Principles of Optimally Placing Data in Tertiary Storage Libraries

Stavros Christodoulakis* Peter Triantafillou* Fenia A. Zioga*
{stavros, peter, fenia}@ced.tuc.gr

Multimedia Systems Institute of Crete (MU.S.I.C.), Technical University of Crete (T.U.C.)
P.O. Box 133, Chania 73100, Greece

Abstract

Recently, technological advances have resulted in the wide availability of commercial products offering near-line, robot-based, tertiary storage libraries. Thus, such libraries have become a crucial component of modern large-scale storage servers, given the very large storage requirements of modern applications. Although the subject of optimal data placement (ODP) strategies has received considerable attention for other storage devices (such as magnetic and optical disks and disk arrays), the issue of optimal data placement in tertiary libraries has been neglected. The latter issue is more critical since tertiary storage remains three orders of magnitude slower than secondary storage. In this paper, we address this issue by deriving such optimal placement algorithms. First, we study the ODP problem in disk libraries (jukeboxes) and subsequently, in tape libraries. In our studies, we consider different scheduling algorithms, different configurations of disk libraries and different tape library technologies (reflecting different existing commercial products) and show how these impact on the ODP strategy.

*Support for this work was provided by the European Community through the ESPRIT Long Term Research Project HERMES no. 9141.

Proceedings of the 23rd VLDB Conference
Athens, Greece, 1997

1 Introduction

In large scale storage servers which are configured to include multiple on-line and off-line storage media and which deal with a large number of requests with unpredictable access patterns, the problem of minimizing the cost of accessing data stored in all media is critical for the performance of the system.

In this paper we examine tape and disk libraries for which we derive data placement algorithms in order to optimize the access cost. Typical tape/disk libraries include a few drives and many more (typically a few hundreds) disks/tapes. Disks/tapes are loaded onto the drives or unloaded from them by a robotic mechanism which typically consists of one arm. When a particular "loaded" object is requested the disk/tape drive head first places the read head onto the disk/tape location where the object resides and then reads the object. The components of the access cost which reflect this sequence of actions during an object access are:

- *Robot cost* (T^{robot}), i.e., the time needed for the robot arm to unload an on-line disk/tape and load a requested off-line one.
- *Head Positioning cost* (T^{head}_{seek}), i.e., the delay due to the placement of the disk/tape drive head on the appropriate disk/tape location (T^{head}). In the case of disks, the positioning cost includes the *seek* to the appropriate track and the *rotation* ($T^{head}_{rotation}$) to the appropriate sector of the track. In the case of tapes, the positioning cost includes only the *search* operation (T^{head}_{search}) to the appropriate tape location where the object resides.
- *Transfer cost* ($T^{transfer}$), i.e., the time needed for the head to read the object.

When a request arrives for an object stored in the library the following events occur:

a) First, the tape/disk that contains the requested object must be located and if it is off-line it must be exchanged with an on-line "victim" tape. The selection of the "victim" tape/disk is determined by a replacement algorithm. This tape/disk exchange process takes T^{robot} seconds and depends on the particular hardware of the library while it has been found to be independent of the particular tape/disk that is requested each time and the distances traveled by the robot arm ([Che94]). Typical values for the total robot delay range from 10 to 30 seconds.

b) Secondly, the tape/disk head locates the requested object within the tape/disk in T^{head}_{search} (or $T^{head}_{seek} + T^{head}_{rotation}$) seconds.

c) Finally, the object is read by the head in $T^{transfer}$ seconds.

The alternative placement schemes impact on some of the components of the access cost, not all of them: the transfer cost is always paid regardless of the placement and hence is not taken into account in the cost minimization[1]. On the contrary, T^{head}_{seek}, T^{head}_{search} and T^{robot} are affected by the placement strategy.

1.1 The Problem

Accessing tertiary libraries is typically at least 3 orders of magnitude costlier than accessing secondary storage. Yet, related research has neglected the issue of optimal data placement in tertiary storage while the issue has received considerable attention for disk based secondary storage ([Chr97c], [Chr91] [Chr97b], [Tri97a], [Tri97b], and [Won83]). In this paper, we address the issue of optimal data placement in tertiary storage in order to minimize the expected access cost.

The access cost in disk libraries is dominated by the disk exchange operation since it takes many seconds (while head positioning takes on average less than 20 ms). In tape libraries, both the *head positioning delay* and the *robot delay* (as defined above) participate in the estimation of the access cost. Even high-end tape drive products have a seek delay of more than one second per GB.

This paper focuses on the issue of minimizing the delay of random accesses in both tape and disk libraries. We consider a disk/tape library consisting of T disks/tapes and D drives and a set of O objects with different access probabilities. The access cost depends on the placement strategies that:

1. determine which objects are placed onto which media, and
2. determine the order with which the objects of some media are placed in it ([Tri97b]).

For disk libraries, only placement strategies that determine which objects are placed on which disks are relevant. For tape libraries the placement strategies must solve the problem at both levels.

The paper is organized in five sections. In section 2 an overview is provided describing the mathematical tools that are used throughout the paper and which are based on the Majorization theory and the theory of Schur functions. In section 3 the optimal placement problem for single and multiple drive libraries and under different scheduling algorithms is examined and solved. In section 4 the problem is solved for tape libraries and optimal algorithms are derived for different tape library technologies. Finally, section 5 contains concluding remarks and summarizes the results.

2 Mathematical Tools

Majorization Theory and Schur Functions

Let us consider two decreasing probability vectors $\vec{\mathbf{p}} = (p_1, \ldots, p_n)$ and $\vec{\mathbf{q}} = (q_1, \ldots, q_n)$. Formally, we say that $\vec{\mathbf{p}}$ majorizes $\vec{\mathbf{q}}$, which is symbolized as $\vec{\mathbf{q}} \prec \vec{\mathbf{p}}$, if $\sum_{i=1}^{j} p_i \geq \sum_{i=1}^{j} q_i$ for all $j < n$ and $\sum_{i=1}^{n} p_i = \sum_{i=1}^{n} q_i$. Intuitively, $\vec{\mathbf{p}}$ majorizes $\vec{\mathbf{q}}$ implies that $\vec{\mathbf{p}}$ is a "more skewed vector" than $\vec{\mathbf{q}}$. The concept of majorization can be used to compare the values of functions of vectors that are of a specific kind (Schur functions).

If a function $\phi : R^n \rightarrow R$ is **Schur convex** (or Schur **increasing**) and $\vec{\mathbf{p}}$ majorizes $\vec{\mathbf{q}}$, then $\phi(\vec{\mathbf{p}}) \geq \phi(\vec{\mathbf{q}})$. If a function $\phi(x)$ is a **Schur concave** (or Schur **decreasing**) and $\vec{\mathbf{p}}$ majorizes $\vec{\mathbf{q}}$, then $\phi(\vec{\mathbf{p}}) \leq \phi(\vec{\mathbf{q}})$.

A necessary and sufficient condition for a continuously differentiable function $\phi : R^n \rightarrow R$ to be Schur convex (concave) is that ϕ is symmetric on R^n and for all $i \neq j$:

$$(x_i - x_j)\left(\frac{\partial \phi}{\partial x_i} - \frac{\partial \phi}{\partial x_j} \right) \geq (\leq) 0$$

where each x_i is a component of the vector $\vec{\mathbf{x}} = (x_1, \ldots, x_n)$ (see [Mar79]).

Using the properties of Schur functions we have proved (see [Chr97a]) the following lemma which will be applied in the cost analysis:

Lemma 1: Let $\vec{p}, \vec{q} \in \mathcal{R}^n$ be two decreasing real-valued vectors and assume that $\vec{p}$ majorizes $\vec{q}$, that is $\vec{q} \prec \vec{p}$. Then

$$\sum_{j=1}^{n} j \cdot p_j \leq \sum_{j=1}^{n} j \cdot q_j$$

[1] Since the rotation component of the head positioning cost does not depend on the placement, the term *head positioning delay* will refer to the seek delay only.

i.e., the sum of the products $j \cdot p_j$ is minimized for the most skewed vector.

Proof

See [Chr97a] and [Mar79].

3 Disk Libraries

Let D denote the number of disk drives in our system and T be the total number of disks. The disks have identical storage and functional characteristics. There are O equally sized objects each having access probability q_i, $i = 1, \ldots, O$, that must be placed onto the T identical disks. Each disk can hold K objects. The number of object locations across all disks is equal to the number of objects O that must be placed in them. We assume that no replication of objects is allowed. We also assume a steady system state in which all drives are loaded. Let d^i denote the i^{th} disk of the library ($i = 1, \ldots, T$) and l_j^i be the j^{th} storage location within the i^{th} disk d^i. Table 1 summarizes the parameters of our system.

Table 1: The parameters of our system.

D	number of drives in the system.
T	number of disks in the system.
O	number of objects to be placed on the disks.
K	number of objects stored per disk.
Z	size of each object (bits).
l_j	the j^{th} location within any disk.
d^i	the i^{th} disk of the library.
l_j^i	the j^{th} location within the i^{th} disk.
q_k	probability of the k^{th} object ($1 \leq k \leq O$).
p^i	cumulative probability of the i^{th} disk.
p_j	cumulative probability of the j^{th} objects in all T disks.
p_j^i	access probability of the j^{th} object stored in the i^{th} disk.
$\vec{P^r}$	$(p^1, \ldots, p^T)$ row probabilities.
$\vec{P_c}$	$(p_1, \ldots, p_K)$ column probabilities.
$\vec{P^i}$	$(p_1^i, \ldots, p_K^i)$ probabilities of i^{th} tape.
$\vec{P_j}$	$(p_j^1, \ldots, p_j^T)$ probabilities of j^{th} column.

We define p_j^i to be the probability that the object stored at location l_j^i is requested. The cumulative probability p^i that the disk d^i is requested is $p^i = \sum_{j=1}^{K} p_j^i$. We define p_j to be the cumulative probability that an object stored at any of the l_j locations of the T disks is requested. The probability p_j is then $p_j = \sum_{i=1}^{T} p_j^i$. The disks of the library can be viewed as a 2-dimensional array in which rows correspond to disks and columns to object locations within the disks. Then we can define the vector consisting of the row probabilities $\vec{P^r} = (p^1, \ldots, p^T)$ and the vector consisting of the column probabilities $\vec{P_c} = (p_1, \ldots, p_K)$ of the array.

We wish to determine the optimal placement of the O objects onto the $T \cdot K$ storage locations of the disk library. This placement problem can be viewed as a two-level problem:

1. Decide which objects must be placed onto which disk.

2. Given the objects to be placed within each disk, determine the precise mapping of these objects to the storage locations of the disk.

For disk libraries, the expression "placement problem" refers to the first level problem of allocating objects to disks. As for the second level problem the related literature (see [Chr91, Chr97c, Tri97a, Won83]) provides various solutions.

We characterize as optimal the placement scheme which results in the minimum expected number of disk exchanges for a random request.

We should note that there are two factors that affect the minimization of the disk exchange cost:

a) The placement strategy itself of the O objects across the T disks.

b) The algorithm employed in selecting the next disk to load (i.e., given a number of requests in a queue waiting for tertiary storage service, which request will be selected for service next and hence which one of the referenced disks will be loaded next).

3.1 Disk Library with a Single Drive

Let us first consider the case of $D = 1$ and $T > 1$, that is, there is only one disk drive in the system and many disks. This means, that each time only one from the T disks can be on-line.

3.1.1 Optimal Placement with FCFS Scheduling

The access cost $C_{disk}^{(1)}$ is the expected disk exchange delay that is caused by a random request. Let T_{robot} be the time taken for an on-line disk to be removed from its drive and be replaced by an off-line disk. T_{robot} is independent from the relative position of the disks on the shelves ([Che94]).

Let $p_{exchange}$ be the probability that a disk exchange is triggered by a random request. Then:

$$p_{exchange} = \sum_{i=1}^{T} p^i (1 - p^i)$$

The product $p^i(1-p^i)$ is the probability that the disk d^i is requested (which occurs with probability p^i) and it is not on-line because the previous request had not requested it (which occurs with probability $(1-p^i)$). The summation over all disks expresses the fact that the requested disk may be any disk. The expected disk exchange delay is:

$$C^{(1)}_{disk}(\vec{P}^r) = T_{robot} \cdot p_{exchange} = T_{robot} \sum_{i=1}^{T} p^i(1-p^i)$$

Theorem 1: The function $C^{(1)}_{disk}(\vec{P}^r)$ is Schur concave. In other words, for all $i \neq j$

$$(p_i - p_j)\left(\frac{\partial C^{(1)}_{disk}}{\partial p_i} - \frac{\partial C^{(1)}_{disk}}{\partial p_j} \right) \leq 0$$

Proof

In our case, the partial derivative of the cost function with respect to p^i is

$$\frac{\partial C^{(1)}_{disk}}{\partial p^i} = \frac{\partial (p^i - {p^i}^2)}{\partial p^i} = 1 - 2p^i$$

Thus: $(p^i - p^j)\left(\frac{\partial C^{(1)}_{disk}}{\partial p^i} - \frac{\partial C^{(1)}_{disk}}{\partial p^j} \right) = (p^i - p^j)[(1 - 2p^i) - (1 - 2p^j)] = 2(p^i - p^j)(p^j - p^i) \leq 0$. The above equation holds for all $i \neq j$. Moreover, $C^{(1)}_{disk}(\vec{P}^r)$ is symmetric. □

This result is interpreted as follows: since the row probability vector $\vec{P}^r \in R^T$ has as components the cumulative probabilities of the disks ($\vec{P}^r = (p^1, \ldots, p^T)$), if S_1, S_2 are two different placement schemes which result in the row probability vectors $\vec{P}^{r^1}$, $\vec{P}^{r^2}$ and $\vec{P}^{r^1} \prec \vec{P}^{r^2}$, then the cost function is minimized when the placement S_2 is enforced (see section 2). This is true, since the row probability vector which is produced by S_2 majorizes the corresponding vector of S_1 scheme and thus $C^{(1)}_{disk}(\vec{P}^{r^2}) < C^{(1)}_{disk}(\vec{P}^{r^1})$. The optimal placement scheme S_{opt} is then the scheme with row vector $\vec{P}^r_{opt}$ such that $\vec{P}^r \prec \vec{P}^r_{opt}$ where $\vec{P}^r$ is the row vector that any placement scheme other than S_{opt} produces. In other words, we must place the objects in the disks in such a way that the resulting row vector majorizes all other possible row vectors. Such an optimal placement algorithm is presented below.

Algorithm 1: ODP in Disk Libraries

Initialize FreeDisks to include all system disks: FreeDisks=$(d^1, \ldots, d^T)$.
Initialize UnallocatedObjects to include all system objects: UnallocatedObjects=$(O_1, \ldots, O_O)$.
Set d^{next} to d^1.
while *the set UnallocatedObjects is not empty* **do**
begin
Select from the set UnallocatedObjects the K objects having the maximum cumulative probability.
Place the selected objects on disk d^{next}.
Remove the selected objects from the set UnallocatedObjects.
Remove disk d_i from the set FreeDisks.
Set d^{next} to d^{next+1}.
end

3.1.2 Optimal Placement with Bypass Scheduling

So far, we have assumed that requests are serviced with First Come First Served (FCFS) scheduling. FCFS scheduling is fair for all requests and simple to implement but different alternatives of request scheduling which take advantage of the on-line disk are likely to perform better.

Bypass scheduling is such a scheduling policy. It gives highest priority to all the requests in the queue that reference the on-line disk regardless of their arrival times. The disk d^i is replaced only when there is no unserviced request for it in the queue.

Assume that the queue of pending requests contains N unserviced requests for the T disks of the library and the drive of the system is loaded with the disk d^i. Let R be the set of all these pending requests. Then, we define the *reference set $R^j \subseteq R$ of disk d^j* to be the set of pending requests that hit disk d^j. If X such non-empty different sets exist in the queue then the number of disk exchanges will be X if the reference set R^i of the on-line disk is empty, otherwise it will be $X-1$. This means that the expected number of disk exchanges to service the N requests in the queue is equal to the expected number of different reference sets that exist in the queue:

$$E(\text{no of disk exchanges}) = \sum_{i=1}^{T} \left(1 - (1-p^i)^N\right) \quad (1)$$

The i^{th} term $1-(1-p^i)^N$ in the summation is the probability[2] that the reference set R^i of disk d^i is non-empty. Recalling that the definition of the access cost is the product of the expected number of disk exchanges times the constant delay of a disk exchange T_{robot}, we can derive the access cost $C^{(1)}_{bypass}$ when bypass scheduling is employed:

$$C^{(1)}_{bypass} = T_{robot} * E(\text{no of disk exchanges}) \quad (2)$$

[2] A selection with replacement modeling is clearly suitable here.

or substituting the expected number of disk exchanges from equation (1):

$$C^{(1)}_{bypass} = T_{robot} * \left(T - \sum_{i=1}^{T} (1-p^i)^N \right) \qquad (3)$$

The access cost of equation (3) is then our cost function which will be examined for minimization. In particular, we will determine the distribution of the probability vector $\vec{P}^r = (p^1, \ldots, p^T)$ which minimizes our cost function.

Theorem 2: The function $\phi(\vec{P}^r) = \sum_{i=1}^{T} (1-p^i)^N$ is Schur-convex.

Proof

The derivative of ϕ with respect to p^i is

$$\frac{\partial \phi}{\partial p^i} = -N(1-p^i)^{N-1}$$

For all $i \neq j$ the product $(p^i - p^j)\left(\frac{\partial \phi}{\partial p^i} - \frac{\partial \phi}{\partial p^j}\right)$ is $(p^i - p^j)\left(-N(1-p^i)^{N-1} + N(1-p^j)^{N-1}\right) > 0$. We can verify that if for example $p^i < p^j$ then $N(1-p^j)^{N-1} - N(1-p^i)^{N-1} < 0$. Moreover, $\phi(\vec{P}^r)$ is symmetric. □

According to equation (3) the cost is minimized when the function $\phi(\vec{P}^r) = \sum_{i=1}^{T} (1-p^i)^N$ is maximized. This occurs when the placement is such, that the probabilities distributed across all T disks compose a $\vec{P}^R_{opt}$ vector which majorizes the vector of any other placement scheme since according to the above theorem $\phi(\vec{P}^r)$ is Schur convex (see section 2). This result is the same as the one that we proved for the case of FCFS scheduling of requests. Therefore, the optimal placement algorithm for Bypass scheduling is the same algorithm that we provided for the case of FCFS scheduling of requests.

3.1.3 Optimal Placement of Variable Sized Objects

The optimal placement algorithm must place the objects onto the disks, so that the resulting $\vec{P}^r_{opt}$ vector majorizes the $\vec{P}^r$ vector of any other placement scheme, in order to accomplish the minimum number of disk exchanges. For equally sized objects it was sufficient to accumulate as big a probability as possible onto the first disk, then onto the second and so on. The objective is the same in this case as well. However, since the object sizes vary, the algorithm should take them into consideration. The construction of the most skewed vector $\vec{P}^r_{opt}$ is reducible to the well-known "0-1" knapsack problem of the algorithmic literature ([Hor78]) which has been solved with dynamic programming. In addition, we have constructed an efficient heuristic algorithm which determines the placement of variable-sized objects within the disk library in polynomial time. The heuristic algorithm is omitted due to space limitations but can be found in ([Chr97a]).

3.2 Multiple Disk Drive Configuration

We now study a system with multiple disk drives and multiple disks. We define the probability $p^{(i,j)}$ to be the probability that the i and j disks are on-line for $i, j = 1, \ldots, T$. Furthermore, we assume that the row probability vector $\vec{P}^R = (p^1, p^2, \ldots, p^T)$ is increasing, i.e., $p^1 < p^2 < \ldots < p^T$. Among the placement algorithms which produce an increasing [3] $\vec{P}^R$ vector we will determine the one with the minimum cost.

The scheme that is proposed below is based on the assumption that when a *miss* occurs (i.e., a request refers to an off-line disk), a disk replacement algorithm replaces the least popular on-line idle disk. This algorithm is called the **Least Popular Disk Replacement Algorithm (LPR)**. In the following sections, we will derive a placement strategy such that when combined with the LPR algorithm the random access cost is minimized.

3.2.1 Access Cost Minimization for the Case of Two Disk Drives and Three Disks

We first concentrate on a restricted example where $D = 2$ and $T = 3$. The row probability vector is $\vec{P}^R = (p^1, p^2, p^3)$ with $p^1 < p^2 < p^3$. Figure (1) shows the corresponding Markov chain state diagram. The arrows show transitions among the states along with the probability of occurrence of the transition.

The node labeled (i, j) represents a system state, in which the disks i and j are on-line. The Markov chain contains a *closed subset of states*, namely, $\{(2,3), (1,3)\}$, since no one-step transition from any of these states to state $(1,2)$ is possible. Therefore, the Markov chain is reducible. The probabilities $p^{(1,3)}, p^{(2,3)}$ can be calculated by solving the system:

$$\begin{aligned} p^{(2,3)} &= p^{(1,3)} * p^2 + p^{(2,3)} * (p^2 + p^3) \\ p^{(1,3)} &= p^{(2,3)} * p^1 + p^{(1,3)} * (p^1 + p^3) \\ p^{(1,3)} + p^{(2,3)} &= 1 \end{aligned}$$

Thus,

$$p^{(1,3)} = \frac{p^1}{p^1 + p^2} \qquad p^{(2,3)} = \frac{p^2}{p^1 + p^2}$$

[3] The fact that we consider placements with increasing $\vec{P}^R$ vectors is not limiting. The same results are obtained when considering all placements with decreasing $\vec{P}^R$ vectors.

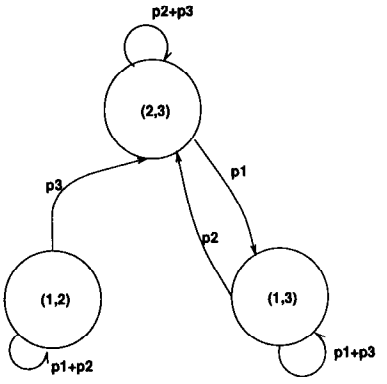

Figure 1: Diagram of Markov-chains for 2 drives and 3 disks ($p^1 < p^2 < p^3$).

Now the probability of an exchange is

$$\begin{aligned} p_{exchange} &= p^{(2,3)} * p^1 + p^{(1,3)} * p^2 \\ &= \frac{p^2}{(p^1+p^2)} * p^1 + \frac{p^1}{(p^1+p^2)} * p^2 \\ &= \frac{1}{(p^1+p^2)} * (p^1 * p^2 + p^1 * p^2) \\ &= \frac{2p^1p^2}{(p^1+p^2)} \end{aligned}$$

and the cost function is

$$C_{disk}^{(D)}(\vec{P}^r) = T_{robot} * p_{exchange} = T_{robot} * \frac{2p^1p^2}{(p^1+p^2)}$$

Theorem 3: The function $C_{disk}^{(D)}(\vec{P}^r)$ is Schur concave.

Proof

It can be easily verified that the partial derivative of the latter cost function is

$$\frac{\partial C_{disk}^{(D)}}{\partial p^1} = \frac{2(p^2)^2}{(p^1+p^2)^2} * T_{robot}$$

The Schur condition $(\mathrm{p^1 - p^2})(\frac{\partial \mathrm{C_{disk}^{(D)}}}{\partial \mathrm{p^1}} - \frac{\partial \mathrm{C_{disk}^{(D)}}}{\partial \mathrm{p^2}})$ for concavity is:

$$(\mathrm{p^1 - p^2}) * \left(\frac{2(\mathrm{p^2})^2}{(\mathrm{p^1+p^2})^2} - \frac{2(\mathrm{p^1})^2}{(\mathrm{p^1+p^2})^2} \right) * \mathrm{T_{robot}}$$

$$= \frac{2(\mathrm{p^1+p^2})}{(\mathrm{p^1+p^2})^2}(\mathrm{p^1 - p^2})(\mathrm{p^2 - p^1}) * \mathrm{T_{robot}} \leq 0$$

for all $p^1 \neq p^2$. Moreover, $C_{disk}^{(D)}(\vec{P}^r)$ is symmetric. □

Therefore, our cost function is minimized when the row probability vector $\vec{P}^R$ is the one that majorizes all other possible vectors that correspond to other placement schemes of the same objects. This placement policy is combined with the replacement algorithm, which maintains the p^3 disk always on-line on the first drive, and switches between the p^1 and p^2 disks on the second drive.

Intuitively, this replacement policy has as result that the disk with the highest probability remains on-line, in the steady system state, continuously. This is why the Markov chain is reducible. In the steady state therefore, the system behaves as if it had one on-line disk drive and two disks to be exchanged, and therefore it behaves like the single drive system.

Generalizing the ODP for T disks and D drives

These results can naturally be expanded to apply to more general configurations of D drives and $T > D$ disks. The corresponding Markov chain will also contain a subset of closed states and will be reducible. In the steady system state, the $D-1$ drives will be continuously occupied by the $D-1$ disks with the highest probabilities. The one drive left will be used to keep one of the $T-D+1$ remaining disks. Hence, the optimal placement policy continues to place the probabilities in such a way that the resulting $\vec{P}^R$ vector majorizes all others and combines this scheme with the replacement policy which maintains the $D-1$ most popular disks on-line always, and performs exchanges in the one remaining disk drive. Such a policy minimizes disk exchanges due to the skewed arrangement of the disk probabilities.

4 Tape Libraries

The factors that influence the estimation of the cost of accessing the tape objects are the *head positioning delay* and the *robot delay*. The *head delay* that is associated with an object access is attributed to two factors: a) the *search delay*, which is the time taken by the head in order to locate the requested object within the tape, and b) perhaps, the *rewind delay*, which is the time taken for the tape to perform the rewind operation. Thus,

$$head\ delay = search\ delay + rewind\ delay \quad (4)$$

The *robot delay* is the delay introduced when a loaded tape is removed from its tape drive and placed on the shelf and an off-line tape containing the newly requested object is subsequently placed on that particular drive.

Given the current technology, two alternatives exist for the estimation of the access cost:

1. Only the head delay is taken into account in the cost determination.

   This is meaningful for environments where the robot delay is negligible as compared to the head

delay and can therefore be ignored in the cost calculations with negligible error. Such environments could for example be the AMPEX DST tape libraries with tape capacities up to 165 GB, search speed equal to 800 MB/sec and resulting average search time of approximately 100 sec. The latter value of the head delay is dominant when compared to the typical value of the robot delay which is reported to be less than 6 sec.

2. Both the head delay and the robot delay contribute to the cost determination.

   The AMPEX DST 810 tape library with cartridge capacity 3 GB is such an example. In this environment, the average search time is 1.5-2.0 sec which is smaller than the robot delay. Hence, exchanges must also be considered in the cost derivation.

Furthermore, the determination of the access cost can vary depending on the operational characteristics of the tape drives. One key issue is the rewind operation. Two alternatives are currently supported:

- In some technologies tape drives must essentially rewind to the PBOT[4], before being ejected ([Amp97, Exa97]). This is the case for example for the Exabyte tapes. In the Ampex DST series this feature of rewinding to the physical beginning of a tape before eject is only optional.

- Other tape drives, such as the AMPEX DST series define "zones" at multiple locations along the tape, enabling rewind to proceed alternatively to any of the zones ([Amp97]) before ejection. Most frequently, the zone nearest to the head is selected in order to minimize the rewind time.

The system parameters included in Table 1 hold for the case of tape libraries, if "disk" is replaced with "tape". For example, the symbol l_j^i denotes the j^{th} location within the i^{th} tape. Moreover, the i^{th} tape is symbolized as t^i. We are interested in determining the optimal placement of the $T \cdot K$ objects across all $T \cdot K$ possible locations of the T tapes[5]. Optimality of placement is achieved when the expected delay of accessing a random object is minimized.

We consider tapes with dominant head delay and therefore the cost function is approximated to be equal to the expected head delay incurred in a random access. According to the definition of the head delay (4), our cost function is the summation of the search and the rewind delay. Let the average search speed of the tape head be $SchSp$, and let $RwSp$ be the average rewind speed. Let also d_{search} be the expected distance (in number of bits) traveled by the head with $SchSp$ to reach a desired tape location. Then, our cost function will be the summation of $\frac{d_{search}}{SchSp}$ (search delay) plus $\frac{d_{search}}{RwSp}$ (rewind delay). In general, $RwSp > SchSp$. Thus, if Rwf, $1 < Rwf < 2$, is a constant such that $(Rwf)\frac{d_{search}}{SchSp} = \frac{d_{search}}{SchSp} + \frac{d_{search}}{RwSp}$ then our cost function is:

$$AccCst = (Rwf)\,(d_{search}/SchSp)$$

Since $SchSp$ and Rwf are constant the cost function depends only on the expected value of the distance of the requested object from the current head position (d_{search}).

As explained previously, the current tape technology supports two alternatives for tape rewinding. Naturally, the methodology of estimating the cost of a random tape access and the corresponding optimal placement algorithm differ depending on the tape rewinding technology, since the search distances are different for different rewinding technologies. Section (4.1) provides the cost analysis and the optimal placement scheme for tapes which rewind to the nearest zone, while section (4.2) examines tapes which rewind to the PBOT. Both analyses are then further specialized in two cases for FIFO and SCAN ([Chr97a]) scheduling algorithms which represent systems under light and heavy load, respectively.

We should recall that optimal placement solutions for single disks have been provided. In [Won83] the optimal placement of objects within a magnetic disk in order to optimize the random access cost is proved to be the organ-pipe arrangement. The organ-pipe arrangement of a set of n probabilities $p_1, \ldots, p_n$ places the largest probability at the middle point. Then, it repetitively places the next largest probability, alternating between the position immediately to the left (right) of those already placed and the position immediately to the right (left). Similar placement algorithms have been developed for CLV disks ([Tri97a]) and multi-zoned CAV disks ([Chr97c]).

4.1 Tapes that Rewind to the Nearest Zone.

We examine tapes which enable tape rewinding to the nearest tape zone as opposed to tapes which require to be rewound to their physical beginning before they can be ejected. The cost analysis is different depending on the request scheduling policy. Therefore, we have separately examined the cases of FCFS and Scan

[4] Physical Beginning of Tape.

[5] We assume that the number of object locations across all tapes and the number of objects that need to be placed in them are equal. We also assume that no replication of objects is allowed. Furthermore, the system is in a steady state i.e., all drives are loaded.

scheduling which are appropriate for lightly and heavily loaded systems respectively and have derived optimal placement algorithms. In the following, we present the cost analysis and the optimal placement for the case where there is a single request for each tape. The case where there are several requests per tape which are scheduled using a SCAN-like algorithm is studied in [Chr97a] and is omitted for space reasons.

4.1.1 Placement of Objects within a Tape

The cost function for a single tape is the expected distance that must be traveled by the tape head in order to serve a random request for an object of this tape and it is derived as follows: let l_w^i and l_z^i be two locations within a random tape t^i. Assume that l_w^i is the current head location (i.e., the location of the object that was previously requested from t^i) and l_z^i is the location where the head must be moved to access the object of the next request for t^i. The distance $d_{search}(l_w^i \rightarrow l_z^i)$ that must be traveled is then:

$$d_{search}(l_w^i \rightarrow l_z^i) = \begin{cases} (w-z+1)Z, & \text{, if } w > z \\ (z-w-1)Z, & \text{, if } w < z \\ 0, & \text{, if } w = z \end{cases}$$

The definition of $d_{search}(l_w^i \rightarrow l_z^i)$ is such that the distance when moving from the tape location l_w^i to l_{w+1}^i is 0, while the distance when moving from the tape location l_w^i to location l_{w-1}^i is $2\,Z$ (i.e., the head always moves from the end of the w^{th} object to the beginning of the z^{th} object).

The expected distance d_{search}^i that must be traveled within t^i to serve a random request that hits t^i is derived by summing the distances of all possible events of moving between any l_w^i, l_z^i tape locations, i.e.:

$$d_{search}^i = \sum_{w<z} p_w^i p_z^i (z-w-1)Z + \sum_{w>z} p_w^i p_z^i (w-z+1)Z =$$
$$\sum_{w,z} p_w^i p_z^i |w-z| Z - Z \sum_{w<z} p_w^i p_z^i + Z \sum_{w>z} p_w^i p_z^i$$

or

$$d_{search}^i = \sum_{w,z} p_w^i p_z^i |w-z| Z + Z \sum_w (p_w^i))^2 \qquad (5)$$

In the search distance function, the term $Z \sum_w (p_w^i))^2$ is independent from the relative placement of probabilities within the tape. Therefore, the only relevant distance component is $\sum_{(w,z)} p_w^i p_z^i |w-z| Z$. This is exactly the cost function that must be minimized when considering the traditional problem of placing objects on a disk. Wong has shown ([Won83]) that this (seek cost) function is minimized when objects are placed in an organ pipe arrangement. Therefore, given the information about which objects must be placed on which tape, the optimal placement of objects within each tape is the one that performs an organ-pipe arrangement of the object probabilities.

4.1.2 Assignment of Objects to Tapes

We are now left with the problem of optimally determining the contents of each tape, i.e., which objects to place in the tapes. The cost function is the expected total distance d_{search} traveled within all tapes when each of the tapes serves a random request for one of the objects that it holds. In other words, the cost function is $d_{search} = \sum_{i=1}^{T} d_{search}^i$ or from equation (5) $d_{search} = \sum_{i=1}^{T} \left(\sum_{w,z} p_w^i p_z^i |w-z| Z + Z \sum_w (p_w^i)^2 \right)$ and $\sum_{i=1}^{T} \sum_{w,z} p_w^i p_z^i |w-z| = \sum_{w,z} \left(\sum_{i=1}^{T} p_w^i p_z^i |w-z| \right)$ $= \sum_{w,z} |w-z| \left(\sum_{i=1}^{T} p_w^i p_z^i \right)$

Theorem 4: The function $\phi(\vec{P}_w) = \sum_{i=1}^{T} p_w^i p_z^i$ is Schur convex for all $w = 1, \ldots, K$.

Proof

Differentiating we get $\frac{\partial \phi(\vec{P}_w)}{\partial p_w^i} = p_z^i$ and $\frac{\partial \phi(\vec{P}_w)}{\partial p_w^m} = p_z^m$. Assuming, for example, decreasing column vectors[6] it holds that for all $i < m$, $p_w^i > p_w^m$, $p_z^i > p_z^m$ and therefore $(p_w^i - p_w^m)(p_z^i - p_z^m) > 0$. It can also be verified that $\phi(\vec{P}_w)$ is symmetric. Thus, $\phi(\vec{P}_w)$ is Schur convex. □ Therefore, (from section 2) for each column vector $\vec{P}_w = (p_w^1, p_w^2, \ldots, p_w^T)$ the sum $\sum_{i=1}^{T} p_w^i p_z^i$ is minimum when the vector $\vec{P}_w = (p_w^1, p_w^2, \ldots, p_w^T)$ is majorized by all other possible vectors (i.e., has as equal components as possible or is as uniform as possible).

Since we have shown that within each tape the organ pipe placement is optimal we know that the q_1 probability (assuming $q_1 > q_2 > \ldots > q_O$) will be placed in the middle location of one tape say in $t_{\frac{K}{2}}^i$ for $1 \leq i \leq T$. A consequence of the "$\vec{P}_w$ uniform" optimality condition that we just derived is the placement of the T biggest probabilities in the T middle locations of all tapes. Then, the only way to produce the most uniform $\vec{P}_{\frac{K}{2}}$ column vector is to fill the rest of its components with $q_2, \ldots, q_T$. Thus, we will have each of the T most popular objects in the middle location of a tape. The same holds for the rest of the locations within the tapes.

The optimal placement scheme is shown in figure (2) for a library consisting of 3 tapes assuming that each

[6] The fact that we consider placements with decreasing column vectors is not limiting. The same results are obtained when considering all placements which result in increasing columns. Our objective is to determine the placement with the minimum cost among the placements which produce decreasing columns.

tape can hold 5 equally sized objects that have different access probabilities.

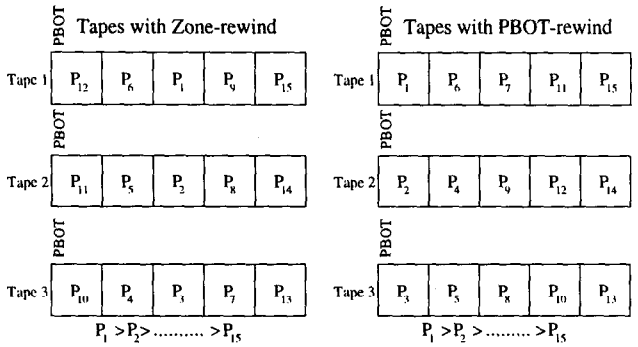

Figure 2: Optimal placement scheme.

4.2 Tapes that Rewind to the PBOT

In this section, we examine tapes which always rewind to their beginning before being ejected. As we have already mentioned the analysis is different depending on the request scheduling policy. In the following, we derive the optimal placement for systems with FCFS scheduling while the analysis for SCAN scheduling can be found in ([Chr97a]).

Let t^i be a single on-line tape and assume that the j^{th} object of that tape is requested. The distance $d_{search}(l_1^i \rightarrow l_j^i)$ that will be traveled by the head during searching for the j^{th} object is equal to the total space occupied by the $j-1$ objects that are placed in front of the requested object:

$$d_{search}(l_1^i \rightarrow l_j^i) = \sum_{l=1}^{j-1} Z = (j-1)Z$$

assuming that the initial head position is at the PBOT (i.e., location l_1^i). The expected distance for randomly accessing any of the K objects located on the tape t^i is

$$d_{search}^i = Z\left(\sum_{j=1}^{K} j \cdot p_j^i - \sum_{j=1}^{K} p_j^i\right)$$

The cost function d_{search} is the sum of the expected distances traveled in all T tapes in order for each tape to serve a request for a random object on the tape:

$$d_{search} = \sum_{i=1}^{T} Z \cdot \left(\sum_{j=1}^{K} j \cdot p_j^i - \sum_{j=1}^{K} p_j^i\right)$$

which can be simplified to:

$$d_{search} = Z \cdot \sum_{i=1}^{T}\sum_{j=1}^{K} j \cdot p_j^i - Z \qquad (6)$$

If we set $p_j = \sum_{i=1}^{T} p_j^i$ then p_j expresses the probability that one of the j^{th} objects of the T tapes is requested and equation (6) can be rewritten as follows:

$$d_{search} = Z \cdot \sum_{j=1}^{K} j \cdot p_j - Z \qquad (7)$$

The expected distance of equation (7) and hence our cost function is minimized when the summation $\sum_{j=1}^{K} j \cdot p_j$ is minimized. Consider the column vector $\vec{P}_c = (p_1, \ldots, p_K)$ in which the component p_j is the aggregate access probability of the objects stored at locations l_j^i for all $1 \leq i \leq T$. From Lemma 2 the summation $\sum_j j \cdot p_j$ is minimized when the vector $\vec{P}_c$ majorizes all others (i.e., is as skewed as possible). The optimal placement scheme must therefore arrange the objects within the tapes so that the resulting $\vec{P}_c$ vector is as skewed as possible. One such scheme for example, is the one which stores each of the T most popular objects first in each of the T tapes, each of the next T most popular objects second in each of the tapes and so on.

Algorithm 2: ODP for PBOT Rewinding Tapes

Sort the UnallocatedObjects in decreasing probability order.
while *the set UnallocatedObjects is not empty* **do**
begin
Scatter randomly the first T objects of the set UnallocatedObjects onto the first free location of each of the T tapes.
Remove the first T objects from the set UnallocatedObjects.
end

The optimal placement is depicted in figure (2) for a simple case of 15 objects with different access probabilities that are allocated onto the 3 tapes (each tape holding 5 objects) of a tape library.

5 Summary

In this paper, we have studied the problem of data placement in robotic disk and tape libraries. This problem is important since tertiary storage is much slower to access than secondary storage.

In the case of disk libraries, the major cost to be optimized is the expected number of disk exchanges, since each disk exchange costs somewhere between 5 and 15 seconds (to find the data on the on-line disk is much less expensive due to the random access mechanism of disks). We showed using the theory of Majorization and Schur functions that the optimal data placement is obtained by placing as many of the most

popular objects as can fit on one disk and by repeating this process for the remaining objects and disks.

In the case of tape libraries, the tape exchange cost is still significant, but another important cost (which can be the dominant cost when the library stores tapes with very large capacities) is the cost of sequentially searching throughout the tape to find the data that is needed. We separately considered tapes of two rewind technologies: zone and PBOT rewind, since the expected distance searched (and the analysis for its minimization) is different depending on the rewind technology. For each tape technology, the analysis was further specialized depending on whether the implemented scheduling policy of requests is Scan or FCFS. However, the analysis under Scan scheduling has been left out since it produces the same results with FCFS scheduling (it can be found in ([Chr97a])).

We showed that when tapes rewind to the nearest zone, the optimal placement must randomly distribute the T highest probabilities in the middle locations of the T tapes, the second and third T highest probabilities to the left and right of the middle locations of the tapes and continue likewise. For tapes that rewind to the PBOT it is optimal to place each of the T highest probabilities on the first location of a tape, each of the second T highest probabilities on the second location of a tape and so on. The above analysis considered only head positioning cost. The exchange cost was omitted for space reasons and because its effect to the optimal data placement strategy is equivalent to that shown for disk libraries.

References

[Amp97] *Personal Communication* with Ampex Co.

[Che94] A. L. Chervenak, *"Tertiary Storage: An Evaluation of New Applications"*, Doctor of Philosophy Thesis, Department of Computer Science, University of California, Berkeley, 1994.

[Chr91] S. Christodoulakis and D. Ford, *"Optimal Data Placement on CLV Optical Disks"*, ACM Transactions on Information Systems (ACM TOIS), 1991.

[Chr97a] S. Christodoulakis, P. Triantafillou and F. A. Zioga, *"Optimal Data Placement in Robotic Disk and Tape Libraries"*, MUSIC Technical Report, No 14.

[Chr97b] S. Christodoulakis and F. A. Zioga *"Principles of Striping and Placement of Delay-Sensitive Data on Disks"*, (submitted). Also available from www.ced.tuc.gr/hermes.

[Chr97c] S. Christodoulakis, P. Triantafillou, and K. Poutos, *"Dependencies of Scheduling on Optimal Data Placement in CLV Optical Disks"*, (submitted). Also available from www.ced.tuc.gr/hermes.

[Exa97] *Personal Communication* with Exabyte Co.

[Hor78] E. Horowitz, S. Sahni, *"Fundamentals of Computer Algorithms"* Computer Science Press, 1978.

[Mar79] A. Marshall and I. Olkin, *"Inequalities: Theory of Majorization and its Applications"*, Academic Press, University of Southern California, 1979.

[Tri97a] P. Triantafillou, S. Christodoulakis, and C. Georgiadis, *"Optimal Data Placement on Disks: A Comprehensive Solution to Different Technologies"*, IEEE Transactions on Knowledge and Data Engineering (conditionally accepted.) Also available from www.ced.tuc.gr/hermes.

[Tri97b] P. Triantafillou and C. Faloutsos, *"Overlay Striping and Optimal Parallel I/O in Modern Applications"*, Parallel Computing Journal, Special Issue on Parallel Data Servers and Applications, (accepted to appear).

[Won83] C. K. Wong, *"Algorithmic Studies in Mass Storage Systems"*, IBM, Thomas J. Watson Research Center, Computer Science Press, 1983.