Effects of Trends in Disk Technology on Disk Arrays

Christopher L. Elford[*] Daniel A. Reed[†]

Department of Computer Science
University of Illinois
Urbana, Illinois 61801

1 Introduction

The personal computer market has created a huge demand for small, inexpensive and reasonably fast disks. The huge market base for these disks easily amortizes the development costs of these disks. Unfortunately, the relatively small market base for high performance computing offers no such amortization of development costs for high performance disks. Consequently the gap between commodity disks and "high performance" disks is shrinking. Disk arrays are one way that high performance computers can potentially capitalize on the commodity disk market.

Improvements in disk technology will help to determine if disk arrays based on commodity disks can provide sufficient performance for high performance, parallel computers. Disk rotational speeds, seek time profiles, media density, platter diameter and disk to controller interface speeds are all changing, but at different rates. We wish to predict the feasibility of disk arrays based on future commodity disks. We begin by examining the effects of changes in disk technology on several disk array architectures. Unfortunately, the space spanned by the system parameters is far too large to examine in a single simulation experiment. Instead, we will define queuing models which can be analyzed to gain insight into system performance.

The remainder of this paper is organized as follows. In §2, we briefly describe disks, trends in disk technology, and the disk array architectures which we analyze. In §3, we derive algebraic expressions for the performance of the disk array models. In §4, we discuss the effects of incremental changes in disk technology on disk array performance. Finally, in §**??**, we conclude with a summary of our results and offer directions for future research.

2 Disk Architectures and Models

We examine the performance of four different disk array architectures to ascertain which configurations provide scalable performance as disk parameters incrementally improve. Before we define the array architectures we must describe our model of disks and disk service.

2.1 Basic Disk Architecture

Variable	Name	Description
S	Seek delay	Head reaches correct cylinder
L	Rotational latency	Head reaches correct sector
T	Media transfer time	Data moves from media to track buffer
I	Disk interface delay	Data moves from track buffer to controller cache

Table 1: Disk Service Time Components

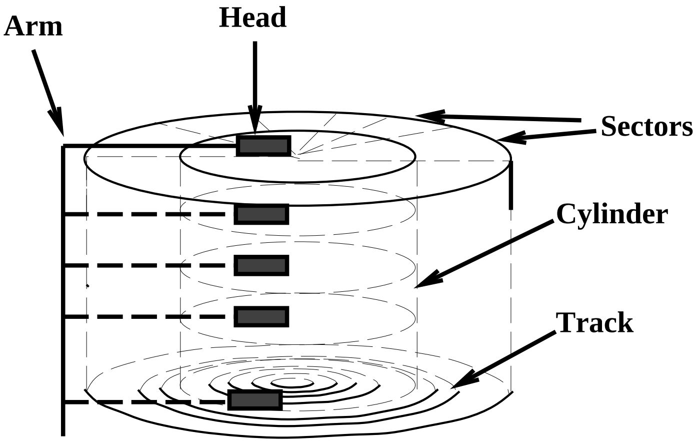


Figure 1: Components of a Typical Disk

A disk, shown in Figure 1 is connected to an input/output controller in the computer via a constant bandwidth interface. The controller either has a local memory which is used as a cache for storing data for later transfer to the requesting processor or it has access to a global memory where it can cache data. We assume a simple model of disk service that includes non-overlapping seek time, rotational latency, media transfer time and disk-controller interface delay; see Table 1.

During a seek, the disk arm, shown in Figure 1, moves the disk heads until they are above the track with the requested data. When a disk arm moves, it first accelerates and then decelerates to reach the correct track. Seek time is not a linear function of seek distance. Figure 2 shows the seek time distribution as a function of seek distance for the Wren 7 and the IBM 0661 [5].

Each track contains the same number of logical subdivisions or sectors where data is stored as shown in Figure 1. After seeking, the disk rotates until the head is positioned at the first of the requested sectors and then rotates through the request's sectors to transfer the data. Rotational latency is solely a function of rotation speed, but the media transfer time is a function of the request size, rotation speed, and the storage density.

When the data are transferred from the media, they are temporarily placed into a disk cache called a track buffer for transfer to the disk controller. After the data are transferred from the media to a track buffer, the controller interface transmits the data to the controller. A direct memory

[*] Supported in part by a Research Assistantship from the Department of Computer Science.

[†] Supported by $\cdots$

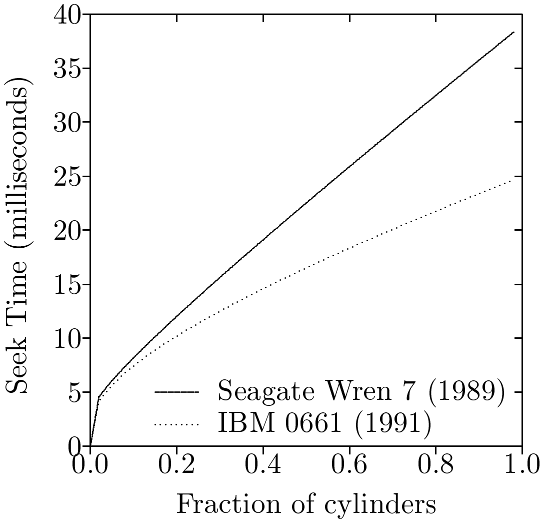


Figure 2: Typical Seek Time Profiles

access (DMA) device on the controller inserts arriving data into the cache without any significant queueing delay at the controller. Interface delay is a function of request size and interface speed.

2.2 Disk Technology

The needs of high performance computing have driven the upper end of disk technology forward. To remove the overhead due to seeks, head per track drives have been developed which electronically switch heads rather than moving to a new track. Optical disks offer vast quantities of storage at the cost of increased response times.

These options have proven very costly because development costs are amortized over a very small market. At the same time, the personal computer market has pushed the "low" end of disk technology forward at a rapid pace. Commodity disks are now smaller, cheaper, have a higher capacity, and are more reliable than ever before.

Table 2 summarizes the evolution of disk technology for several representative disk types. The IBM 3380, introduced in 1981, is the classic large expensive disk. Reduction in the distance between read/write heads and the disk surface, coupled with better disk surface coating have made possible higher data densities and closer platter spacing.

This increase in density led to reduced disk diameters for equivalent amount of storage. Smaller diameters, coupled better better control over disk head position helped to reduce head seek times. These trends made the development of drives such as the Wren 5, 6, and 7 possible.

The Fujitsu 2652 and IBM 0661 are representative of the technology currently driving the market. The Fujitsu 2652 is a full height 5.25 inch disk while the IBM 0661 is a 3.5 inch half height drive. The Fujitsu 2652 has a higher bit density and rotates and seeks faster than the IBM 0661.

2.3 Disk Array Organizations

Name	**Year**	**Speed** (RPM)	**Mean Seek Time** (ms)	**Density** (Mbits/inch^2)	**Diameter** (Inches)	**Capacity** (Megabytes)	**Interface Speed** (Mbytes/sec)
IBM 3380	1981	3600	18	2.27	14	645	3
Wren 5	1987	3600	16.5	15.1	5.25	613	5
Wren 7	1989	3600	15	25.9	5.25	1050	5
Wren 9	1991	3600	12.9	45.1	5.25	1830	10
Fujitsu M2652	1991	5400	11	32.4	5.25	1800	10
IBM 0661	1991	4320	12.6	19.4	3.5	320	10

Table 2: Disk Parameters

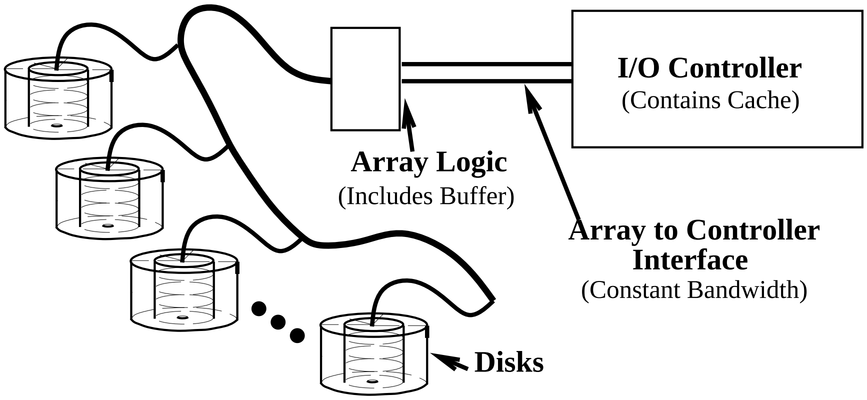


Figure 3: Basic Disk Array Configuration

High performance system developers seeking to capitalize on strides in commodity disk technology have proposed and built various forms of disk arrays [6, 7]. Groups of disks can be physically coupled via hardware or logically coupled via software. We examine four different disk array configurations with different approaches to coupling and synchronization. Figure 3 shows a conceptual view of a disk array which is not bound to any one model.

Fully Synchronous Array

In a fully synchronous array [3], disks are both seek and rotationally synchronized. This model effectively combines the D disks via hardware to form a single disk with a transfer rate D times faster than that of a single disk, but with the same seek and rotational latencies.

Partially Synchronous Array

In a partially synchronous array, the disks seek together but rotate independently. Data is striped across the same physical locations on all disks. All disks cooperate to service each request. This architecture is a software implementation of a fully synchronous array.

Fully Asynchronous Array

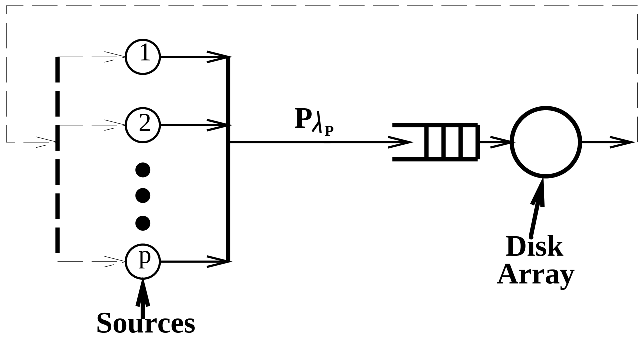


Figure 4: Queuing Model

In a fully asynchronous disk array [4], disks seek and rotate independently. Data is striped across the disks. Unlike synchronous and partially synchronous arrays, only those disks containing data participate in servicing a request, potentially improving performance for small requests.

Fully Decoupled Disks

For the previous architectures to be feasible, they must offer a response time which is at least as low as that of an architecture where each disk contains a disjoint file system. A fully decoupled disk array architecture differs from the asynchronous architecture in that data is not striped across the disks. The request arrival rate at each disk is $\frac{1}{D}$th of the arrival rate for the preceding models. Decoupled disk systems trade lower response times on single requests for inter-request parallelism and potentially higher throughput.

3 Disk Queuing Models

Having briefly described disks, disk trends, and disk arrays, we now derive a mathematical representation of the issues discussed above. For each of the four disk array architectures proposed above, we derive algebraic expressions for their performance. Our approach is to derive the mean service time and response time for each of the disk array organizations described in §2.3. We want to use these expressions to predict the performance of future disk arrays composed of varying numbers of disks with lower seek delays, higher rotational speeds and media densities, and different interface speeds.

3.1 Definitions

Table 3 summarizes the key variables for the disk array performance model. As shown in Figure 4, we assume the requests arrive at rate $P\lambda_p$ and follow a Poisson process. From this arrival rate and the disk array models, we derive expressions for service rate, utilization, queuing lengths and delays and throughput.

Table 4 defines the key components of disk system service time. We assume that seek time averages $1/\beta$ seconds, that each disk completes a full rotation every c seconds, and that each disk track contains t blocks of data. Space on each disk is allocated in units of s byte blocks. The disk

Variable	Definition
λ_p	Processor request rate (requests/second)
P	Number of processors
λ	Aggregate request rate $= P\lambda_p$
W	Mean queue delay (seconds)
N	Mean queue length
X	System throughput (MBytes/second)
R	Total request delay (seconds)

Table 3: Queuing Variables

Variable	Definition
β	Reciprocal of average seek time (seconds)
c	Time for full disk rotation (seconds)
t	Size of disk track (blocks)
l	Fixed Request size (blocks)
s	Block size (disk allocation unit) (bytes)
i	Interface delay per byte (seconds)
D	Number of disks

Table 4: Disk System Parameters

to controller interface operates at a rate of i seconds per byte. Disk arrays consist of D disks. Each request is for l blocks. For this analysis, we do not examine the effects of request size variation.

3.2 Basic Service Model

We begin our models with a mathematical description of the components of service time for a single disk. These expressions will be used as building blocks for our disk array models.

A viable statistical model for expected seek delay must reflect the common case of short but slow seeks as well as the less common case of long but faster seeks. To approximate this distribution and maintain a tractable analytic model, we approximate seek time as an exponential random variable with mean $1/\beta$ and probability density function $f_s(x|\beta)$, given by

$$f_s(x|\beta) = \begin{cases} \beta e^{-\beta x} & \text{for } x > 0, \\ 0 & \text{for } x \leq 0. \end{cases} \tag{1}$$

Integrating (1) gives an expected seek delay of

$$S \quad = \quad \int_0^\infty \beta x e^{-\beta x} dx \quad = \quad \frac{1}{\beta}, \tag{2}$$

and variance

$$Var(S) \quad = \quad \int_0^\infty \beta x^2 e^{-\beta x} dx \quad = \quad \frac{1}{\beta^2}. \tag{3}$$

Disk rotation speed is constant, and any possible rotational latency is equally likely at the time a request is issued. Therefore, rotational latency is uniformly distributed on the interval $[0, c]$ where c is the time for a full disk rotation,[1] and the probability density function is

$$f_r(x|c) = \begin{cases} \frac{1}{c} & 0 \leq x \leq c, \\ \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

The expected rotational delay is obtained by integrating (4) on the interval [0,c]:

$$L \quad = \quad \int_0^c \frac{x}{c} dx \quad = \quad \frac{c}{2}, \tag{5}$$

and the variance of the rotational delay is

$$Var(L) \quad = \quad \int_0^c \frac{x^2}{c} dx - L^2 \quad = \quad \frac{c^2}{12}. \tag{6}$$

The total disk transfer time depends on request size, rotation speed and data density. Because rotational speed and data density are fixed by the disk architecture, and request size does not vary in our analyses, disk transfer time is constant. Finally, array-controller interface delay depends on request size and interface speed. Because interface speed is fixed by the disk architecture and request size does not vary, interface delay is constant.

3.3 Fully Synchronous

We begin with the simplest case: all disks rotationally and seek synchronized. For this model, the heads on all D disks are positioned identically to effectively provide a virtual disk with D times the transfer speed of a single disk. Similarly, the disks rotate in unison, and the same sector passes under the heads of all disks simultaneously.

Seek requests are sent to all disks concurrently, making the seek delay equivalent to the seek delay of a single disk. The rotational latency of the group of disks is the same as the rotational latency of a single disk. Each disk contains only $\frac{1}{D}$th of the request data, reducing the media transfer time by a factor of D.

The seek delay of a synchronous array is the same as the seek delay of a single disk. The expected seek delay and its variance are given in Equations (2) and (3). Likewise, the expected rotational latency and its variance are given in Equations (5) and (6).

Data transfer time is the fraction of a track transferred multiplied by the disk rotation time. For a request of size l blocks, the data transfer time is

$$T \quad = \quad \frac{c}{t} \lceil \frac{l}{D} \rceil,$$

where t is the size of a track, and c is the time for a disk rotation. Because we assume all requests are of fixed size, the transfer time variance is zero.

After media transfer is complete, data are transferred to the controller cache. We assume that the interface delay per byte is a constant i, yielding a total interface delay for a request of size ls bytes of

$$I \quad = \quad lis.$$

[1] Note that this presumes rotational position sensing.

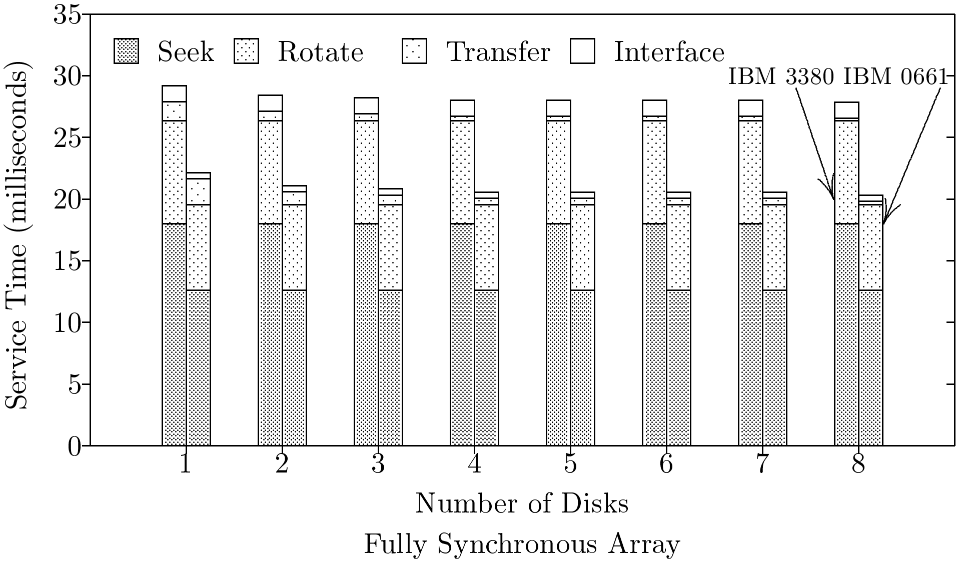


Figure 5: Service Time Components (4K Byte Requests 512 Byte Blocks)

Combining the mean seek, rotational latency, transfer time, and interface delay, the expected time B to service a request when there is no resource contention is

$$\begin{aligned} B &= S+L+T+I \\ &= \frac{1}{\beta}+\frac{c}{2}+\frac{c}{t}\lceil\frac{l}{D}\rceil+lis. \end{aligned} \tag{7}$$

Figures 5 and 6 show how the synchronous disk array service time varies with the number of disks and request size for two different disks, the IBM 3380 and the IBM 0661. For small requests, seek and rotational latency dominates, and multiple disks provide little benefit. However, for larger requests, the D-fold increase in transfer rate makes multiple disks attractive.

As disk diameters shrink, seeks tend to be faster, and rotational speeds tend to increase. Smaller disks are not just tiny replicas of their larger counterparts. Insufficient improvements in media density when compared to changes in disk diameter dictate that tracks will have a lower capacity than their larger counterparts. As track capacity decreases, transfer time increases. Increased rotational speeds reduce rotational latency and help to offset the increase in transfer time due to reduced track size. The relative magnitude of seek time, rotational speed and data density determine the contribution of transfer time to total service time. This in turn dictates the feasibility of this synchronous disk array model for new generations of disks.

In addition to analyzing the mean request service time, one must also consider queuing delays in order to obtain a true estimate of system performance. Queuing delays depend on service rate and request arrival rate. For M/G/1 queuing analysis, we need expressions for average service time and the second moment of the service time.

In this model, disks operate synchronously at all times. The components of service time do not overlap. Because the components of service are independent, and the service time variance is

$$Var(B) \quad = \quad Var(S)+Var(L)+Var(T)+Var(I)$$

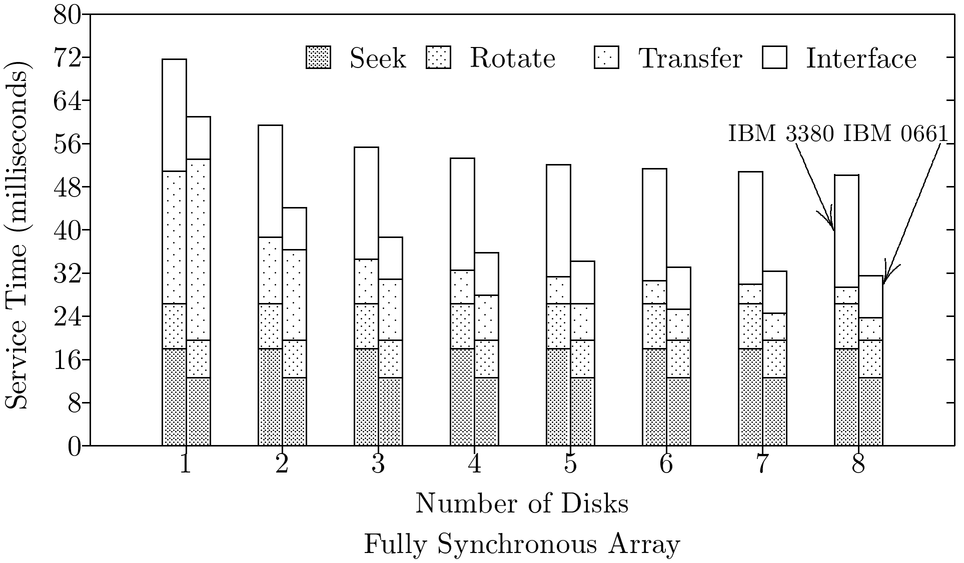


Figure 6: Service Time Components (64K Byte Requests 512 Byte Blocks)

Disk Parameters	$\lambda = 2/sec$				$\lambda = 4/sec$				$\lambda = 8/sec$				$\lambda = 10/sec$			
IBM 0661	2	4	6	8	2	4	6	8	2	4	6	8	2	4	6	8
Halved Average Seek	2	2	3	3	2	2	3	3	2	2	3	3	2	2	3	3
Double Rotation Speed	1	2	2	3	1	2	2	3	1	2	2	3	1	2	2	3
Halved Seek/Rotate Times	1	1	2	2	1	1	2	2	1	1	2	2	1	1	2	2

Table 5: Disk Counts for Equivalent Array Performance (64 K Byte Requests)

$$= \frac{1}{\beta^2} + \frac{c^2}{12}.$$

The second moment of the service time is

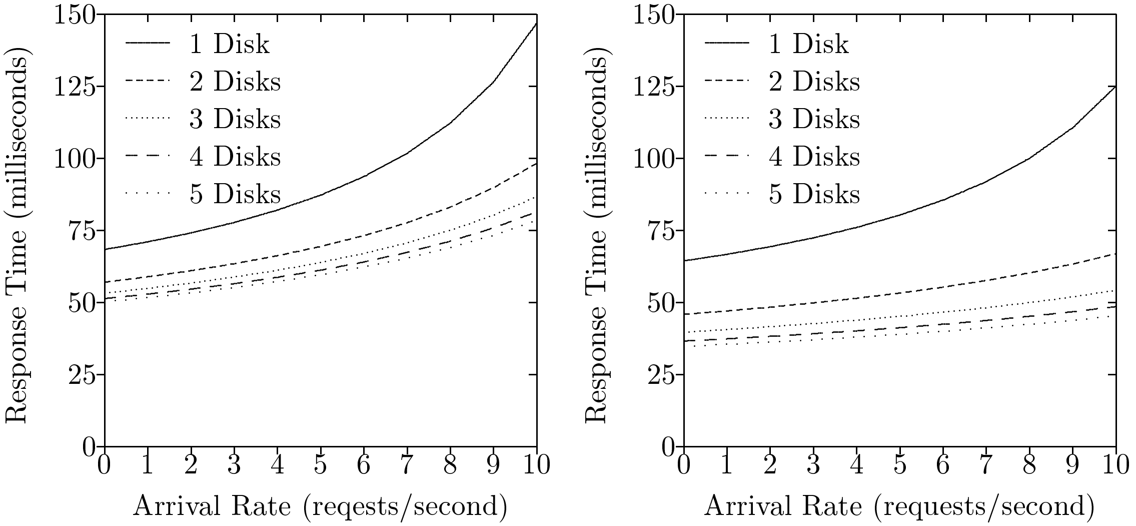
$$\begin{aligned} Sec(B) &= Var(B) + B^2 \\ &= \frac{1}{\beta^2} + \frac{c^2}{12} + \left(\frac{1}{\beta} + \frac{c}{2} + \frac{c}{t}\lceil\frac{l}{D}\rceil + lis\right)^2. \end{aligned} \quad (8)$$

Finally, the average response time is simply the sum of queuing delay and service time

$$R = W + B = \frac{\lambda Sec(B)}{2(1 - \lambda B)} + B. \quad (9)$$

Figure 7 shows how response time varies with request arrival rate and the number of disks for the IBM 3380 and IBM 0661. The rate of increase in response time as request rate rises is less for the IBM 0661 than the IBM 3380.

By setting (9) for a disk array with one set of disk parameters equal to that of an array with another set of disk parameters, one can solve for how many disks of the new disk type it takes to



a) IBM 3380

b) IBM 0661

Figure 7: Response Time for Fully Synchronous Array (64K Byte Requests 512 Byte Blocks)

have a synchronous disk array with performance equal to the original array's performance. Table 5 shows exactly how some "simple" improvements to the IBM 0661 result in smaller synchronous disk arrays with the same performance.

3.4 Partially Synchronous

We next consider a partially synchronous disk array architecture. Data is block striped across the disks. In order to ensure that all disk heads are always, on identical tracks, all disks participate in servicing all requests regardless of request size. Because there is no physical synchronization between the disks in the array, disk rotations are independent and the time to satisfy a request is the maximum across the disks.

When a request is serviced, all disks seek in unison to the same physical track, and the mean seek time and variance is identical to that of a fully synchronous array.

Rotational latency on each disk is uniformly distributed on the range $[0, c]$. Because disks rotate independently, the rotational latency of the disk array is the maximum of the rotational latencies of the individual disks.

The probability density function of the maximum of n i.i.d. random variables with probability density function $f(x)$ and distribution function $F(x)$ [1] is

$$f_{max_n}(x) = n[F(x)]^{n-1}f(y). \tag{10}$$

The probability density function of the uniform distribution is given in (4),

$$F(x) = \int \frac{1}{c}dx = \frac{x}{c}.$$

Substituting in (10), we obtain the probability density function of the maximum of D i.i.d. uniform random variables:

$$f_{max_D}(x) = \frac{D}{c}\left(\frac{x}{c}\right)^{n-1}. \tag{11}$$

The expected rotational latency for a D disk request is obtained by integrating (11) on the range $[0, c]$

$$\begin{aligned} L &= \int_0^c \frac{Dx}{c}\left(\frac{x}{c}\right)^{n-1} dx \\ &= \frac{cD}{D+1}, \end{aligned}$$

and the variance is

$$\begin{aligned} Var(L) &= \int_0^c \frac{Dx^2}{c}\left(\frac{x}{c}\right)^{n-1} dx - L^2 \\ &= \frac{c^2 D}{D+2} - \left(\frac{cD}{D+1}\right)^2. \end{aligned}$$

When the disk with the maximal rotational latency finishes, it must perform its data transfer. Because all disks transfer the same amount of data, when this last disk finishes its transfer, all disks have completed. Therefore, the data transfer time is

$$T = \frac{c}{t}\lceil\frac{l}{D}\rceil.$$

After media transfer is complete, data is transferred from the buffer to the controller. This delay is identical to the synchronous case above.

Combining the mean seek, rotational latency, transfer time, and interface delay, the expected time B to service a request when there is no resource contention is

$$\begin{aligned} B &= S + L + T + I \\ &= \frac{1}{\beta} + \frac{cD}{D+1} + \frac{c}{t}\lceil\frac{l}{D}\rceil + lis. \end{aligned} \tag{12}$$

Figures 8 and 9 show how the components of service time vary with the number of disks and request size for two different disks. In Figure 8, transfer time is a small fraction of service time. The benefits gained from reduced transfer time is masked by increased rotational latency. More disks leads to worse response time. In Figure 9, transfer time dominates service time for few disks. The decrease in transfer time as D increases offsets the increase in rotational latency and leads to a performance improvement.

Because components of service are independent, the variance of service time is given by

$$\begin{aligned} Var(B) &= Var(S) + Var(L) \\ &= \frac{1}{\beta^2} + \frac{c^2 D}{D+2} - \left(\frac{cD}{D+1}\right)^2. \end{aligned}$$

The second moment of service time is

$$\begin{aligned} Sec(B) &= Var(B) + B^2 \\ &= \frac{1}{\beta^2} + \frac{c^2 D}{D+2} - \left(\frac{cD}{D+1}\right)^2 + \left(\frac{1}{\beta} + \frac{cD}{D+1} + \frac{c}{t}\lceil\frac{l}{D}\rceil + lis\right)^2. \end{aligned} \tag{13}$$

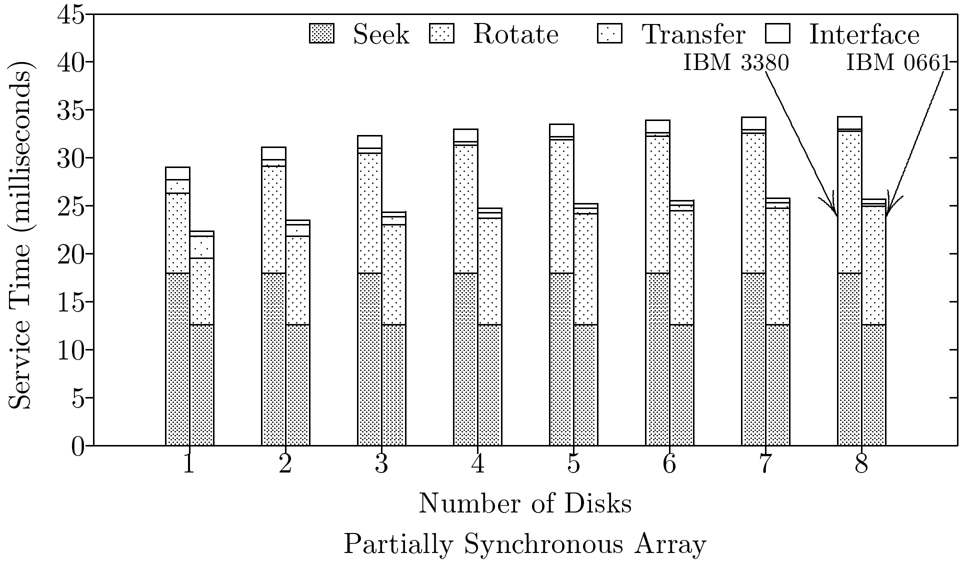


Figure 8: Service Time Components (4K Byte Requests 512 Byte Blocks)

Substituting (12) and (13) into (9) gives response time for a partially synchronous array.

Figure 10 shows how response time varies with request arrival rate and the number of disks in the array for the IBM 3380 and the IBM 0661. Transfer time dominates service time more on the IBM 0661 than on the IBM 3380 due to the latter's slower rotational speed and larger track capacity. Therefore, the IBM 0661 disk array is able to accommodate a higher arrival rate before response time rises excessively.

By comparing Figures 7 and 10, we see that, for this workload, partially synchronous disk arrays based on the IBM 3380 degrade faster than fully synchronous disk arrays. This is due to the increased rotational latency in the partially synchronous array. This degradation is less noticeable in disk arrays based on the IBM 0661.

3.5 Fully Asynchronous Array

We now consider an asynchronous disk array architecture. Like the previous disk array architectures, data is striped across the disks in one block units. With this model, one does not assume that all disks start and end each transaction on identical tracks. In fact, some disks may not even participate in servicing some requests. In order to maintain an solvable analytic model, however, we assume the disk array only services a single request at a time even if the disks for two requests do not overlap.

The disks seek and rotate independently. When a request is serviced, disks with request data are sent transactions. When all disk transactions are complete, the array forwards the data to the controller.

For a request of size l blocks, the number of disks n involved in servicing the request is

$$n = min\,(D, l)\,.$$

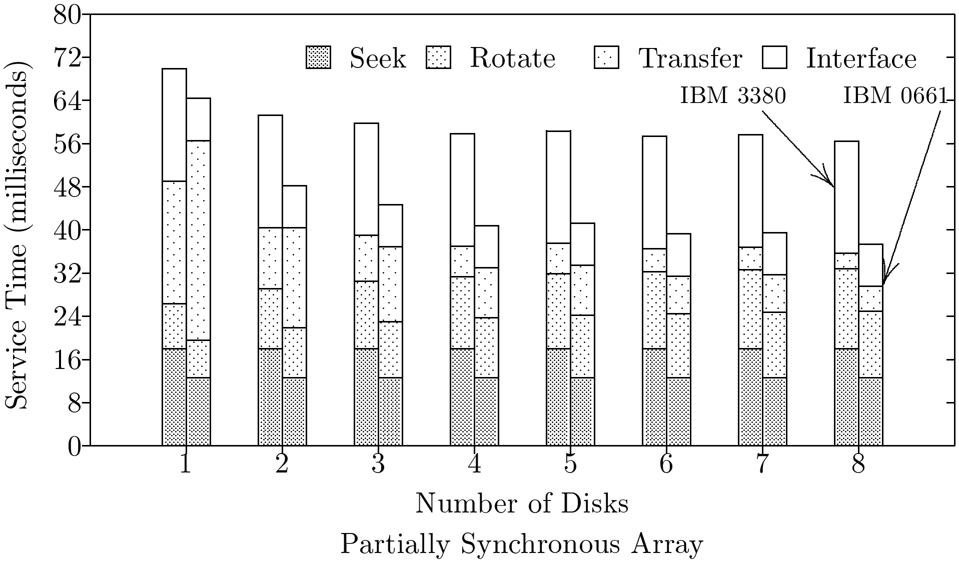


Figure 9: Service Time Components (64K Byte Requests 512 Byte Blocks)

Each of the n disks seek according to (1) and rotate according to (4). The expected seek time is the maximum of the n independent seek delays and the expected rotation time is the maximum of the n independent rotational latencies.

Simply adding the expected seek delay to the expected rotational delay would imply that all disks finish seeking before any rotational latencies begin. Because this is not the case, we find the distribution of seek and rotation together. The probability density function of seek with latency is obtained by convolving (1) and (4) [3] as

$$\begin{aligned} f_{s+r}(z|\beta,c) &= \int_0^z f_s(x|\beta) f_r(z-x|c)dx \\ &= \begin{cases} \frac{1}{c}\left(1-e^{-\beta z}\right) & \text{for } 0 \le z \le c, \\ \frac{e^{-\beta z}}{c}\left(e^{\beta c}-1\right) & \text{for } z > c. \end{cases} \end{aligned} \tag{14}$$

The associated distribution function is obtained by integrating (14) [3]:

$$\begin{aligned} F_{s+r}(t|\beta,c) &= \int_0^t f_{s+r}(z|\beta,c)dz \\ &= \begin{cases} \frac{t}{c}\frac{1}{\beta c}\left(1-e^{-\beta t}\right) & \text{for } 0 \le t \le c, \\ 1-\frac{e^{-\beta t}}{\beta c}\left(e^{\beta c}-1\right) & \text{for } t > c. \end{cases} \end{aligned} \tag{15}$$

Substituting (14) and (15) into (10) gives the probability function of the maximum seek and rotational latency:

$$f_{max_n}(x) = nF_{s+r}^{n-1}(x) f_{s+r}(x). \tag{16}$$

Integrating (16) gives an expected seek and rotational latency of

$$S+L = \int_0^\infty xnF_{s+r}^{n-1}(x) f_{s+r}(x)dx,$$

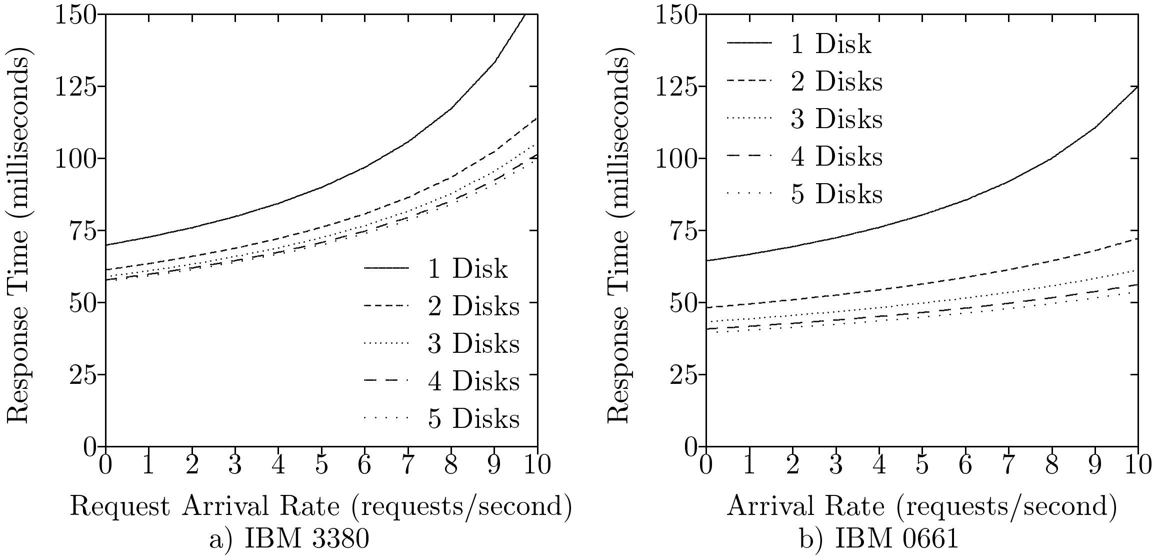


Figure 10: Response Time for Partially Synchronous Array (64K Byte Requests 512 Byte Blocks)

and variance

$$Var(S+L) = \int_0^\infty x^2 n F_{s+r}^{n-1}(x) f_{s+r}(x) dx - (S+L)^2 .$$

Because obtaining a closed form for these integrals is not straightforward, we use numerical approximations for the quantity $S+L$ and its variance.

When the disk with the maximal seek and rotational latency finishes, it must perform its data transfer. Because all disks involved in the transaction transfer the same amount of data, when this last disk finishes its transfer, all disks have finished. Therefore, the data transfer time is

$$T = \frac{c}{t} \lceil \frac{l}{n} \rceil .$$

Interface delay is identical to the cases above.

Combining the expected seek and rotational latency, transfer time, and interface delay, the expected time B to service a request when there is no resource contention is

$$\begin{aligned} B &= (S+L) + T + I \\ &= \int_0^\infty x n F_{s+r}^{n-1}(x) f_{s+r}(x) dx + \frac{c}{t} \lceil \frac{l}{n} \rceil + lis. \end{aligned} \tag{17}$$

Figures 11 and 12 show how the components of service time vary with the number of disks and request size for two different disks. Note that, unlike the other models, seek and rotational latencies are combined. Like the partially synchronous disk array model shown in Figures 8 and 9, for small requests, the increase in overhead as the number of disks increase offsets the benefits of reduced transfer time. Unlike the partially synchronous array, however, service time does not necessarily improve for larger request sizes as the number of disks increases.

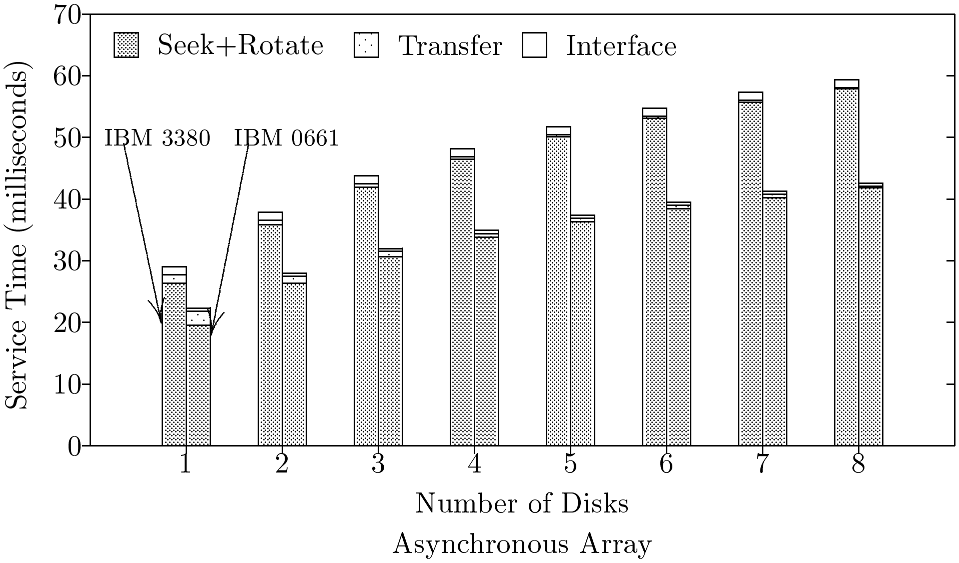


Figure 11: Service Time Components (4K Byte Requests 512 Byte Blocks)

In Figure 12, large request performance improves as the number of disks increases until the increasing overhead of seek and rotational latency is greater than the reduction in transfer time due to parallelism. For this workload, asynchronous IBM 3380 disk arrays suffer from increased service time for arrays with more than two disks while IBM 0661 disk arrays have improved service time for arrays containing up to four disks.

If the block size is increased, small requests will be distributed over fewer disks. In particular, any request smaller than the block size will access only one disk regardless of the number of disks in the array. The performance degradation evident in Figure 11 would be eliminated with a block size s of at least four kilobytes. Large requests would still benefit from reduced transfer delays because they would still span all of the disks in the array.

Because components of service are independent, the variance of service time is given by

$$\begin{aligned} Var(B) &= Var(S+L) \\ &= \int_0^\infty x^2 n F_{s+r}^{n-1}(x) f_{s+r}(x) dx - (S+L)^2 . \end{aligned}$$

The second moment of service time is

$$\begin{aligned} Sec(B) &= Var(B) + B^2 \\ &= \int_0^\infty x^2 n F_{s+r}^{n-1}(x) f_{s+r}(x) dx - (S+L)^2 + \left((S+L) + \frac{c}{t} \lceil \frac{l}{n} \rceil + lis \right)^2 . \end{aligned} \tag{18}$$

Substituting (17) and (18) into (9) gives response time for an asynchronous array.

Figure 13 shows how response time varies with request arrival rate and the number of disks in the array for the IBM 3380 and the IBM 0661. By comparing Figures 7 and 10 to Figure 13, one sees that for this workload, asynchronous disk arrays with more disks can suffer from increased response time. Figure 13a shows that for this workload, asynchronous disk arrays consisting of more

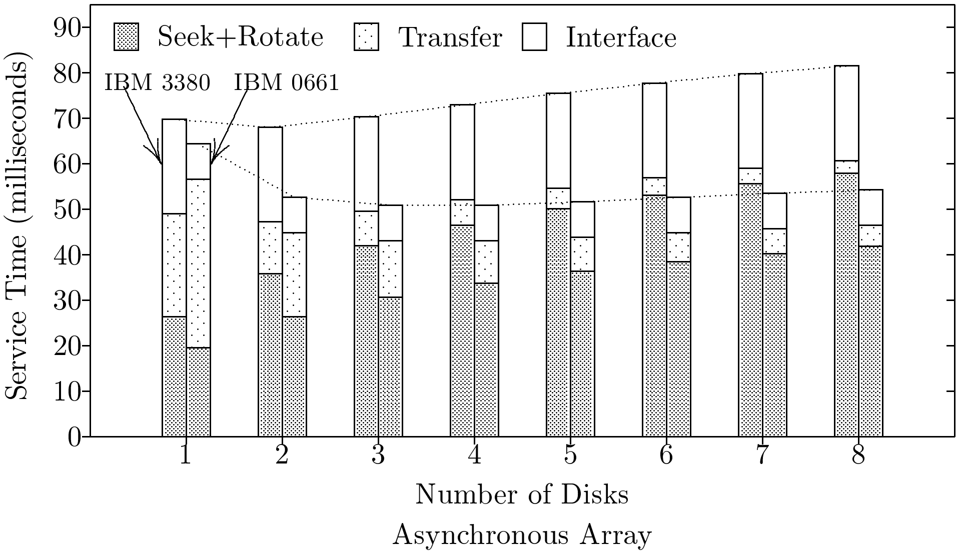


Figure 12: Service Time Components (64K Byte Requests 512 Byte Blocks)

than two IBM 3380 disks actually have worse response time than a single IBM 3380 disk working alone. Unlike Figures 7b and 10b, Figure 13b shows no reduction in service time for asynchronous IBM 0661 disk arrays consisting of more than four disks.

3.6 Fully Decoupled Disks

Our final array model does not split individual transactions among the disks. Instead, it assumes that the request stream is divided evenly among the disks so that each disk services a request stream with with arrival rate λ/D.

This model differs from the other models in that data is not striped across the disks. Because each request is serviced by a single disk, the seek and rotational latencies are those of a single disk. In addition, transfer time is not amortized by the number of disks.

Because each disk acts on an independent request stream, disks do not wait for the other disks to finish between requests. Seek and rotational delays are individual disk delays. As described in §3.2, we assume that seek time is exponentially distributed with parameter β. This gives us the expected seek delay shown in (2) with variance (3). Rotational latency is uniformly distributed on the range $[0,c]$. This yields the expected rotational latency shown in (5) with variance (6).

As in the other models, data transfer time is the fraction of a track transferred times the disk rotation time. Because requests are not divided across the disks, a request of size l blocks has a data transfer time of

$$T = \frac{cl}{t}.$$

After media transfer is complete, data is transferred from the buffer to the controller. We assume that interface delay is of unit cost i, yielding a total interface delay for a request of size l

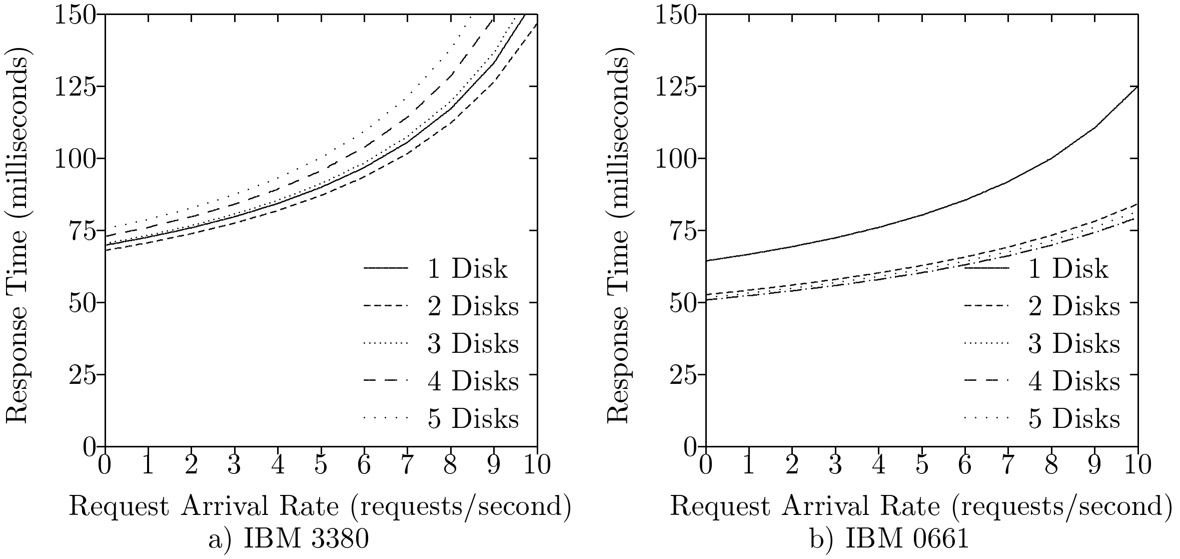


Figure 13: Response Time for Fully Asynchronous Array (64K Requests)

blocks of

$$I = lis.$$

Based on these assumptions on components of service time, the expected base service time is

$$\begin{aligned} B &= S + L + T + I \\ &= \frac{1}{\beta} + \frac{c}{2} + \frac{cl}{t} + lis. \end{aligned} \tag{19}$$

Figure 14 shows how the components of service time vary with request size for two different disks. This figure differs from figures 5, 6, 8, 9, 11 and 12 because the number of disks is irrelevant to service time distribution. With disk arrays consisting of eight IBM 0661 disks, service time for 64 kilobyte requests takes about twice as long with this model as with the other disk array architectures. On the IBM 3380, about the same amount of time is spent on media transfer as is spent on controller-interface delay.[2] Conversely, the IBM 0661 becomes strictly media transfer bound as request size grows due to its faster SCSI interface.

Because components of service are independent, the variance of service time is given by

$$\begin{aligned} Var(B) &= Var(S) + Var(L) \\ &= \frac{1}{\beta^2} + \frac{c^2}{12}. \end{aligned}$$

The second moment of service time is

$$\begin{aligned} Sec(B) &= Var(B) + B^2 \\ &= \frac{1}{\beta^2} + \frac{c^2}{12} + \left(\frac{1}{\beta} + \frac{c}{2} + \frac{cl}{t} + lis\right)^2. \end{aligned} \tag{20}$$

[2] The IBM 3380 has a 3 megabyte per second external interface.

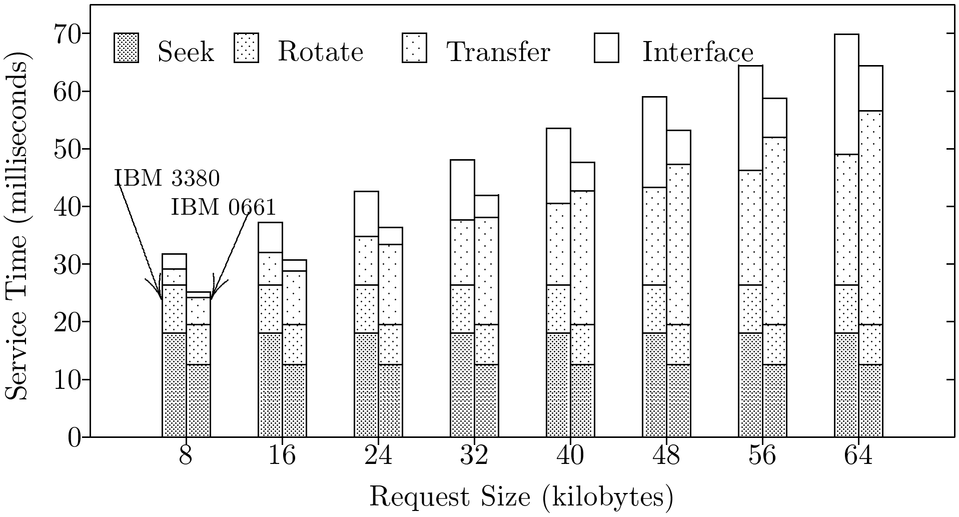


Figure 14: Service Time Components for Decoupled Disks

Substituting (19) and (20) into (9) gives response time for a partially synchronous array.

Figure 15 shows how response time varies with request arrival rate and the number of disks for the IBM 3380 and the IBM 0661. Unlike the other models, for low request arrival rates, there is little difference among the response times for varying numbers of disks. For these arrival rates, the other disk array models outperform this configuration.

By comparing Figure 15 to Figures 7, 10, and 13, one sees that as the request arrival rate increases for this workload, response time for decoupled disks rises more slowly than with either synchronous arrays or partially synchronous arrays. The region where other disk models have superior response time is smaller for IBM 3380 disks than for IBM 0661 disks.

4 Evaluation

Having derived the response times for the different disk array types, we can now compare them to each other. By setting the response time of two array types equal to each another, we can solve for how many disks of one array type are necessary to have a response time equivalent to that of a disk array of another type. It is instructive to examine how many disks a synchronous, partially synchronous, or asynchronous disk array need to equal the performance of some fixed size decoupled disk system.

4.1 Comparison of Disk Arrays Based on Current Disks

As described in §3.3, Equation (9) can be set equal to itself for different disk parameters and disk array types in order to determine the "equivalence" of different disk arrays. For example, suppose one sets the response time of a D disk decoupled IBM 0661 disk array equal to the response time of a n disk synchronous IBM 0661 disk array. One can solve for the number of disks, n, necessary to match the decoupled response time performance with a synchronous disk array.

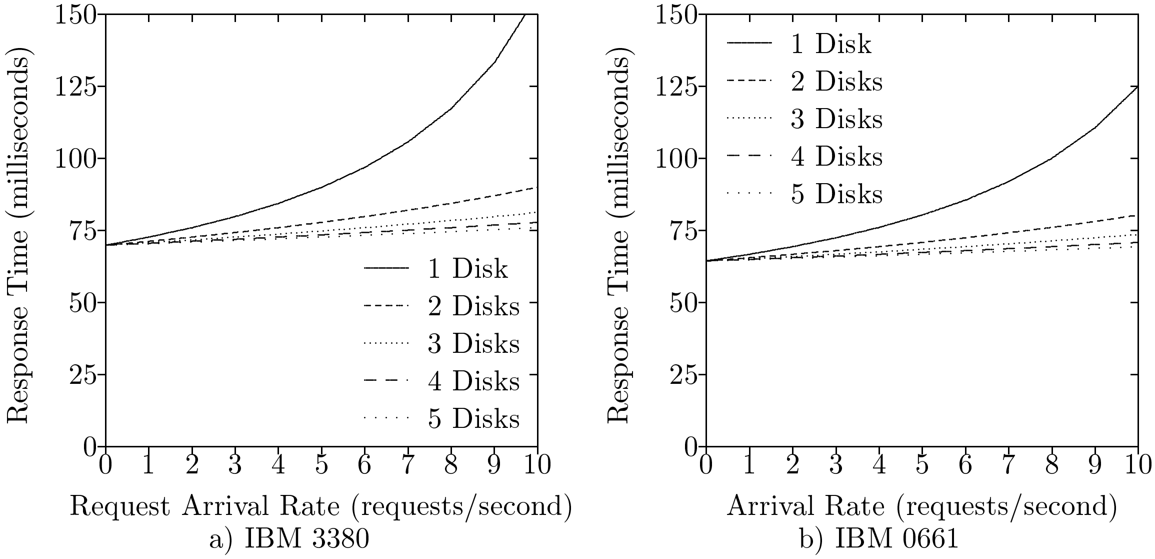


Figure 15: Response Time for Decoupled Disks (64K Byte Requests)

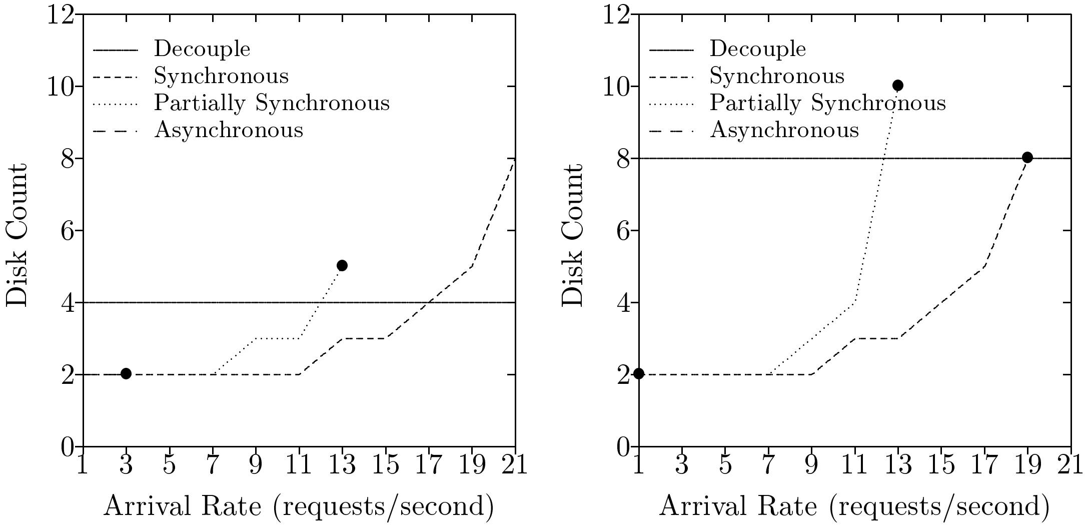
Figure 16 shows how many Fujitsu 2652 disks must be in a disk array to equal the response time performance of a decoupled disk system consisting of four and eight disks. The horizontal axis represents the rate at which requests arrive at the array and the vertical axis represents the disk count. When the arrival rate prohibits a disk array type from matching the performance of a decoupled disk system, a bullet is shown to indicate this maximum arrival rate.

For this request size and disk type, asynchronous disk arrays are unable to match the response time performance of decoupled disks for arrival rates larger than three requests per second. This is due to the increase in seek and rotational latencies which exceeds any decrease in transfer time. Partially synchronous disk arrays are able to achieve better response time than decoupled disk systems with fewer disks for arrival rates less than 13 requests per second. At this point, the increase in queuing delay on the partially synchronous array makes it impossible to compete with the 'reduced' arrival rate of the decoupled disk system. Synchronous disk arrays fare better, achieving better response time for arrival rates up to about 17 requests per second.

Similarly, Figure 17 shows how many IBM 0661 disks are needed to equal the performance of a decoupled disk system consisting of four or eight disks. The lower capacity tracks on the IBM 0661[3] coupled with its slower rotation speed[4] dictate that transfer time is more dominant in the IBM drive. This improves the effects of disk arrays which capitalize on reduction in transfer time. The most significant difference between Figures 16 and 17 is that, in Figure 17, asynchronous disk arrays are able to beat the performance of decoupled disk systems with fewer disks for arrival rates up to seven requests per second.

[3] The IBM 0661 has 24K Byte tracks while the Fujitsu 2652 has 44K Byte tracks.

[4] IBM 0661 disks rotate 72 times per second while Fujitsu 2652 disks rotate 90 times per second.



a) Decouple: Four Disks

b) Decouple: Eight Disks

Figure 16: Response Time Equivalences for Fujitsu 2652 Disk Arrays. (64K Byte Requests, 512 Byte Blocks)

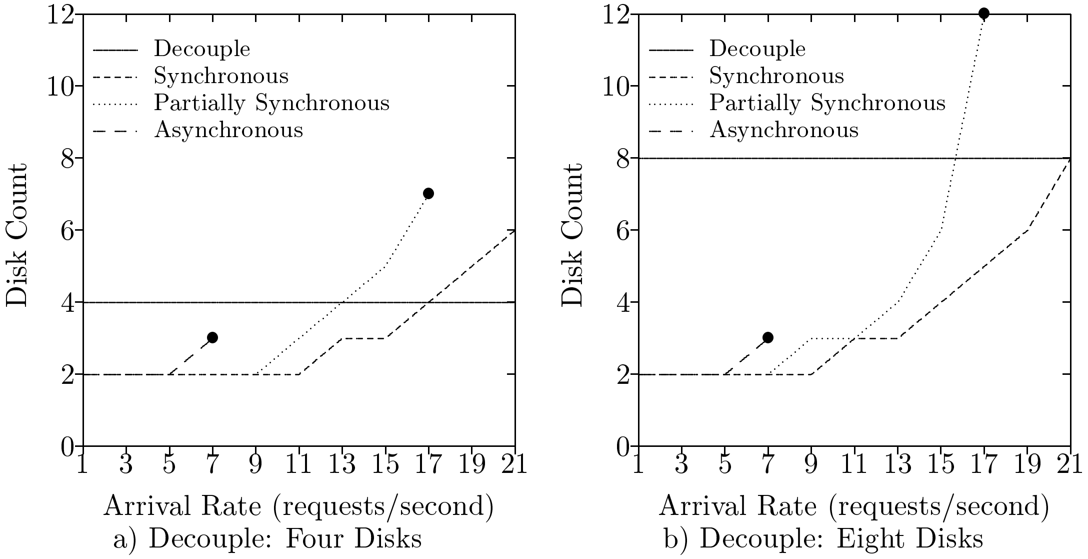
4.2 Increased Rotational Speeds

As new disks are developed, improvements in components and packaging have made possible increases in disk rotation speeds. This improvement reduces rotational latency which tends to increase the fraction of service time attributable to transfer. At the same time, however, transfer time is decreased. Figures 18 and 19 show the effects of doubling the rotational speeds the Fujitsu 2652 and the IBM 0661 respectively. While it is unlikely that rotational speeds will double within a single disk generation, one can easily see the asymptotic effects produced by improvements in rotational speeds.

By comparing Figures 16 and 18, one can see the effects that an increases in rotation speed has on disk arrays composed of Fujitsu 2652 disks. The net effect is that the range where disk arrays compete with a decoupled disk system is slightly reduced under higher rotational speeds. The high seek and rotational latency of asynchronous disk arrays is never amortized by reduced transfer time. Partially synchronous disk arrays are unable to achieve a better response time than a decoupled disk system for request arrival rates greater than eleven per second. Synchronous disk arrays are competitive until the arrival rate time exceeds about 17 requests per second.

Similarly, by comparing Figures 17 and 19, one sees how increased rotation speed affects the performance of IBM 0661 disk arrays. The range where asynchronous disk arrays can match the performance of a decoupled disk system is reduced to three requests per second. Partially synchronous and synchronous disk arrays are also slightly impaired.

One can see by comparing Figure 16 to 18 and Figure 17 to 19, that IBM 0661 disks are less affected by the increase in rotational speed than the Fujitsu 2652 disks. For a five disk array consisting of IBM 0661 disks, each disk transfers 13 kilobytes of data for a 64 kilobyte request, or about half of a track. Transfer time is roughly equivalent to rotational latency. The reduction in transfer time due to increased rotation speed approximately equals the reduction in rotational



a) Decouple: Four Disks

b) Decouple: Eight Disks

Figure 17: Response Time Equivalences for IBM 0661 Disk Array Types. (64K Byte Requests, 512 Byte Blocks)

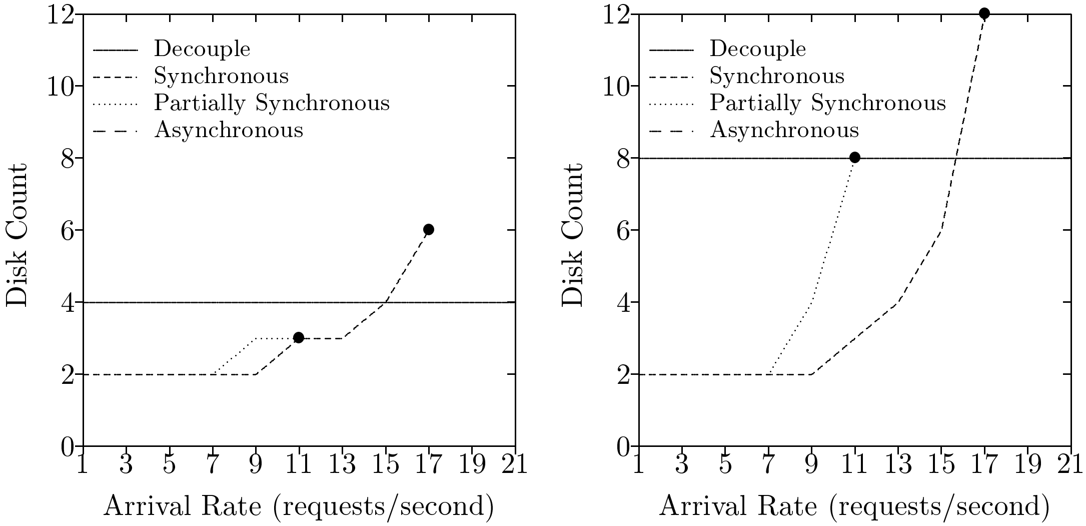
latency. This equalization of reduction helps make the IBM 0661 disk arrays less susceptible to changes in rotational speed than Fujitsu 2652 disk arrays.

4.3 Reduced Seek Delays

In addition to improving rotational speeds, considerable effort is placed into reducing average seek time in new disks. Seek times are almost half of what they were when the IBM 3380 was released; see Table 2. Unlike increases in rotation speeds, reduction in seek time strictly increases the fraction of service time attributable to data transfer. This in turn improves the performance of disk arrays which reduce transfer time. Figure 20 shows the effects of halving the seek time of the Fujitsu 2652 disk arrays. These effects are similar to the effects of a reduction in seek time to the IBM 0661 disk arrays.

By comparing Figures 16 and 20, one sees how much a reduction in seek time on the Fujitsu 2652 affects the fraction of service time attributable to transfer time. Asynchronous disk arrays are able to offer response times superior to a four disk decoupled system with only two disks for arrival rates up to nine requests per second.

Asynchronous disk arrays can compete with eight disk decoupled systems for arrival rates up to 15 requests per second. Partially synchronous disk arrays are competitive with decoupled systems for arrival rates up to about 17 requests per second. Synchronous disk arrays have better response time than decoupled disk systems with fewer disks even when the request arrival rate is as high as 21 requests per second.



a) Decouple: Four Disks
b) Decouple: Eight Disks

Figure 18: Response Time Equivalences for Fujitsu 2652 Disk Arrays With Doubled Rotation Speeds. (64K Byte Requests, 512 Byte Blocks)

4.4 Increased Track Density

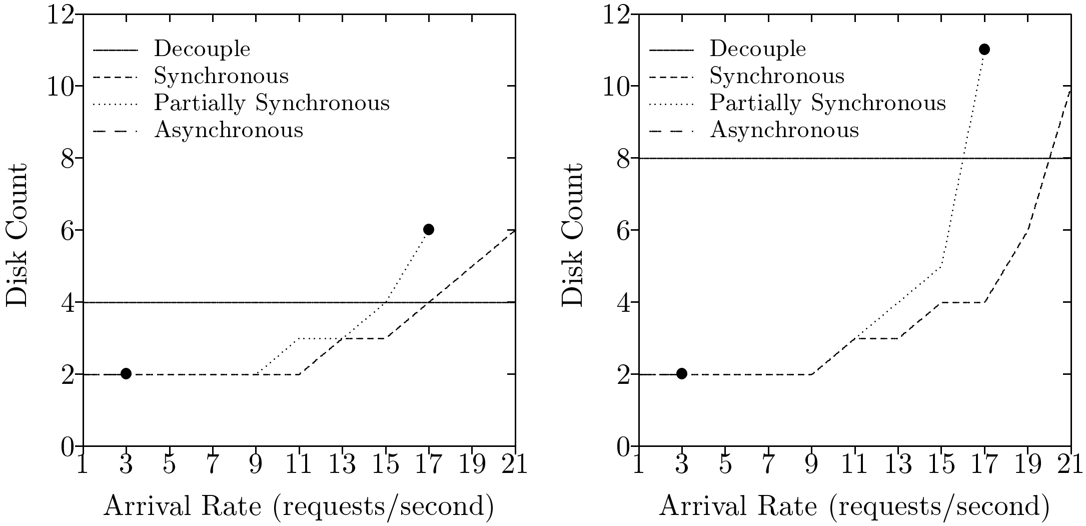
Since the IBM 3380 was released in 1981, data storage densities have doubled several times. This trend is likely to continue in the future. Increases in track capacity reduce the amount of service time attributable to transfer delay. Figure 21 summarizes the effects of doubling the track capacity of the the Fujitsu 2652 disk arrays. These effects are similar to those of the IBM 0661 arrays.

By comparing Figures 16 and 21, one sees how much an increase in track capacity affects the performance of disk arrays. Doubling the track capacity halves the transfer time, decreasing the benefit of the transfer time reduction offered by synchronous, partially synchronous, and asynchronous disk arrays. Partially synchronous disk arrays can only meet the response time of decoupled disk arrays for request rates less than five requests per second. Queues start to grow, preventing low response times, for synchronous disk arrays when arrival rate exceeds 13 requests per second.

4.5 Combined Disk Changes

Having examined the effects of changes in rotation speed, seek time, and data density, we now look at the how disk arrays perform when all of these disk improvements are made at the same time. Figures 22 and 23 show the effects of these improvements on disk arrays based on the Fujitsu 2652 and the IBM 0661 respectively.

By comparing Figure 16 to 22, one sees that partially synchronous disk arrays based on the 'improved' Fujitsu 2652 disks are less able to compete with decoupled disk arrays than arrays based on the original Fujitsu disks. The increase in disk array performance due to reduced seek time is smaller than the negative disk array effects produced by increased rotation speeds and higher track capacity. Synchronous disk arrays are less affected and are able to maintain their advantages over



a) Decouple: Four Disks

b) Decouple: Eight Disks

Figure 19: Response Time Equivalences for IBM 0661 Disk Arrays With Doubled Rotation Speeds. (64K Byte Requests, 512 Byte Blocks)

decoupled disk arrays for request arrival rates less that about 17 requests per second.

By comparing Figure 17 to 23, one sees how the disk array performance of IBM 0661 disks changes with increased rotation speed, reduced seek time, and higher track density. IBM 0661 disk arrays were less affected than Fujitsu 2652 disk arrays by increases in rotation speed; see §4.2. Therefore, one would expect that IBM 0661 disks arrays would perform better in the face of combined technology improvements than Fujitsu 2562 disk arrays. As expected, partially synchronous and synchronous disk arrays with fewer disks are able to match the performance of decoupled disk systems for slightly higher request arrival rates for the improved IBM 0661 disks.

5 Conclusions

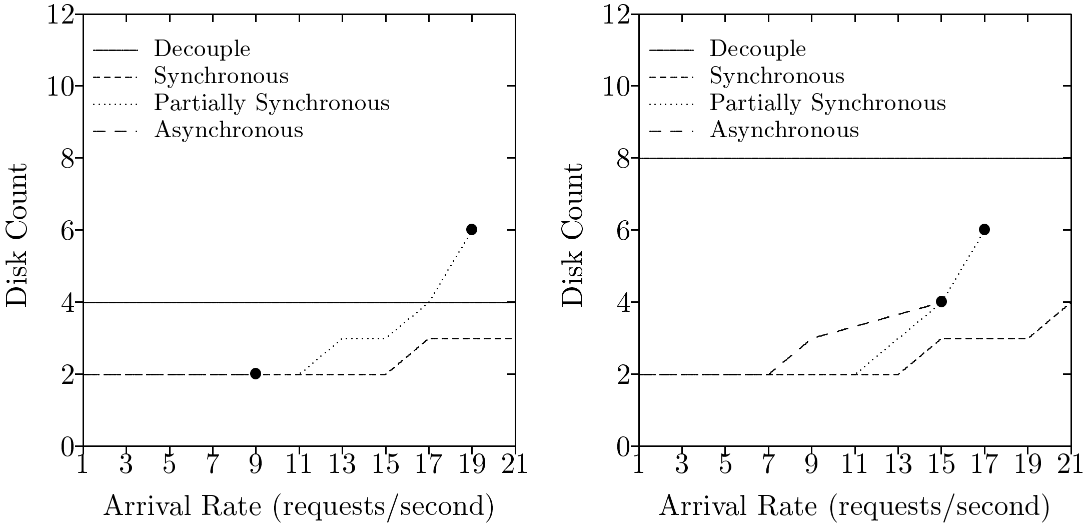
References

[1] DeGroot, M. H. *Probability and Statistics.* Addison-Wesley, Reading, MA, 1987.

[2] Gottlieb, A., and Almasi, G. *Highly Parallel Computing.* Benjamin/Cummings, Redwood City, CA, 1989.

[3] Kim, M. Y. Synchronized disk interleaving. *IEEE Transactions on Computers C-35* (1986), 978–988.

[4] Kim, M. Y. Asynchronous disk interleaving: Approximating access delays. *IEEE Transactions on Computers 40* (July 1991), 801–810.



a) Decouple: Four Disks

b) Decouple: Eight Disks

Figure 20: Response Time Equivalences for Fujitsu 2652 Disk Arrays With Halved Seek Delays. (64K Byte Requests, 512 Byte Blocks)

[5] Lee, E., and Katz, R. An Analytic Performance Model of Disk Arrays. In *ACM SIGMETRICS* (May 1993), pp. 98–109.

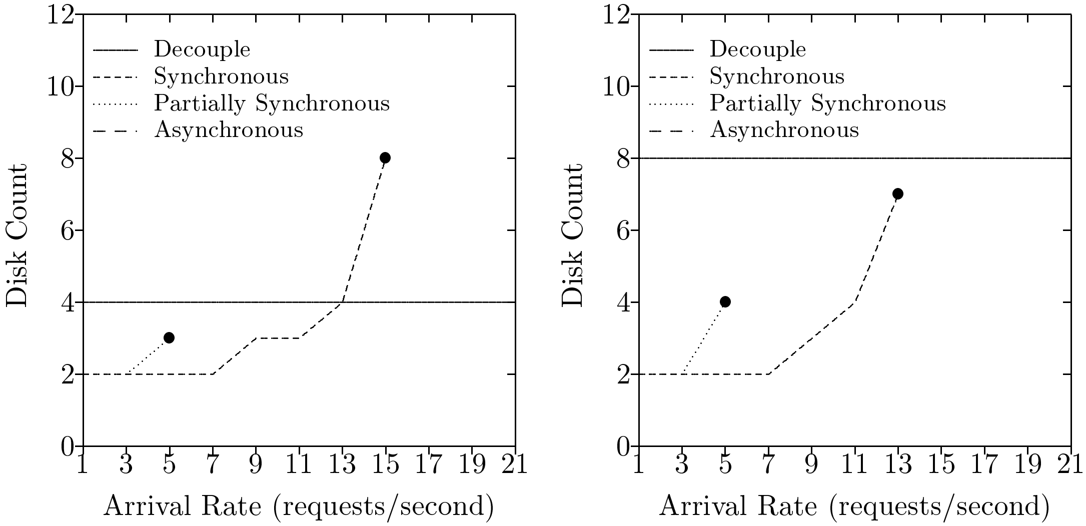
[6] Lee, E. K. Software and Implementation Issues in the Implementation of a RAID Prototype. Tech. Rep. UCB/CSD 90/573, UC Berkley/CSD, May 1990.

[7] Patterson, D., Gibson, G., and Katz, R. A Case For Redundant Arrays of Inexpensive Disks (RAID). In *Proceedings of ACM SIGMOD* (December 1988).

[8] Ziff Communications Company. *Data Sources.* New York, NY, 1991 V.11 no.1.

[9] Ziff Communications Company. *Data Sources.* New York, NY, 1992 V.11 no.2.

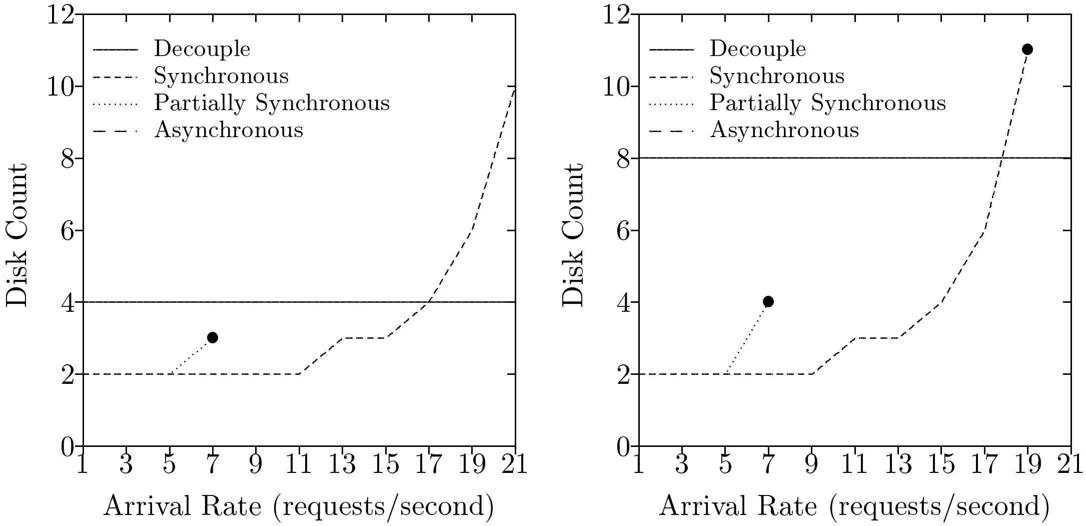
[10] Ziff Communications Company. *Data Sources.* New York, NY, Summer 1983.



a) Decouple: Four Disks

b) Decouple: Eight Disks

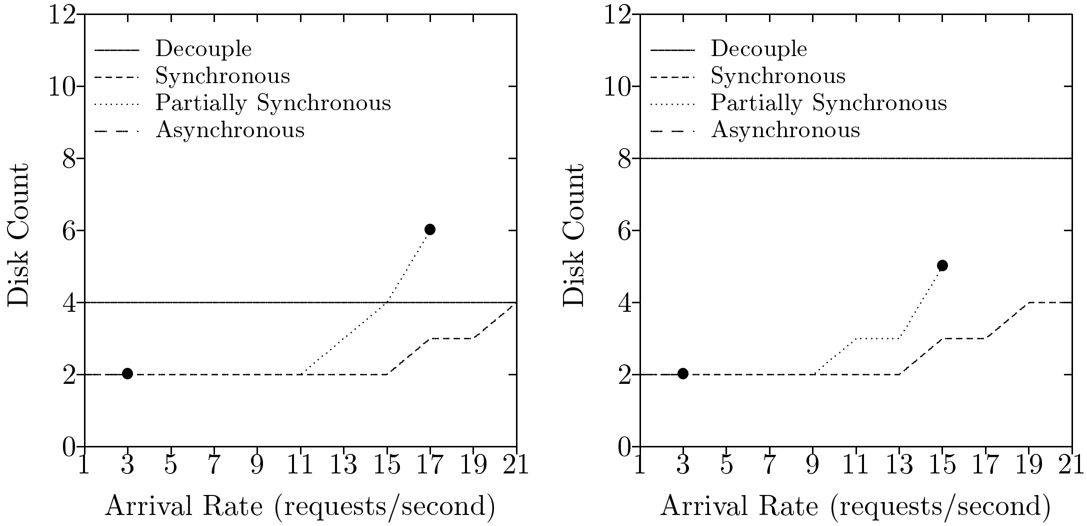
Figure 21: Response Time Equivalences for Fujitsu 2652 Disk Arrays With Doubled Track Capacity. (64K Byte Requests, 512 Byte Blocks)



a) Decouple: Four Disks

b) Decouple: Eight Disks

Figure 22: Response Time Equivalences for Fujitsu 2652 Disk Arrays With Combined Improvements in Rotation, Seek, and Density. (64K Byte Requests, 512 Byte Blocks)



a) Decouple: Four Disks

b) Decouple: Eight Disks

Figure 23: Response Time Equivalences for IBM 0661 Disk Arrays With Combined Improvements in Rotation, Seek, and Density. (64K Byte Requests, 512 Byte Blocks)