Impact of Media Exchanges in Robotic Storage Libraries*

Sunil Prabhakar Divyakant Agrawal Amr El Abbadi

Abstract

The large storage requirements of many commerical and scientific applications cannot be met by magnetic disks due to their high cost and low storage density. Consequently, cheaper and more dense tertiary storage systems are being intergrated into the storage hierarchies of these applications. Although tertiary storage can accomodate large amounts of data, the access latency is very high due to the need to load and unload media from the read/write drives. These media exchanges are very slow with typical times in the order of tens of seconds. In order to reduce this high latency research efforts have focussed on minimizing media exchanges under the assumption that it is always beneficial to eliminate exchanges. We analyze the validity of this assumption. It is shown that there are instances when the assumption does not hold. It is further shown that various factors influence the effect of an exchange, making it difficult to establish simple heuristics for scheduling media exchanges. Simulation is used to validate the theoretical results.

1 Introduction

Many commercial and scientific applications such as digital libraries, data warehousing, data mining, multimedia applications etc., have very large data storage requirements. Due to the high cost and low storage density (bytes/volume) of magnetic disk storage, it is not feasible for these applications to store data solely on magnetic disks. Tertiary storage systems are two orders of magnitude cheaper and denser than disk systems with equivalent capacity. It is therefore no surprise that many data intensive applications are incorporating tertiary storage into their storage hierarchies. In the current commercial arena, there are two prevalent technologies – magnetic tapes and optical (or magneto-optical) disks. Throughout this paper we will use the terms *Medium* and *Media* to refer to a single or multiple removable units of tertiary storage such as a tape(s) or optical disk(s).

The performance characteristics of these devices are very different from those of magnetic disks. Their cost advantage over magnetic disks comes from the fact that they are removable media and therefore large numbers of tapes or disks share the same drive(s). However, this feature is also responsible for their major drawback because in order to perform I/O on a tape or optical disk, it is necessary to first load the medium onto the drive. Since this involves the physical movement of the tape or disk, it is a slow operation. In order to reduce this time, automatic tape and disk libraries (e.g. [EXA96, SON96, Qua96]) which operate much faster than humans, are commercially available. Typical exchange times for these automatic libraries are in

*Work supported by a research grant from NSF/ARPA/NASA IRI9411330.

tens of seconds. The access latency of tertiary storage can be very high due to these large media exchange times whereas magnetic disks involve no media exchanges. Another significant difference is that magnetic tape, the leading tertiary technology, offers only sequential access as opposed to both magnetic and optical disks. Seek delays for tapes can be as high as a few minutes in contrast to hundreds of milliseconds for disks.

Due to the high cost of media exchanges, reducing the number of media exchanges promises to be an effective mechanism to reduce the latency. Reordering of I/O requests or intelligent scheduling of pending requests have been proposed for achieving this reduction. Most of the solutions are applicable only to specific applications such as video-on-demand [CPK95, KDST95], relational databases [ML95, SS93], and multidimensional arrays [SS94, CDK$^+$95]. Generally applicable techniques for reducing the access latency include scheduling of I/O requests for a single medium [HS96, LO96, ABZ96] and altering the order in which media are loaded onto the drives [Ano96].

Given the magnitude of the exchange time, it is generally assumed that reducing the number of media exchanges will always be beneficial. However, this assumption is not always valid, since there are instances where having extra media exchanges results in better performance. In particular, for technologies such as tapes which can have seek or transfer times greater than exchange times the assumption may be invalid. It is interesting to note that most of the reseach work on tertiary storage has focussed on tapes. In this paper we conduct a complete study of the impact of media exchanges in a non-application-specific setting. Given a batch of I/O requests for a robotic library, we would like to determine whether schedules that minimize the media exchanges perform better than schedules that allow more exchanges to be made and if not, under which conditions do they not perform better. In contrast to previous studies [HS96, LO96], our study considers requests for more than a single medium.

The rest of the paper is organized as follows. In Section 2 we summarize earlier work done on finding optimal schedules that minimize the number of media exchanges. In Section 3 an analytic evaluation of introducing extra media exchanges is presented. Section 4 presents and analyses results of experiments conducted to validate our analysis. Section 5 concludes the paper.

2 Scheduling With Minimum Media Exchanges

In this section, we summarize earlier results of I/O scheduling to minimize the number of media exchanges. The technique of scheduling batches of I/O requests is based upon the fact that I/O traffic is bursty in nature [ABZ96, RW93], i.e., I/O requests are generated in a short period of time followed by periods during which little or no reqeusts are generated. This technique has also been used for scheduling batch I/O requests for single media such as linear tapes [LO96], serpentine tapes [HS96] and magnetic disks [ABZ96]. Scheduling of bursty I/O requests can be simplified by considering the requests in each burst independently. It can be assumed that these requests arrive together, and thus the problem can be viewed as off-line scheduling with requests arriving together rather than on-line scheduling with requests arriving continuously. This technique has also been used for studying other scheduling algorithms [HS96, LO96, ABZ96]. Given the current improvements in processor speeds and the high access latency of tertiary storage devices, it is tempting to consider an exhaustive search of all possible schedules in order to obtain an optimal solution instead of looking for efficient scheduling policies. This approach however, is feasible only for very small numbers (< 15) of media and requests due to the exponential number of options that need to be considered. In our experiments, it took almost half an hour to do an exhaustive search for an optimal schedule for only 12 requests. We now present a description of the problem, followed by our proposed solution for scheduling with minimum media scheduling.

2.1 Problem Statement

We begin with a formal description of the problem of scheduling I/O requests for robotic storage libraries.

Given a set of media, S, a number of drives, d, a number of robotic arms, A, a set of requests, R, each request $r \in R$ consisting of a medium number, a start block, a number of blocks and a read/write flag, find a schedule for servicing these requests such that the average waiting time for all requests is minimized.

It is assumed that requests do not span more than one medium. Though techniques such as striping could be used to split data items across several media, it has been shown that due to the small number of drives in tertiary libraries, striping is not beneficial in the presence of concurrent users [GM95, DK93]. If a data item is too large to fit on a single medium, we can split a request for such a large item into separate requests for each contiguous segment.

In order to compare the efficiency of potential scheduling policies, it is necessary to define a quantitative measure of performance. This measure, or optimization function, could be one of several possibilities. Some of the popular optimization functions that have been studied in the literature include [Pin95, MP93]:

- *Makespan:* – the time at which the last request is completed.
- *TWCT:* – the Total Weighted Completion Time (each job has a weight, e.g. a priority).
- *AWT:* – the Average Waiting Time for all jobs.
- *Throughput:* – the average number of requests serviced per second.

Since the major concern for tertiary storage performance is the high latency, reducing the amount of time that a given request has to wait before it is serviced is important. We have therefore focused on minimizing the average waiting time over all the requests as our optimization function. The waiting time for a request is taken to be the time from the arrival of the request to the beginning of the data transfer for the request. This includes the time to perform necessary seek operations. Note that for a given set of requests, the average waiting time is equal to the total waiting time for all requests divided by the number of requests. Thus minimizing the total waiting time is the same as minimizing the average waiting time.

Clearly, if the set of requests to be scheduled are located on m different media, then in order to service all requests it is necessary to load each medium at least once. Therefore, we see that the lower bound on the number of media exchanges is m (assuming that none of the requested media is initially loaded on the drive). Thus any schedule that services all the requests for a medium once it is loaded, before unloading it will incur a minimum number of exchanges (m). The problem now simplifies to finding two sub-schedules: a *Media Loading* schedule to determine the order in which to load the different media and a *Single Medium* schedule to determine the order in which to process all the requests on a loaded medium. The goal of both these schedules is to minimize the total waiting time.

The problem of finding efficient *Single Medium* schedules has already been addressed by a number of researchers. Hillyer and Silberschatz [HS96] have looked at efficient methods of scheduling a batch of requests for a single serpentine tape. Li and Orji [LO96] have studied the problem of scheduling a batch of request for linear tapes. Scheduling algorithms for magnetic disks have been extensively studied [Teo72] and these could readily be applied to optical and magneto-optical disks. Any of these scheduling algorithms can be used to determine the *Single Medium* schedule for each medium.

The problem of finding the *Media Loading* schedules to determine the optimal order in which to load the media onto the drive has not been studied. In [Ano96] we investigated this problem and developed an $O(m \log m + N)$ (m is the number of media and N is the number of requests) algorithm to find the optimal schedule for loading the media for a single drive. The problem of finding an optimal media schedule for multiple drives was shown to be NP-complete. The algorithm obtained for the single drive was shown to be a very effective heuristic for the multiple drive problem as well. We now present a summary of these results based upon which we will investigate the benefits of introducing extra media exchanges.

2.2 Media Schedules on Single Drive Libraries

For a single drive, given a schedule of loading the media onto the drive, the total waiting time seen by all requests, W, can be expressed as:

$$W = \sum_{i=1}^{m} C_i + \sum_{i=1}^{m} E \cdot i \cdot n_i + \sum_{i=1}^{m} \left(n_i \sum_{k=1}^{i-1} P_k \right) \tag{1}$$

where,

- m is the number of media for which requests have been made,
- E is the exchange time,
- n_i is the number of request for the ith medium in the schedule,
- C_i is a constant for the ith medium in the schedule,
- P_k is a constant for the kth medium in the schedule.

The media exchange time, E, consists of the time required for the drive to unload the currently loaded medium, the robot to take up this medium and put it on the shelf, then move to the location where the new medium is placed, pick up the new medium and place it on the drive and finally for the drive to load the new medium. We assume that this time is constant as given in most product specifications. The details of the derivation can be found in [Ano96]. The constant C_i represents the wait incurred by the requests for medium i after it has been loaded onto the drive. This time is independent of when the medium gets loaded and does not depend upon the schedule. The second factor in the above equation, $\sum_{i=1}^{m} E \cdot i \cdot n_i$, represents the wait incurred by each request for medium i due to the exchanging of media that are loaded before it. This time depends only on the number of media that are loaded before and not on which media are loaded before. The third factor, $n_i \sum_{k=1}^{i-1} P_k$, is the wait incurred by the requests for medium i due to the processing of the requests for media that are loaded earlier. P_k is the total time required to process the requests of medium k after the medium has been loaded onto the drive. The value of this time is dependent upon the schedule used to service requests for a single medium. We do not restrict this schedule, any efficient schedule such as those described in [HS96, LO96] for tapes or in [ABZ96, Teo72] for disks could be used. We only require an estimate of the total processing time, P_k.

The total waiting time can therefore be rewritten as:

$$W = \sum_{i=1}^{m} \left(C_i + E \cdot i \cdot n_i + n_i \sum_{j=1}^{i-1} P_j \right) \tag{2}$$

Based upon the above equation, the following theorem was established:

Theorem 1 *Any schedule of m media which satisfies Equation 3 is optimal*

$$\forall i \;\; 1 \leq i < m, \;\; \frac{n_i}{E + P_i} \geq \frac{n_{i+1}}{E + P_{i+1}} \tag{3}$$

The proof of this theorem is presented in [Ano96]. Thus we see that for a single drive, the problem of finding an optimal schedule for loading the media onto the drive that minimizes the number of media exchanges, reduces to the problem of sorting the media in the order of non-increasing ratio of the number of requests to the sum of the exchange time and the processing time for each medium. We call this the OPT algorithm. It is important to note that no restriction has been placed on the value of the processing times P_i. We only need to know a good estimate of this value to compute an optimal schedule for loading the media.

In the case of multiple drives, the problem of finding an optimal schedule for loading the media onto the drives that minimizes the number of media exchanges was shown to be NP-complete in [Ano96]. In the absence of efficient optimal solutions, heuristics were proposed. It was shown that the algorithm described above, OPT, which schedules media in order of non-increasing ratio of the number of requests to the sum of the exchange time and the processing time for each medium is an effective heuristic for multiple drives.

Experiments were also conducted to assess the effectiveness of scheduling to minimize media exchanges. It was found that the OPT algorithm achieves 85% to 94% savings over a simple first-come-first-serve scheduling policy. The gains from reducing media exchanges are significant.

3 Impact of Media Exchanges

In the last section we saw that minimizing media exchanges results in large savings. The next question that arises is whether eliminating unnecessary exchanges is always beneficial? We expect that for certain media such as optical disks which have very small seek times compared to exchange times, the answer to the question will be yes. On the other hand, for media such as tapes, for which the exchange time can be smaller than the seek or transfer times, the answer is not obvious. Consider the following example.

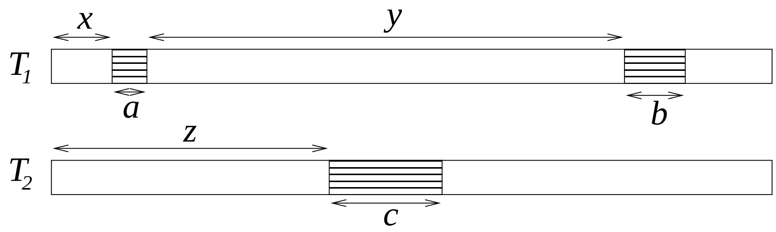

Figure 1: Layout of Requests on Tapes (Not drawn to Scale)

Example Consider only two linear tapes, T_1 and T_2 with three requests of lengths a, b and c bytes as shown in Figure 1. The separation of the requests is given by x, y and z in bytes. If we consider minimal exchanges, then each tape is loaded once and its requests are serviced. The optimal order for loading the two tapes is given by the algorithm OPT described in Section 2, i.e. load tapes in non-increasing order of the ratio of number of requests to the sum of the exchange time and the processing time. Let us assume that the ratio for T_1 is higher than that for T_2, i.e.

$$\frac{n_1}{E+P_1} \geq \frac{n_2}{E+P_2} \tag{4}$$

where P_1 and P_2 are the processing times for the two tapes and E is the exchange time. We assume that the drive head is located at the beginning of the tape (i.e. at the left most end) just after the tape is loaded onto the drive. It is clear that the optimal order for processing a and b is to seek to a, service it and then to seek to b, service B, followed by rewinding to the beginning of the tape so that it can be offloaded.

Request c has to wait for the processing of both a and b and the rewinding of tape T_1. If the gap between a and b is very large, then the time spent in seeking to reach b and then rewinding from the end of b to the beginning of the tape will be very large. If this time is large compared to the exchange time, then it is possible that processing c just after a and then processing b could result in an overall lower waiting time. Let us consider the requirements for such gains. The following notation will be used:

- v is the overhead for seeking or rewinding on the tape in seconds,
- s is the seek rate in bytes/second,
- r is the rewind rate in bytes/second,
- t is the transfer rate in bytes/second

The seek and rewind rates are modeled as a constant startup time followed by a linear search rate. The total waiting time for the three items when processed in the order a, b, c is given by W_1:

$$\begin{aligned}
\text{Waiting time for } a = \quad & E+v+\tfrac{x}{s} \\
\text{Waiting time for } b = \quad & E+v+\tfrac{x}{s}+\tfrac{a}{t}+v+\tfrac{y}{s} \\
\text{Waiting time for } c = \quad & E+v+\tfrac{x}{s}+\tfrac{a}{t}+v+\tfrac{y}{s}+\tfrac{b}{t}+v+\tfrac{x+y+a+b}{r}+E+v+\tfrac{z}{s} \\
\text{Total Waiting time } W_1 = \quad & 4E+7v+x(\tfrac{3}{s}+\tfrac{1}{r})+a(\tfrac{2}{t}+\tfrac{1}{r})+y(\tfrac{2}{s}+\tfrac{1}{r})+b(\tfrac{1}{t}+\tfrac{1}{r})+\tfrac{z}{s}
\end{aligned}$$

If we exchange T_1 after processing a, then the total waiting time for the three items when processed in the order a, c, b is given by W_2. By an analysis similar to the one above, we get:

$$\text{Total Waiting time } W_2 = \quad 6E+9v+x(\tfrac{4}{s}+\tfrac{2}{r})+a(\tfrac{2}{t}+\tfrac{1}{s}+\tfrac{2}{r})+z(\tfrac{2}{s}+\tfrac{1}{r})+c(\tfrac{1}{t}+\tfrac{1}{r})+\tfrac{y}{s}$$

In order to have a lower waiting time after making the extra exchange, we need $W_1 > W_2$, which gives us the following result:

$$y > 2(E+v)\frac{sr}{s+r}+x+z+a+(c-b)\frac{s(t+r)}{t(s+r)} \tag{5}$$

Therefore, in order for the extra exchange to be beneficial, the value of y must be large. If we assume that $s=r$ then y must be at least as large as the number of bytes that the drive can seek in the time it takes to exchange a medium. Can y be arbitrarily large in practice? The value of y is limited by two factors. The first is the length of the tape, the second factor arises from the requirement of Equation 4. If this equation is not satisfied, then the optimal schedule would be to process tape T_2 before T_1. In which case the question of an extra exchange does not make sense. Therefore, Equation 4 requires that $\frac{n_1}{E+P_1} \geq \frac{n_2}{E+P_2}$, i.e.:

$$\frac{2}{E+v+\frac{x}{s}+\frac{a}{t}+v+\frac{y}{s}+\frac{b}{t}+v+\frac{x+y+a+b}{r}} \geq \frac{1}{E+v+\frac{z}{s}+\frac{c}{t}+v+\frac{z+c}{r}}$$

$$y \leq (E+v)\frac{sr}{s+r} + 2z - x + (2c - a - b)\frac{s(t+r)}{t(s+r)} \tag{6}$$

Equation 6 sets an upper limit on the value of y. Let us now substitute actual values for the various parameters in order to construct an example where exchanging is beneficial. Using the parameters specified for a commercial tape library (Exabyte EXB-480) with a single drive (Exabyte EXB Mammoth), we set the values as shown in Table 1. By substituting the values into the above formulae, we see that Equation 4 requires that $y \leq 16010MB$ which is the case. The total waiting time without the extra exchange, $W_1 = 416.02$, and the total waiting time with the extra exchange, $W_2 = 405.72$. Therefore we see that indeed there is a reduction in the total waiting time by introducing an extra medium exchange. Note however, that the gain is not very high (2.47% reduction), and the value of y needs to be in a small range ($14020 < y < 16010$).

Parameter	*Value*
v	0.1 seconds
E	40 seconds
$s = r$	200 MB/s
t	10 MB/s
$a = b = c$	1 MB
x	0 MB
y	15000 MB
z	6000 MB

Table 1: Table of values

Motivated by the above example which shows us that introducing extra media exchanges can be beneficial, we now analyze the conditions under which exchanging will be beneficial in general. We will focus on a single drive. Given a set of requests, the OPT algorithm is used to determine an optimal sequence with minimum exchanging as described in the previous section. The schedule for the requests on a single medium after the medium is loaded is left unspecified. As mentioned earlier, any of the existing schedules can be used. We require to know only the sequence in which the requests will be serviced and the total processing time. Let us now consider the benefit of exchanging the ith medium in this sequence after the jth request for the medium has been serviced (instead of servicing the $j+1$th request, we offload the medium). The remaining requests for this medium will be serviced at a later time when the medium is reloaded. Other media could be loaded and serviced completely before the exchanged medium is reloaded. Any of the m media could be split in this manner. Figure 2(a) shows the optimal sequence of the media before the split. Let M be the medium that we choose to offload and reload. In general there will be some set of media A that precede M and some set B that follow M in the optimal minimum exchange schedule generated by OPT. The offloading and reloading of M can be viewed as the replacement of medium M by two media, M_1 and M_2, where M_1 contains the data for the first j requests of M and M_2 contains the data for the rest of the requests. The locations of the requested data on M_1 and M_2 are the same as the locations of the respective data on M.

Having split tape M into two tapes, we can now look for the optimal sequence of this new set of tapes using the OPT algorithm. This new order would give us the best schedule for the tapes if we allowed tape M to be exchanged after its jth request. Since the optimal order is based upon the ratio for each tape, the tapes in the sets A and B will remain in the same relative order as before since their ratios are unchanged.

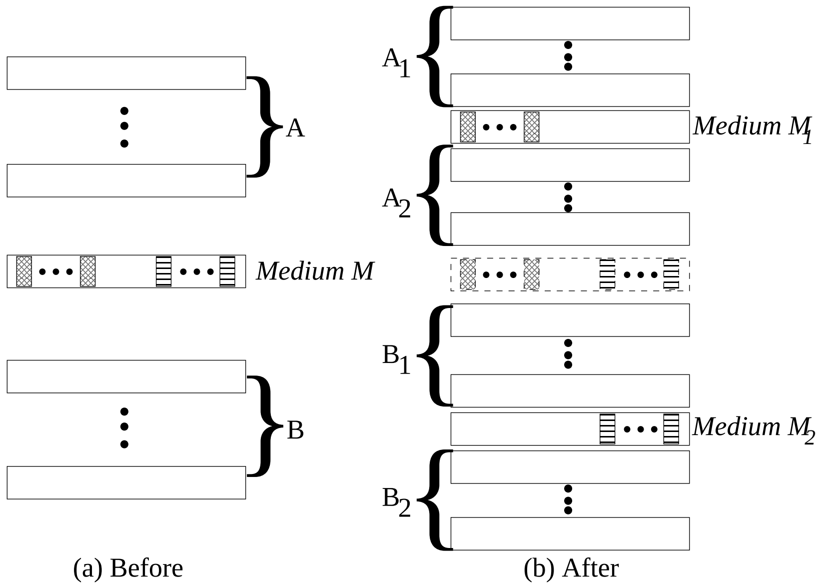

Figure 2: Optimal sequence of Tapes (a) Before Split (b) After Split

Tapes M_1 and M_2 will be inserted into the schedule depending upon their ratios. Due to the large variability in the relative values of the various tertiary technology parameters such as, exchange time, seek and rewind rates and overheads, transfer rates, it is possible that the ratios of the three tapes M, M_1 and M_2 could have any of the six possible orderings. For example, M_1 could have the highest ratio, followed by M, and M_2 could have the lowest ratio, or M_2 could have the highest ratio and M could have the lowest. In order to demonstrate the complexity of the factors involved, we will analyze only a single representative case. Since we are interested in introducing extra exchanges only for the purpose of reducing the average waiting time, we choose to analyze the case that we expect to be most likely to result in reduction of total waiting time. This case is shown in Figure 2(b). The order of non-increasing ratios for the three tapes is M_1, M and M_2. Since M_1 has a higher ratio than M, it is possible that it moves ahead of some of the tapes in the set A. Thus set A is now split into two sets, A_1 which is scheduled before M_1, and A_2 which is scheduled after M_1. Similarly, M_2 has a ratio that is lower than M, therefore it is possible that it gets scheduled after some of the tapes in set B. Thus set B is now split into two sets, B_1 which is scheduled before M_2 and B_2 which is scheduled after M_2. It should be noted that any of the four sets of tapes A_1, A_2, B_1 and B_2 could be empty as long as $A = A_1 \bigcup A_2$ and $B = B_1 \bigcup B_2$. Note that medium M, shown dotted in Figure 2(b), is not present in that schedule - it is only shown to indicate the division between A_2 and B_1. Also, M_1 and M_2 are the same medium (M) being loaded at different times in the same schedule, they are not independent media.

Let us now consider the effect of this split of medium M on the total waiting time. The requests on the media in set A_1 will see no change in their waiting times. Each request on the media in set A_2 will experience an increase in waiting time of an extra exchange and the processing time of medium M_1, i.e. $E + P_{M_1}$. The change in waiting time for each request can be analyzed similarly. More formally, using n_x to denote the number of request in set x, P_x to denote the total processing time required for all requests in set x and N_x to denote the number of media in set x and ΔW_x to denote the total change in the waiting time for all requests in set x:

$$\begin{aligned}
\Delta W_{M_1} &= -n_{M_1}(N_{A_2}E + P_{A_2}) \\
\Delta W_{A_2} &= n_{A_2}(E + P_{M_1}) \\
\Delta W_{B_1} &= n_{B_1}(P_{M_1} - P_M) \\
\Delta W_{M_2} &= n_{M_2}(N_{B_1}E + P_{B_1} + E + P_{M_1} + X_2 - X) \\
\Delta W_{B_2} &= n_{B_2}(E + P_{M_1} + P_{M_2} - P_M)
\end{aligned}$$

where X_2 is the time required to seek from the start of M_2 to the first request on M_2 and X is the time required to seek from the end of the jth request on M to the beginning of the $j+1$th request.

In order for the extra exchange to be beneficial, the total change in waiting time should be negative. Thus from the above equations, we get:

$$\begin{aligned}
E(n_{A_2} + n_{B_2} + n_{M_2}N_{B_1} + n_{M_2}) + P_{M_1}(n_{A_2} + n_{B_1} + n_{B_2} + n_{M_2}) + P_{M_2}n_{B_2} + n_{M_2}P_{B_1} + n_{M_2}X_2 \\
< n_{M_1}N_{A_2}E + P_M(n_{B_1} + n_{B_2}) + n_{M_1}P_{A_2} + n_{M_2}X \qquad (7)
\end{aligned}$$

Also, because of the order in which the media are scheduled by OPT, there are limitations on the value of their ratios. In particular:

$$\left.\frac{n_i}{E+P_i}\right|_{i\in A_1} \geq \frac{n_{M_1}}{E+P_{M_1}} \geq \left.\frac{n_i}{E+P_i}\right|_{i\in A_2} \geq \frac{n_M}{E+P_M} \geq \left.\frac{n_i}{E+P_i}\right|_{i\in B_1} \geq \frac{n_{M_2}}{E+P_{M_2}} \geq \left.\frac{n_i}{E+P_i}\right|_{i\in B_2} \qquad (8)$$

For the split to be beneficial, Equations 7 and 8 need to be satisfied. Equation 8 sets limits on the relative vales of the parameters and Equation 7 requires that:

- n_{M_1} should be large, P_{M_1} should be small,
- N_{A_2} should be large, n_{A_2} should be small, P_{A_2} should be large,
- N_{B_1} should be small, n_{B_1} should probably be large, P_{B_1} should be small,
- n_{M_2} should probably be small, P_{M_2} should be small, and
- n_{B_2} should be small

Some of the factors that determine whether a switch will be beneficial or not depend upon requests for other media, which makes it difficult to determine when an exchange will be effective. Therefore, it is not obvious when the processing of the next request for the loaded medium should be postponed and requests of other media be processed. In particular, this decision is dependent upon the other media in the schedule. Similar expressions can also be derived for the other possible orderings of M, M_1 and M_2. Due to the large number of variables, the analysis is complex and we expect that simple heuristics will not be easy to find. Therefore, we choose to approach the problem through simulation. In the next section, we investigate experimentally the advantages and disadvantages of media exchanges.

4 Experimental Results

From the preceding analysis, we see that there are many factors which determine whether exchanging a medium before completing all requests will result in an increase or decrease of the total waiting time. In this section we use simulation to evaluate the impact of such media exchanges.

4.1 Experimental Setup

The experiments were conducted using an event driven simulator for robotic libraries developed using the CSIM [Sch86] simulation package. A model of a general robotic library consisting of a configurable number of drives, robot arms and media has been developed. The performance of the drives and robot arms and the characteristics of the media are also configurable. Table 2 lists some of the parameters that can be adjusted for the model. The model mimics an actual robotic library. The drives operate independently of each other, except that they share the robot arms to pick and put media.

Parameter	*Meaning*
NDRIVES	Number of Drives
NROBOTS	Number of Robotic arms
NUM_MEDIA	Number of media in the library
NUM_REQ	Number of requests for each experiment
PICK_TIME	Time required by robot to remove medium from drive
PUT_TIME	Time required by robot to place medium on drive
MOVE_TIME	Time required by robot to move between media on shelf
SEEK_OVHD (REW_OVHD)	Constant overhead for seek(rewind) (sec)
SEEK_RATE (REW_RATE)	Rate at which the drive can seek(rewind) (MB/s)
XFER_SPEED	Transfer rate of the drive (MB/s)
REQ_SIZE	Size of a single request (KBytes)
EJECT_TIME	Time required by the drive to eject a medium (sec)
LOAD_TIME	Time required by the drive to load a medium (sec)

Table 2: Table of Model Parameters

Experiments were conducted using a single drive setting only. Two sets of input request patterns were used for our experiments. The first set consisted of requests distributed randomly across the media in the library. The second set consisted of a skewed "hot-cold" distribution where 80% of the media were accessed 20% of the time and the remaining 20% of the media were accessed 80% of the time. For each experiment, the number of media was varied from 10 to 100, and the number of requests was always set to be ten times the number of media, e.g. for 20 media, we generated 200 requests. Each experiment was repeated thrice using different random number seeds for the request generator. Due to the lack of space, not all results are presented, only representative results are shown and discussed in the paper.

4.2 Validation of Hypothesis

Based upon the large value of the exchange time as compared to the average seek and transfer times for optical and magneto-optical disks, we expect that introducing extra exchanges will not be beneficial. Several experiments were conducted for a typical optical disk juke box (SONY OSL-2001 with SMO F541 drives). The values of the various parameters used are given in Table 3. As expected, in all our experiments we found that introducing an extra exchange was always harmful. The effect of introducing an exchange was to increase the average waiting time. The percentage increase varied depending upon the number of disks and requests, being greater for smaller numbers. The increase was greater for the hot-cold distributions than for the random distributions. We found that for 10 disks with the hot-cold distribution, an extra exchange could result in as much as 23% increase in the average waiting time for all requests. Figure 3(a) shows the results

Parameter	*OPTICAL DISKS*	*TAPES*
NDRIVES	1	1
NROBOTS	1	1
NUM_MEDIA	10 - 100	10 - 100
NUM_REQ	100 - 1000	100 - 1000
PICK_TIME	1 sec	10 sec
PUT_TIME	1 sec	10 sec
MOVE_TIME	1 sec	2 sec
SEEK_OVHD (REW_OVHD)	0.0083 (0.0083) sec	0.1 (0.1) sec
SEEK_RATE (REW_RATE)	103 (103) MB/s	193 (188) MB/s
XFER_SPEED	2.0 MB/s	3.0 MB/s
REQ_SIZE	100 KBytes	2560 KBytes
EJECT_TIME	1 sec	8 sec
LOAD_TIME	2 sec	10 sec

Table 3: Parameter values used for the experiments

for the case of 20 disks with a hot-cold distribution and Figure 3(b) shows the results for the case of 20 disks with a random distribution of requests. In each experiment the effect of introducing an exchange after each request of each medium (except the last request to be processed on the medium) was evaluated. The x-axis of the plots shows the request number following which the exchange was introduced. The requests were totally ordered by ordering the disks. For each disk the order was chosen to be the order of block numbers. The y-axis shows the percentage increase in the total waiting time as a result of the extra exchange. Each peak in the plots represents the requests for one disk. Thus for the optical disk, introducing extra exchanges is not beneficial. The schedules developed by the OPT algorithm for optical disks are therefore optimal.

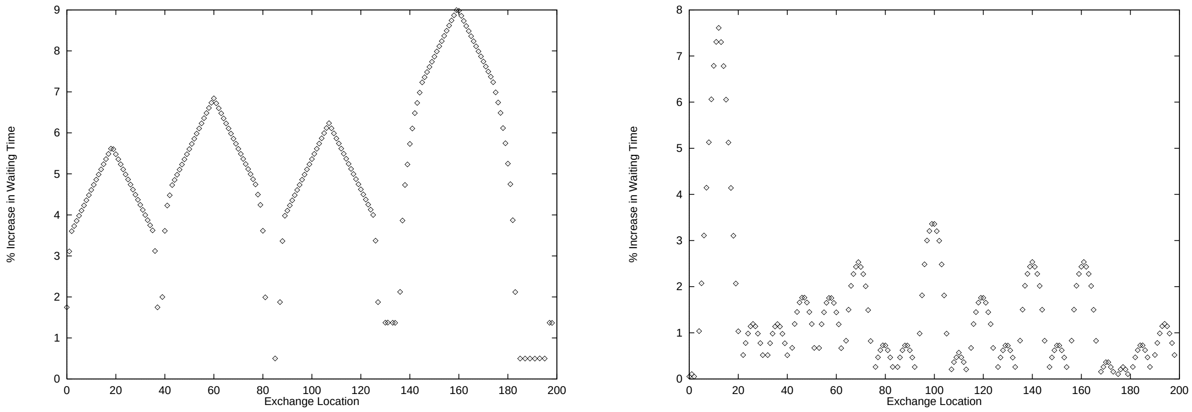

Figure 3: Plot for (a) 20 Disks with Hot-Cold Requests and (b) 20 Disks with Random Requests

For tapes however, introducing exchanges did result in improvements over the minimal exchanging schedules. Figure 4 shows the results for experiments based upon the data for an Exabyte EXB-480 library and EXB Mammoth drive. The values of the parameters used are given in Table 3. Figure 4(a) shows the results of 20 tapes with a hot-cold distribution and Figure 4(b) shows the results for 20 tapes with a random distribution. Although gains are observed, they are very small (less than 0.6%) and occur only for a few cases. A natural question to ask is how one could identify exchange points that would result in reduction of the total waiting time.

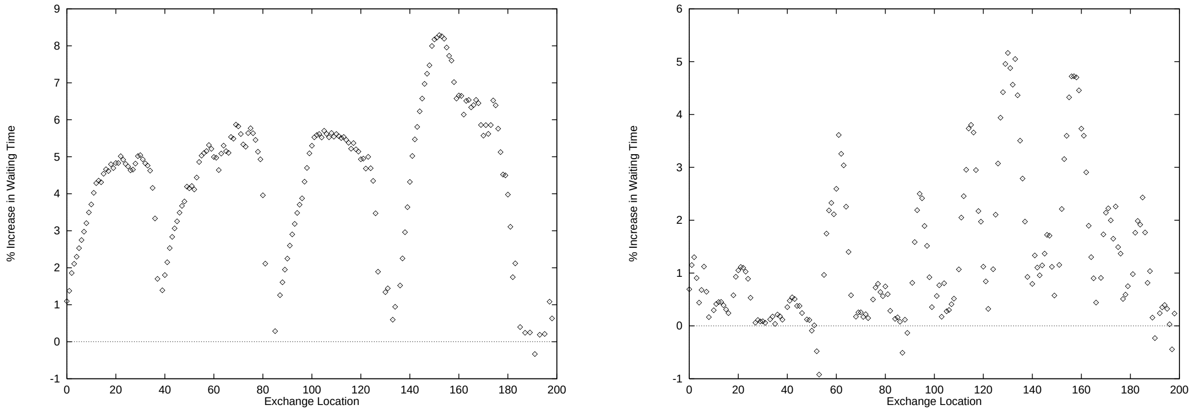

Figure 4: Plot for (a) 20 Tapes with Hot-Cold Requests and (b) 20 Tapes with Random Requests

As pointed out earlier, we expect that reductions will be seen when the gap between the location of the request after which the exchange is made and the location of the next request on the same tape is large. To test this hypothesis, in Figure 5, we plot the percentage increase in waiting time versus the distance (in blocks) between the exchange location and the next request. The performance with the hot-cold distribution fits the hypothesis, whereas the performance with the random distribution does not. With the hot-cold distribution, exchanging is not beneficial when the gap is small, but as the gap gets larger, exchanging begins to be useful. This behavior was observed for all experiments with the hot-cold distribution. With the random distribution however, there are instances where exchanging with even a small gap results in reduction of the waiting time. This is due to the influence of the other media in the schedule. With a hot-cold distribution, most of the requests are located on a few tapes and there are tapes with very few requests. Since the dominant factor in the ratio used by OPT is the number of requests, we see that the schedules have the hot tapes loaded before the cold ones. For the random distribution, the number of requests per tape is more uniform and therefore even the tapes at the tail end of the schedule have large numbers of requests. Using the notation of Section 3, we see that the location of M_2 in the schedule after splitting will be determined by its ratio, which will also be largely governed by the number of requests. With the random distribution, the number of requests in M_2 will usually be smaller than the number of requests in the rest of the tapes. Consequently, it will often be scheduled last, i.e. the set B_2 will be empty. With a hot-cold distribution, the set B_2 will usually be non-empty. This factor makes it more likely that extra exchanges will be more beneficial with the random distribution than for the hot-cold distribution.

It should be noted that reduction in the total waiting time occurs rarely and when it does occur, the magnitude of the improvement is not large. On the other hand, most of the extra exchanges lead to increases in the total waiting time, the magnitude of which can be very large (up to 23%). By eliminating unnecessary exchanges, the OPT algorithm was able to achieve significant improvements over first-come-first-serve scheduling, whereas, the gains of introducing extra exchanges are not significant. In fact, for some technologies, such as disks, there are no gains. Due to the many factors involved, it is not easy to determine a criterion that determines when an extra exchange should be introduced.

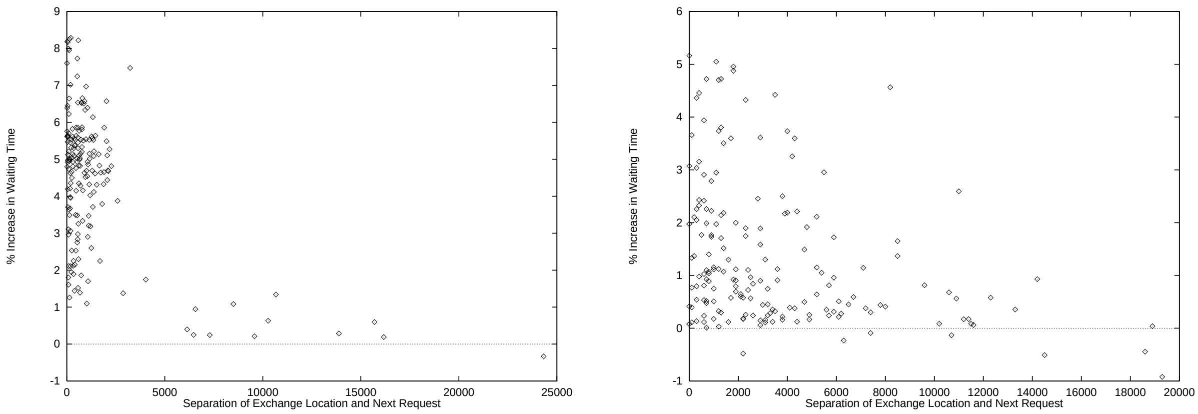

Figure 5: Plot for (a) 20 Tapes with Hot-Cold Requests and (b) 20 Tapes with Random Requests

5 Conclusion

We have investigated the impact of media exchanges on the performance of automatic libraries. For certain technologies, such as optical and magneto-optical disks, we showed that media exchanges result in increased waiting times and should therefore be minimized. Optimal schedules for such technologies can therefore be obtained by focussing only on scheduling algorithms that service all requests for a loaded medium before offloading it. For technologies such as tapes, which have seek and transfer times comparable to the switch time, media exchanges can be beneficial. The conditions under which media exchanging results in gains do not occur very often. There are many factors which govern the impact of a media exchange and therefore it is not easy to establish simple heuristics for predicting the impact of an exchange. Moreover, the magnitude of the gains is not large.

References

[ABZ96] M. Andrews, M. A. Bender, and L. Zhang. New algorithms for the disk scheduling problem. In *Foundations of Computer Science*, 1996.

[Ano96] Anonymous. Efficient I/O scheduling in tertiary libraries. Technical Report TRCS96-26, Dept. of Computer Science (Details omitted to facilitate blind review), 1996.

[CDK+95] L. T. Chen, R. Drach, M. Keating, S. Louise, D. Rotem, and A. Shoshani. Efficient organization and access of multi-dimensional datasets on tertiary storage systems. In *Information Systems*, volume 20, pages 155–83. Elsevier Science, 1995.

[CPK95] A. L. Chervenak, D. A. Patterson, and R. H. Katz. Storage systems for movies-on-demand video servers. In *Proceedings of the Fourteenth IEEE Symposium on Mass Storage Systems*, pages 246–56, Los Alamitos, CA, Sep. 1995. IEEE Computer Society Press.

[DK93] A. L. Drapeau and R. H. Katz. Striping in large tape libraries. In *Proc. of Supercomputing*, pages 378–387, Portland, Oregon, 1993. ACM.

[EXA96] EXABYTE. Products. http://www.Exabyte.COM:80/Products/, Oct. 1996.

[GM95] L. Golubchik and R. Muntz. Analysis of striping techniques in robotic storage libraries. In *Proceedings of the Fourteenth IEEE Symposium on Mass Storage Systems*, pages 225–38, Monterey, CA, 1995.

[HS96] B. K. Hillyer and A. Silberschatz. Random I/O scheduling in online tertiary storage. In *Proc. ACM SIGMOD Int. Conf. on Management of Data*, Canada, 1996.

[KDST95] M.G. Kienzle, A. Dan, D. Sitaram, and W. Tetzlaff. Using tertiary storage in video-on-demand servers. In *Digest of Papers. COMPCON '95*, pages 225–33, 1995.

[LO96] J. Li and C. Orji. I/O scheduling in tape-based tertiary systems. *Journal of Mathematical Modelling and Scientific Computing*, 6, 1996.

[ML95] J. Myllymaki and M. Livny. Disk-tape joins: Synchronizing disk and tape access. In *Joint International Conference on Measurement and Modeling of ComputerSystems. SIGMETRICS '95/PERFORMANCE '95*, pages 279–90, Ottawa, Canada, 1995.

[MP93] T. E. Morton and D. W. Pentico. *Heuristic Scheduling Systems with applications to Production Systems and Project Management.* John Wiley and Sons, Inc., 1993.

[Pin95] M. Pinedo. *Scheduling Theory, Algorithms and Systems.* Prentice-Hall International Series in Industrial and Systems Engineering. Prentice-Hall, Englewood Cliffs, NJ, 1995.

[Qua96] Quantum. Products and technology. http://www.quantum.com/products, Oct. 1996.

[RW93] C. Ruemmler and J. Wilkes. UNIX disk access patterns. In *USENIX*, pages 405–420, Winter 1993.

[Sch86] H. D. Schwetman. CSIM: A C-based, process-oriented simulation language. In *Proceedings of the 1986 Winter Simulation Conference*, pages 387–396, December 1986.

[SON96] SONY. Recording media products. http://sony.sosin.com.sg/record-media/, Oct. 1996.

[SS93] S. Sarawagi and M. Stonebraker. Single query opimization for tertiary memory. Technical Report s2k-94-45, Computer Science Div. U.C. Berkeley, December 1993.

[SS94] S. Sarawagi and M. Stonebraker. Efficient organization of large multidimensional arrays. In *IEEE Int. Conf. on Data Engineering*, pages 328–336, Houston, TX, USA, Feb. 1994. IEEE Comput. Soc. Press.

[Teo72] T. J. Teorey. Properties of disk scheduling policies in multiprogrammed computer systems. In *Proceedings of the AFIPS Fall Joint Computer Conference*, pages 1–11, 1972.