

THE APPLICATION OF RESTRICTED COUNTER SCHEMES TO THREE MODELS OF LINEAR SEARCH

Micha Hofri[†] and Hadas Shachnai[‡]

December 1991

ABSTRACT

The mechanism of the *Counter Scheme (CS)* has been shown to be an effective statistical approach for the reorganization of linear lists, where the records in the list are referenced independently with a time homogeneous multinomial distribution. In this paper we show that derivative schemes can be used effectively in other contexts as well.

Specifically, we consider (a) linear lists that are doubly-linked, so that they may be accessed at both ends; (b) multilists, which result from dissecting a linear list into several pieces which are accessed independently and reside in WORM (write-once-read-many store), and (c) reorganizing a disk, by copying its contents to another disk, so as to minimize the expected seek time required to access a record.

1. List Reorganization with the Restricted Counters Scheme – Concepts and notation

Finding an expedient order for the elements of a linear list is a well-studied problem. Most of the work in the area considered the following elementary data structure:

$L = \{R_1, \dots, R_n\}$ is a linear (singly-linked) list of n records, initially linked in an arbitrary order. The set of records is fixed in time, with no additions or deletions.

The requests to access the list obey the independent reference model (*irm*), and use a reference-probabilities vector (*rpv*) $\mathbf{p} \equiv (p_1, \dots, p_n)$ where p_i is the fixed (time-homogeneous) probability that an access request is for the record R_i , $1 \leq i \leq n$.

Typically, each access requires a sequential search starting at the head of the list, until the specified record is encountered. We define the cost of a reference to an element in position j in L as j . The objective of managing the list is to minimize the expected cost of access.

In the ideal case where the access probabilities are all known, this model calls for an optimal static arrangement of the records in decreasing order of their reference probabilities, i.e.

$$p_i > p_j \iff \sigma_o(i) < \sigma_o(j),$$

where $\sigma_o(i)$ denotes the location of R_i in the optimal permutation σ_o of the list L . Ties are resolved arbitrarily.

However, it is more common that no initial information is available on the *rpv*, and the sequence of requests may be considered as a learning process. Then, during a history of

[†] Department of Computer Science, University of Houston, Houston Tx 77204-3475, USA.

[‡] Department of Computer Science, Technion – Israel Institute of Technology, Haifa 32000, Israel.

references, the list is dynamically reorganized by a permutation rule, which may use any information accumulated during the process to achieve the desirable order.

Three reorganization methods give rise to a large set of rules presented in previous work: *Move To the Front (MTF)* shifts the accessed record to the head of the list, leaving the relative order of the other elements unchanged. *Transpose (TR)* advances the accessed element one step ahead by an interchange with its immediate predecessor. The *Counter Scheme (CS)* keeps a reference count for each record, which tracks the number of references to it. The list is maintained sorted in nonincreasing order of the counter values.

A comprehensive account of the policies considered in last two decades appears in [6]. Some variations and more recent analyses of the above rules appear in [3,5,8,10,12].

We note that all the reorganization methods allow us to model the list order as an ergodic Markov chain of $n!$ states. Let $C_m(PR|\mathbf{p})$ and $C(PR|\mathbf{p})$ denote the expected access cost to the list using the permutation rule PR and the m th request and in the limiting state respectively. The initial order of the list plays a role in $C_m(PR|\mathbf{p})$ only; the limiting value is independent of it. In most of the published works, and in *all* the analyses and results shown below, the initial state, where it matters, is assumed to be uniformly distributed over all $n!$ possibilities.

Courcoubetis and Weber present in [4] a chain inequality, which characterizes the relations between any pair of the four ordering methods mentioned above in the limit of a long reorganization sequence:

$$C(OPT|\mathbf{p}) = C(CS|\mathbf{p}) \leq C(TR|\mathbf{p}) \leq C(MTF|\mathbf{p}) \leq 2 C(OPT|\mathbf{p}) . \quad (1.1)$$

Both the *MTF* and *TR* are *memory free*, and they do not produce convergence to the optimal ordering for nonuniform rpv 's. Moreover, for any reference sequence which keeps referencing at least two distinct keys, both rules never stop reordering the list. On the other hand, the *CS*, which asymptotically attains the minimal average cost, suffers from a space problem, since the counters are unbounded. Hence it is impractical for long reference sequences and large values of n .

There are however two ways of curbing the space requirements of the *CS*. Both produce suboptimal rules, where the departure from optimality can be controlled by parameters. One approach is the *Terminating Counter Scheme (TCS)*, according to which the list is reorganized for a finite, predetermined number of times, m^* . The *TCS* guarantees that the expected access cost is within a factor of $(1 + \alpha)$ of the optimal access cost. It was introduced in [8], and we show there¹ that $m^* = n(n-1)(1+\alpha)^2/16\alpha^2$.

The second approach, called the *Limited Counters Scheme (LCS)*, also analyzed in [8], is defined there as follows:

Each record R_i $1 \leq i \leq n$, is associated with a frequency counter C_i , which may not exceed the value c_{\max} . As long as $C_i < c_{\max}$, it is incremented at each request to R_i . At the same time R_i is shifted forward if necessary, so as to precede any R_j with $C_j < C_i$ (we can only have such $C_j = C_i - 1$). When C_i reaches c_{\max} it remains fixed, and the location of R_i is unaffected by any subsequent references to the list.

Hence, the dynamic reorganization process involves at most nc_{\max} changes in record positions.

If $c_{\max} = 1$, the *LCS* produces the same expected access cost as the *MTF*, i.e.

¹Special situations—in particular, extra information about \mathbf{p} —may admit substantially lower bounds.

$$C_m(LCS|\mathbf{p}, 1) = C_m(MTF|\mathbf{p}) \quad \forall m \geq 1. \quad (1.2)$$

The asymptotic access cost to the list under *LCS* satisfies the relation

$$\frac{C(LCS|\mathbf{p}, c)}{C(OPT|\mathbf{p})} \leq 1 + \max_{a \geq 1} \frac{(a-1) \left[\sum_{r=0}^c \binom{2c}{r} a^r - \frac{1}{2} \binom{2c}{c} a^c \right]}{(1+a)^{2c}}, \quad (1.3)$$

which is bounded by 1.2175 already for $c_{\max} = 3$. (Proofs of equations(1.2) and (1.3) appear in [8]).

Some exact computations of that relation, for a few known distribution functions lend strong support to the conjecture that *LCS* performs well, with very modest space requirements.

In the following sections, we consider the use of *TCS* or *LCS* on three other models of sequential search.

2. The CS for Doubly Linked Lists

An immediate elaboration of the above model is the doubly linked list. We assume the following layout:

A set of n records $\{R_i, 1 \leq i \leq n\}$, identified uniquely by their keys, is held in a doubly linked list D . Each element R_i is accessed with fixed probability p_i , and the search for it may begin either at the “left” end of D , with probability p_{iL} , or at the “right” end, with probability $p_{iR} = 1 - p_{iL}$. A reference in this scheme specifies both a key and a starting point for the search. The access cost, as above, is defined to be the number of key-comparisons required to locate a record in the list.

The referenced key and the starting point for each search are both chosen independently of the past accesses or all previous states of the list. This may typically result when the searching mechanism takes part also in other activities, and does not serve this reference string only. The initial ordering of D is assumed random, with equal probability for each permutation.

Let $\mathbf{p} = (p_1, p_{1L}, \dots, p_n, p_{nL})$ be the *rpv*. When \mathbf{p} is known, the average access cost is minimized when D is kept in a static optimal ordering, which is in decreasing order (from left to right) of the values

$$p_i^* = p_i (p_{iL} - 1/2) = \frac{p_i(p_{iL} - p_{iR})}{2}, \quad (2.1)$$

as shown by Matthews *et al.*, in [11].

From now on we assume that \mathbf{p} is *unknown*, and D is dynamically rearranged during the reference sequence.

Some special degenerate instances of \mathbf{p} are considered in [11], for which the strategies adopted for singly linked lists are as effective in the present context.

Lemma 2.1: ([11])

i) If $p_{iL} = p = 1/2$, $1 \leq i \leq n$, then no rearrangement is necessary or can improve the expected access cost.

ii) If $p_{iL} = p > 1/2$, $1 \leq i \leq n$, and H^* is an optimal policy for the corresponding sequential list of n elements when $p = 1$ (which behaves precisely as the singly linked list), then H^* is optimal for D as well. (In [7] we proved that $H^* = CS$). \square

We examine the general case, in which there exists at least one pair of indices $1 \leq i, j \leq n$, such that $p_{iL} \neq p_{jL}$. For that case, two memory-free rules were presented:

Move To the End (MTE) – an accessed record is moved to the position where the search for it began.

Transpose Toward End (TTE) – an accessed record is shifted one step towards the start point of its search.

A result, analogous to the one derived by Rivest [13] for linear lists, is stated in

Lemma 2.2: ([11])

If $C(OPT|\mathbf{p})$ is the minimum expected search cost when D is optimally arranged, from left to right, then

$$C(MTE|\mathbf{p}) < 2C(OPT|\mathbf{p}) . \quad \square \quad (2.2)$$

We now define an equivalent to CS for doubly linked lists (DCS), which keeps a difference count D_i for each record R_i , $1 \leq i \leq n$: $D_i = C_{iL} - C_{iR}$, where C_{iL} and C_{iR} denote the number of searches for R_i starting at the left end and at the right end respectively. D_i is changed whenever R_i is accessed: when search starts at the left end, D_i is incremented, otherwise it is decremented. The DCS maintains the list in nonincreasing order of D_i from left to right.

Lemma 2.3: The average access cost to D under DCS after the m th request, $m \geq 1$, satisfies

$$\begin{aligned} C_m(DCS|\mathbf{p}) &= 1 + (n-1) \left(1 - \sum_{i=1}^n p_i p_{iL}\right) + 2 \sum_{i=1}^n p_i (p_{iL} - 1/2) \sum_{j \neq i}^m \sum_{k=0}^m \binom{m}{k} (1 - p_i - p_j)^{m-k} \\ &\times \left[\sum_{r=\lfloor k/2 \rfloor + 1}^k \binom{k}{r} (p_i p_{iL} + p_j p_{jR})^{k-r} (p_i p_{iR} + p_j p_{jL})^r + \frac{1}{2} I_k \binom{k}{k/2} (p_i p_{iL} + p_j p_{jR})(p_i p_{iR} + p_j p_{jL})^{k/2} \right] \end{aligned} \quad (2.3)$$

where an empty sum on r vanishes, and $I_k = \begin{cases} 1 & \text{when } k \text{ is even} \\ 0 & \text{otherwise.} \end{cases}$

Proof: Let $l(j, i)$ be the probability that R_j is located to the left of R_i after the m th request, when D is maintained under DCS . Then, since $l(j, i) + l(i, j) = 1$,

$$\begin{aligned} C_m(DCS|\mathbf{p}) &= 1 + \sum_{i=1}^n p_i p_{iL} \sum_{j \neq i} l(j, i) + \sum_{i=1}^n p_i p_{iR} \sum_{j \neq i} l(i, j) \\ &= 1 + 2 \sum_{i=1}^n p_i (p_{iL} - 1/2) \sum_{j \neq i} l(j, i) + (n-1) \left(1 - \sum_{i=1}^n p_i p_{iL}\right) \\ &= 1 + \sum_{1 \leq j < i \leq n} [(p_i p_{iL} + p_j p_{jR}) + 2(p_j^* - p_i^*) l(i, j)] . \end{aligned} \quad (2.4)$$

Note, that under DCS R_j is positioned to the left of R_i whenever $D_j > D_i$. In terms of $C_{iL} \equiv C_{iL}^{(m)}$ and $C_{iR} \equiv C_{iR}^{(m)}$,

$$D_j > D_i \iff C_{jL} - C_{jR} > C_{iL} - C_{iR} ,$$

hence

$$l(j, i) = \text{Prob} \{ C_{jL} + C_{iR} > C_{iL} + C_{jR} \} .$$

As $C_{jL} + C_{iR} \sim \text{Bin}(m, p_i p_{iR} + p_j p_{jL})$, writing explicitly $l(i, j)$ into the second line of (2.4) produces equation (2.3). \square

Lemma 2.4: DCS converges asymptotically to the optimal ordering, i.e.

$$C(DCS|\mathbf{p}) \equiv \lim_{m \rightarrow \infty} C_m(DCS|\mathbf{p}) = C(OPT|\mathbf{p}) . \quad (2.5)$$

Proof: Under the optimal arrangement, R_i will be placed to the left of R_j if

$$p_i(p_{iL}^{-1/2}) > p_j(p_{jL}^{-1/2}).$$

By the strong law of large numbers,

$$\frac{C_{iL} - C_{iR}}{m} \xrightarrow{m \rightarrow \infty} p_i p_{iL} - p_i p_{iR} > p_j p_{jL} - p_j p_{jR} = \lim_{m \rightarrow \infty} \frac{C_{jL} - C_{jR}}{m}.$$

Hence, under *DCS*, each pair of records will be ordered properly in the limiting state.

□

Consider now the adaptation of *TCS* to the doubly-linked list. In a sense, the following result measures the rate with which $C_m(\text{DCS}|\mathbf{p})$ converges to $C(\text{OPT}|\mathbf{p})$. It enables the restriction of *DCS* to finite reference sequences, thus avoiding possible overflows of the D_i 's with bounded storage overhead cost.

Theorem 2.5: For any doubly linked list D with n elements and the probabilities vector \mathbf{p} , $C_m(\text{DCS}|\mathbf{p})$ approaches $C(\text{OPT}|\mathbf{p})$ to within a difference of $x = n\alpha$, $0 < \alpha < 1$, after at most m^* accesses, where

$$m^* = \left\lceil \frac{(n-1)^2 \log 2}{2\alpha^2} \right\rceil. \quad (2.6)$$

Proof: Assume a renumbering of the records, such that $p_i^* \geq p_j^*$, $\forall 1 \leq i < j \leq n$, with p_i^* defined in equation (2.1). Then,

$$C(\text{OPT}|\mathbf{p}) = 1 + 2 \sum_{1 \leq i < j \leq n} p_j^* + (n-1) \left(1 - \sum_{i=1}^n p_i p_{iL}\right),$$

and rearranging equation (2.3),

$$\begin{aligned} C_m(\text{DCS}|\mathbf{p}) &= 1 + (n-1) \left(1 - \sum_{i=1}^n p_i p_{iL}\right) + 2 \sum_{1 \leq i < j \leq n} \left(p_j^* + (p_i^* - p_j^*) \sum_{k=0}^m \binom{m}{k} (1 - p_i - p_j)^{m-k}\right. \\ &\times \left. \left[\sum_{r=\lfloor k/2 \rfloor + 1}^k \binom{k}{r} (p_i p_{iL} + p_j p_{jR})^{k-r} (p_i p_{iR} + p_j p_{jL})^r + \frac{1}{2} I_k \binom{k}{k/2} (p_i p_{iL} + p_j p_{jR})(p_i p_{iR} + p_j p_{jL})^{k/2} \right] \right) \end{aligned}$$

where I_k is defined in Lemma 2.3. Note that the factor multiplying $(p_i^* - p_j^*)$ is the probability we denoted by $l(j, i)$. Obviously, for a given value of x , the inequality

$$C_m(\text{DCS}|\mathbf{p}) - C(\text{OPT}|\mathbf{p}) \leq x \quad (2.7)$$

is satisfied, when for every pair of indices $1 \leq i < j \leq n$ we have

$$\begin{aligned} &2(p_i^* - p_j^*) \sum_{k=0}^m \binom{m}{k} (1 - p_i - p_j)^{m-k} \times \left[\sum_{r=\lfloor k/2 \rfloor + 1}^k \binom{k}{r} (p_i p_{iL} + p_j p_{jR})^{k-r} (p_i p_{iR} + p_j p_{jL})^r \right. \\ &\left. + \frac{1}{2} I_k \binom{k}{k/2} (p_i p_{iL} + p_j p_{jR})(p_i p_{iR} + p_j p_{jL})^{k/2} \right] - \frac{2x}{n(n-1)} \leq 0. \end{aligned} \quad (2.8)$$

Moreover, if p_i^* and p_j^* are close enough, so that

$$(p_i^* - p_j^*) - \frac{x}{n(n-1)} < 0,$$

then inequality (2.8) holds already at $m = 0$. (We use without proof the fact—proved in [8] for the original CS—that $C_m(\text{DCS}|\mathbf{p})$ is monotonically decreasing in m). Hence we may reduce the search only to those pairs i, j for which

$$(p_i^* - p_j^*) > \frac{x}{n(n-1)}. \quad (2.9)$$

Define, for a fixed pair i, j $q \equiv p_i p_{iL} + p_j p_{jR}$, and $p \equiv p_j p_{jL} + p_i p_{iR}$. From inequality (2.9) follows $0 < p < q < 1$. Clearly,

$$\sum_{r=\lfloor k/2 \rfloor + 1}^k \binom{k}{r} p^r q^{k-r} + \frac{1}{2} I_k \binom{k}{k/2} (pq)^{k/2} \leq 2^{k-1} \sum_{r \geq \lfloor k/2 \rfloor} p^r q^{k-r} = \frac{(2\sqrt{pq})^k}{2(1 - \frac{p}{q})}.$$

Therefore, it is sufficient to find, for each pair of indices, the first m for which the inequality

$$q(1 - p_i - p_j + 2\sqrt{pq})^m \leq \frac{4x}{n(n-1)}$$

is satisfied.

To obtain a universal bound, that holds for all $rpvs$, we look for a value of m that just exceeds

$$\max_{p,q} \frac{\log\left(\frac{4x}{qn(n-1)}\right)}{\log(1 - p - q + 2\sqrt{pq})}, \quad (2.10)$$

under the constraint

$$q - p > \frac{2x}{n(n-1)}.$$

The last inequality also indicates that the function we need to maximize is majorized by

$$g(p, q) \equiv \frac{\log 2}{\log(1 - p - q + 2\sqrt{pq})},$$

and since $g(p, q)$ obtains its maximum where $(\sqrt{q} - \sqrt{p})^2$ is minimal, we let $p = q - 2x/n(n-1)$.

Since $p + q = p_i + p_j \leq 1$, the extreme value is at $q^* = 1/2 + x/n(n-1)$, $p^* = 1/2 - x/n(n-1)$. Then

$$g(p^*, q^*) = \frac{\log 2}{\frac{1}{2} \log\left(1 - \frac{4x^2}{(n(n-1))^2}\right)} \approx -\frac{(n(n-1))^2}{2x^2} \log 2. \quad (2.11)$$

Substituting $x = n\alpha$ into (2.11), we derive the desired value of m^* . \square

The generality of the theorem and the liberality of some of the bounds in the proof should lead us to expect $C_{m^*}(DCS|\mathbf{p})$ to exceed $C(OPT|\mathbf{p})$ usually by far less than $n\alpha$.

The alternative way to reduce the size of the field allocated to the D_i 's is by modifying the DCS as follows:

For a chosen value of $r_{\max} \geq 1$, the *Limited DCS (LDCS)* updates D_i only during the first r_{\max} requests for R_i . When a difference count changes its value, the accessed record may be moved, either to the right or to the left, so as to keep the list arranged in nonincreasing order of the D_i values (note that even when r_{\max} is not small, the values of D_i may wobble around zero).

Theorem 2.6: The asymptotic average access cost to D under the LDCS with $r_{\max} = d$ is given by

$$C(LDCS|\mathbf{p}, d) = 1 + \sum_{1 \leq j < i \leq n} [(p_i p_{iL} + p_j p_{jR}) + 2(p_j^* - p_i^*) b(i, j)], \quad (2.12)$$

where

$$\begin{aligned}
 b(i, j) = & \sum_{k=0}^d \binom{d}{k} p_{iL}^k p_{iR}^{d-k} \sum_{r=0}^{k-1} \binom{d}{r} p_{jL}^r p_{jR}^{d-r} + \sum_{k=0}^d \binom{d}{k} \binom{d-1}{k} (p_{iL} p_{jL})^k (p_{iR} p_{jR})^{d-k} \\
 & + \frac{\sum_{k=0}^d \binom{d}{k}^2 \left(\frac{2k}{d} - 1\right) (p_{iL} p_{jL})^k (p_{iR} p_{jR})^{d-k} \left(s(d, a) - \frac{1}{2} \binom{2d}{d} a^d\right)}{(1+a)^{2d}}, \tag{2.13}
 \end{aligned}$$

$$\text{where } a \equiv \frac{p_j}{p_i} \quad \text{and } s(d, a) \equiv \sum_{r=0}^d \binom{2d}{r} a^r.$$

We use in the proof the following combinatorial identity:

Lemma 2.7: For any $d \geq 1$ and $0 \leq p, q \leq 1$,

$$\sum_{l \geq 2d} \frac{\binom{l-1}{d-1} q^d p^{l-d}}{(p+q)^l} = \frac{\sum_{r=d+1}^{2d} \binom{2d}{r} p^r q^{2d-r} + \frac{1}{2} \binom{2d}{d} (pq)^d}{(p+q)^{2d}} \tag{2.14}$$

Proof: To prove (2.14), it is sufficient to show that

$$\sum_{r=0}^d \binom{r+d-1}{d-1} \left(\frac{q}{p+q}\right)^d \left(\frac{p}{p+q}\right)^r = \sum_{r=0}^d \binom{2d}{r} \left(\frac{p}{p+q}\right)^r \left(\frac{q}{p+q}\right)^{2d-r}. \tag{2.15}$$

Assume we have an urn, and the probability of finding in it a black (white) ball is $q/(p+q)$ ($p/(p+q)$, respectively). From this urn we draw up to $2d$ balls with replacement. Equation (2.15) displays the probability of getting at most d white balls in two ways: The right-hand side is the expression given by the binomial distribution $\text{Bin}(2d, p/(p+q))$ for that probability. The left-hand side gives an alternative way of accepting at most d white balls: we stop drawing as soon as the number of black balls drawn out reaches d . Hence, the total number of balls drawn in this way may be in the range $[d, 2d]$, and the summand is the probability of drawing d blacks in $r+d$ trials, with the last one being black². \square

Proof of Theorem 2.6: The expression in (2.12) is equivalent to (2.4), where $b(i, j)$ replaces $l(i, j)$ defined for DCS . Since both D_i and D_j are determined by the first r_{\max} references to each of the records, $D_i > D_j \iff C_{iL} > C_{jL}$.

Now, let

- m = number of references to the list, after which both $C_{iL} + C_{iR}$ and $C_{jL} + C_{jR}$ are $\geq r_{\max}$, (necessarily the m th reference brings the total count of one of these records to r_{\max}),
- l = number of requests to both R_i and R_j by the m th reference, and $k = C_{iL}$.

Then,

$$\begin{aligned}
 b(i, j) = & \text{Prob}_{LDCS} \{ R_i \text{ is to the left of } R_j \text{ after the } m\text{th request} \} \\
 = & \sum_{m \geq 2d} \sum_{l=2d}^m \binom{m-1}{l-1} \binom{l-1}{d-1} (1-p_i-p_j)^{m-l} \\
 & \times [p_i^d p_j^{l-d} \text{Pr}_i \{ C_{iL} \geq C_{jL} \} + p_j^d p_i^{l-d} \text{Pr}_j \{ C_{iL} \geq C_{jL} \}], \tag{2.16}
 \end{aligned}$$

where Pr_i stands for the probability of the indicated event when R_i was the target of the m th reference, and is given by

²Another way of showing (2.14) is by comparing the coefficients of $(q/p)^i$, for any i in the range $[-d, 0]$, on both sides of the equation.

$$\Pr_i\{C_{iL} \geq C_{jL}\} = \sum_{k=0}^d \binom{d}{k} p_{iL}^k p_{iR}^{d-k} \sum_{r=0}^{k-1} \binom{d}{r} p_{jL}^r p_{jR}^{d-r} + \sum_{k=0}^d \binom{d}{k} \binom{d-1}{k} (p_{iL} p_{jL})^k (p_{iR} p_{jR})^{d-k} \quad (2.17)$$

and similarly

$$\Pr_j\{C_{iL} \geq C_{jL}\} = \sum_{k=0}^d \binom{d}{k} p_{iL}^k p_{iR}^{d-k} \sum_{r=0}^{k-1} \binom{d}{r} p_{jL}^r p_{jR}^{d-r} + \sum_{k=0}^d \binom{d}{k} \binom{d-1}{k-1} (p_{iL} p_{jL})^k (p_{iR} p_{jR})^{d-k} .$$

The first term in each of these expressions reflects the probability of $C_{iL} > C_{jL}$, and the second one equals the probability that R_i is to the left of R_j when $C_{iL} = C_{jL}$, under the specified termination event. Using the summation formula

$$\sum_{m \geq 2d} \sum_{l=2d}^m \binom{m-1}{l-1} (1-p_i-p_j)^{m-l} \binom{l-1}{d-1} p_j^d p_i^{l-d} = \sum_{l \geq 2d} \frac{\binom{l-1}{d-1} p_j^d p_i^{l-d}}{(p_i+p_j)^l} , \quad (2.18)$$

we eliminate one summation. Lemma 2.7, with some algebraic manipulations, gives the expression in (2.13) which is better for numerical evaluation. \square

Tables 2.1 and 2.2 present the ratio $C(LDCS|\mathbf{p}, d)/C(OPT|\mathbf{p})$ for various values of n and d with the p_i 's determined by Zipf's law and the geometric distribution. $C(LDCS|\mathbf{p}, d)$ was computed by (2.12) and (2.13). We allowed the p_{iL} s to be uniformly and independently distributed over $[0, 1]$. Evaluating the expectation of the right-hand side of equation (2.12) with respect to this distribution provides

$$C(LDCS|\mathbf{p}, d, \text{uniform } \mathbf{p}_L) = \frac{1}{2} (n+1) - \sum_{i \neq j} \frac{A_{ij}}{(p_i+p_j)^{2d}} \left[\frac{1}{2} \binom{2d}{d} (p_i p_j)^d + \sum_{r=d+1}^{2d} \binom{2d}{d} p_j^r (2_i^{2d-9}) \right]$$

where $A_{ij} = [(d+1)p_j + (d-1)p_i]/6(d+1)$. The integration required to evaluate $C(OPT|\mathbf{p})$ is

$\lambda \backslash d$	2	3	4	5
0.1	0.761	0.761	0.76	0.76
0.3	0.534	0.527	0.526	0.525
0.5	0.404	0.394	0.391	0.389
0.7	0.289	0.28	0.268	0.263
0.9	0.19	0.156	0.136	0.122
0.99	0.152	0.115	0.092	0.078

Table 2.1

The excess cost $C(LDCS|\mathbf{p}, d)/C(OPT|\mathbf{p}) - 1$ for a doubly linked list of 25 items and geometrical access probabilities:

$$p_i = (1-\lambda)\lambda^{i-1}/(1-\lambda^n) \text{ and } p_{iL} \sim U(0, 1).$$

$d \backslash n$	5	10	15	20	25
2	0.125	0.166	0.198	0.201	0.22
3	0.104	0.142	0.171	0.178	0.191
4	0.092	0.128	0.158	0.154	0.177
5	0.085	0.119	0.146	0.153	0.17

Table 2.2

The excess cost $C(LDCS|\mathbf{p}, d)/C(OPT|\mathbf{p}) - 1$ for lists of items accessed with a Zipf r_{pv} , given by $p_i = 1/iH_n$, and $p_{iL} \sim U(0, 1)$.

much harder, and its value was estimated from a set of 500 samples for each value of d and λ in Table 2. 1, and 500 samples for each value of d and n in Table 2. 2.

Both tables show a significant decrease in the above ratio already for low values of r_{\max} , though it is far less pronounced than with singly-linked lists.

LDCS is a generalization of *Move Once To the End (MOTE)*, which shifts each record precisely once – after the first access to it. The record is then moved towards the end from which its search started, only to precede those records, that were never requested.

Lemma 2.8:

$$C_m(\text{LDCS}|\mathbf{p}, 1) = C_m(\text{MOTE}|\mathbf{p}) = C_m(\text{MTE}|\mathbf{p}), \quad \forall m \geq 1. \quad (2.20)$$

Proof: Let $r_i = (k_i, u_i)$ be the i th reference, where $k_i \in (1, \dots, n)$, and $u_i \in \{L, R\}$. The arrangement obtained under *MTE* after the sequence r_1, \dots, r_m , is also produced by *MOTE* – following the sequence r_m, \dots, r_1 . Due to the independence of the r_i 's both events have the same probability, $\prod_{i=1}^m p_{r_i} p_{r_i u_i}$. \square

Hence, using Lemma 2. 2, we find

Corollary 2.9: For any $n \geq 2$, a probabilities vector \mathbf{p} and $d \equiv r_{\max} \geq 2$,

$$\frac{D(\text{LDCS}|\mathbf{p}, d)}{C(\text{OPT}|\mathbf{p})} \leq \min\left(2, 1 + \max_{1 \leq j < i \leq n} \left(\frac{p_j p_{jL} + p_i p_{iR}}{p_i p_{iL} + p_j p_{jR}} - 1 \right) b(i, j) \right), \quad (2.21)$$

with $b(i, j)$ as defined in Theorem 2. 6.

3. Limited Counters for Multiple Lists

Assume a set of n elements is maintained in k linear lists. Each request involves an extraction of the specified item, after which it is moved to the front of one of the lists. As before, we assume the i th element E_i , $1 \leq i \leq n$, may be requested with a fixed probability p_i ($\sum_{i=1}^n p_i = 1$).

We limit the discussion to a special case, where spaces are not compacted after an item is extracted (even if it is in the first position). This model fits for example update activities of a WORM, such as an optical disk. Thus the cost of a single access is the number of items and spaces encountered along a sequential search starting from the head of the list, until the specified item is reached.

Courcoubetis and Weber [4] study that model, and show, that if the retrieval probabilities are known, and the number of required sublists is prescribed, then the optimal policy first sorts the elements in non-increasing order of their access probabilities, and then partitions the sorted list into k sublists. The sublists remain fixed: when an element E_i is requested, it is shifted to the front of its own sublist, adding 1 to the cost of retrieving each of the other elements in that sublist. Observe that in the limit, all the access costs can be ‘charged’ to these additions. Hence,

Lemma 3.1: ([4]) Finding an optimal partition for the above model is equivalent to dividing n elements to k distinct sets l_1, \dots, l_k , such that

$$C(\text{OPT}|\mathbf{p}, k) = \sum_{i \in l_1} p_i \sum_{\substack{j \in l_1 \\ j \neq i}} 1 + \dots + \sum_{i \in l_k} p_i \sum_{\substack{j \in l_k \\ j \neq i}} 1 = |l_1| \sum_{i \in l_1} p_i + \dots + |l_k| \sum_{i \in l_k} p_i - 1 \quad (3.1)$$

is minimized, when the $\{p_i\}$ are renumbered so that $p_i \geq p_j$, $\forall 1 \leq i < j \leq n$. \square

Let the optimal partition be given by a sequence of $k + 1$ indices $1 = i_1 \leq i_2 \leq \dots \leq i_{k+1} = n$, such that

$$l_j = (i_j, i_j + 1, \dots, i_{j+1} - 1), \quad \forall 1 \leq j \leq k. \quad (3.2)$$

Consider the scenario where the ordered probabilities vector (possibly renumbered as above) $\mathbf{p} = (p_1, \dots, p_n)$ is **known up to a permutation**, i.e. we are given the distribution function which generates the retrieval sequences, but we do not know to which record each probability relates.

Therefore, an estimation of that mapping from the **ordered** vector \mathbf{p} to the set $\{1, \dots, n\}$ is required, for applying the initial partition on the elements.

We propose for this purpose a strategy based on the *LCS*:

We start with an arbitrary permutation of the elements, and perform a preliminary partition to k sublists, using the known set of indices. At the same time we keep an “image linear list” L , of all the n records in the same initial permutation. Each element E_i is allocated a reference counter C_i . E_i then participates in the reorganization process of L until C_i reaches a fixed value, c_{\max} . At that stage E_i receives its estimated index, i.e. its position in the arrangement of the elements used to create the sublists: the index is determined by the location E_i occupies in L . E_i then changes places with the element in the corresponding sublist previously allocated to that position. We only consider the limiting state, where all counters reach c_{\max} , with probability 1. The time to reach this state has a finite expectation.

We need a few technical results:

Lemma 3.2: The partition which minimizes $\sum_{i=1}^k |l_i| \sum_{j \in l_i} p_j$, satisfies

$$(a) \quad |l_i| \leq |l_{i+1}| \quad \forall \quad 1 \leq i \leq k-1. \quad (3.3)$$

$$(b) \quad \sum_{j \in l_i} p_j \geq \sum_{j \in l_{i+1}} p_j \quad \forall \quad 1 \leq i \leq k-1. \quad (3.4)$$

Proof: By way of contradiction (for (a) and (b) separately), using interchange arguments.

□

Lemma 3.3: For any ordered frequency vector $\bar{C}^{(m)}$, the function

$$f^{(m)}(k) = \sum_{i=1}^n p_i Pr_{CS}(\sigma_m(i)=k \mid \mathbf{p}, \bar{C}^{(m)}), \quad 1 \leq k \leq n$$

is monotone decreasing in k .

A proof is given in [9]; it is quite simple but requires surprisingly heavy notation, and is therefore omitted.

Corollary 3.4: If the list is organized by the *LCS* with $c_{\max} = c$, then

$$f(k) = \sum_{i=1}^n p_i Pr_{LCS}(\sigma(i)=k \mid \mathbf{p}, c), \quad 1 \leq k \leq n$$

is monotone decreasing in k .

Proof: The *LCS* with any $c \geq 1$ arranges the list in the limiting state as the *CS* would order the records after the m th request, that achieves the frequency vector $\bar{C}^{(m)} = (c, \dots, c)$, and the limits c are reached at the same order. Hence we can use Lemma 3.3. □

Let $C(MLCS|\mathbf{p}, k, c)$ and $C(OPT|\mathbf{p}, k)$ denote the asymptotic average access cost for retrieving an element from the k -sublist structure, where the partitions are performed by using the *LCS* with $c_{\max} = c$ and the optimal partition, respectively. We can relate them when certain natural conditions on the optimal partition hold, and k is sufficiently large (roughly, at least \sqrt{n}).

Theorem 3.5: For $p_1 \geq p_2 \geq \dots \geq p_n$, if the following conditions hold:

- (i) The number of sublists, k , is large enough: $2n \leq k(k+1)$, and
- (ii) the optimal partition is such that there exists a value $t \in [1, k]$, where

$$i \geq |l_i| \quad \forall 1 \leq i \leq t, \quad (3.5)$$

then

$$\frac{C(MLCS|\mathbf{p}, k, c)}{C(OPT|\mathbf{p}, k)} \leq (1 + x_c) \times \frac{\sum_{i=1}^k i \sum_{j \in l_i} p_j}{\sum_{i=1}^k |l_i| \sum_{j \in l_i} p_j} \quad (3.6)$$

where

$$x_c = \max_{1 \leq j < i \leq n} \frac{(a_{ij} - 1) \left[\sum_{r=0}^c \binom{2c}{r} a_{ij}^r - \frac{1}{2} \binom{2c}{c} a_{ij}^c \right]}{(1 + a_{ij})^2}, \quad a_{ij} = \frac{p_j}{p_i} \geq 1. \quad (3.7)$$

Proof: Observe, that $C(LCS|\mathbf{p}, c)$ for a list of k elements, may be written as

$$C(LCS|\mathbf{p}, c) = \sum_{i=1}^k i E[p_i], \quad (3.8)$$

where $E[p_i] = \sum_{j=1}^k p_j \text{Prob}_{LCS} \{R_j \text{ is } i\text{th in the limiting state}\}$ – the expected access probability of the record in position i .

We first collapse the rpv to an rpv of a list of k “super-elements” and use the notation π_i for $\sum_{j \in l_i} p_j$. Then, by Corollary 3.4,

$$\pi_i \geq \pi_{i+1} \implies E[\pi_i] \geq E[\pi_{i+1}] \quad \forall 1 \leq i \leq k-1.$$

Now, if there exists $1 \leq t \leq k$ satisfying the inequalities (3.5), then by Lemma 3.2(b)

$$\sum_{i=1}^t (i - |l_i|) E[\pi_i] \geq \sum_{i=1}^t (i - |l_i|) E[\pi_t] \geq \sum_{i=t+1}^k (|l_i| - i) E[\pi_{t+1}] \geq \sum_{i=t+1}^k (|l_i| - i) E[\pi_i]$$

is also satisfied, and so we get

$$\begin{aligned} C(LCS|\mathbf{p}, c) - C(MLCS|\mathbf{p}, k, c) &= \sum_{i=1}^k i E[\pi_i] - \sum_{i=1}^k |l_i| E[\pi_i] \\ &= \sum_{i=1}^t (i - |l_i|) E[\pi_i] - \sum_{i=t+1}^k (|l_i| - i) E[\pi_i] \geq 0. \end{aligned}$$

Hence, writing

$$\frac{C(MLCS|\mathbf{p}, k, c)}{C(OPT|\mathbf{p}, k)} = \frac{C(MLCS|\mathbf{p}, k, c)}{C(LCS|\mathbf{p}, c)} \cdot \frac{C(LCS|\mathbf{p}, c)}{C(OPT|\mathbf{p})} \cdot \frac{C(OPT|\mathbf{p})}{C(OPT|\mathbf{p}, k)},$$

we get the bound in the Theorem. \square

Comment: Typically, distributions have to be fairly skew to satisfy condition (ii). A marginal case occurs for example when $\mathbf{p} = (0.4, 0.3, 0.1, 0.1, 0.05, 0.05)$. When $k=3$ we find the optimal partition is $l_i = i$.

The quantity x_c of equation (3.7) is bounded from above by the right-hand-side of equation (1.3). For the above example of \mathbf{p} we find that x_c , for $1 \leq c \leq 8$ has the values

$$x_c = (0.7778, 0.3125, 0.2099, 0.1733, 0.1448, 0.1221, 0.1036, 0.09529). \quad (3.9)$$

For a distribution which is even more skew, $\mathbf{p} = (0.6, 0.3, 0.04, 0.03, 0.02, 0.01)$, we find

$$x_c = (0.9672, 0.3125, 0.2099, 0.1733, 0.1448, 0.1233, 0.1144, 0.1066). \quad (3.10)$$

Curiously enough, the values here for $c = 4$ coincide (at this precision) with the absolute bound in equation (1.3). \square

4. Disk Rearrangement Using Limited Frequency Counts

A major component in the access time to auxiliary storage is seek time, which is directly related to the length of travel the read/write head makes when moving to a requested position. We discuss the problem of ordering records or files on cylinders of a disk, so as to minimize this motion.

The access model is defined as follows:

A set of n records is ordered by its access probabilities $p_1 \geq p_2 \geq \dots \geq p_n$, ($\sum_i p_i = 1$). The records are placed on n separate cylinders of a disk, and the head moves from one location to another, to service the requests.

The objective is finding an arrangement which minimizes the expected distance traveled by the head, when successive requests are serviced. Assuming the *irm*, the cost function for a single access is given by

$$E[D|\mathbf{p}] = \sum_{i,j} p_i p_j d(i, j),$$

where $d(i, j)$ is the distance from R_i to R_j , which depends on the permutation of the records, as well as on the metric used.

It is known [2], that the above problem is intractable when accesses are not probabilistically independent. However, if the requests are generated independently of the past and the arrangement of the records, the permutation which minimizes the expected head travel is the ‘organ pipe’ (Fig. 4.1), which places the records in decreasing order of their access probabilities, starting at the middle cylinder and advancing alternately to both sides.

Hence, denoting by $E_{OPT}[D|\mathbf{p}]$ the minimal average seek length,

$$E_{OPT}[D|\mathbf{p}] = \sum_{i,j} p_i p_j \left(\lfloor \frac{|i-j|}{2} \rfloor (1 - \text{mod}(i+j, 2)) + \lfloor \frac{|i+j|}{2} \rfloor \text{mod}(i+j, 2) \right).$$

We assume the access probabilities are **unknown**, and the records are initially arranged in an

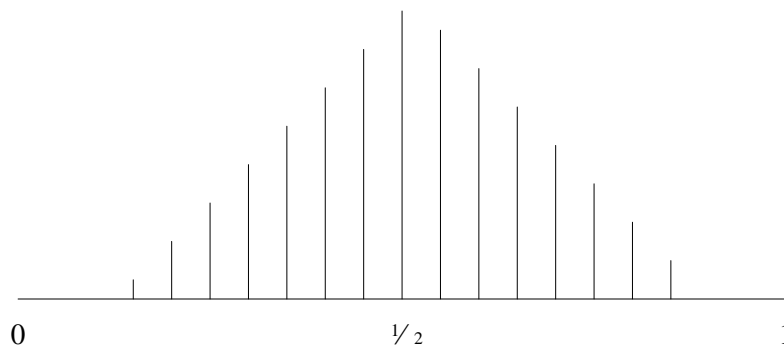


Fig 4.1: Sample ‘organ pipe’ arrangement
(vertical bar lengths are proportional to access probabilities)

arbitrary order on n cylinders of a disk. When the reference sequence is processed, the records are rearranged on another disk by the *CFR* (*Copy at First Request*) rule, defined as follows:

Each record is copied to the second disk, when it is first accessed. The records are placed in an ‘organ pipe’ order, according to their relative locations in the sequence, i.e. the record requested first is placed at the middle, the one accessed second – to its right, and the third – to its left, etc..

Theorem 4.1: The asymptotic average seek length under the above policy satisfies

$$E_{CFR}[D|\mathbf{p}] \leq 2 \sum_{k=1}^n k p_k . \tag{4.1}$$

Proof: Assume a given initial ordering of the records and a reference sequence, and that following the first reference to each record, it is copied twice: once to a second disk, by the *CFR*, and once to a linear list, arranged by the *LCS* with $c = 1$.

Let d_i and l_i denote the distance of R_i from the middle cylinder on the second disk, and the position of R_i in the linear list respectively. Then, $d_i = \lfloor \frac{1}{2} l_i \rfloor$. The distance traveled for each request from R_j to R_i may be shown to be at most $d_j + d_i$: we will certainly do no worse than when requiring the head to visit the middle position between successive references.

Let $\mathbf{i} = (i_1, \dots, i_k)$ be a sequence of k references, then the expected cost generated by this sequence satisfies

$$E_{CFR}[D|\mathbf{p}, \mathbf{i}] = \sum_{j=1}^k E[d(i_{j-1}, i_j)] \leq \sum_{j=1}^k E[d_{i_{j-1}} + d_{i_j}],$$

where the expectation is over all orderings of the list elements on the disk, and i_0 is the initial position of the arm. We may assume it is the middle cylinder – it would not matter in the long run, and then

$$E_{CFR}[D|\mathbf{p}, \mathbf{i}] \leq 2 \sum_{j=1}^k E[d_{i_j}] \leq \sum_{j=1}^k E[l_{i_j}] = E_{LCS}[D|\mathbf{p}, \mathbf{i}; c = 1]. \tag{4.2}$$

Using (1.2) and the rightmost inequality in (1.1) we obtain the inequality of the Theorem. \square

The *CFR* may be generalized to the *Copy at the cth request* (*CcR*), under which a record is copied to the second disk only when it has accumulated c requests. Along the reference sequence, each record acquires its position in the *organ pipe*, by keeping a linear list of the keys, managed by CS. A record copied to the second disk is deleted from the list. When the list empties, the second disk becomes the permanent storage device for the records, and the first disk may be

$n \setminus c$	1	2	3	4
10	3.185	2.095	1.939	1.869
20	2.861	1.882	1.742	1.679
50	2.617	1.721	1.593	1.535
100	2.504	1.647	1.524	1.469
200	2.425	1.595	1.476	1.422
300	2.389	1.572	1.454	1.402

Table 4.1
 Bounds on the excess cost $E_{CFR}[D|\mathbf{p}]/E_{OPT}[D|\mathbf{p}]$ and $E_{CcR}[D|\mathbf{p}, c]/E_{OPT}[D|\mathbf{p}]$ for lists of items accessed with a Zipf rpv , given by $p_i = 1/iH_n$.

overwritten.

Corollary 4.2: For any $c \geq 2$

$$E_{CCR}[D|\mathbf{p}, c] \leq (1 + x_c) \sum_{i=1}^n i p_i, \quad (4.3)$$

where x_c is defined in equation (3.7). \square

Table 4.1 presents bounds on the ratio $E_{CFR}[D|\mathbf{p}]/E_{OPT}[D|\mathbf{p}]$ and $E_{CCR}[D|\mathbf{p}, c]/E_{OPT}[D|\mathbf{p}]$ for access probabilities distributed by Zipf's law, computed by equations(4.1) and (4.3).

ACKNOWLEDGMENT

The idea of using the formalism of reorganizing linear lists to handle disk reorganization was suggested by prof. Alon Itai.

REFERENCES

- [1] Bellow M.E.: An investigation of self-organizing heuristics for doubly linked lists. *Allerton 25th Conf.*, pp. 64–65, 1987.
- [2] S.D. Carson, P. Vongsathorn: Error bounds on disk arrangement using frequency information. *Inf. Proc. Lett.*, **31**, pp. 209–213, (1989).
- [3] F.R.K. Chung, D.J. Hajela, P.D. Seymour: Self-organizing sequential search and Hilbert's inequalities. *J. Comp. System Sc.*, **36**, pp. 148–157, (1988).
- [4] C. Courcoubetis, R.R. Weber: The Move-To-Front rule for multiple lists. *Probab. Engin. and Inf. Sc.*, **4**, pp. 19–27, (1990).
- [5] P. Flajolet, D. Gardy, L. Thimonier: The birthday paradox, coupon collectors, caching algorithms and self-organizing search lists. *INRIA*, RR#724, (1987).
- [6] J.H. Hester, D.S. Hirschberg: Self-organizing linear search. *ACM Comput. Surv.* **17**, #3, pp. 295–312, (1985).
- [7] M. Hofri, H. Shachnai: On the optimality of the counter scheme for dynamic linear lists. *Inf. Proc. Lett.*, **37**, 175–179, (1991).
- [8] M. Hofri, H. Shachnai: Self-organizing lists and independent references – a statistical synergy. *Jour. of Alg.*, **12**, 533–555, (1991).
- [9] M. Hofri, H. Shachnai: Partial information and its limited utility – the case of the reorganizing lists. TR #UH–CS–91–11, The Department of Computer Science, The University of Houston, May 1991.
- [10] E. Makinen, On linear search heuristics. *Inf. Proc. Lett.*, **29**, pp. 35–36, (1988).
- [11] D. Matthews, D. Rothen, E. Bretholz E.: Self-organizing doubly linked lists. *Intern. J. of Computer Math.*, **8**, pp. 99–106, (1980).
- [12] B.J. Oommen, E.R. Hansen: List organizing strategies using stochastic move-to-front and stochastic move-to-rear operations. *SIAM J. Comput.* **16**, #4, pp. 705–716, (1987).
- [13] R. Rivest: On self-organizing sequential search heuristics. *Commun. ACM*, **19**, #2, 63–67, (1976).