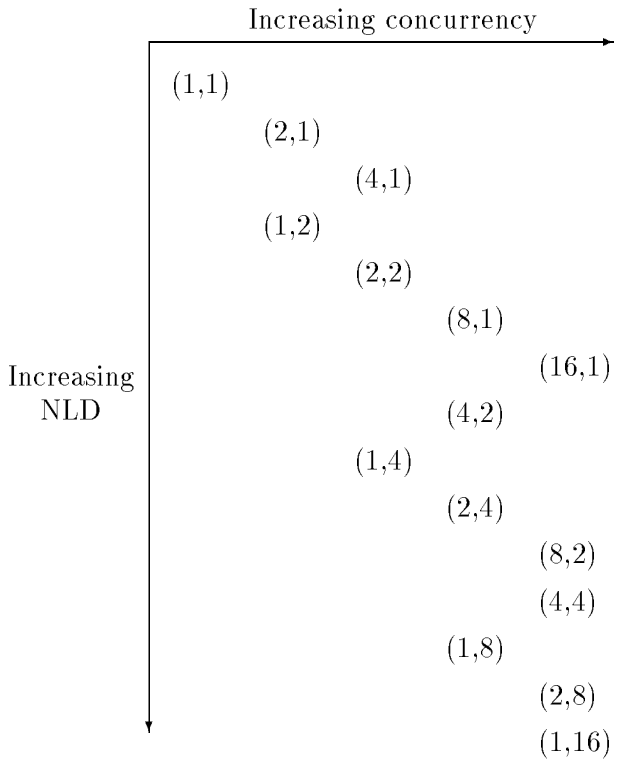
(2). If r is the same for two configurations, a larger q induces a smaller or equal NLD. This is because q striped physical devices act as q logical devices for very small files and behave as less than q logical devices for large files. Then,

$$\begin{array}{r|ccccccccc}
 & \multicolumn{9}{c}{\text{Decreasing NLD} \longrightarrow} \\
\hline
 & (1,1) & \geq & (2,1) & \geq & (4,1) & \geq & (8,1) & \geq & (16,1) \\
\text{Increasing} & (1,2) & \geq & (2,2) & \geq & (2,4) & \geq & (2,8) & & \\
\text{concurrency} & (1,4) & \geq & (2,4) & \geq & (4,4) & & & & \\
\downarrow & (1,8) & \geq & (2,8) & & & & & &
\end{array}$$

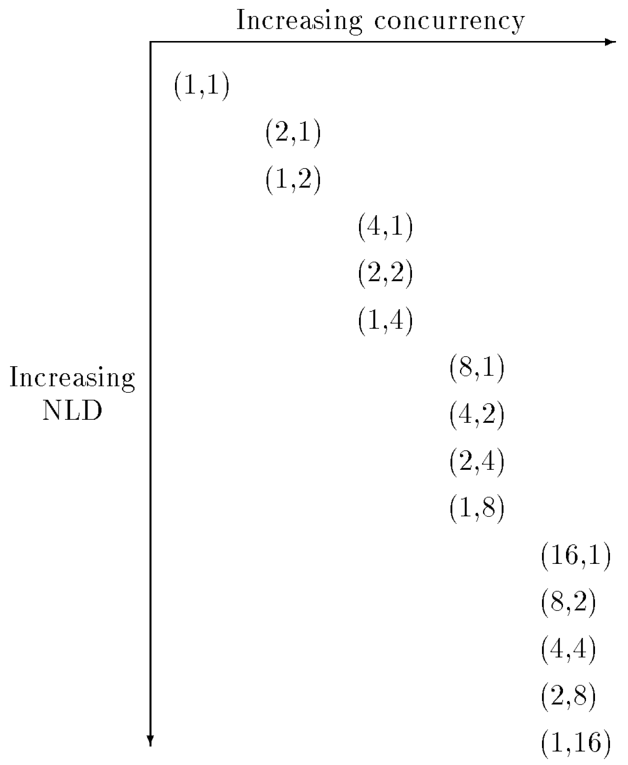
(3). If the concurrency is the same for two configurations, a larger number of synchronized devices induces a smaller NLD than the equivalent number of striped devices, since synchronized disks don't allow more than one logical device even in the case of a very small file. Then,



(4). Also, analytically we get the following results from equations (1) and (2). For small files (peak 4 [KB]), NLD is only partially determined by concurrency (see figure 9). The order of NLD for this file distribution is



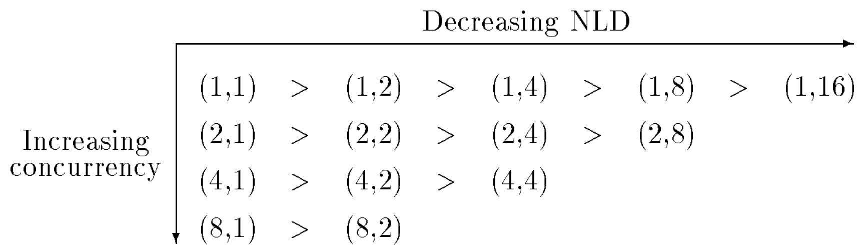
(5). For large file size, configurations with the same NLDs have approximately the same concurrency. Workload1 with peak file size 32 [KB] induces the order of NLD



C. Service Time (S)

Logically, service time behaves over all configurations as NLD. The performance of a disk array system is basically a tradeoff between lower service time and a smaller number of logical devices.

(1). If q is the same for two configurations, larger r induces smaller S. This is because r synchronized physical devices allow r concurrent data transfer. That is,



(2). If r is the same for two configurations, larger q induces smaller or equal S. For small files, q striped physical devices allow one concurrent data transfer while large files use more than one concurrent data transfer.

Figure 10: Service Time of Device

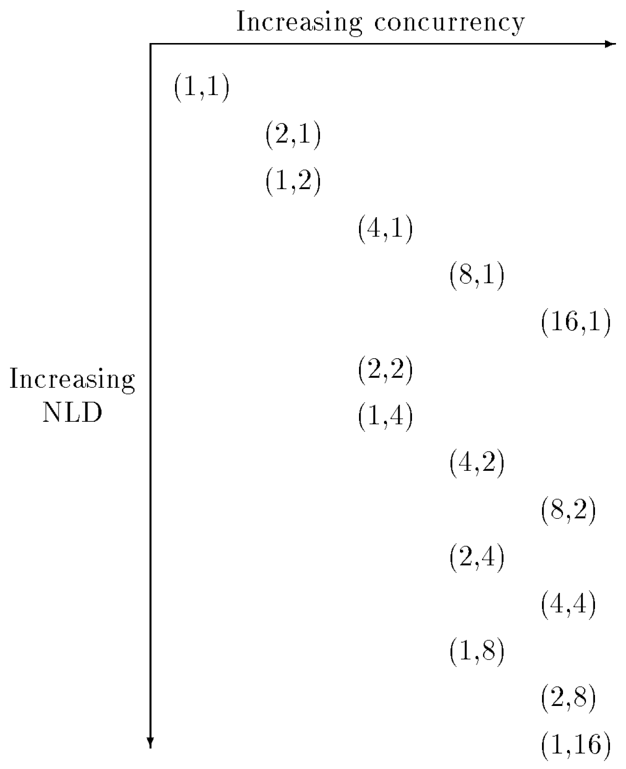
$$\begin{array}{c|ccccccccc}
 & \multicolumn{9}{c}{\text{Decreasing NLD} \longrightarrow} \\
\hline
\downarrow & (1,1) & \geq & (2,1) & \geq & (4,1) & \geq & (8,1) & \geq & (16,1) \\
\text{Increasing} & (1,2) & \geq & (2,2) & \geq & (2,4) & \geq & (2,8) & & \\
\text{concurrency} & (1,4) & \geq & (2,4) & \geq & (4,4) & & & & \\
 & (1,8) & \geq & (2,8) & & & & & &
\end{array}$$

(3). If the concurrency is the same for two configurations, larger synchronized devices induce smaller S than the equivalent number of striped devices. This is because striped disks use a larger interleaving unit than the equivalent number of synchronized disks. That is,

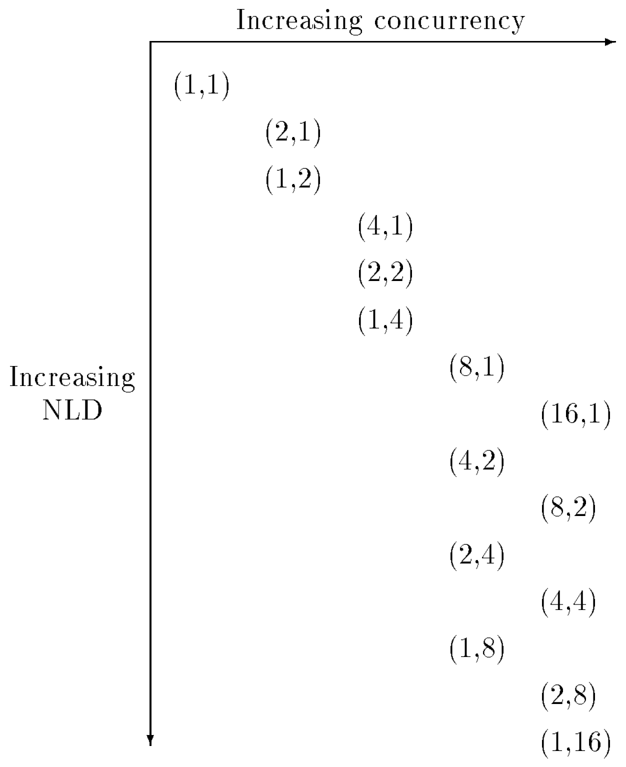
$$\begin{array}{c|ccccccccc}
 & \multicolumn{9}{c}{\text{Decreasing NLD} \longrightarrow} \\
\hline
\downarrow & (2,1) & \geq & (1,2) & & & & & & \\
\text{Increasing} & (4,1) & \geq & (2,2) & \geq & (1,4) & & & & \\
\text{concurrency} & (8,1) & \geq & (4,2) & \geq & (2,4) & \geq & (1,8) & & \\
 & (16,1) & \geq & (8,2) & \geq & (4,4) & \geq & (2,8) & \geq & (1,16)
\end{array}$$

Also, analytically we get the following.

(4). For small files, S is only partially determined by concurrency (see figure 10). For peak size 4 [KB], the order of S is



(5). For large file size, configurations with the same concurrency have approximately the same S. Peak size 32 [KB] induces the ordering:



D. Utilization of logical device

Roughly, disk synchronization or disk striping for n physical devices improves data transfer by a factor of n, while logical device number decreases by a factor of $1/n$. However, response time is determined not just by data transfer time but also by total device occupancy time. It consists of a data transfer time and some fixed bias time such as seek or latency. Hence, for performance, disk synchronization or disk striping makes logical device number smaller by a factor of $1/n$ while service time is shortened by a factor larger than $1/n$. This is the most important feature of disk array system. If the factors for logical device number and service time were the same, the utilization of logical device which is defined by $\lambda \times S/n$ would be the same over all configurations. This also would make the tradeoff of disk array system much easier.

We can see this imbalance of factors for service time and logical device number in figure 11.

E. Response time

Figures 12 and 13 show the response time for files of peak size 4 [KB] and 32 [KB], respectively. From these figures,

- The configuration with smaller service time has the best performance at low traffic rate.
- The configuration with larger NLD has the best performance at high traffic rate.

Figure 11: NLD vs. S

In addition,

- For small file size, S and NLD are only partially determined by concurrency.
- For large file size, S and NLD are primarily determined by concurrency.

NLD is more strongly determined for large files than S (figures 9 and 10). Therefore, in the case of 32[KB], performance is dominated by the mainly smaller S. On the other hand, for small files, S does not strongly affect performance because the role of S in the total service time is small. Therefore, in the case of peak size 4 [KB], performance is dominated by larger NLD.

From these graphs, the maximum performance configuration in order of traffic is (1,16),(1,8),(2,4),(1,4),(2,2)... in the case of 4 [KB], and (1,16),(1,8),(1,4)... in the case of a 32 [KB] file. NLD is the key performance factor for small files and S is that for large files.

3.2 System behavior

Figure 14 shows throughput versus processing rate when $K = 8$ (K is the number of logical processes). Processing rate is defined as $1/U$ in our model. For example, processing rate=0.1

Figure 12: Response Time for 4 [KB] File

means 100 [ms] interval between requests by CPUs. From this figure, the system throughput defined by equation (13) increases as the processor becomes faster (processing rate increases), but later (after point A in figure) the throughput approaches saturation due to longer response time caused by increased waiting time, and finally system throughput saturates with respect to processing rate (after point B). We call point A, a *throughput slowdown point* and point B, a *throughput saturation point.* A is approximately 0.1 [logical processes/ms] for a 4 [KB] file and 8 CPUs and 0.01 [logical processes/ms] for 256 [KB] file and 8 CPUs. B is approximately 1.0 [logical processes/ms] for 4 [KB] file and 0.1 [logical processes/ms] for 256 [KB] file. The reason that A and B for 256 [KB] file size occur at a lower processing rate than those for 4 [KB] file size is that the data transfer time for 256 [KB] is larger than that for 4 [KB].

The best performance configurations after throughput saturation, in the case of an I/O bottleneck, are (1,1), (2,1), (4,1), (1,2) for a 4 [KB] file and (1,4), (2,2), (4,1), (1,8) for a 256 [KB] file. The worst configurations are (1,16), (2,8), (1,8), (4,4) for 4 [KB] file and (1,1), (16,1), (8,2), (4,4) for 256 [KB] file. The fact that higher concurrency does not offer the best configuration is remarkable.

Figure 13: Response Time for 32 [KB] File

4 Effects of factors

In this section, we investigate the best performing configurations for a fixed workload and fixed parameters. We consider the effect of the following several factors.

- $1/U$: Processing rate
- K: The number of logical processes
- d: Depth
- File distribution

Figure 14: Throughput Saturation

4.1 The effect of processing rate

From equation (10) in section 2, the response time of disk array is

$$R = \frac{KS}{\rho_{sys}} - U = \frac{\rho(1+c^2)}{2(1-\rho)}S + S = W + S$$

System behavior can be best understood by considering the asymptotic behavior first for low, then for high traffic.

A. Low Traffic Asymptotic Behavior

For low traffic, waiting time does not affect system performance because the CPU has some idle time. Therefore, a smaller service time offers better performance. For low traffic rate, we let *waiting time* (W) in the preceding equation approach zero. Then, we get

$$R = \frac{KS}{\rho_{sys}} - U = S$$

This equation can be solved for λ,

$$\lambda = \frac{\rho_{sys}}{S} = \frac{K}{S+U} \tag{15}$$

Therefore, the performance at low traffic is dominated by S under fixed K and U. As mentioned earlier, larger concurrency ($q \times r$) induces smaller S. For low traffic, the configurations with larger concurrency are faster than those with smaller concurrency.

B. High Traffic Asymptotic Behavior

For high traffic, total waiting time and service time dominate system performance. Increased waiting time is primarily caused by small NLD. So, in this case, the tradeoff of NLD and S is more sensitive. For high traffic, we can let $U \to 0$ in equation (12). Then, we get

$$\lim_{U \to 0} \lambda = \frac{-(n+K) + \sqrt{n^2 + K^2 + 2nc^2K}}{(c^2-1)S} \tag{16}$$

Especially, when $c = 1$ (ex. exponential service time)

$$\lim_{U \to 0} \lambda = \frac{n}{S} \cdot \frac{K}{n+K} \tag{17}$$

While, when $c = 0$ (ex. uniform service time)

$$\lim_{U \to 0} \lambda = \frac{n + K - \sqrt{n^2 + K^2}}{S} \tag{18}$$

From these formulas, at high traffic the performance depends not only on S but also on n (NLD).

4.2 The effect of the number of logical processes

For K (number of logical processes), we can get the following relationship from equations (15) to (17).

$$\frac{n + K - \sqrt{n^2 + K^2}}{S} \geq \frac{-(n+K) + \sqrt{n^2 + K^2 + 2nc^2K}}{(c^2-1)S} \geq \frac{n}{S} \cdot \frac{K}{n+K}$$

Since the performance (λ) improves as the coefficient of variation (c) decreases. When $K \to \infty$, both the left term and right term converge to n/S. Hence, we can get

$$\lim_{K \to \infty} \frac{-(n+K) + \sqrt{n^2 + K^2 + 2nc^2K}}{(c^2-1)S} = \frac{n}{S} \tag{19}$$

The performance asymptotically approaches the value n/S as K increases. As we mentioned before, changing disk configuration generally causes the pair of (n,S) to move on the linear line of $S = \alpha \cdot n + \beta$, where $\alpha, \beta > 0$ (see figure 11). Then, n/S is

$$\frac{n}{S} = \frac{1}{\alpha} - \frac{1}{\alpha S}$$

Figure 15: Best Concurrency

Hence, the performance improves as S increases, that is, as concurrency decreases. Configurations with less concurrency have better performance at high K values.

Figure 15 shows which system is the best for a small files (peak size 4 [KB] of workload1). Figure 15 is partitioned into regions of optimum concurrency, where

- N-dominant: the configurations in which concurrency is equal to 1 or 2.
- 1/S-dominant: the configurations in which concurrency is equal to 16.
- N,S tradeoff: the configurations in which concurrency is equal to 8 or 4.

From this figure,

- High concurrency is optimum at low traffic rate system.
- Low concurrency is optimum when there are a number of logical processes.
- Concurrency 4 or 8 is optimum for low K and high traffic.
- Concurrency 4 or 8 is optimum for intermediate traffic where CPU and I/O rates are balanced.

Figure 16: Effect of Depth (a),(b)

Also, we get the following tables from our experiment.

In the case of filesize = 4[KB],

K	CPU-bound	CPU,I/O balanced	I/O bound
1	4,8,16	4,8	4,8
4	4,8,16	4,8	4,8
16	1-16	1,2	1,2

In the case of filesize = 256[KB],

K	CPU-bound	CPU,I/O balanced	I/O bound
1	16	16	4,8
4	16	4,8	4,8
16	16	1,2	1,2

Figure 17: Effect of Depth (c),(d)

4.3 The effect of depth

The performance of a disk array subsystem also depends on n (or NLD) and S. The amount of data for one I/O request (D) is, from section 2.3 **E**,

$$
\begin{aligned}
D &= \lceil \frac{\lceil \frac{l}{d} \rceil}{simul} \rceil && simul \neq 1 \\
&\lceil \frac{l}{0.5} \rceil && simul = 1
\end{aligned}
$$

$$simul = min(q, \lceil \frac{l}{d} \rceil)$$

From equation (6), let $Tlat + Tseek = \alpha$; recall that $Tbuf = Txfer$. Then using equation (5),

$$
\begin{aligned}
S &= \alpha + \frac{2dD}{vr} && simul \neq 1 \\
&\alpha + \frac{2 \cdot 0.5D}{vr} && simul = 1
\end{aligned}
$$

From these equations, an increase in d causes both n (or NLD) and S to increase. Figures 16 (a),(b) and 17 (c),(d) show the effect of an increase in d on system performance.

Choosing the appropriate depth is an effective means for maximizing performance for some file distributions and some types of configurations.

A. Approximate model of depth effect

In order to evaluate the effect of depth, we use the approximate formulas for n (or NLD) and S. We approximate NLD by

$$\begin{array}{llll} n = & \frac{16}{q\times r} & d < \dfrac{l}{q} & (simul = q) \\ & \frac{16d}{lr} & \dfrac{l}{q} \leq d \leq l & (simul = \lceil \dfrac{l}{d} \rceil) \\ & \frac{16}{r} & d > l & (simul = 1) \end{array} \tag{20}$$

Also we can approximate S by

$$\begin{array}{llll} S = & \alpha + \frac{2lt}{qr} & d < \dfrac{l}{q} & (simul = q) \\ & \alpha + \frac{2dt}{r} & \dfrac{l}{q} \leq d \leq l & (simul = \lceil \dfrac{l}{d} \rceil) \\ & \alpha + \frac{2lt}{r} & d > l & (simul = 1) \end{array} \tag{21}$$

Here, for purposes of explanation, we consider only the case of $c = 1$ (the worst case c under consideration).

B. Effect of depth

(1). The case of $d < \frac{l}{q}$

We get from equation (15),

$$\begin{aligned} \lambda &= \frac{n}{S} \cdot \frac{K}{K+n} \\ &= \frac{16Kqr}{(\alpha qr + 2lt)(Kqr + 16)} \end{aligned} \tag{22}$$

Note that here the performance is not a function of d.

(2). The case of $\frac{l}{q} \leq d \leq l$

$$\lambda = \frac{16dKr}{(\alpha r + 2dt)(16d + lKr)} \tag{23}$$

Also,

$$\frac{\partial \lambda}{\partial d} = \frac{16Kr(\alpha lKr^2 - 32d^2t)}{(\alpha r + 2dt)^2(16d + lKr)^2} \tag{24}$$

Hence, the performance goes up until d reaches the value $r\sqrt{\alpha lK}/4\sqrt{2t}$ (determined by $\partial\lambda/\partial d = 0$) achieves the maximum performance, and decreases. that is,

$$maximum \;\; performance = \frac{4r\sqrt{2t\alpha lK}K}{(2\alpha\sqrt{2t} + \sqrt{r\alpha lK})(4r\sqrt{\alpha lK} + \sqrt{2t}lK)} \tag{25}$$

is achieved when

$$d = \frac{r\sqrt{\alpha l K}}{4\sqrt{2t}} \tag{26}$$

(3). The case of $d > l$

We get

$$\lambda = \frac{16r}{\alpha r + 2lt} \cdot \frac{K}{Kr + 16} \tag{27}$$

In this region, again the performance is not a function of d.

Figures 16 (a),(b), and 17 (c),(d) show the performance of the system as a function of d. Here, we define three values of d for convenience.

- point A: the value d of l/q
- point B: the value d of l
- point M: the value d of $\frac{r\sqrt{\alpha l K}}{4\sqrt{2t}}$

From figure 16 (a),(b) and figure 17 (c),(d), we get the following table for (16,1) and (4,1) configurations.

figure	configuration	A	B	M	depth relationship
(a)	(16,1)	0.84	13.4	$69\sqrt{K}$	A < B < M < Cyl
(b)	(16,1)	16.0	256.0	$75.9\sqrt{K}$	A < M < B < Cyl
(c)	(16,1)	64.0	1024.0	$151.8\sqrt{K}$	A < M < Cyl < B
(d)	(4,1)	64.0	256.0	$75.9\sqrt{K}$	A < M < B < Cyl

(where Cyl = Cylinder size.)

In figure 16 (a), the throughput rises from (A) to (B) continuously because (B) $\leq$ (M). In figure 16 (b), the throughput is flat until (A), rises until (M), achieves the maximum throughput at (M), and is flat after (B). In figure 17 (c), the throughput is flat until (A), achieves the maximum throughput at (M), and decreases as d increases. In figure 17 (d), the throughput is flat until (A), achieves the maximum at (M), and is flat after (B).

In figure 17 (d), the throughput decreases until (A) and increases after (B) with high K. Notice that asymptotic approximations can be inaccurate. Figures 16 and 17 are plotted using equation (12). Using the approximation, we would expect the throughput for $d <$ (A) to be flat, but this is not the actual case due to block boundary losses.

Figure 18: Approximate Model

C. Evaluation of approximate model

On the other hand, approximate models for the behavior of depth seem to work well. Figure 18 shows our approximate model compared to an exact calculation. Equation (24) can be used to easily determine the most suitable value of depth.

D. Areas where deep depth is effective

Figure 19 (a),(b) shows the area of l and K in which the most effective depth is reached. This is based on equation (14). As a summary, we get the following tables.

In the case of a (16,1) configuration,

K	4[KB]	64[KB]	256[KB]
1	medium	medium	minimum
4	medium	medium	medium
16	maximum	maximum	medium

Figure 19: Effect of Deep Depth (a),(b)

In the case of a (4,1) configuration,

K	4[KB]	64[KB]	256[KB]
1	medium	minimum	minimum
4	medium	minimum	minimum
16	maximum	maximum	minimum

Where "minimum," "medium" and "maximum" mean that d should be chosen as smallest block size such as page size, $d = \frac{r\sqrt{\alpha l K}}{4\sqrt{2t}}$, largest block size such as cylinder size, respectively. Generally, we use the following depth guideline. The depth should be

- large file size and low sharing — smallest block size such as page size.
- large file size and high sharing — $d = \frac{r\sqrt{\alpha l K}}{4\sqrt{2t}}$.
- middle file size and low sharing — $d = \frac{r\sqrt{\alpha l K}}{4\sqrt{2t}}$.
- middle file size and high sharing — largest block size such as cylinder size.
- small file size and low sharing — $d = \frac{r\sqrt{\alpha l K}}{4\sqrt{2t}}$.

Figure 20: BEST CONFIGURATION for Workload2

- small file size and high sharing — smallest block size such as page size.

The area in which a medium size of d is the most effective is smaller for smaller striping configurations.

4.4 The effect of file distribution

Generally, the performance tradeoff of a disk array system strongly depends on the file distribution. We consider two additional types of file distribution in this section. We then develop some guidelines for the best performance.

A. Workloads

1. workload2
   - Normal distribution
   - Average of file size = l
   - Standard deviation = $l/3$
2. workload3

- Bimodal distribution
- Small file size = 32[KB]
- Large file size = 256[KB]
- Probability of small file access = p [%]
- Probability of large file access = $1 - p$ [%]

These workloads might characterize such environments as:

- Transaction processing, such as banking service, is reflected in workload2 with small average file size.
- Heavy scientific computation is reflected in workload2 with large average file size.
- Data base maintenance processing is reflected in workload3 since there are both small index and large data files.

B. Best configuration tradeoff for synchronized configurations

In order to examine the concurrency tradeoff for the synchronized configurations, we use the approximate formulas for n (or NLD) and S under the assumption of $c = 1$. By substituting $q = 1$ in equations (19) and (20) for synchronized configurations, we get the following formulas.

$$n = \frac{16}{r}$$

$$S = \alpha + \frac{2lt}{r}$$

Then,

$$\lambda = \frac{16Kr}{(\alpha r + 2lt)(Kr + 16)} \tag{28}$$

This function has its maximum at $r = \sqrt{32lt/\alpha K}$. Further, optimum r increases with file size l and decreases with the number of logical processes K. The boundaries for optimum synchronized configurations are shown in figure 20, obtained from equation (15).

C. Best configuration tradeoff for striped configurations

There are two cases of interest:

(1). $l > q \times d$

From equations (19) and (20), we get the same equation as synchronized configurations. We

Figure 21: NLD or Ts vs. File Size

expect the concurrency boundaries for striped configurations to act similarly to synchronized configurations and to follow figure 20 (where q replaces r).

(2). $l < q \times d$

For this case, the relationship of n (or NLD) and S between two striped configurations $(q_1, 1)$ and $(q_2, 1)$ is shown in figure 21. We assume $q_1 \geq q_2$.

Comparing the NLD between these two configurations, recall that the effect of large NLD on system performance is greater when K is large (section 3). Secondly, this effect is also relatively stronger in the area of large l because NLD does not change much for smaller file size. From figure 21, for small file size, NLD is independent of q but this is not true for large files. From this, we can determine regions of optimum striped configurations (figure 22).

D. Best configuration tradeoff for striped and synchronized configuration

Figure 23 shows the relationship of n(or NLD) and S between striped configurations and synchronized configurations. The synchronized configurations have lower service time S than the counterpart striped configuration (large file sizes) because of smaller S and similar NLD. In addition, the synchronized configurations are slower (larger S) with large K and small file size because NLD decreases. These two facts result in tradeoff boundaries between synchronized configurations and striped configurations, shown in figure 22.

Figure 22: BEST CONFIGURATION for Workload3

5 Best configurations

From the discussions above, we get the following guidelines for optimizing a disk array system.

A. Best concurrency for traffic

We can determine the best disk concurrency as a function of traffic rate from figure 15. For our workload, the best concurrency was

Traffic level	File size 4[KB] av.	File size 256[KB] av.
CPU bound	16	4,8,16
CPU-I/O balanced	1,2	1,2
I/O bound	1,2	1,2

B. Best configuration for one peak file distribution

Synchronized configurations are the best for one peak (unimodal) file distributions.

Figure 23: NLD or S vs. File Size

sharing level	file : 4[KB]	file : 32[KB]	file : 64[KB]	file : 128[KB]	file : 256[KB]
1	(1,2)	(1,8)	(1,8)	(1,8)	(1,16)
4	(1,1)	(1,2)	(1,4)	(1,4)	(1,8)
16	(1,1)	(1,1)	(1,1)	(1,2)	(1,2)

C. Best configuration for bimodal file distribution

Synchronized or mixed configurations are the best for bimodal file distributions.

sharing level	file : 32[KB]	file : 64[KB]	file : 128[KB]	file : 256[KB]
1	(1,8)	(2,8)	(2,8)	(1,16)
4	(1,2)	(4,2)	(4,2)	(2,4)
16	(1,1)	(4,1)	(4,1)	(2,2)

6 Comparisons with other reported results

6.1 Comparison with Reddy's result

For validating our model, we compare our results with Reddy's simulation [5]. Reddy simulates disk arrays in a CPU bound environment, that is, response time does not affect CPU request rate (traffic). For comparison, we use our open queue model and adjust our model.

Reddy assumes the following:

- The file system has 800 active files.
- Each file is 4 [MB] in size.
- Total number of physical devices is 16.
- Capacity per sector (RA81 disk) is 512 [Bytes].
- Number of tracks per cylinder is 14.
- Number of sectors per track is 52.
- Seek policy is SCAN instead of FIFO.
- Seek time per track is 0.04 [sec.].
- 4 [KB] block can be transferred in 2.6 [ms].

Thus, we make the following adjustments:

A. $Tseek$: seek movement time to desired track

Because Reddy used SCAN policy instead of FIFO policy, we use the following formula for average seek time [7].

$$Tseek = S_{min} + \frac{S_{max} - S_{min}}{L + 1}$$

where

S_{min} : seek time for minimum distance

S_{max} : seek time for maximum distance

L : mean number of requests serviced by SCAN in one sweep across the disk surface

The number of cylinders for user file can be calculated (see section 2.3 D) as:

$$N = \frac{800 \times 4 \times 1024 \times 1024}{16 \times 512 \times 14 \times 52} = 563 \text{ [cylinders]}$$

Also, we assume seek time is linear in seek distance and mean number of requests in one sweep (L) is 1.

Then, since 1 track seek time is 0.04 [sec.] in Reddy's model, we approximate Reddy's seek time as

$$Tseek = 0.04 \times \frac{563}{2}$$

B. $Txfer$: data transfer time between data buffer and CPU memory

Because Reddy does not allow concurrent data transfer for striped configurations, we approximate the data transfer time as:

$$Txfer = \frac{l}{v \times r}$$

where v is calculated from Reddy for a 4 [KB] block as:
$v = \frac{4}{2.6} = 1.5$ [MB/sec.]

C. $Tbuf$: data transfer time between disk and data buffer

Reddy's file size varies from 1 to 16 blocks (block size is 4 [KB]). Thus, we cannot neglect the effect of sector boundary losses in computing data transfer time. We approximate the disk-buffer transfer time as:

$$Tbuf = \frac{\lceil \frac{\lceil \frac{\lceil \frac{l}{d} \rceil}{simul} \rceil \times d}{0.5 \times r} \rceil \times 0.5}{v}$$

D.$Var(S)$: Variance of S

$$Var(S) = Var(Tlat) + Var(Txfer) + Var(Tbuf)$$

Table 1 shows the comparison between our calculated value and Reddy's simulated value. The upper values are from Reddy, the lower from our model.

The difference between our calculated response time (the lower row of R column) and Reddy's simulated response time (the upper row of R column) is shown below (relative to Reddy's data).

Minimum difference is 0.3 [%].

Maximum difference is -10.3 [%].

Mean difference is -2.9 [%].

Table 2 also shows the comparison between our calculated and Reddy's simulated value of Utilization rate. The upper values are from Reddy, the lower values from our model.

Table 1: Response Times

Configuration	R [ms]	S [ms]	W [ms]	$Txfer$ [ms]	$c * c$	$Utilization$	n [dev.]	$Difference$ [%]
(1,16)	32.28	20.16	11.68	0.44		0.502		
	31.78	20.25	10.97	0.55	0.0570	0.5063	1.00	-1.5
(1,8)	26.49	21.07	4.50	0.92		0.265		
	25.59	20.67	3.81	1.11	0.0592	0.2583	2.00	-3.4
(1,4)	26.28	22.28	2.22	1.78		0.139		
	25.82	21.77	1.84	2.21	0.0715	0.1361	4.00	-1.8
(1,2)	29.41	24.62	1.22	3.57		0.076		
	29.50	23.99	1.09	4.43	0.1227	0.075	8.00	0.3
(1,1)	36.66	28.53	0.83	7.29		0.045		
	38.11	28.41	0.84	8.85	0.2747	0.0444	16.00	4.0
(16,1)	29.85	22.28	1.72	5.84		0.118		
	29.98	22.16	1.56	6.25	0.0556	0.1177	4.71	4.4
(8,1)	29.70	22.95	1.35	5.40		0.095		
	29.66	22.58	1.24	5.83	0.0591	0.0946	5.97	-1.3
(4,1)	30.03	24.57	1.07	4.39		0.069		
	29.32	23.41	0.91	5.00	0.0733	0.0673	8.70	-2.4
(2,1)	32.43	26.80	1.03	4.60		0.055		
	29.17	25.08	0.76	3.33	0.1233	0.0509	12.31	-10.0
(8,2)	28.87	21.63	2.68	4.56		0.182		
	27.88	21.07	2.38	4.43	0.0543	0.1765	2.99	-3.4
(4,4)	29.06	21.33	4.35	3.39		0.246		
	26.08	20.52	3.34	2.21	0.0557	0.2360	2.17	-10.3
(2,8)	29.51	20.93	5.16	2.42		0.348		
	26.61	20.25	5.25	1.11	0.0570	0.3291	1.54	-9.8

Table 2: Utilization Rates

Average Utilization at Inter-arrival times [ms]				
Configuration	80 [ms]	40 [ms]	20 [ms]	10 [ms]
(1,16)	25.4	50.2	99.4	sat
	25.3	50.6	sat	sat
(1,8)	13.2	26.5	51.4	95.0
	12.9	25.8	51.7	sat
(1,4)	6.9	13.9	28.0	54.5
	6.8	13.6	27.2	54.4
(1,2)	3.8	7.6	15.2	29.8
	3.8	7.5	15.0	30.0
(1,1)	2.2	4.5	8.9	17.6
	2.2	4.4	8.9	17.8
(16,1)	5.9	11.8	23.7	47.2
	5.9	11.8	23.6	47.1
(8,1)	4.7	9.5	19.2	38.5
	4.7	9.5	18.9	37.8
(4,1)	3.5	6.9	14.1	27.9
	3.4	6.7	13.5	26.9
(2,1)	2.8	5.5	11.1	21.9
	2.6	5.1	10.2	20.4
(8,2)	9.1	18.2	36.7	73.2
	8.8	17.7	35.3	70.6
(4,4)	12.2	24.6	48.6	97.5
	11.8	23.6	47.2	94.4
(2,8)	17.4	34.8	66.7	sat
	16.5	32.9	65.8	sat

The difference between our calculated Utilization (the lower row) and Reddy's simulated Utilization (the upper row) is shown below (relative to Reddy's data). We regard "sat" (meaning saturated) as 100 [%] utilization.

Minimum difference is 0.0 [%].

Maximum difference is -8.1 [%].

Mean difference is -2.0 [%].

6.2 Comparison with Chen's result

Our evaluation on the effect of depth in section 4.3 shows a similar behavior to Chen's simulation [6].
However, figure 19 indicates that more than 1 block depth does not always offer the best performance. In particular,

- Deep depth is less effective for lower concurrency configurations.
- Deep depth is less effective for large file size and low multitasking environment.

Chen's simulation did not consider these configurations.

7 Conclusions

Using a queuing theoretic model we evaluated disk array configurations with various parameters and under several environments. The queuing model seems to agree with several earlier reported disk array simulations.

We define disk array concurrency as the product of the degree of disk striping (q) and the degree of disk synchronization (r). For a fixed number of disks, creating a "disk array" or high degree of concurrency does not always provide the best performance. While high degrees of concurrency ($q \times r$) are desirable for low traffic environments, the opposite is true for high traffic with large degrees of multitasking. Indeed, partitioning disks into independent groups of moderate concurrency seems optimum for either:

1. Environments with high traffic but low degrees of multitasking, or
2. Balanced CPU–I/O traffic environments.

File depth is the unit (number of blocks) of file distribution for striped disks. Increasing file depth is effective primarily for small files with high degrees of multitasking. In comparing striped versus synchronized disk configurations, synchronized configurations are preferred for uni-modal file distributions, while striped configurations are preferred for bimodal (or multimodal) file distributions.

References

[1] M. Y. Kim, "Synchronized Disk Interleaving," *IEEE Transactions on Computers*, vol. C-35, no. 11, pp. 978–988, Nov. 1986.

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

[3] R. H. Katz, G. A. Gibson, and D. A. Patterson, "Disk System Architecture for High Performance Computing," *Proc. of the IEEE*, vol. 77, no. 12, pp. 1842–1858, Dec. 1989.

[4] D. A. Patterson, G. A. Gibson, and R. H. Katz, "A Case for Redundant Arrays of Inexpensive Disks (RAID)," *UCB EECS*, no. CSD 87/391, pp. 1–18, Dec. 1987.

[5] A. L. Reddy and P. Banerjee, "An Evaluation of Multiple-Disk I/O Systems," *IEEE Transactions on Computers*, vol. 38, no. 12, pp. 1680–1690, Dec. 1989.

[6] P. M. Chen and D. A. Patterson, "Maximize Performance in a Stripped Disk Array," *Proc. Computer Architecture*, pp. 322–331, Jun. 1990.

[7] P. J. Denning, "Effects of scheduling on file memory operations," *Proc. AFIPS 1967 SJCC*, vol. 30, pp. 9–21, 1967.

[8] T. J. Teorey and T. B. Pinkerton, "A Comparative Analysis of Disk Scheduling Policies," *Comm. of the ACM*, vol. 15, no. 3, pp. 177–184, Mar. 1972.

[9] M. Y. Kim and A. N. Tantawi, "Asynchronous Disk Interleaving," *IBM Technical Report*, vol. RC-12497, pp. 1–61, Jan. 1987.

[10] D. Bitton and J. Gray, "Disk Shadowing," *Proc. of the 14th VLDB Conference*, pp. 331–338, 1988.

[11] D. Bitton, "Arm Scheduling in Shadowed Disks," *Proc. Compcon 89*, pp. 132–136, 1989.

[12] R. A. Scranton and D. A. Thompson, "The Access Time Myth," *IBM Technical Report*, vol. RC-10197, pp. 1–8, Sep. 1984.

[13] H. Kobayashi, "Modeling and Analysis," *Addison-Wesley*, pp. 144–148, 1978.

[14] S. Ng, "Some Design Issue of Disk Arrays," *Proc. Compcon 89*, pp. 137–142, 1989.