A SCSI Disk Model for Multimedia Storage Systems

Meng Chang Chen, Jan-Ming Ho, Ming-Tat Ko and Shie-Yuan Wang
Institute of Information Science, Academia Sinica, Taiwan
{mcc,hoho,mtko,shieyuan}@iis.sinica.edu.tw

October 24, 1996

Abstract

Modern magnetic disks support advanced mechanisms, such as read-ahead and caching, to optimize disk access performance. However, most of the proposed disk models in the literature of multimedia storage systems are too simplified to include the advanced feature that result in inaccurate performance prediction. In this paper, we propose an accurate, detailed SCSI disk model customized for multimedia storage systems. The methodologies of extracting the parameters of the proposed model are presented. Disk throughput is analyzed and verified by experiments under various contexts. The model and methodologies proposed in the paper can benefit the design of multimedia storage system in many ways, such as performance prediction, file layout design and disk I/O scheduling policy.

1 Introduction

Modern magnetic disks play a very important role in multimedia computing, such as file cache in a hierarchical multimedia storage system[7, 10], and storage for multimedia clips[6] and (long) continuous media[4]. The objectives of many research projects aim to allow a disk or a group of disks[1] to provide high bandwidth, as well as to respond as predictable as expected[12]. For both goals, it requires an accurate performance model of disk as foundation to understand and to predict the behavior of the disk. In addition, the model cannot be too complicated to be implemented as on-line decision supports for multimedia storage systems, such as performance prediction, benefit scheduling policy decision[14] and file layout design[5].

However, most of the disk models in the previous research of multimedia storage systems are oversimplified. Some assume disk transfer rate is constant [3, 11], while the others use the maximum delay as an assurance of service quality [4]. The former (the "average" model) does not take the fluctuating characteristics of disk access into consideration, and the latter (the "worst case" model) is conservative that it under-utilizes the capacity of the disks. Moreover, most disk models fail to model the sophisticated mechanisms and controls of modern disk that allow caching and performing read-ahead to reduce response time. The parallelism among disk operation, data transfer and host machine processing is another area often ignored. As a result, the accuracy of predicted performance is questionable.

An accurate, detailed model of modern disk consists of many parameters, including seeking time, transfer rate, cache full/empty ratio, and is able to describe the complicated mathematical relationships among the parameters. Some parameters have fixed values, such as size of track skew, while some are tunable, such as cache full/empty ratio, that can influence the disk throughput. Some are functions of other dependent parameters, such as the disk head seeking time is a function with seeking distance as input. Not all the parameters can be obtained in technical manuals from disk manufacturers, as they are context dependent or simply not provided by the disk manufacturers. Some parameters varies from drive to drive due to the physical discrepancy of disks, not possible to be documented in the technical manuals. For instance, each disk has different number of bad sectors from manufacturing defection. Even found in technical manuals, the published parameters may be neither accurate, nor in the desired representation forms. An example is the seeking time function that generally only average and/or maximum seeking time are published, while seeking time function with seeking distance as input parameter is desired by system designers. Furthermore, some performance parameters depend on the hardware and/or software of the host machines that makes it impossible for disk manufacturers to provide the numbers. Consequently, there is a need to have feasible methodologies to extract the parameters from the disks.

The goal of this paper is different from [8, 12] which focus on building a complete and accurate disk simulator. Its drawback is that a disk simulator cannot provide on-line advice due to its complexity. Rather, this paper emphasizes on finding a simple and useful disk model that still maintains a high degree of accuracy, specially customized for multimedia storage systems. The major differences between a multimedia storage system and a conventional text file system are their access unit sizes and access controls. The access unit of a multimedia storage system is generally large, up to several tracks, while in a conventional text file system, the access unit is a line or a record, just hundreds or thousands bytes long. Due to the real time constraint of most multimedia systems, the multimedia storage system has to be designed so that the data can be read and transmitted in time as predicted. In consequence, most multimedia storage systems are designed to have control on both of the number and bandwidth of access streams, while the text file system plays no active role in access control. A carefully designed disk model can take advantage of the unique characteristics of multimedia storage systems to reduce the complexity. However, the model has to be easily built and utilized, and still accurate and useful enough to help the multimedia storage systems in their on-line decision makings, such as disk requests scheduling.

In this paper, Section 2 introduces the SCSI disk architecture, and proposes a performance model. The performance model carefully depicts the parallelism of disk, SCSI controller and data transfer. In Section 3, an acquisition methodology of each parameter in the performance model is presented. The acquisition methodology selection criteria include accuracy of the parameter and ease of use. Throughput issues related to multimedia applications are discussed, and formulas are proposed and verified by experiments in Section 4. Conclusions are in Section 5.

2 A SCSI Disk Model

2.1 SCSI Controller and Disk

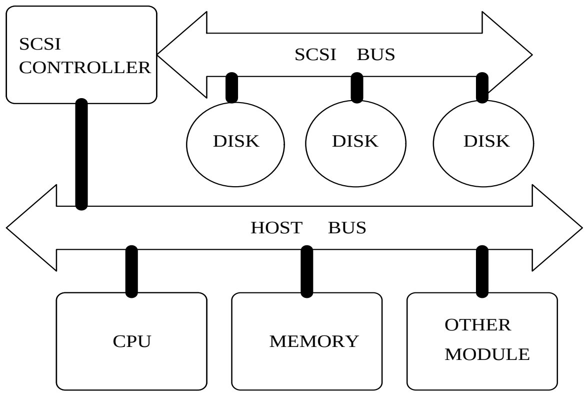

Figure 1: SCSI Bus Configuration.

SCSI (Small Computer System Interface) [13] disk I/O system is popular for its low cost, flexibility, versatility and high performance. As depicted in Figure 1, a SCSI bus connects a SCSI controller to multiple SCSI peripherals, and the SCSI controller is attached to the host bus.

In Figure 2, the structure of disk is depicted. In a disk, there are several platters that both sides can be used for data storage and each side has its own read/write head. On each surface, there are several thousands tracks, and the tracks of each platters with the same radius from the spindle form a cylinder. In the most modern disks, the tracks in the outer zone have larger capacity that the tracks in the inner zone. Contiguous data structure is

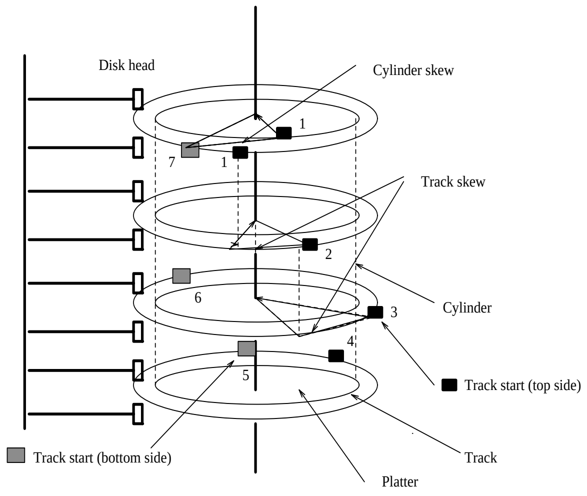

Figure 2: SCSI Disk Structure.

usually stored in one or more track of a cylinder to save the read/write head seeking motion. The tracks of a cylinder have different starting positions; the distance of the starting position of two neighboring tracks of a cylinder is called track skew. The purpose of disk skew is designed to offset the time of head switch, (i.e. the settlement operation of a read/write head due to out of alignment between tracks) and host processing time that may allow sequential read of tracks as fast as disk rotates (called non-stop media access). Similarly, cylinder skew is the distance between the starting positions of neighboring cylinders, and serves the same purpose. A detailed introduction of magnetic disk mechanism and model can be found in [8].

The analysis in this section is valid for both read and write operations. For simplicity, only read operation is used as example, and only one zone is considered. Without loss of generality, cylinder skew is considered the same size as track skew in the analyses of the paper to reduce replications of similar formulas. Since the discussion in this article is intended for multimedia storage system, the unit of access is track, instead of sector as in other papers[9]. Due to the limitation of length, the proof of formula is ignored when the formula is clear and self-explained. Interested readers can refer to [2] for details.

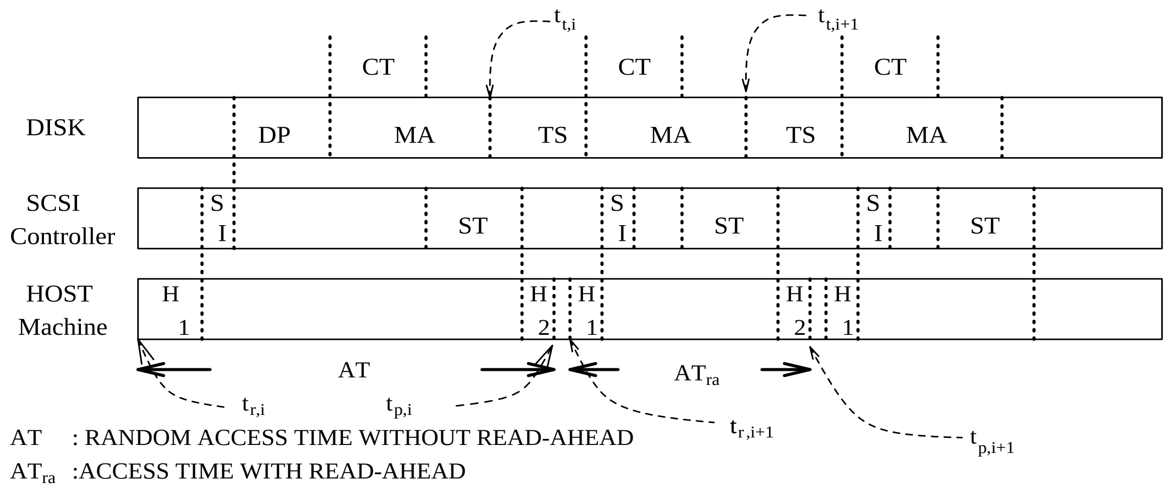

Figure 3: Timing Diagram of SCSI Disk Access.

2.2 SCSI Disk Access Model

The access time of a SCSI disk comprises host machine pre-access processing time, SCSI controller command interpretation time, disk head positioning time, waiting time before reading from disk platter, media access time, SCSI transmission time, and host machine post-access processing time. The timing diagram of disk access is depicted in Figure 3.

Machine pre-processing (H1) time starts from the time user process calls I/O functions until operating system issues access commands to SCSI controller. When SCSI controller receives a command from host machine, it interprets the command, initiates connection and forward the SCSI commands to target hard drive. Generally, the interpretation time (SI) is short, in comparison with the whole I/O processing time.

Hard disks have to position the read/write head to the right cylinder that the duration is known as seek time, before data read/write. The seek time is a linear function of square root of seeking distance when the distance is short, and for long seeking distance, it is proportional to the distance. Before the read/write head settles down, some minor adjustment is needed to have the head right on the top of the target cylinder that the adjustment time is called head settle time. In addition, read/write cannot be started unless the target sector is rotated to the position underneath the head. The time of rotation is naturally called rotation latency. Altogether, the sum of seek time, settle time and rotation time is called disk head positioning (DP) time.

Once data is read from disk platter, it is stored in the cache of disk controller. The time of data read, called as media access time (MA), is exactly the rotation time of the target data block. The purpose of the caching is not only for balancing transmission rates between disk and SCSI bus in order to optimize the overall performance of SCSI devices, but also for data storage during read-ahead operation. Data cannot be transferred from disk drive to SCSI controller until the amount of data cached is higher than a pre-specified threshold value. The time to fill the cache to the threshold level is called cache filling time (CT). The transfer time of cached data, via SCSI controller, to host machine is denoted as ST. Finally, the host machine spends H2 time to process the incoming data and to move it to the user address space.

The older disk drives do not have caching and read-ahead functions that its access response time is determined by the disk head position as the request is processed. While disk head position is considered a random function for random access, it is obtainable from other parameters in a sequential access.

In Figure 3, $t_{r,i}$ denotes the time user process issues i-th disk access, $t_{t,i}$ denotes the time disk completes media access of the i-th disk access, and $t_{p,i}$ is the time user process receives data of i-th disk access. IR_i is the inter-request delay after the i-th access. I.e. $t_{r,i+1} = t_{p,i} + IR_i$. We have formula Eq. (1) representing the access time (without read ahead) AT.

$$AT = H1 + SI + DP + CT + ST + H2 \tag{1}$$

From Figure 3, it can be observed MA is subsumed by CT and ST. Therefore, MA does not present in the equation.

For sequential access (without read ahead), the first read can apply Eq. (1). Afterward, the time of each disk access $AT_{seq,i}$ is as in Eq. (2).

$$AT_{seq,i} = H1 + SI + DP_{seq,i} + CT + ST + H2 \tag{2}$$

where

$$DP_{seq,i} = \begin{cases} RES_i * (MA + TS) & if\ RES_i\ \geq\ 0 \\ (\lceil |RES_i| \rceil - |RES_i|) * (MA + TS) & if\ RES_i\ <\ 0 \end{cases} \tag{3}$$

$$RES_i = \frac{MA + TS - CT - ST - H2 - IR_i - H1 - SI}{MA + TS} \tag{4}$$

$$t_{p,N} = t_{r,1} + AT + \sum_{i=2}^{N} (IR_{i-1} + AT_{seq,i}) \tag{5}$$

If $RES_i \geq 0$, the next media access can proceed immediately at the next rotation. Otherwise, disk will rotate $\lceil |RES_i| \rceil$times without media access. Eq.(5) shows the completion time of N sequential accesses.

2.3 Disk Read Ahead

$$\boxed{t_{p,i} = \begin{cases} t_{t,i-1} + TS + CT + ST + H2 & if\ t_{r,i} + H1 + SI \leq t_{t,i-1} + TS + CT \\ t_{r,i} + H1 + SI + ST + H2 & if\ t_{t,i-1} + TS + CT \leq t_{r,i} + H1 + SI \end{cases} \qquad (6)}$$

Figure 4: Access Completion Time with Read Ahead.

In order to increase the throughput of disk I/O system, modern disk provides a function called *read-ahead* that reads the next data block immediately and store the data in the cache memory of disk after the current read is done. When the next data block is requested, it can be accessed immediately. From Figure 3, the read-ahead operation overlaps the data transfer and host machine processing. If arranged properly, the disk access rate can be as fast as the disk rotation rate times track density. During read-ahead, if the incoming access requests other data block, the disk controller is intelligent enough to interrupt the current read-ahead operation and start to read the requested target data block.

The disk access time with read-ahead is not as straightforward as without read-ahead. The reason is that the states of disk and cache affect the response time. The read operation completion time, $t_{p,i}$ is represented as in Eq. (6) in Figure 4. The first equation shows the case that the requested data block is not cached when disk controller receives the request, and the second shows otherwise. The response time with read ahead, AT_{ra}, can be calculated by

$$AT_{ra} = t_{p,i} - t_{r,i}. \tag{7}$$

For a sequence of sequential access, the read-ahead function can greatly reduce the overall response time if the read request can be issued in time. As read-ahead operation only reads one more data block and stops once completed, the read command of the next data block must arrive at the disk controller before the completion of read-ahead operation (i.e. $t_{r,i+1} + H1 + SI \leq t_{t,i} + TS + MA$, for the i-th access) in order to maintain the non-stop media access. Otherwise, the disk has to wait for one complete revolution before it can read the next data block. For the sequential access without read-ahead, the window for the read command to arrive to achieve non-stop media access is narrower. The condition is $t_{r,i+1} + H1 + SI \ \leq t_{t,i} + TS$ for the i-th access.

3 Parameters Acquisition

Only having accurate performance parameters warrants accurate disk models. Generally an easy way to obtain performance parameters is to consult technical references published by disk manufacturers. However, the performance data on the technical reference may be nominal (e.g. average access time instead of a function of context variables) or not mentioned (e.g. cache filling algorithm)i Furthermore, performance parameters may be host machine dependent. As a result, the performance data has to be either derived from model, measured or obtained via interrogations with SCSI disk and bus controllers.

There may be more than one way to acquire the value of a parameter. Our preference is as the order of the sequence: derivation from model, user-level measurement, kernel-level measurement and interrogation, and external measurement(e.g. using SCSI bus analyzer). The rationale is to allow ordinary users to acquire the parameters as easy and accurate as possible. The interrogation method is preferred for the specification of logical structure of disk, such as the number of cylinders and the length of track skew, since the data stored in the disk controller is generally correct, and by doing so can save much trouble in measuring. The acquisition methodologies can be applied to many other environments with different hardware and software.

3.1 Environment

The parameters extraction and measurement environment in this paper is on a 486 DX2-33 PC with SCSI bus (AHA2842VL) and SCSI disk drive (HPC3323A). A fine grained micro timer is implemented for event timing recording, which will not be discussed in this paper. The disk driver is modified to allow interrogating peripheral directly for parameter extraction, and recording the time of events. A SCSI analyzer is used for analyzing and measuring the signals on SCSI bus.

3.2 Methodologies

Pre-read Host Processing Time

Pre-read host processing time (H1) is the time from user process issues I/O command until disk driver sends out SCSI commands, including the time of context switch, memory mapping and management activities, and driver execution. Both the starting and finishing time are logged using the micro timer, and H1 can be calculated as their difference. Generally, the length of pre-read host processing time is proportional to the size of data block, and is host machine dependent.

SCSI Command Interpretation Time

SCSI command interpretation time (SI) is the time SCSI controller runs its firmware to interpret the SCSI command and performs initial set up with the target disk. As SCSI controller executes as a black box, SI cannot be measured directly with the tools we have. Alternatively, we measure the processing time of SCSI controller of commands without interacting with disks (called non-disk SCSI commands) and the time of SCSI controller performs handshaking with disk separately, and add up the two numbers as a reference value of SI. For the former, we choose a dozen of non-disk SCSI commands and measure the processing time by the micro timer invoked from the device driver. The handshaking time can be measured directly using SCSI analyzer. The largest sum is selected as SI, just about 1% of the total disk access time, in order not to underestimate. The other way to obtain the SI value is to subtract all other processing times from the total access time. As SI is relatively small to the whole disk access time, the imprecision in other measurements will result in high error rate of SI. Thus, this approach is not adopted.

Disk Head Position Time

Disk head positioning time (DP) includes seeking time and rotation latency[*]. Seeking time is a function of seeking distance that is synthesized from the measurements for each seeking distance. First, the time to complete the seek operation of the current cylinder (called self-seek operation) is measured at the user-level. The time of self-seek is spent on host machine and controller processing, but the mechanical disk head movement. Then seek commands of each different distance are issued and measured. Subtracting the measured times of each distance by the self-seek time, the results are the pure disk head seeking times for different distances. The process is repeated until the seeking times for all distances are obtained. The seeking time in our model is shorter than the published numbers and other work since the handshaking time between SCSI controller and disk is counted in SI in this model.

[*]Head switch time is subsumed by track/cylinder skew and head settle time is measured as part of seeking time that both do not show in this model.

The rotation latency is the result of dividing the distance between the disk head and the target sector by the revolution speed of disk. The revolution time published in the technical manual is generally accurate. However, in order to amend the adverse situation of having a bad disk, the method in [12] is adopted that the revolution time is obtained by reading same sector without read-ahead many times. The difference between two completion times is the time for one revolution.

Track/Cylinder Skew

ANSI SCSI command set[13] provides commands to interrogate SCSI controller and disk to obtain the track/cylinder skew factor, which is represented in the unit of sector. Track/cylinder skew (TS/CS) time can be calculated by dividing track/cylinder skew by the revolution speed.

Cache Filling Time

Caching filling time (CT) is determined by the threshold value of cache full ratio, is controlled by a SCSI variable, called Buffer Full Ratio(BFR) for read operation[†]. Only when the cached data is above the threshold value, data can be transferred from disk to SCSI bus. The usage of Buffer Full Ratio is a grey area in ANSI SCSI-2 specification that disk manufacturers may have their own designs. Thus, the associated cache filling time of each value of BFR (range from 00 to FF) needs to be measured.

In the measurement, a SCSI analyzer is used to measure the time from disk receives read command (of the same track) to data transfer starts. The measured time is the sum of rotation latency and cache filling time. In order to remove effect of rotation latency, the measurements are taken many times with different inter-request intervals. The smallest measurement is considered as the cache filling time. This process continues until all the BFR values are examined.

[†]Similarly, Buffer Empty Ratio is for writing data from SCSI controller to disk.

Media Access Time

Media access time (MA) for a track is exactly the time of one revolution.

SCSI Bus Transfer Time

SCSI bus transfer time (ST) is decided by the size of data and BFR, and can be measured directly using SCSI analyzer. With smaller BFR, the times of data transfer is larger. Thus, the transfer time is longer due to more initializations. Note that ST is measured for each different bfr value.

Post-Read Host Processing Time

Post-read host processing time, H2, is generally proportional to the block size. H2 is the difference between the time of disk interrupt occurs and user process receives data, logged by calling the micro timer.

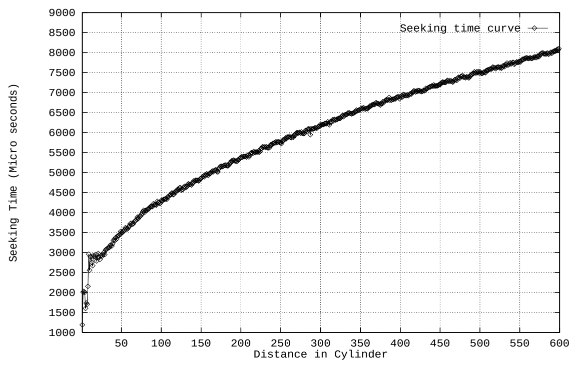

Figure 5: Seeking Time from 1 to 600 Cylinders.

Example 1 The extracted values of parameters following the methodologies, averaged from 100 measurements, are listed in Table 1. The seeking function with distance from 1 to 600 cylinders are plotted in Figure 5, each time point averaged from 100 random experiments. ■

H1	SI	MA	TS	CS	H2
1.42ms	0.10ms	11.1ms	1.48ms	3.05ms	0.58ms

BFR	CT	ST
01	1.86ms	12.72ms
80	6.64ms	7.97ms
FF	13.00ms	7.98ms

Table 1: Parameters Measured for 60KB Read on HPC3323A.

4 Disk Model and Measurements for Multimedia Storage Systems

There are several factors affect the throughput of disk, including access pattern (i.e., sequential or random access), read-ahead or not, inter-request period, and values of buffer ratio variables. In this section, the performance analyses for random and sequential accesses are presented. Applications of the model on multimedia storage systems are also discussed.

4.1 Random Access

For random access, Eq. (1) is applicable for the response time. The values of seeking distance, rotation latency and track size are needed to calculate AT. Rotation latency depends on the distance between the read/write head position and the target sector. There are two ways to keep track of the read/write head position. One way is to interrogate the SCSI and disk controller, and the other is to maintain a counter in host machine to keep the disk head position and have the counter updated every a short period. However, both approaches cannot work effectively that the former suffers from inaccuracy due to processing delays and the latter has high counter update overhead and synchronization problem. The difficulty to obtain correct position of read/write head in time leads to the problem in deciding rotation latency. Thus, average response time is a practical alternative for system designers. The average random access time, $\overline{AT}_{random}$, is as below.

$$\overline{AT}_{random} = H1 + SI + DP_{random} + CT + ST + H2 \tag{8}$$

where DP_{random} is DP with average rotation latency (i.e. half a revolution). The throughput of random access, TP_{random}, can be calculated as

$$TP_{random} = \frac{\text{track size}}{(\overline{AT}_{random} + IR_{avg})} \tag{9}$$

where IR_{avg} is the average inter-request time that equals to the time of half revolution of disk.

	Response Time	Throughput
Model	24.26ms	2579KB/s
Measurement	24.11ms	2489KB/s

Table 2: Response Time and Throughput of 60KB Read.

Example 2 Applying Eq. (8) on the data in Table 1, the average random access time of 60KB read (with 2 cylinders seek and BFR=80) is 24.26ms. Note the average seeking time for 2 cylinders is 2ms from Figure 5. An experiment with the same setting is conducted 1000 times randomly, and the measured average response time is 24.11ms that is within 1% error rate from the time derived from our model. The throughput data is also listed in Table 2. ■

4.2 Sequential Access

The response time of sequential access without read-ahead is shown in Eq. (2). The throughput of sequential access without read-ahead is represented in Eq. (10), which is similar to Eq. (9).

$$TP_{seq} = \frac{\text{track size}}{(\overline{AT}_{seq} + IR_{avg})} \tag{10}$$

The difference is that $\overline{AT}_{seq}$ is the average AT_{seq}, based on a given distribution of arrival time of next request. The response time of disk access without read ahead is generally not predictable, as the rotation latency is not predictable.

In the case of sequential access with read-ahead, Eq. (6) and Eq. (7) formulate the response time. Generally, the maximum throughput of disk equals to the disk rotating speed times track size and adjustment. For a disk with NP data platters, due to the existence of track skews, there needs NP+1 revolution in order to read all the NP tracks in a cylinder that can be observed from Figure 2. In addition, cylinder skew, which is generally longer than track skew, needs to be discounted from the equation. Eq. (11) shows the maximum throughput of a disk.

$$TP_{max} = \frac{\text{track size}}{MA} * AJ \tag{11}$$

where

$$AJ = \frac{NP * MA}{((NP+1) * MA) + (CS - (MA - (NP-1) * TS))} \tag{12}$$

The adjustment AJ in Eq. (12) is to compensate the time spent on track and cylinder skews.

When the inter-request delay is long, according to the second part of Eq. (6), there is a delay for data transfer from disk to host machine. The accumulation of several late read requests may cause a miss[‡] and disrupt the read-ahead chain. As a result, one or more revolution is wasted and the disk cannot achieve its maximum throughput. The following lemma shows a condition for maximum throughput.

Lemma 1 If for every i,

$$t_{p,i} = t_{t,i-1} + TS + CT + ST + H2, \tag{13}$$

the host machine receives data as fast as the maximum throughput of disk. **Proof** $t_{p,i}$ is the time when host machine finishes the i-th disk access and $t_{t,i-1}$ is the time when disk finishes the $(i-1)$-th media access. It can be observed from Figure 3 that

$$t_{p,i} - t_{t,i} = CT + ST + H2 - MA. \tag{14}$$

[‡]I.e. the read/write head passes the start of the track before the access request arrives at the disk controller.

Replacing $t_{p,i}$ in Eq. (14) using Eq. (13), we have

$$t_{t,i-1} - t_{t,i} + TS = -MA, \tag{15}$$

or

$$t_{t,i} - t_{t,i-1} = TS + MA. \tag{16}$$

Eq. (16) means that disks has no miss in the sequential read. I.e. at its maximum throughput. ■

In the following, we will show the condition of the length of inter-request period (IR_i) so that host machine can obtain the maximum throughput from the disk.

Lemma 2 If for every i,

$$IR_i \leq TS + MA - H1 - SI - ST - H2, \tag{17}$$

then the host machine can read the disk at its maximum throughput. **Proof** Adding $t_{p,i}$ to both side of Eq.(17), we have

$$t_{p,i} + IR_i \leq t_{p,i} + TS + MA - H1 - SI - ST - H2. \tag{18}$$

Since $t_{r,i+1} = t_{p,i} + IR_i$ and $t_{t,i} = t_{p,i} - H2 - ST - CT + MA$, Eq. (18) becomes

$$t_{r,i+1} \leq t_{t,i} + TS + CT - H1 - SI, \tag{19}$$

or

$$t_{r,i+1} + H1 + SI \leq t_{t,i} + TS + CT. \tag{20}$$

Applying Eq. (6), we have

$$t_{p,i} = t_{t,i-1} + TS + CT + ST + H2 \tag{21}$$

According to **Lemma 1**, the host machine can access the disk at disk's maximum speed. ■

To obtain the maximum throughput, a video server has to keep inter-request delay under $TS + MA - H1 - SI - ST - H2$, that can be a critical factor for scheduling. When the condition does not hold, the response time is increased, and eventually the read-ahead chain will be broken. In consequence, the total throughput is reduced. The following formula is an approximation of the throughput that modifies Eq. (11) by multiplying a degradation factor.

$$TP_{appx} = \begin{cases} \frac{\text{track size}}{MA} * AJ * \frac{BC}{BC+1} & if \;\; BC \; > \; 0 \\ TP_{max} & if \;\; BC \; \leq \; 0 \end{cases} \tag{22}$$

where

$$BC = \frac{MA + TS}{IR_i - (TS + MA - H1 - SI - ST - H2)} \tag{23}$$

BC is the number of disk reads before the read-ahead chain is broken and $\frac{BC}{BC+1}$ is the degradation factor.

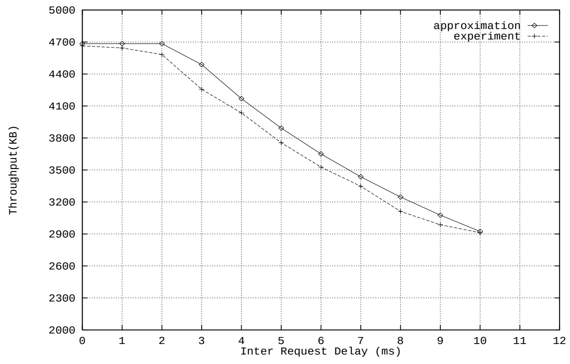

Figure 6: Throughput vs. Inter-Request Delay.

Example 3 In this example, accuracy of the approximation throughput function Eq.(22) is examined. The outcomes of approximation function are compared with experiments for delay from 0ms to 10ms, depicted in Figure 6. Each data point is the average from 1000 random tests. ■

4.3 Applications in Multimedia Storage Systems

There are decision support issues, such as performance prediction, file layout, and disk I/O scheduling of multimedia storage system can benefit from the proposed model. The model can offer an accurate prediction for most of the accesses. For instance, accessing 30 MB on HPC3323A disk continuously, the measurement from experiment is 6400ms. If using the average access time from the reference manual, the predicted response time is 9642ms, while it is 6520ms when applying this model. The reason that the prediction time from our model is higher than the measurement is that our model does not put bad sectors and other unexpected incidents (such as parity error) into account. The error rate is mostly within 5% which agrees to the findings in [8]. Another application of the accurate response time estimation is disk I/O scheduling that scheduling algorithms can be more effective when the cost of each request can be estimated accurately.

An important video playback design issue is the disk I/O planning problem of accessing a collection of tracks. Now assume tracks T_0, T_1, ..., T_N are to be read, where tracks T_i and T_{i+1} are not necessarily adjacent, and T_i is closer to the disk outer boundary than T_j, if i < j. The planner needs to partition tracks into groups, and issues one I/O request for a group of tracks. Having a model which can estimate the response times of all alternatives accurately, the planner can make the optimized plan.

Intuitively, there are at least two choices. The first approach is to read the target tracks one by one, and the other is access the target tracks together with the non-targeted tracks lying among them in one multi-tracks read. The access time for a contiguous N tracks (with read-ahead) can be derived as in Eq. (24), if Lemma 2 is satisfied.

$$AT_N = H1 + SI + DP + (N-1) * (MA + TS) + CT + ST + H2 \tag{24}$$

The initialization cost is only counted once, and for the other N-1 reads, the system may perform non-stop disk read. The time to read the N data tracks using a sequence of N data reads (i.e. one read command only reads one track) is in Eq. (25).

Example 4 A user process is to read track i and track i+3 that each track is 60K bytes

$$t_{p,i} = \begin{cases} t_{p,i} \text{ as in Eq. (6)} & if\ \ T_i\ and\ T_{i-1}\ are\ adjacent \\ t_{t,i-1} + TS_{i-1,i} + CT + ST + H2 & if\ \ t_{d,i} \le t_{t,i-1} + TS_{i-1,i} \\ t_{t,i-1} + TS_{i-1,i} + nr * (MA + TS) + CT + ST + H2 & if\ \ t_{d,i} > t_{t,i-1} + TS_{i-1,i} \end{cases} \quad (25)$$

where

$$\begin{aligned} seek(T_{i-1}, T_i) &= seeking\ time\ from\ track\ T_{i-1}\ to\ T_i \\ TS_{i-1,i} &= track\ and\ cylinder\ skews\ between\ tracks\ T_{i-1}\ and\ T_i \\ nr &= \left\lceil \frac{t_{p,i-1} + IR_{i-1} + H1 + SI + seek(T_{i-1}, T_i) - t_{t,i-1} - TS_{i-1,i}}{TS + MA} \right\rceil \\ t_{d,i} &= t_{p,i-1} + IR_{i-1} + H1 + SI + seek(T_{i-1}, T_i) \end{aligned}$$

Figure 7: The Equation of Access Time of N-Tracks Data Read.

long. The disk scheduler can issue a 4-tracks read to read in track i, track i+1, track i+2 and track i+3 in one I/O request, or to read track i and track i+3 separately. Assume initially disk head is on the target cylinder, and the host machine processing time H1 for 240KB is four times as for 60KB. Using the data in Table 1 (with BFR=80), Eq. (24) and Eq. (25), the elapse times of both approaches reading track i and track i+3 are listed in Table 3. The averages of the outcomes from experiments, conducted 1000 times randomly, are also listed in Table 3. ■

From the example, it can be found no plan is better than the other for all the cases. Every single parameter, such as revolution speed, seeking time, and inter-request interval may reverse the decision that shows the importance of an on-line algorithm for making the optimized plan. In Figure 8, the chart shows the derived response times from the model of reading 4 blocks in a read request and read i-th and (i+3)-th blocks separately under different revolution speeds and inter-request intervals.

Interleaving partitions of a file or several files is popular in the near-video-on-demand

Access Type	Model		Experiment
Read 4 tracks	64.26ms		66.07ms
Read track i and i+3	IR = 1ms	47.42ms	49.03ms
	IR = 10ms	60.00ms	62.12ms
	IR = 20ms	72.58ms	74.68ms

Table 3: One Multi-Track Read vs. Multiple Reads of Single Track.

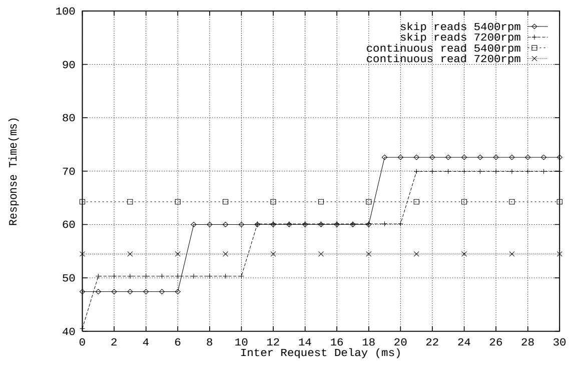

Figure 8: Response Times with Different Parameter Values

server design [5] in order to support many group of users. Generally, each of n files, f_1, f_2, ..., f_n, is partitioned into m elements, stored sequentially as $f_{1,1}$, $f_{2,1}$, ..., $f_{n,1}$, $f_{1,2}$, $f_{2,2}$, ..., $f_{1,m}$, ..., $f_{n,m}$ (that $f_{i,j}$ is the j-th element of the i-th file), and each group of users access to one file. Since video data is played in a high rate, disk access has to catch up the playing rate for every group. Moreover, it is normal that only some files are served at one time.

In the file layout design, decisions include the number of files, the number of partitions and the size of element with the constraint that the access time of any subset of $\{f_{1,j}$, ..., $f_{i,j}$, ..., $f_{n,j}\}$ should not exceed the time a video player consumes one file element, i.e. $f_{i,j}$. Eq. (24) and Eq. (25) can be used to verify the legitimacy of the selected parameters.

However, the algorithms for the decisions are beyond the scope of this paper.

5 Conclusion

In this paper, we present an accurate and useful, but not tedious, SCSI disk model for multimedia storage systems, and a methodology for acquiring the parameters. The decision supports of multimedia storage systems can utilize this performance model to predict performance, design file layout and I/O scheduling policy. Although the model is not claimed as accurate as the disk simulator in [8, 12], for its compactness it can be implemented in the disk driver for on-line decision supports. Currently, the model is implemented in the disk device driver to explore various file layouts and I/O schedule algorithms of a video server.

There are still many issues to be resolved. The first issue is to extend the performance model and acquisition approach to the environment with multiple disks and/or with multiple buses. The second issue is regarding the acquisition process. Although our initial intention is to develop a methodology for ordinary users, it still has to resort to device driver modification and use of special equipment, such as SCSI bus analyzer. It is hoped that SCSI disk and controller manufacturers can provide Vendor Unique Commands for programmers to extract needed parameters.

References

[1] Steven Berson, Richard Muntz, Shahram Ghandeharizadeh, and Xiangyu Ju. Staggered striping in multimedia information systems. In *Proc. ACM SIGMOD*, 1994.

[2] Meng Chang Chen, Jan-Ming Ho, Ming-Dat Ko, and Shie-Yuan Wang. A scsi disk model for video playback. Technical report, Institute of Information Science, Academia Sinica, 1995.

[3] D. James Gemmell, Jiawei Han, Richard Beaton, and Stavros Christodoulakis. Delay-sensitive multimedia on disks. *IEEE Multimedia Magazine*, Fall 1994.

[4] P. Lougher and D. Shepherd. The design of a storage server for continuous media. *The Computer Journal*, 36(1), 1993.

[5] Yen-Jen Oyang, Meng-Huang Lee, Chun-Hung Wen, and Chih-Yuan Cheng. Design of multimedia storage systems for on-demand playback. In *Proc. 11th International Conference on Data Engineering*, 1995.

[6] P. Venkat Rangan and Harrick M. Vin. Efficient storage techniques for digital continuous multimedia. *IEEE Transactions on Knowledge and Data Engineering*, 5(4), August 1993.

[7] P. Venkat Rangan and Harrick M. Vin. Designing an on-demand multimedia service. *IEEE Communications Magazine*, July 1992.

[8] Chris Ruemmler and John Wilkes. An introduction to disk drive modeling. *IEEE Computer Magazine*, March 1994.

[9] Bernhard Seeger, Per-Ake Larson, and Ron McFadyen. Reading a set of disk pages. In *Proc. VLDB*, 1993.

[10] W. D. Sincoskie. System architecture for a large video on demand service. *Computer Networks and ISDN Systems*, (22), 1991.

[11] Fouad Tobagi, Joseph Pang, Randall Baird, and Mark Gang. Streaming raid^{TM}–a disk array management system for video files. In *Proc. First ACM International Conference on Multimedia*, 1993.

[12] Bruce Worthington, Gragory Ganger, Yale Patt, and John Wilkes. On-line extraction of scsi disk drive parameters. In *Proc. ACM SIGMETRICS*, 1995.

[13] ANSI X3T9.2. *Small Computer System Interface-2.* 1993. Draft Revision 10k.

[14] Philip Yu, Mon-Song Chen, and Dilip Kandlur. Design and analysis of a grouped sweeping scheme for multimedia storage management. In *Proc. Third International Workshop on Network and Operating System Support for Audio and Video*, 1992.