Proceedings of the 1990 Winter Simulation Conference
Osman Balci, Randall P. Sadowski, Richard E. Nance (eds.)

DISK I/O, A STUDY IN SHIFTING BOTTLENECKS

Charles E. Knadler, Jr.
Ralph M. May

IBM Corporation
704 Quince Orchard Road
Gaithersburg, Maryland 20878

ABSTRACT

Disk memory systems are electromechanical storage systems whose performance has not keep pace with the rapid increases in performance experienced by the solid state implementations of processor logic and main memory. As a result, there is a growing body of research into methods of better matching disk system and processor performance. This paper demonstrates, using a few example techniques from current research, that the underlying result of this work is to move the performance bottlenecks of disk systems between the seek, latency and data transfer phases. The paper does not address all current work, but suggests that through using analysis of the type illustrated, queueing network bottleneck analysis and simple simulations can accurately predict system performance enhancements.

1. INTRODUCTION

Discrete event simulation and network queueing theory are utilized to provide a unifying discussion of the effects of file structure and hardware organization on the performance of disk systems. Advances in disk performance can be characterized as shifting the performance bottleneck from the seek to the latency to the data transfer phase.

As a result of the cost advantage of disk memory over magnetic core and semiconductor memories and the performance differential between disks and central processing units, a large body of research has attempted to find architectural techniques to improve effective disk access speed. Two complementary approaches to improving disk performance have been proposed. One approach has focused on hardware sophistication of disk access and control: (1) a single disk per path, (2) multiple disks on a single path, (3) multiple disks on a single path with rotational position sensing (RPS), and (4) multiple synchronized disks connected via multiple paths.

The other approach has focused on data organization. Aided by improvements in hardware, an evolution in approaches to file-to-disk mapping is evident: (1) all task files on a single disk, (2) files distributed over multiple disks, (3) records distributed over multiple disks (disk striping or record level interleaving), and (4) disk striping with synchronized disks (byte-level interleaving).

The relative success of these approaches at improving disk performance is shown using discrete event simulation augmented by queueing network analysis. This work selectively addresses the three major components of a disk access: the seek phase, the latency phase and the data transfer phase.

2. SYSTEM PARAMETERS

The use of simulation and network queueing analysis to study disk and file organization, requires that definite system characteristics be chosen in order to achieve quantitative results. Tables 1 and 2 contain the disk drive performance characteristics and format assumptions that are used as the basis for this work. The values are designed to be representative of the performance of existing disk drives, not just one particular system.

3. SIMULATION EXPERIMENTS

A series of three simulation experiments were conducted to evaluate the effectiveness of traditional solutions to improving system throughput in I/O bound systems. The sections that follow provide a summary of the assumptions and results of these experiments. Where possible, our results are compared to others in the literature.

Table 1. 115MB Disk Drive Characteristics

Access time:	
single cylinder seek	6 ms
average seek	28 ms
maximum seek	60 ms
Disk rotational speed:	
3600 revolutions per minute	
average latency	8.33 ms
maximum latency	16.67 ms
Formatted characteristics:	
115,000,000 bytes	
18 sectors per track	
1024 bytes per sector	
915 cylinders	

Table 2. Seek Time in Milliseconds, as a Function of the Number of Tracks Traversed

$Ts = 5.6774194 + 0.3225806n,$	$1 < n < 32$
$Ts = 14.593408 + 0.0439560n,$	$33 < n < 305$
$Ts = 11.973745 + 0.0525451n,$	$306 < n < 914$

3.1 Experiment 1: Single Disk

A series of five measurements were made to determine the average system throughput (transactions per second) and the average disk access time for a single processor accessing a single disk of the type characterized in Tables 1 and 2. For the purposes of this analysis, a transaction will be defined as a sequence of 8 disk accesses, each of 1024 bytes and consisting of 7 reads and 1 write. The results of this experiment are presented in Table 1, where the times are expressed in milliseconds and seek lengths are expressed in number of tracks moved. These results were obtained with model runs of 100 seconds duration.

Table 3. Experiment 1 Simulation Results

RUN	XACTS/ SEC.	MEAN RESPONSE TIME	MEAN DISK UTILIZATION	MEAN SEEK TIME	MEAN SEEK LENGTH
1	3.18	314.1	0.97	28.2	299.9
2	3.15	316.6	0.97	28.4	304.2
3	3.14	317.8	0.97	28.5	305.8
4	3.13	318.7	0.97	28.5	305.7
5	3.15	317.1	0.97	28.6	307.0

In order to perform this experiment, additional information was needed about the disk accesses and computer system. What is the distribution of the data base over the disk? What is the pattern of seek lengths? What is the data bus bandwidth? What is the CPU processing time per transaction?

Assume the data base occupies the entire disk and that accesses are uniformly distributed over the entire data base. Data bus bandwidth is 1.5 megabytes per second with a 20% loss of efficiency due to overhead, effective bandwidth of 1.2 megabytes per second. The cpu has an execution time of 1.0 milliseconds per access (transaction processing and I/O setup).

The results of this experiment present little information that could not be deduced from the disk characteristics, however this experiment serves as a validation of the disk model. This model was then used to carry out additional experiments, whose results are not so readily deduced from the disk performance parameters.

3.2 Experiment 2: Multiple Disks

A series of measurements were made to determine the improvement in system performance as additional disks are added to the system. Consider 1, 4, 16, and 24 processors (or a single multitasking system with 1, 4, 16, and 24 concurrent tasks) accessing 1, 2 and 4 disks over the same data bus. The same characteristics are used as in experiment 1, however the data base does not occupy the entire disk for systems containing 2 or 4 disks, but one half of each of two disks and one quarter of each of 4 disks. The results of this experiment are presented in Table 4. These results are based on model runs of 200 seconds duration. Times are expressed in milliseconds and seek lengths are expressed in units of tracks moved.

Table 4. Experiment 2 Simulation Results

No. OF USERS	No. OF DISKS	XACTS/ SEC.	MEAN RESP. TIME	MEAN DISK UTIL.	MEAN SEEK TIME	MEAN SEEK LENGTH
1	1	3.16	315.8	0.97	28.3	302.0
1	2	3.94	254.1	0.48	20.7	151.1
1	4	4.49	222.6	0.24	16.7	76.0
4	1	3.24	1231.2	1.00	28.3	302.2
4	2	6.82	585.6	0.92	20.8	153.0
4	4	11.01	362.7	0.64	16.8	77.0
16	1	3.20	4920.3	1.00	28.4	302.4
16	2	7.11	2092.1	0.97	20.8	152.1
16	4	14.64	1090.4	0.88	16.8	76.8
24	1	3.17	7402.5	1.00	28.5	305.5
24	2	7.64	3121.2	0.97	20.7	151.6
24	4	15.49	1543.6	0.94	16.8	76.6

The results for 1 user illustrate the significantly improved performance resulting from shorter seeks. As the data base is spread across multiple disks, it occupies a smaller fraction of the disk and the average seek length is reduced. The mean seek time is seen to be reduced from 28.3 msec. for the single disk case to 16.7 msec. for the 4 disk case. Multiple disks can also be used to improve performance for a single user in other ways, which will be discussed later.

Examining the results for multiple users, it is observed that full use is not made of the multiple disks in this configuration. In the case of 16 users, 2 disks are only 97% utilized and 4 disks are only 88% utilized. While in the case of 24 users, 2 disks are only 97% utilized and 4 disks are only 94% utilized. Based on previous experience analyzing computer systems, two possible explanations suggest themselves to account for this performance: 1. during a portion of the experiment disks are free, but all users are queued for other disks and 2. a system bottleneck exists.

First let's examine explanation 1. The simulation repo rtsprovide the queueing statistics in Tables 5 and 6. Queue length is here defined as the number of customers waiting for service. Thus when queue length is 15, one customer is in service so that a total of 16 users are waiting for service by that disk. Examining the results for 16 users, we see from the maximum queue lengths that during the measurement period there were at least 3 periods with only one disk in use, thus certainly this theory partially explains the failure to fully utilize the disks. The increase in disk utilization for the 24 user case supports this explanation, since the probability that only one disk, or only two disks, or only three disks will be in use decreases with the larger number of users.

An examination of our system model, indicates that the single shared resource is the data bus. Could contention for the bus be reducing system throughput? The simulation model implements a design similar to rotation position sensing, RPS, [Lazowska et al. 1984; Kim 1986]. When the requested sector is available to be read or written, the disk controller determines if the bus is available or not. If the bus is available it is seized and the data transfer takes place. If the bus is busy, the disk rotates one revolution and the process is reiterated. Thus each time a disk finds the bus busy, the data transfer is delayed for one rotational period or 16.67 msec.

Table 5. Experiment 2 Disk Queueing Results (16 Users)

QUEUE	MAXIMUM QUEUE LENGTH	MEAN QUEUE LENGTH	DISK UTILIZATION
DISK1	15	3.700	0.9099
DISK2	14	3.313	0.8895
DISK3	15	2.902	0.8834
DISK4	15	2.426	0.8578

Table 6. Experiment 2 Disk Queueing Results (24 Users)

QUEUE	MAXIMUM QUEUE LENGTH	MEAN QUEUE LENGTH	DISK UTILIZATION
DISK1	22	5.699	0.9600
DISK2	19	4.139	0.9241
DISK3	23	6.270	0.9500
DISK4	20	4.011	0.9214

Model results show that with 24 users, bus contention occurs 668 times in 200.0 seconds for the two disk case and 5167 times in the same period for the four disk case. This latter result implies that approximately 10% of the simulation time is lost to bus contention and that only 90% of the time is available for useful work by the disks.

How do these results compare with other research? Our results are quite different from those reported by Olson [1989] in two areas. Olson finds little reduction in seek length when the data base is distributed over multiple disks and substantially poorer performance with multiple disks than our simulation model (also using a 1.5 megabyte per second bus).

Olson attributes the seek length performance to the success of the UNIX operating system's disk driver algorithm in ordering disk requests to minimize seek times. Olson's paper does not provide performance characteristics for the disk drives used, therefore it is not possible to completely assess the results. However for the same type of transactions used in our experiments, his graphs indicate that he observes throughputs of approximately 3.3 tx/sec. for a single user accessing a single disk, 3.8 tx/sec. for a single user accessing two disks, and 3.9 tx/sec for a single user accessing 4 disks. These results indicate a four disk throughput of 118% of the single disk throughput; the corresponding result for our simulation experiment is a four disk throughput of 142%. Olson's results appear to indicate a greater nonlinearity in seek time as a function of seek length, than those illustrated in Table 2 or in Baer's [1980] book; i.e., seek times for short seeks are longer than those calculated from Table 2.

Olson also observes only a 2.8 factor improvement for 24 users as the number of disks is increased from 1 to 4 and he attributes this to bus contention. The experiment 2 simulation yields a 4.9 factor improvement. Part of the difference in throughput improvement between experiment 2 and Olson's work can be attributed to the difference in mean seek times for the four disk case. This factor can be eliminated by considering the ratio of 24 users accessing four disks to that of one user accessing 4 disks. Olson's graph provides a ratio of 2.5 and the results for experiment 2 provide a ratio of 3.4, these results can be more accurately attributed to bus contention than the ratios of 2.8 and 4.9.

The difference between simulation results and hardware measurements can be explained in several ways. One explanation is that Olson's SCSI bus has greater overhead than the 20% overhead modelled in experiment 2. Another explanation is that the system holds the bus for a greater length of time per access than the RPS approach modelled.

Experiment 2 was repeated using a bus overhead of 50% for an effective bus bandwidth of 0.75 Mbytes per second and yielded a ratio of 3.3. Increasing the overhead to 75% for an effective bus bandwidth of 0.375 Mbytes per second still yielded a ratio of 3.0. Bus inefficiency of this magnitude is not reasonable, thus the time the disk holds the bus must be greater than modelled.

In order to investigate this second possibility, the experiment 2 model was modified to reflect a non-RPS architecture, in which the disk holds the bus for the entire latency period and queues the request if the bus is not available. With a bus overhead of 20%, effective bus bandwidth of 1.2 Mbytes per second, the model yielded a ratio of 2.7 comparable to Olson's results. Thus to fully take advantageof multiple disks either multiple buses or RPS controllers are required. Simply increasing the bus bandwidth will not significantly improve the throughput, since the bus is held during the latency period.

3.3 Experiment 3: Multiple Disks, Interleaved Files

Another series of experiments was conducted to investigate the performance of systems containing nonsynchronized interleaved arrays of disks.

3.3.1 Case 1. Interleaved Files

Kim [1986] suggests that I/O bandwidth may be increased by interleaving data on multiple disks (her preferred configuration is synchronized disks). Experiment 3 begins with a series of simulations designed to study the performance of a multitasked computer system with a DASD subsystem of 8 disks connected via a single RPS controller with a single 1.5 megabyte per second channel.

Using the disk performance characteristics given in Tables 1 and 2, with the exception of disk capacity, the performance of interleaved and noninterleaved disks was contrasted. The assumption was made that there were sufficient tracks in a cylinder that each request to a single disk requires a single seek. In the interleaved case, the interleaving was on a sector basis; e.g. sectors 1, 9, 17, 12, ... are on disk 1 and sectors 3, 11, 19, 27, ... are on disk 3. Using the request statistics reported by Kim: block size was varied between 4K to 240K bytes using a uniform distribution. In the noninterleaved disk case, the requests were uniformly distributed over the disks. It was assumed that only 25% of each disk was utilized for data (this effectively reduced seek lengths). Model runs of 200 seconds duration were used to generate the results shown in Table 7.

The results in Table 7 reflect a deterioration in performance with disk interleaving rather than a performance enhancement. How can this be? The interleaving of data sets over disks, called disk striping, has been shown empirically to improve performance and is an integral feature of the IBM Transaction Processing Facility (TPF) operating system [IBM 1988; Kim 1986; Olson 1989].

Table 7. Experiment 3 Simulation Results

No. OF USERS	Inter-leaved	XACTS/ SEC.	RESP TIME	MEAN DISK UTIL	MEAN SEEK TIME	MEAN SEEKS/ SEC.
1	YES	3.78	263.9	0.56	16.7	20.0
1	NO	4.20	238.4	0.12	17.0	4.2
4	YES	4.00	999.1	0.83	16.8	31.6
4	NO	4.59	869.0	0.40	16.8	4.6
16	YES	4.12	3845.5	0.94	16.7	33.0
16	NO	4.58	3464.9	0.73	16.7	4.6
24	YES	4.18	5655.6	0.9	16.9	33.6
24	NO	4.47	5273.2	0.83	17.0	4.5

TPF is a high volume transaction processing system, which normally accesses smaller blocks than the average 122K bytes accessed in experiment 3. Disk striping serves to make disk accesses more evenly distributed over the disk units. The assumption of even distribution of accesses is not normally a good model of computer systems. Non-interleaved disk systems normally exhibit highly skewed accessing patterns. Also TPF configurations normally include multiple disk controllers and higher speed channels than those modelled.

Kim assumes a disk accessing skew, in which 25% of the disks receive 68% of the accesses, and that the average seek time is 7.2 milliseconds. If either or both of these assumptions are true that could explain the difference in results. The impact of these assumptions is explored in the following variations to Experiment 3.

3.3.2 Case 2. Skewed Accesses

Experiment 3 was rerun with a skewed accessing pattern for the non-interleaved cases, but with the same seek patterns. Kim's probability distribution for choosing disks was used. [P(1) = 0.388, P(2) = 0.225, P(3) = 0.153, P(4) = 0.102, P(5) = 0.068, P(6) = 0.054, P(7) = 0.010, and P(8) = 0.001] Additionally the disk data transfer rate was doubled by doubling the data density per track while maintaining the same rotation rate.

The results for this simulation are presented in Table 8. These results were generated in model runs of 200 seconds duration. Again the interleaved disks yield lower performance than the non-interleaved disks.

Table 8. Experiment 3, Case 2, Simulation Results

No. OF USERS	Inter-leaved	XACTS/ SEC.	MEAN RESP. TIME	MEAN DISK UTIL.	MEAN SEEK TIME	SEEKS/ SEC.
1	YES	4.81	207.7	0.56	16.7	38.1
1	NO	5.36	186.8	0.12	17.0	5.4
4	YES	5.40	739.2	0.84	16.8	42.8
4	NO	5.97	669.0	0.31	16.8	6.0
16	YES	5.50	2891.0	0.94	16.7	43.8
16	NO	5.96	2649.9	0.34	17.1	6.0
24	YES	5.39	4396.4	0.98	16.8	42.9
24	NO	5.94	3948.8	0.36	17.0	6.0

3.3.3 Case 3. Shorter Seek Distances

To further illustrate that seek time is not the bottleneck to improved performance Case 2 was rerun with even shorter seek lengths. The data base was constrained to 12.5% of each disk's cylinders to reduce seek lengths.

Table 9. Experiment 3, Case 3, Simulation Results

No. OF USERS	Inter-leaved	XACTS/ SEC.	MEAN RESP. TIME	MEAN DISK UTIL.	MEAN SEEK TIME	SEEKS/ SEC.
1	YES	4.87	205.2	0.56	13.9	38.6
1	NO	5.44	183.7	0.10	14.2	5.4
4	YES	5.24	761.7	0.83	13.9	41.7
4	NO	5.78	686.4	0.31	14.2	5.8
16	YES	5.54	2867.0	0.95	13.9	43.9
16	NO	5.85	2680.8	0.36	14.1	5.9
24	YES	5.47	4347.5	0.97	13.9	43.5
24	NO	5.98	3887.6	0.37	14.1	6.0

Again the interleaved disks yield lower performance than the non-interleaved disks. Interleaving the disks results in a larger number of seeks for the system. These seeks become the system bottleneck. If the disk system were synchronized as suggested by Kim, the number of seeks would be reduced and results would reflect those of a single disk with a much higher data bandwidth. However the authors of this paper feel it is not practical to maintain synchronization for a large number of disk drives and the authors have helped implement near real-time systems incorporating over one hundred disk units of the largest capacity available at the time. Systems of this size do not appear to be a good candidate for synchronization. This feeling is not shared by Reddy and Banerjee [1989] who state that synchronization is a good solution when large files need to be transferred.

3.3.4 Case 4. Multiple Disk Controllers

In order to illustrate that the data transfer time (and contention for the data transfer path) is the bottleneck, case 2 was rerun with two disk controllers utilized to reduce controller contentions, the four (4) most popular disks on the same controller and the four (4) least favorite disks on a second controller. This allocation of disks to controllers provides the least favorable possible results for an equal number of disks on each controller.

The results of this last variation of Experiment 3 are presented in Table 10. Once again, model runs of 200 second duration were used to generate the data. Table 11 presents some additional information about controller utilization for the configuration being simulated.

Table 10. Experiment 3, Case 4, Simulation Results

No. OF USERS	Inter-leaved	XACTS/ SEC.	MEAN RESP TIME	MEAN DISK UTIL	MEAN SEEK TIME	SEEKS/ SEC.
1	YES	8.16	122.6	0.62	16.6	64.4
1	NO	5.35	186.8	0.12	17.0	5.4
4	YES	9.66	413.5	0.89	16.7	76.3
4	NO	6.71	594.8	0.30	16.8	6.7
16	YES	9.91	1608.0	0.94	16.8	78.7
16	NO	6.78	2326.6	0.34	16.7	6.8
24	YES	9.85	2421.3	0.94	16.7	78.7
24	NO	6.75	3494.6	0.33	16.7	6.8

Table 11. Experiment 3, Case 4, Controller Utilization

No. OF USERS	Inter-leaved	XACTS/ SEC	CONTROLLER1 UTILIZATION	CONTROLLER2 UTILIZATION
1	YES	8.16	66.58	63.89
1	NO	5.35	73.91	12.05
4	YES	9.66	80.02	76.97
4	NO	6.71	94.60	15.03
16	YES	9.91	80.91	77.75
16	NO	6.78	95.37	15.09
24	YES	9.85	81.30	78.28
24	NO	6.75	95.14	14.55

Here the interleaved disks yield higher performance than the non-interleaved disks as a result of contention for the controllers. The interleaved case results in virtually equal utilization for the two controllers while the non-interleaved case reflects the popularity of the disks and the first controller is much more heavily utilized as shown above.

4. CONCLUSIONS

A disk system can be viewed as a queueing system whose service time is the sum of three independent components: seek time, latency time, and data transfer time (neglecting the impact of a RPS controller on latency time or else considering the RPS induced delay to be part of the transfer time). Queueing network theory can be applied to provide insights into the problems that we have been examining.

4.1 Queueing Network Models

Mean value analysis provides a methodology for expressing certain system performance averages in terms of other more fundamental system performance parameters. Following Brandwajn [1985], the time required to satisfy an I/O request may be expressed as:

I/O Time = Queueing Time + Service Time.

Each of the terms on the right hand side of this equation can be expressed as sums of other averages:

Queueing Time = Device Wait + Path Wait,

Service Time = Seek Time + Path Wait + Latency + Channel Contention + Data Transfer.

Normally the Path Wait components associated with setting up the disk I/O request are viewed as negligible. Thus, queueing time is reduced to simply Device Wait time. For queueing network models, the device wait is calculated as queue length times the service time. Queue length is a function of service time and total number of accesses required. Similarly, the path wait associated with servicing the request is ignored. Thus, service time can be expressed as

Service Time = Seek + Latency + Contention + Transfer.

The Contention component of the Service Time represents the time lost due to contention for the channel at data transfer time.

Using an iterative application of mean value analysis (MVA) methods, see Lazowska et al. [1984], one can build queueing network models based on the three basic disk parameters: seek, latency, and data transfer. The forth component of interest, channel contention, is approximated in a series of model runs. (The use of such iterative techniques has a history stretching back to the work of Wilhelm [1977].)

The queueing network models are structured as follows. There is one queueing center that represents the CPU, and there is a queueing center for each disk in the system. The models do not include centers that correspond to channel controllers. Instead, the models account for channel contention effects by estimating the effect of channel contention on disk service demand.

The process starts by assuming that the contention component is zero. The model is run and the total network throughput is calculated. This throughput calculation is used to calculate the channel utilization for each disk:

UCH(Disk k) = Throughput * (No. Disk k Accesses per Transaction) * Transfer Time.

Total channel utilization, UCH, is calculated by simply summing over the disks allocated to the channel. These values are then used to calculate the contention component of the disk service time as

Contention = Retries * Rotation

where

Retries = (UCH - UCH(Disk k))/(1 - UCH).

The adjusted disk service time is used to conduct a new model run. The process is continued until the estimates for system throughput remain stable. The models are limited in that they can only be applied to non-saturated systems.

This queueing network model can be applied to the analysis of the Olson performance measurements and analysis conducted in Experiment 2. The results of the modeling are summarized in Table 12. The right hand column of the Table shows that this model yields results that are close to those generated by the simulations.

Table 12. Comparison of Simulation and Queueing Model Results

No. of Users	No. of Disks	Simulation XACTS/ SEC.	QN Model XACTS/ SEC.	Ratio Simulation to Model
1	1	3.16	3.27	.97
1	2	3.94	3.98	.99
1	4	4.49	4.53	.99
4	1	3.24	3.36	.96
4	2	6.82	6.51	1.05
4	4	11.01	10.38	1.06
16	1	3.20	3.36	.95
16	2	7.11	7.66	.93
16	4	14.64	15.16	.97
24	1	3.17	3.36	.94
24	2	7.64	7.82	.98
24	4	15.49	15.96	.97

4.2 Shifting Bottlenecks

The various strategies used to improve disk access performance can be understood as attempts to reduce the effects of performance bottlenecks in disk I/O access. Thus, Olson's strategy of distributing the contents of a file across multiple disks can be understood as an attempt to insure that a single disk does not become a bottleneck. One reduces the effects of device queueing while suffering an increase in the device service time due to channel contention effects.

Spreading records across the disks should enable one to reduce average seek times, but Olson's findings and the above modelling results indicate that this may not be automatic.

Kim's synchronous disk access method offers a solution to the channel congestion problems associated with large record transfers. For large records, even 3 MByte/sec data transfer rates can not eliminate channel congestion. The decision to adopt disk synchronization eliminates channel contention and reduces the data transfer by a factor of 1/n, where n is the number of synchronized disks. As Kim points out, the synchronized disk system has the same seek and latency characteristics as a single larger capacity unsynchronized disk. This means that synchronization should only be considered in situations where the channel contention reduces throughput to levels below that of a single disk with reduced transfer rate:

$$(n-1)/n \ * \text{Transfer} > \text{Seek} + \text{Latency}.$$

The results presented above show that in situations where there is wide variation in record size, synchronization does not help and, in fact, reduces performance.

APPENDIX

A.1 Simulation Structure

It is relatively simple to construct a disk system simulation, adequate to evaluate and study techniques for improving disk system performance. The following set of eight events forms the basis for a versatile multiuser, disk array model.

Table 13. Basic Events

Event#1. I/O Request: an application or system program requests a read or write and the I/O manager (simulated operating system software) generates a disk access request event in response.
Event#2. Disk Access Request: the I/O manager causes the disk controller to initiate a disk access.
Event#3. Seek Start: the disk controller initiates the seek operation.
Event#4. Seek complete: the disk controller completes the seek operation.
Event#5. Start Rotation Wait: the beginning of the latency period.
Event#6. Sector Ready: the end of the latency period.
Event#7. Start Data Transfer: correct sector starts to pass under the read/write heads.
Event#8. Complete Data Transfer: data transfer complete.

A.2 Model Design

A Pascal language disk system model can be structured as follows:

```
program main;
 {model of disk arrays}
  uses p_smpl, rand_p; {simulation and random number
                                              routines}
 const
  {simulation parameters, disk characteristics etc.}
 type
  {definition of model data structures}
 var
  {model variables}
  { Pascal procedures used by the model }
 begin {body of simulation}
  {simulation initialization}
  while time < stop_time do begin { main simulation loop}
    CAUSE(event,transaction); {take next event off of the
                                   model future event chain}
   case event of {event handling logic}
    1: begin {i/o request}
         {call i/o manager to generate disk access request}
         {create i/o request data control block for the disk access,
          this block contains the data needed for the model to
          control the requested disk access}
         SCHEDULE event #2 for this access
       end; {end of event 1 logic}
    2: begin {disk access request}
         if requested disk is idle then SCHEDULE
                                   event #3 for this access
         else ENQ (queue) this request for the disk
       end; {end of event 2 logic}
    3: begin {seek start}
         REQUEST (seize) disk, calculate seek time,
          and SCHEDULE event #4 for this disk access
       end; {end of event 3 logic}
    4: begin {seek complete}
         SCHEDULE event #5 for this disk access
       end; {end of event 4 logic}
    5: begin {start latency period}
         calculate latency time and SCHEDULE event #6
         for this disk access
       end; {end of event 5 logic}
    6: begin {sector ready}
         if data bus is available for the disk transfer then
             REQUEST (seize) the bus and SCHEDULE
               event #7
         else SCHEDULE event #6 with latency time = 1 rotation
       end; {end of event 6 logic}
    7: begin {start data transfer}
         SCHEDULE event #8 after data transfer
       end; {end of event 7 logic}
    8: begin {complete data transfer}
         RELEASE the data bus;
         RELEASE the disk;
         if any accesses queued for this disk then
             DEQ (dequeue) the first access and SCHEDULE
                          event #2 for the new access;
         SCHEDULE event #1 to start next access for
                     the user completing a disk access;
       end; {end of event 8 logic}
   end; {end of event handling logic}
   REPORT; {print simulation reports}
  end; {end of main simulation loop}
 end. {end of model}
```

REFERENCES

Baer, J. (1980), *Computer Systems Architecture*, Computer Science Press, Potomac, MD, 255-257.

Brandwajn, B. (1983), "Models of DASD Subsystems with Multiple Access Paths: A Throughput-Driven Approach," *IEEE Transactions on Computers*, Vol. C-32, No. 5, 451-463.

Brandwajn, B. (1985) "Issues in Mainframe System Modelling," *Modelling Techniques and Tools for Performance Analysis*, D. Potier, Ed., Elsevier Science Publishers B.V., New York, NY. 259-277.

IBM (1988), "Transaction Processing Facility, Version 2 Rel. 4,Concepts and Structure Manual," IBM Corp., GH20-7447.

Kim, M.Y., (1986) "Synchronized Disk Interleaving," *IEEE Transactions on Computers*, Vol. C-35, No. 11, 978-988.

Lazowska, E.D., J. Zahorjan, G.S. Graham, and K.C. Sevcik (1984), *Quantitative System Performance*, Prentice Hall, Englewood Cliffs, NJ.

Olson, T.M. (1989), "Disk Array Performance in a Random I/O Environment," *Computer Architecture News*, Vol. 17, No. 5, ACM Press, 71-77

Reddy, A.L.N. (1989) and P. Banerjee, "An Evaluation of Multiple Disk I/O Systems," *IEEE Transactions on Computers*, Vol. 8, No. 12, 1680-1690.

Wilhelm, N.C. (1977), "A General Model of the Performance of Disk Systems," *Journal of the ACM*, Vol. 24, No. 1, 14-31.