SELF-ORGANIZING LISTS AND INDEPENDENT REFERENCES - A STATISTICAL SYNERGY

by

M. Hofri and H. Shachnai

Technical Report #524

October 1988
ABSTRACT

Let $R_1, \ldots, R_n$ be a linear list of $n$ elements. We assume the independent reference model, with a fixed but unknown access probability vector. The problem of dynamically reorganizing the list, on the basis of accrued references, with the objective of minimizing the expected access-cost is surveyed. The Counter Scheme (CS) is known to be asymptotically optimal for this purpose. This paper concentrates on the CS, with the aim of reducing its storage requirements. We start with a detailed exposition of its cost function. We then point out that it interacts with the access model to produce some remarkable synergistic effects. These make it possible to use very effective 'truncated versions' of the CS, which have very modest space requirements. The versions we consider are:

1. Introduction

A linear list is a data structure in which there is little structure. Thus it is simple to build and maintain, with the consequent penalty that it is relatively inefficient in use. Nevertheless, under certain (quite common, actually) circumstances, it is the structure of choice.

We assume the following layout:

A list $L$ contains $n$ records ($R_i$, $1 \leq i \leq n$), that are uniquely identified by their keys ($K_i$). The content of the list is assumed to remain unchanged over time.

The list structure implies the following sequential search access scheme: in order to access $R_i$, the keys in positions 1, 2, ..., are compared successively against $K_i$ until an equality is obtained. The number of these comparisons is defined as the cost of the access.
We assume that whichever algorithm that needs to reference the list generates access requests according to the following reference model: each request is for the record \( R_i \) with a time-homogeneous probability \( p_i > 0, 1 \leq i \leq n \). This holds true independently of the list order and the history of past accesses. Such a scheme is known as the (stationary) independent-reference model (irm).

A natural objective is to have the smallest possible cost per access. If the reference-probability vector \( \{ p_i \} \) were known, the optimal policy would be to keep the list sorted by the size of the record reference-probabilities, in non-increasing, static order.

The adjective "self-organizing" in the title bespeaks of the situation where the \( \text{rpv} \) is not known, and the access mechanism activates periodically – typically following each access – a procedure that may change the order of the records, based on the reference history.

Various reorganization rules, or heuristics, have been proposed for this purpose. Rivest (1976) calls them "permutation rules". Their purpose is to try and decrease, using information produced by the reference string up to any given time, the expected cost of future references. Hester and Hirschberg (1985) survey most of the work in this area published by 1983.

We describe below three of the better known reorganization- or update-schemes. They are activated each reference. To describe their operation we use the symbol \( \sigma(i) \) to denote the location of \( R_i \) in the list. The state of the list is correspondingly said to be the permutation \( \sigma \).

**Move to the front method (MTF):** Following an access to record \( R_i \), in position \( \sigma(i) \), it is placed at the head of the list, pushing the records in locations 1, 2, \( \ldots \), \( \sigma(i) - 1 \) one step back. This heuristic has been extensively studied. Two measures of its operation were examined. Most researchers referred to its asymptotic cost per access, under various \( \text{rpv's} \) [Bitner (1979), Burville and Kingman (1973), Hendricks (1976), Knuth (1973), McCabe (1965) and Rivest (1976)]. They show that the limiting expected cost per access is given by

\[
E_{MTF}(C) = 1 + 2 \sum_{1 \leq i < j \leq n} \frac{p_i p_j}{p_i + p_j}.
\]

Bitner (1979) considered also the number of requests required until the cost function converges to its asymptotic value. More precisely, he defined the "overwork" (OW) for any heuristic \( H \) as

\[
OW_H \equiv \sum_{m \geq 0} (E_H^{(m)}(C) - E_H(C)),
\]

where \( E_H^{(m)}(C) \) is the expected cost of the \( m + 1 \)st access, assuming the initial state of the list is any of the \( n! \) possible permutations with probability \( 1/n! \), and \( E_H(C) = \lim_{m \to \infty} E_H^{(m)}(C) \). The expectation is evaluated with respect to the initial state and the history of references. Notice that in general it need not even be finite for all \( H \); however, in all the cases we shall examine, the list state can be described by a first-order Markov chain, and then \( OW_H \) is guaranteed to be finite. Bitner used the overwork measure to compare MTF to other known heuristics.

Recently, there have appeared some work [Sleator and Tarjan (1985), Bentley and McGeogh (1985)] that considers not the expected cost of MTF, but rather the worst possible cost, when averaged (or – as called in that context – amortized) over a long reference sequence. Since the irm is irrelevant there, these works are outside the scope of the present paper. For one reason or another, MTF is the heuristic that has been most assiduously studied, and is commonly used as a yardstick by which other rules are measured. We shall follow this usage to some extent.
Transposition rule (TR): A record \( R_i \), when accessed at location \( \sigma(i) > 1 \) is transposed with its neighbor in location \( \sigma(i) - 1 \), getting thus one step closer to the head of the list. This rule was found to be more efficient asymptotically than the MTF scheme [Rivest (1976)]. However, the rate of convergence to its desirable limiting cost is known to be lower [Bitner (1979), Hester and Hirschberg (1985)].

Counter Scheme (CS): Each of the records \( R_i \) is associated with a frequency counter \( c_i \), which is incremented whenever the record is accessed. The record is then moved forward (if needed) to keep the list records sorted in non-increasing order of their counters.

Let \( m \) be the total number of requests made to the list, then by the strong law of large numbers, for all \( 1 \leq i \leq n \)

\[
\frac{c_i}{m} \xrightarrow{m \to \infty} p_i.
\]

This implies that, in the limit, the list will be optimally ordered. Clearly, this is the best among the asymptotic costs of all possible heuristics. This advantage is not only a property of the limiting cost: Lam et al. (1981) have shown, in analyzing a slightly generalized CS (of which the above CS is a special case), that the expected access-cost under the GCS is lower than that produced by any permutation rule which is not a CS, at every stage. In spite of its desirable cost-function evolution, the CS has not attracted much attention in the published literature. The reason appears to be real enough – compared with MTF, TR and similar varieties, it is the only one requiring the maintenance of additional information – the counters \( c_i \).

Also, when compared with certain rules that do maintain some administrative records, the CS has been denigrated for having "infinite space complexity", meaning that it requires an unbounded amount of storage space, when the rule is implemented indefinitely, since the counters may grow beyond any limit. Bitner (1979) suggested a few variations on the CS in order to overcome this problem. First, he considered the limited-difference rule, which stores the differences between successive counters and, in addition, imposes an upper bound on the size of the stored values. Once a certain difference reaches the limit, the stored value remains unchanged, unless the difference decreases. This version of CS does not converge to the optimal ordering, but its asymptotic access-cost approaches the minimum as the upper bound on the difference fields is increased (and so does their space requirement).

Secondly, he examined two classes of rules which are combinations of CS with other permutation rules. In 'wait c, move and clear', each record \( R_i \) has a counter \( c_i \) which is initially set to zero. When a counter \( c_i \) obtains the maximal value \( c \), the record \( R_i \) is moved according to the specified permutation rule (MTF, TR, etc.) and all the counters are reset. The class of 'wait c and move' rules is basically the same with the exception that when a record \( R_i \) is moved, only \( c_i \) is reset. Both classes improve the asymptotic access-costs of the original rules (for example 'wait c and MTF' rule performs better – in terms of the limiting expected access-cost – than MTF), but as Bitner points out, their asymptotic cost is higher than that of the limited difference rule and their rate of convergence is low.

The latter two modifications may be viewed as an attempt to correct the main shortcoming of MTF: when a "rare" record (one having a low \( p_i \)) is referenced, it is moved all the way to the front, and will take a long time to dribble back to an inexpensive position.

Oommen and Hansen (1987) suggested the seemingly quaint "stochastic move to the rear" scheme, according to which an accessed record \( R_j \) is moved to the rear of the list with probability \( q_j \). The value of \( q_j \) decreases each time the record is accessed. Eventually, as the \( q_j \)'s become negligible, the list reaches a steady state, in the sense that the records tend to stay in place when accessed. Since records
with low \( p_i \) are accessed less frequently, they are the ones expected to keep moving at later times, and ultimately land at the rear of the list. Instead of keeping the values of the \( q_j \)'s, it is sufficient to maintain the number of accesses made to each of the records. The upper limit on the size of counter fields is determined by the assumption that the \( q_j \)'s cannot assume a value smaller than some \( 0 < x_{\text{min}} < 1 \). The value of \( x_{\text{min}} \) was chosen by the authors to be the "smallest positive real number produced by a random-numbers generator accessible to the user". Hence the operational and storage overhead is like that of a bounded CS. One disadvantage of this method is that the greater part of the time the scheme is active, before settling down, would be spent on locating properly those records with the smaller \( p_i \)'s. Those probabilities contribute little to the expected access-cost and therefore need not be known too precisely, or sorted with much circumspection; it is enough if they are at the rear of the list and "out of the way" most of the time. Moreover, during the initial phase, when popular records still land often at the rear, the cost here is abysmal.

The last point touches on the main thesis of the present work: a reorganization rule should estimate the correct order of the important records, and do it as promptly as possible. With this in mind, recall that frequencies are sufficient statistics for the estimation of the \( r_p v \) of a multinomial process. Moreover, they are also the (most) efficient ones.

There is a substantial amount of published statistical work on discriminating multinomial probabilities, under various requirements [a comprehensive account is provided in Gibbons et al. (1977)]. Here, however, the requirement is merely to minimize the expected value \( E[\sigma(I)] \), where \( I \) is the index of the accessed record. The \( r_m \) and this cost function interact and give rise to the following synergistic effects:

- Broadly speaking, only the records with large probabilities make any significant contribution to the cost function. However, it is precisely those records for which the counters increase fast and provide good estimates early on.

- Actually it is not even the probabilities proper we need to estimate, but only their sorted order. Now, when the difference between two probabilities, say \( p_i - p_j > 0 \) is substantial, again after a relatively short reference sequence there will be a very small probability for the event \( \{ c_i < c_j \} \). On the other hand, when \( p_i - p_j > 0 \) is very small, while it is true that the number of references required to order them correctly, with high probability, is huge [Gibbons et al. (1977)], correspondingly the penalty of incorrect order is minute (even when \( p_i \) and \( p_j \) proper are not small — since it is proportional to \( p_i - p_j \)). We conclude then that the CS is well-positioned to utilize this synergy.

The rest of the paper justifies and quantifies in various ways these claims, and outlines efficient procedures to take advantage of their promise.

This paper presents two ways for limiting the space complexity of the counter scheme without damaging its superiority over other permutation rules.

One idea is to fix a maximal value \( c_{\text{max}} \) which the counters may not exceed. The other method stops the list-reorganization process after a finite, predetermined number of requests. This second approach not only bounds the space requirements, but also sets an upper bound on the time required to implement the counter scheme. No bound of that kind has been presented in previous work, for any other permutation rule. Limiting the activation period is attractive in itself, by reducing the overhead involved in the usage of these heuristics.
In Section 2 we analyze in detail the characteristics of CS. We define the expected access-cost to the list of each request and show it to be monotonic in \( m \), the number of references and, of course, that it converges to the expected access-cost of the optimal ordering as \( m \to \infty \).

In Section 3 we examine this convergence in closer detail, and show that for any \( \alpha > 0 \) the expected access-cost gets to within a factor of \( 1 + \alpha \) of the minimum (optimal) expected cost within a finite number of requests that we can bound.

We derive two estimates for the required number of references. The first depends closely on the values of the access probabilities, and then we also exhibit a "non-parametric" bound: while it depends on the values of \( \alpha \) and \( n \), it does not depend on the rpv \((p_1, \cdots, p_n)\). This fact which may appear strange at first results from the structure of the cost function; we like to think of it as the essence of the above synergy.

Finally, we present a few bounds on the 'activation period' of the scheme for specific classes of distribution functions (geometric, Zipf's law, ...). These bounds are useful when there is some a priori information on the rpv.

In Section 4 we present a variation of CS which limits the space allocated to the counter fields by limiting their size to \( c_i \leq c_{\max} \), where \( c_{\max} \) may be any integer larger than or equal to 1. Its performance is shown to compare surprisingly well with the optimal static ordering, even for quite small \( c_{\max} \). Moreover, we show that even for \( c_{\max} = 1 \), the expected access-cost for each request is as good as that of the MTF rule, which is considerably more expensive to implement.

We conclude with a discussion of the assumptions of the model and point out that there are many questions concerning it, and its immediate extensions, that are as yet unanswered.

2. Definitions and Preliminary Results

The main measure we use to compare the CS to other heuristics (and, in particular, to the optimal static ordering) is the expected access-cost at each stage.

Let \((p_1, \cdots, p_n)\) be the reference probabilities vector (rpv). Without loss of generality, we may assume a renumbering such that \( p_1 \geq p_2 \cdots \geq p_n \). Also, let \( E^{(m)}_{\Lambda}(C) \) denote the expected cost to access the list under the scheme \( \Lambda \) for the \( m + 1 \)st reference, \( m \geq 0 \).

**Theorem 2.1:** If the initial order of the list is random with uniform probability for each permutation of \( L \), then

\[
E^{(m)}_{\text{CS}}(C) = 1 + \sum_{i=1}^{n} p_i \sum_{j=1}^{m} \begin{pmatrix} m \\ k \\ end{pmatrix} (1-p_i-p_j)^{m-k} \left( \sum_{r=k/2+1}^{k} \begin{pmatrix} k \\ r \\ end{pmatrix} p_j p_i^{k-r} + \frac{1}{2} I_k \right) (p_i p_j)^{k/2},
\]

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:** Showing equation (2.1) requires a straightforward computation, but also some tedious bookkeeping. Let us introduce a suitable notation:
\( \sigma^{(m)} \) - a permutation-valued random variable (rv); the state of the list following the \( m \)-th access (and update, if one was required).

\( H_i^{(m)} \) - the number of records in \( \sigma^{(m)} \) ahead of \( R_i \).

In terms of \( \{H_i^{(m)}, 1 \leq i \leq n\} \) the expected cost is

\[
E_{CS}^{(m)}(C) = 1 + \sum_{i=1}^{n} p_i E[H_i^{(m)}].
\]

(2.2)

Since only the first moment is to be computed, we can afford to adopt the same approach as in Bitner (1979) and evaluate the expected cost by summing over pairs of records. This requires writing \( H_i^{(m)} \) as a sum of binary rv's

\[
H_i^{(m)} = \sum_{j=1 \atop j \neq i}^{n} A_{ji}^{(m)},
\]

(2.3)

where \( A_{ji}^{(m)} \) is 1 (0) when \( R_j \) is ahead of (behind) \( R_i \) in \( \sigma^{(m)} \).

Now, \( A_{ji}^{(m)} \) has the distribution of the binary rv \( B(q_{ji}^{(m)}) \), where \( q_{ji}^{(m)} = \text{Prob}(R_j \text{ is ahead of } R_i \text{ in } \sigma^{(m)}) \), and from equations (2.2) and (2.3)

\[
E_{CS}^{(m)}(C) = 1 + \sum_{i=1}^{n} p_i \sum_{j=1 \atop j \neq i}^{n} q_{ji}^{(m)}.
\]

(2.4)

Let \( X_i^{(m)} \) be the number of times \( R_i \) is accessed in the first \( m \) references, and denote \( X_i^{(m)} + X_j^{(m)} \) by \( K_{ij}^{(m)} \). Clearly, \( K_{ij}^{(m)} \sim B(m, p_i + p_j) \), a binomial rv, and given that \( K_{ij}^{(m)} = k \) we also have \( X_i^{(m)} \sim B(k, p_i/(p_i + p_j)) \).

Using the CS means that given \( K_{ij}^{(m)} = k \), \( R_j \) is ahead of \( R_i \) whenever

\[
0 \leq k \leq m
\]

(2.5)

By definition,

\[
q_{ji}^{(m)} = \sum_{k=0}^{m} q_{ji}^{(m)} \cdot \text{Prob}(K_{ij}^{(m)} = k).
\]

(2.6)

Substituting (2.5) and (2.6) into (2.4) yields (2.1).  

\[ \Box \]

It will be convenient now to introduce some shorthand notation: For \( 1 \geq p_i \geq p_j \geq 0 \) (s.t. \( p_i + p_j \leq 1 \)) and the integers \( m, k \geq 0 \), define

\[
E(k; p_i, p_j) = p_i \sum_{r=\lfloor k/2 \rfloor + 1}^{k} \left[ \frac{k}{r} \right] p_i^{k-r} p_j^{r} + \frac{p_i + p_j}{2} \cdot I_k \cdot \left[ \frac{k}{k/2} \right] (p_i p_j)^{k/2} + p_j \sum_{r=\lfloor k/2 \rfloor + 1}^{k} \left[ \frac{k}{r} \right] p_i^{k-r} p_j^{r}.
\]

(2.7)
Thus,

\[ E_{\text{CS}}^{(m)}(C) = 1 + \sum_{1 \leq i < j \leq n} E^{(m)}(p_i, p_j). \]  

The following theorem brings a property of CS that should be intuitively obvious; the proof is however rather tedious, and relegated to the Appendix.

**Theorem 2.2:** For every multinomial distribution function given by the rpv \((P_1, \ldots, P_n)\), \(E_{\text{CS}}^{(m)}(C)\) is monotonically decreasing in \(m\), i.e.

\[ E_{\text{CS}}^{(m)}(C) \leq E_{\text{CS}}^{(m-1)}(C) \quad \text{for all } m \geq 1 \quad \Box \]  

Consider now the limiting state, when the number of accesses \(m\) increases. As mentioned above the convergence of the list state to the optimal order results from the strong law of large numbers. It may be shown however directly in the cost function, by letting \(m \to \infty\) in equations (2.7-8). This yields, for all \(1 \geq p_i, p_j \geq 0\),

\[ E^{(m)}(p_i, p_j) \to_{m \to \infty} p_j. \]  

Substituting this limit in equation (2.9) is enough in order to verify that

\[ \lim_{m \to \infty} E_{\text{CS}}^{(m)}(C) = \sum_{i=1}^{n} p_i = E_{\text{opt}}(C). \]  

### 3. A Stopping Point for the Counter-Scheme

In the previously published accounts of self-organizing lists it was tacitly assumed that any organizing method is to be implemented "till convergence to a steady state". In effect, this means letting an unbounded number of rearrangements of the list to be made, which at least in the case of CS is impractical and unnecessary. In fact, after a predictable (in a sense we discuss below) number of requests, the cost function is already close enough to the optimal cost, and any subsequent updates serve only to reduce it slowly, by an insignificant amount. This property of the cost function, which holds for any access probabilities vector \((P_1, \ldots, P_n)\), is the essence of the synergy of counter scheme that we discussed in Section 1. One can intuitively argue for it as follows: The utility of CS can be justified for two broad categories of rpv’s, or access distributions. For a 'steep' distribution function (one in which there is a substantial variation between the \(P_i\)s), the records with higher access probabilities are the first ones to organize properly, at the front, whereas the records which are less frequently accessed remain longer unordered in the rear of the list. But the latter contribute—precisely because they are rarely referenced—much less to the expected access-cost. Therefore, with high probability \(E_{\text{CS}}^{(m)}(C)\) reaches a low value at an early stage, even though only a small part of the records are expected to be correctly ordered by then. On the other hand, when the distribution function is flat, the rate of convergence to the optimal ordering is slow (due to the hard-to-detect slight differences between the access probabilities), but then, *any* ordering would provide a similar, relatively high, expected cost. Again, implementing the scheme for an infinite length of time would not improve much the expected access-cost computed nearly
Hence

The following theorem presents an upper bound on the number of requests required in order to guarantee that the expected cost has converged to within an $\alpha$-fraction of the minimal cost.

Theorem 3.1: Let $(p_1, \cdots, p_n), n \geq 2$ be a rpv by which a linear list is accessed as above. Then

$$E_{CS}^{(m)}(C) \leq (1 + \alpha)E_{opt}(C) \text{ for all } m \geq m^*$$

where $m^*$ is the integer part of

$$\frac{\max_{1 \leq i < j \leq n} 1}{2(2p-1)\alpha\left[q(1-p) + 2/n(n-1)\right]},$$

where $q \equiv p_i + p_j, p = p_i/q \geq 1/2, 1-p = p_j/q$, and the search for the maximum is over the entire rpv, so long as the probability pair $(p_i, p_j)$ satisfy

$$p > \frac{1}{2} + \frac{\alpha [1 + 4/n(n-1)q]}{1 + \alpha}.$$  

Proof: There is really no "natural" way to compare the two sides of relation (3.1), hence we proceed in a round-about way. As usual we use the renumbering $1 > p_1 \geq p_2 \cdots \geq p_n > 0$.

We need to show $\Delta^{(m)}(C) \equiv E_{CS}^{(m)}(C) - (1 + \alpha)E_{opt}(C) \leq 0$. Using equations (2.9) and (2.12) we write

$$\Delta^{(m)} = 1 + \sum_{1 \leq i < j \leq n} [E^{(m)}(p_i, p_j) - (1 + \alpha)p_j] - (1 + \alpha)$$

The dependence of this expression on $m$ is much too complex to tackle directly. We shall simplify our task in several stages; begin by bringing the whole of the right-hand side of (3.4) under the summation sign, by observing that

$$\sum_{1 \leq i < j \leq n} \frac{2}{n(n-1)} = 1.$$  

Hence

$$\Delta^{(m)} = \sum_{1 \leq i < j \leq n} \Delta_{ij}^{(m)} \equiv \sum_{1 \leq i < j \leq n} \left[E^{(m)}(p_i, p_j) - \frac{2\alpha}{n(n-1)} - (1 + \alpha)p_j\right].$$

Since we cannot obtain the minimal value of $m$ that makes the entire sum negative, we consider each term separately, and use for the Theorem the maximal value, where each individual term is negative. In each such term the pair of indices $i$ and $j$ is fixed; define $q \equiv p_i + p_j, p \equiv p_i/q \geq 1/2$. Corresponding to this notation we also write $\Delta^{(m)}(p, q), E^{(m)}(p, q)$ and $q^{k+1}E(k; p)$ for $\Delta_{ij}^{(m)}, E^{(m)}(p_i, p_j)$ and $E(k; p_i, p_j)$, respectively.

From the proof in the Appendix it follows that $\Delta^{(m)}(p, q)$ is monotonic nonincreasing in $m$. It would
The feasibility of this proof stems from the obliging manner equation (3.9) separates terms that do or do not depend on \( m \). Introducing this into the summation:

\[
G(x; p, q) = \sum_{m=0}^{x-1} 2(p-\frac{1}{2})^2 \sum_{k=0}^{m} \binom{m}{k}(1-q)^{m-k} q^{k+1} \sum_{r>k_1} \binom{2r}{r} p^r (1-p)^r
- x \alpha q (1-p) + \frac{2}{n(n-1)}
\]

For the sum, to be denoted by \( A \), we obtain a simple, if coarse bound by letting the summation extend to infinity (all the terms are positive):

\[
A \leq 2(p-\frac{1}{2})^2 \sum_{k=0}^{m} \sum_{m \geq k} \binom{m}{k}(1-q)^{m-k} q^{k+1} \sum_{r>k_1} \binom{2r}{r} p^r (1-p)^r
\]
Theorem 3.2: For an $n$-long linear list, the CS approaches the optimal cost to within a factor of $1 + \alpha$ after at most $m^\ast$ accesses, where

$$m^\ast = \frac{n(n-1)(1 + \alpha)^2}{16\alpha^2}.$$  \hfill (3.13)

Proof: All we need to do is to let $p$ and $q$ range over all their possible values in equation (3.2). The form of the bound there is such that by inspection this maximization can be done essentially independently.

For $p$ we let it approach $\frac{1}{2}$ as much as permitted by the lower bound specified there:

$$p^\ast = \frac{1}{2} + \frac{\alpha}{2} \left( \frac{1}{1 + \alpha} + \frac{4}{q(1 + \alpha)n(n-1)} \right),$$  \hfill (3.14)

and substituting this into (3.2) we find that we need

$$m^\ast = \max_{0 < q < 1} \left[ \frac{(1 + \alpha)^2}{2\alpha^2} \left( \frac{2}{2} + \frac{4}{n(n-1)} + \frac{8}{qn^2(n-1)^2} \right)^{-1} \right].$$  \hfill (3.15)

The maximum is immediately obtained at

$$q = \frac{4}{n(n-1)},$$  \hfill (3.16)

and the value obtained upon substitution is given in equation (3.13).  \hfill ⊓⊔

We now present
Theorem 3.3 (A specialization of Theorem 3.1):

a) If the access probabilities are determined by a geometric distribution with parameter \(0 < \lambda < 1\), that is:
\[
p_i = \frac{(1-\lambda)\lambda^{i-1}}{1-\lambda^n} = a_n \lambda^{i-1},
\]
then relation (3.1) holds for all \(m \geq m^*(\lambda)\), where
\[
m^*(\lambda) = \begin{cases} \frac{n(n-1)(1+\lambda)}{2\alpha(1-\lambda)[\frac{2\alpha\lambda}{\frac{n}{2} - \lambda/2 - \alpha\lambda} + 2]} & \lambda < \frac{1}{1+2\alpha}, \\ 0 & \text{otherwise.} \end{cases}
\]

Comment: When \(\lambda > 1/(1+2\alpha)\) the distribution is too flat for any gain, beyond the \(\alpha\) factor, to be achieved by a reorganization.

b) For an rpv governed by Zipf's law, \(p_i = 1/i^\alpha H_n\), \(1 \leq i \leq n\), relation (3.1) holds by the \(m^*-\)th request, where
\[
m^* = \frac{2i + 1}{2\alpha [\frac{1}{H_n(i+1)} + \frac{2}{n(n-1)}]}
\]
and \(i\) is the largest integer in the range \([1, n]\) that does not exceed
\[
i^* = \frac{-(1 + \alpha/B) + \sqrt{(1 + \alpha/B)^2 + 2/B}}{2} B = \frac{2\alpha H_n}{n(n-1)}.
\]

Proof: Both proofs consist in observing that the maximum in equation (3.2) will be obtained by picking adjacent indices (i.e., \(j = i+1\)), and then finding the largest value of \(i\) (that leads to the smallest \(q\)) that still satisfies the bound (3.3). We leave out the details.

Comment: While Theorems 3.1 and 3.2 fill the claim we made, they were in an important sense disappointing: this is particularly evident in the way the bound of Theorem 3.2 (and equations (3.18) and (3.19)) were obtained: they were determined by a \(p\) chosen for its closeness to one half, and typically, a very low value for \(q\). However, our understanding of this process tells us that it is precisely with pairs of records that produce such values, that the correct order is quite unimportant — still, in the only general approach that was open to us, such bounds are handed out.

An idea as to how unsatisfactory these bounds really are is furnished by a case which is so simple that inequality (3.1) can be solved directly for \(m^*\):

Theorem 3.4: If the rpv is \(l\)-uniform, i.e
\[
p_i = \begin{cases} \frac{1}{l} & 1 \leq i \leq l \\ 0 & l < i \leq n \end{cases}
\]
then inequality (3.1) is satisfied for all \(\alpha > 0\) with
A "uniform" bound, that is useful when the value of $l$ is not known \textit{a priori}, and that holds for all $\alpha \geq 0.01$ is $m^* = \alpha n$, $e = 2.7182818...$

Proof: For $rpv$'s in that category, equations (2.7-9) give

$$E_{CS}^{(m)}(C) = 1 + \sum_{1 \leq i < j \leq l} \sum_{k=0}^{m} \left[ \frac{m}{k} \right] \left( 1 - \frac{2}{l} \right)^{m-k} \frac{2^k}{l} \cdot \frac{1}{l+1} \sum_{j=1}^{l} \frac{1}{2l} (1 - \frac{1}{l})^m$$

$$= l + \frac{1}{2} \left[ \frac{(n - l)(1 - \frac{1}{l})^m}{2} \right].$$

Thus,

$$E_{CS}^{(m)}(C) = E_{opt}(C) = 1 + \frac{(n - l)(1 - \frac{1}{l})^m}{l+1},$$

and equation (3.22) follows. Equation (3.23) is obtained by replacing the logarithm in the denominator by its first-order approximation. For the "uniform" bound: Let $l = \alpha n$, $0 < x \leq 1$, $m = \alpha n$ and define

$$f(n) = \frac{n (1-x)(1-\frac{1}{x\alpha})e^n}{x + 1}.$$

For a fixed given $x$,

$$f(n) = \frac{n (1-x)(1-\frac{1}{x\alpha})^x}{x + 1} \leq \frac{n (1-x)(e^{-x})^{1/x}}{x + 1} = g(n).$$

As $g(n)$ is monotonically increasing in $n$, we get

$$f(n) \leq \lim_{n \to \infty} g(n) = \frac{(1-x)(e^{-x})^{1/x}}{x} \leq \max_{0 < x \leq 1} \left( \frac{(1-x)(e^{-x})^{1/x}}{x} \right) = 0.008905 < 0.01.$$

As we do not know how to demonstrate analytically such tight bounds for any other, less degenerate $rpv$'s, we tried a numerical example, to assess the quality of Theorems 3.1 through 3.4.

Parts (a) and (b) of the enclosed table represent two ways of obtaining the value of $m^*$ when the $rpv$ followed a Zipf distribution: The first was obtained by computing $\Delta(m)(p, q)$ numerically, and searching for the lowest value of $m$ that made it negative, whereas the second is obtained directly from the bound of equation (3.19). The irregular change of the values there with $n$ results from the truncation when $i^*$ is computed. Two things are immediately clear: First, the bounds are substantially higher. In view of the methods used to generate the bounds, this is hardly surprising. It is not obvious yet whether this is a coincidence, but the ratio between the bound and the measured values is rather consistently close to $\alpha^{-1}$. Secondly, as far as dependence on $n$ and $\alpha$ shows through these values, both tables exhibit very close
The required number of reorganizations under CS to approach the optimum to within $1 + \alpha$. The values are given as multiples of $n^2$.

4. The Limited-Counters Scheme

Lam et al. (1981), showed that the expected access-cost at each stage, under the counter scheme, is lower than that of any other permutation-heuristic. The generalization inherent in their treatment may be viewed as restricted to the selection of a particular distribution for the initial state of the list ordering, and hence their theorem is also relevant for our formulation.

We present now a 'truncated version' of the original rule, and compare its performance with that of the MTF and the optimal ordering. Consider the 'Limited-Counters Scheme' (LCS) in which $c_{\text{max}}$ is the maximal value any of the counters may reach: A counter $c_i$ which gets to the value $c_{\text{max}}$ remains unchanged when further requests are made for $R_i$. From this follows, inter alia, that $R_i$ does not change its position when further accessed. Thus, as $m$ increases, a longer and longer prefix of the list stays "frozen".

Let $E_{LCS}^{(m)}(C | c_{\text{max}})$ denote the expected access-cost of the $m + 1$st request under LCS.

**Theorem 4.1:** If the list is initially ordered arbitrarily, so that each of the arrangements is equi-probable and $c_{\text{max}} = c$, then:

(i) for $1 \leq m \leq 2c$ \[ E_{LCS}^{(m)}(C | c) = E_{CS}^{(m)}(C). \] (4.1)

(ii) for $m > 2c$:
where

\[ f_{ij}(k) = \sum_{r=\lceil k/2 \rceil + 1}^{k} \binom{k}{r} p_i^r p_j^{k-r} + \frac{1}{2} \binom{k}{k/2} (p_i p_j)^{k/2}. \]  

(4.3)

Note that \( f_{ij}(k) \) is the same as \( (p_i + p_j)^k q_{ji}^{(m)} \).

**Proof:**

(i) Clearly, when \( m \leq 2c \), the LCS produces exactly the same permutations as the original CS rule does. Hence, the equality (4.1) follows.

(ii) For \( m > 2c \) we follow the steps used in the proof of Theorem 2.1. Considering a pair of records \( R_i, R_j \), the value of \( q_{ji}^{(m)} \) is to be calculated *a novo*:

Let \( k \) denote the number of requests made to \( R_i \) and \( R_j \); for \( 0 \leq k \leq m \), there are two cases to be distinguished:

a) \( 0 \leq k \leq 2c \); then \( R_i \) and \( R_j \) will be ordered by the values of \( c_i \) and \( c_j \) just as under the CS.

b) \( 2c < k \leq m \); in that case, the order between the records is determined within the first \( 2c \) requests. At this stage one of the counters (or both) would attain the value \( c \) and the relation \( c_i \geq c_j \) or \( c_j \geq c_i \) will be thereafter preserved.

Hence, now

\[
q_{ji}^{(m)} = \begin{cases} \sum_{r=\lceil k/2 \rceil + 1}^{k} \binom{k}{r} p_i^r p_j^{k-r} + I_k \frac{1}{2} \binom{k}{k/2} (p_i p_j)^{k/2} & 0 \leq k \leq 2c \\ 2c \sum_{r=c+1}^{2c} \binom{2c}{r} p_i^r p_j^{2c-r} + \frac{1}{2} \binom{2c}{c} (p_i p_j)^c & 2c < k \leq m \end{cases}
\]  

(4.4)

(Note that the right-hand side of equation (4.4) does not depend explicitly on \( m ! \)) Summing up over \( k \), and substituting again into (2.4) provide \( E_{\text{LCS}}^{(m)}(C \mid c) \) as stated in the theorem. □

Now consider the limiting cost:

**Theorem 4.2:** The asymptotic expected access-cost under LCS is given by:

\[
E_{\text{LCS}}(C \mid c) = \lim_{m \to \infty} E_{\text{LCS}}^{(m)}(C \mid c) = 1 + \sum_{i=1}^{n} p_i \sum_{j=1}^{n} \sum_{r=c+1}^{2c} \binom{2c}{r} p_i^r p_j^{2c-r} + \frac{1}{2} \binom{2c}{c} (p_i p_j)^c (p_i + p_j)^{2c}. \]  

(4.5)

**Proof:** We derive the asymptotic cost directly from the cost function defined in Theorem 4.1.
Using the notation \( f_{ij}(k) \) defined in equation (4.3), and letting \( m \) increase, we find

\[
\lim_{m \to \infty} E_{\text{LCS}}^{(m)}(C \mid c) = 1 + \sum_{i=1}^{n} p_i \sum_{j=1}^{n} \frac{f_{ij}(2c)}{(p_i + p_j)^{2c}} + \sum_{k=0}^{2c} \lim_{m \to \infty} \left[ \binom{m}{k} \right] (1 - p_i - p_j)^{m-k} \times [f_{ij}(k) - (p_i + p_j)^{k-2c}f_{ij}(2c)] .
\]

(4.6)

For the sum over \( k \) in equation (4.6), note that the number of terms is fixed and finite, and that for each particular \( k \) the difference \([f_{ij}(k) - (p_i + p_j)^{k-2c}f_{ij}(2c)]\) is bounded. Let \( U_m = \binom{m}{k} \times (1 - p_i - p_j)^{m-k} \). Since \( \lim_{m \to \infty} U_{m+1}/U_m = (1 - p_i - p_j) \), the sum over \( k \) in equation (4.6) vanishes in the limit, and we obtain the desired result, equation (4.5). This equation has a very simple intuitive interpretation, and could be written at once, without going through Theorem 4.1: as the number of references increases, every pair reaches the limit of \( 2c \) references, that determine their relative order, and then the last equation can be read as a rewrite of equation (2.4).

Clearly, one expects the performance of the LCS to improve monotonically as \( c \) increases, since we are using, in a consistent way, more information. A formal proof is easy to do, and promptly reduces to showing the monotonicity in \( c \) of the second line in the right-hand side of equation (4.4). This in turn follows by defining \( a_{ij} = \frac{p_i}{p_j} \); the above expression is homogeneous in \( a_{ij} \), and the monotonicity can be shown on a power-by-power basis.

Certainly we shall have for all \( c \geq 1 \) and \( m > 2c \)

\[
E_{\text{LCS}}^{(m)}(C \mid c) \geq E_{\text{LCS}}^{(m)}(C).
\]

(4.7)

The extent of the difference is not easy to characterize in any formal way, beyond the explicit expressions. Some understanding of its possible range will be provided by the bounds we give below. We bring now a result that suggests that the limited LCS is of interest, even in its most stunted version.

**Theorem 4.3:** Let \( E_{\text{MTF}}^{(m)}(C) \) denote the expected cost of the \( m+1 \)st request under the MTF rule. Then for all \( c \geq 1, m \geq 1 \)

\[
E_{\text{LCS}}^{(m)}(C \mid c) \leq E_{\text{MTF}}^{(m)}(C).
\]

(4.8)

The proof uses the following technical lemma:

**Lemma 4.3.1:** If \( \frac{1}{2} \leq p \leq 1 \) and \( k \geq 1 \) then

\[
2p(1-p) - E(k; p, 1-p) \geq 0.
\]

(4.9)

**Proof:** The monotonicity property of \( E(k; \cdot, \cdot, \cdot) \), shown in the Appendix—see (A.4)—implies that it suffices to show (4.9) for \( k=1 \). Then, from equation (2.7),

\[
2p(1-p) - E(1; p, 1-p) = 2p(1-p) - p(1-p) - (1-p)p = 0,
\]

(4.10)

which establishes the lemma.

**Proof of Theorem 4.3:** For \( n \geq 2, m \geq 1 \) and a given rpv \((p_1, \cdot, \cdot, p_n)\), write the value of \( E_{\text{MTF}}^{(m)}(C) \) in a form which facilitates comparison with \( E_{\text{LCS}}^{(m)}(C \mid c) \):
\[ E_{\text{MTF}}^{(m)}(C) = 1 + \sum_{i} p_i \sum \left[ m_i \right] \frac{\sum_{i=1}^{n} \left(1 - p_i - p_j\right)^m}{p_i + p_j} + \frac{1}{2} \frac{(1 - p_i - p_j)^m}{1 - p_i - p_j} \]  
(4.11)

\[ = 1 + \sum_{1 \leq i < j \leq n} \left[ 2p_i p_j \sum_{k=1}^{m} \left[ \left(1 - p_i - p_j\right)^{m-k}(p_i + p_j)^{k-1} + \frac{1}{2} (p_i + p_j)(1 - p_i - p_j)^m \right] \right]. \]  
(4.12)

For each pair of records \( R_i, R_j \) with the access probabilities \( p_i, p_j \) respectively, let \( q_{ij} = p_i + p_j \) and \( e_{ij} = p_i / q_{ij} \). Following the previous assumption, \( p_i \geq p_j \) for \( i < j \), and then \( 1 \geq e_{ij} \geq \frac{1}{2} \).

We derive the inequality (4.8) directly, considering separately the two ranges of \( m \).

(i) \( 1 \leq m \leq 2c \): using equations (4.12), (2.8), (2.9) (the \( k=0 \) term is split off and cancels out), and Lemma 4.3.1:

\[ E_{\text{MTF}}^{(m)}(C) - E_{\text{LCS}}^{(m)}(C \mid c) = \]

\[ = \sum_{1 \leq i < j \leq n} \left[ \sum_{k=1}^{m} \left(1 - q_{ij}\right)^{m-k}q_{ij}^{k-1}(2e_{ij}(1-e_{ij}) - E(k; e_{ij}, 1-e_{ij})) \right] \geq 0. \]

(ii) Similarly, for \( m \geq 2c + 1 \)

\[ E_{\text{MTF}}^{(m)}(C) - E_{\text{LCS}}^{(m)}(C \mid c) = \sum_{1 \leq i < j \leq n} \left[ \sum_{k=2c+1}^{m} \left(1 - q_{ij}\right)^{m-k}q_{ij}^{k-1}(2e_{ij}(1-e_{ij}) - E(k; e_{ij}, 1-e_{ij})) \right] \geq 0. \]

Remark: An alternative proof, which the reader may find more appealing starts with the monotonicity of \( E_{\text{LCS}}^{(m)}(C \mid c) \) in \( c \), mentioned above; followed by possibly the most surprising aspect of Theorem 4.3; namely, when \( c = 1 \), the relation (4.8) realizes the equality – that is, the LCS with \( c = 1 \) provides exactly the asymptotic cost of the MTF scheme. Section 5 brings the idea behind an immediate proof of this. Using the LCS with \( c = 1 \) means a record will be moved at most once – when it is first accessed: it is then moved to the tail of the sublist of records that have been accessed before. The Theorem does not claim this is a particularly good scheme; Theorem 4.2 implies we could do better, the following Theorem outlines by how much better.

Theorem 4.4: For any \( n \geq 2 \) and \( c \geq 2 \) the asymptotic expected cost under the LCS is bound as follows:

\[ \frac{E_{\text{LCS}}(C \mid c)}{E_{\text{opt}}(C)} \leq 1 + \max_{a \geq 1} \frac{(a - 1)\left[ \sum_{r=0}^{c-1} \binom{2c}{c} a^r - \frac{1}{2}(\frac{2c}{e})^c \right]}{(1 + a)^{2c}} \]  
(4.13)

Proof: The proof consists in manipulating expressions we have obtained above. First, from equation (4.5) we find, by splitting the sum over the index \( j \) and using the shorthand notations \( a_{ij} = p_j / p_i \) and \( g(c, a) = \sum_{r=0}^{c} \binom{2c}{c} a^r \).
Note that due to the convention of numbering the records in decreasing order of access probabilities, $a_{ij} \geq 1$, for all $i$ and $j$. Using equation (2.12) and the relation (that holds for all positive numbers)

$$\sum_{i} a_{ij} / \sum_{j} \leq \max\{a_{k} / b_{k}\},$$

we find

$$E_{LCS}(C | c) = \sum_{i=1}^{n} \sum_{j=1}^{n} p_{i} \frac{(1 + a_{ij})^{2c} + (a_{ij} - 1)[g(c, a_{ij}) - \frac{1}{2}(2c)a_{ij}^{c}]}{(1 + a_{ij})^{2c}},$$

Note that due to the convention of numbering the records in decreasing order of access probabilities, $a_{ij} \geq 1$, for all $i$ and $j$. Using equation (2.12) and the relation (that holds for all positive numbers)

$$\sum_{i} a_{ij} / \sum_{j} \leq \max\{a_{k} / b_{k}\},$$

we find

$$E_{LCS}(C | c) = \sum_{i=1}^{n} \sum_{j=1}^{n} p_{i} \frac{(1 + a_{ij})^{2c} + (a_{ij} - 1)[g(c, a_{ij}) - \frac{1}{2}(2c)a_{ij}^{c}]}{(1 + a_{ij})^{2c}},$$

and by replacing the sum over $j$ by $i$ times its maximum, we get

$$E_{LCS}(C | c) \leq \max_{1 \leq i \leq n} \frac{(1 + a_{ij})^{2c} + (a_{ij} - 1)[g(c, a_{ij}) - \frac{1}{2}(2c)a_{ij}^{c}]}{(1 + a_{ij})^{2c}}.$$

Finally, replacing the double maximization over $i$ and $j$ by allowing any $a \geq 1$, we obtain the relation (4.13).

Comment: Theorem 4.4 provides a bound that is valid for all rpv's; also, for reasonably low values of $c$, it is expressed in terms of low-degree polynomials. Thus, for example, it is easy to show, by directly finding the maxima of the respective right-hand sides of equation (4.13), that the ratios $E_{LCS}(C | c)/E_{opt}(C)$ for $c = 2, 3, 4$, are bound by $1.3156, 1.2175$ and $1.1733$, respectively.

We may expect the universality of the bound to mean that for most distributions the results would even be better. This was tested out for two cases, the geometric and Zipf distributions—the asymptotic cost was computed from equation (4.5) and compared with $E_{opt}(C)$ of equation (2.12). Tables 2 and 3 show the results. The most interesting feature is possibly the behavior of the ratio—or the excess cost as displayed in Table 2—with $\lambda$: $E_{LCS}(C | c)$ is very close to the optimal cost when the distribution is either very skew or nearly flat, and less so in intermediate cases.

5. Discussion and Open Problems

We have argued the case for the Counter Scheme as the best heuristic for organizing a linear list when a stationary irm may be assumed to hold. Under such a condition it is the preferred method of list reorganization, with the further provision that there is adequate storage in the data structure to hold the

<table>
<thead>
<tr>
<th>$\lambda c$</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>0.019</td>
<td>0.005</td>
<td>0.002</td>
</tr>
<tr>
<td>0.3</td>
<td>0.081</td>
<td>0.045</td>
<td>0.027</td>
</tr>
<tr>
<td>0.5</td>
<td>0.125</td>
<td>0.080</td>
<td>0.058</td>
</tr>
<tr>
<td>0.7</td>
<td>0.158</td>
<td>0.055</td>
<td>0.079</td>
</tr>
<tr>
<td>0.9</td>
<td>0.180</td>
<td>0.132</td>
<td>0.104</td>
</tr>
<tr>
<td>0.99</td>
<td>0.038</td>
<td>0.037</td>
<td>0.036</td>
</tr>
</tbody>
</table>

Table 2

The excess cost $E_{LCS}(C | c)/E_{opt}(C) − 1$ for a list of 25 items and geometrical access probabilities as given in equation (3.17)
Note that due to the convention of numbering the records in decreasing order of access probabilities, 
\( a_{ij} \geq 1 \), for all \( i \) and \( j \). Using equation (2.12) and the relation (that holds for all positive numbers)
\[ \sum_{i} a_{ij} \leq \max_{k} (a_{k}/b_{k}) \text{, we find} \]
\[ \frac{E_{LCS}(C|c)}{E_{opt}(C)} \leq \max_{1 \leq i \leq n} \sum_{j=1}^{i} \frac{(1 + a_{ij})^{2c} + (a_{ij} - 1)[g(c,a_{ij}) - \frac{1}{2}(2c)a_{ij}^{c}]}{(1 + a_{ij})^{2c}}. \tag{4.14} \]
and by replacing the sum over \( j \) by \( i \) times its maximum, we get
\[ \frac{E_{LCS}(C|c)}{E_{opt}(C)} \leq \max_{1 \leq i \leq n} \max_{1 \leq j \leq i} \frac{(1 + a_{ij})^{2c} + (a_{ij} - 1)[g(c,a_{ij}) - \frac{1}{2}(2c)a_{ij}^{c}]}{(1 + a_{ij})^{2c}}. \tag{4.15} \]
Finally, replacing the double maximization over \( i \) and \( j \) by allowing any \( a \geq 1 \), we obtain the relation (4.13).

Comment: Theorem 4.4 provides a bound that is valid for all rpv's; also, for reasonably low values of \( c \), it is expressed in terms of low-degree polynomials. Thus, for example, it is easy to show, by directly finding the maxima of the respective right-hand sides of equation (4.13), that the ratios \( E_{LCS}(C|c)/E_{opt}(C) \) for \( c = 2, 3, 4 \), are bound by 1.3156, 1.2175 and 1.1733, respectively.

We may expect the universality of the bound to mean that for most distributions the results would even be better. This was tested out for two cases, the geometric and Zipf distributions – the asymptotic cost was computed from equation (4.5) and compared with \( E_{opt}(C) \) of equation (2.12). Tables 2 and 3 show the results. The most interesting feature is possibly the behavior of the ratio—or the excess cost as displayed in Table 2—with \( \lambda \): \( E_{LCS}(C|c) \) is very close to the optimal cost when the distribution is either very skew or nearly flat, and less so in intermediate cases.

5. Discussion and Open Problems

We have argued the case for the Counter Scheme as the best heuristic for organizing a linear list when a stationary irm may be assumed to hold. Under such a condition it is the preferred method of list reorganization, with the further provision that there is adequate storage in the data structure to hold the

<table>
<thead>
<tr>
<th>( \lambda \cdot c )</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>0.019</td>
<td>0.005</td>
<td>0.002</td>
</tr>
<tr>
<td>0.3</td>
<td>0.08</td>
<td>0.045</td>
<td>0.027</td>
</tr>
<tr>
<td>0.5</td>
<td>0.125</td>
<td>0.080</td>
<td>0.058</td>
</tr>
<tr>
<td>0.7</td>
<td>0.158</td>
<td>0.055</td>
<td>0.079</td>
</tr>
<tr>
<td>0.9</td>
<td>0.180</td>
<td>0.132</td>
<td>0.104</td>
</tr>
<tr>
<td>0.99</td>
<td>0.038</td>
<td>0.037</td>
<td>0.036</td>
</tr>
</tbody>
</table>

Table 2
The excess cost \( E_{LCS}(C|c)/E_{opt}(C) - 1 \) for a list of 25 items and geometrical access probabilities as given in equation (3.17)
counters. When the scheme is chosen, we address the question of determining the size of the counter fields when the reference probability vector is not known a priori. We discussed two possible solutions. The first one, the LCS, may reduce significantly the space complexity while nearly maintaining the high performance level of the full scheme; the upper bound on its possible expected cost, compared with the optimal scheme is very promising indeed. The second approach also sets an upper bound on the space required for each of the counters, when the original counter scheme is implemented, by fixing a finite point of time at which the list reorganization may be stopped. Remembering that the linear list is acceptable as a data structure for frequently referenced data only when \( n \) is rather small – 6? 20? 50? – unlikely to be much larger, we realize that the required storage, under the improved schemes, is indeed hardly a serious consideration.

One of the interesting aspects of the analysis was the light it shed on tacit assumptions. For example, why is the result of Theorem 4.3, specialized for \( c = 1 \) surprising – as it is, at first sight? Technically, it holds because under the condition that a pair of records were at all accessed, the probability of a particular one of them to be the last one to be referenced is the same as that of its being the first, due to the stationarity of the \( irn \). The reason then we find it surprising that the "move once" LCS (with \( c = 1 \)) is as good as the ever-busy MTF, is that we tend to prize recent information, knowing that stationary processes are no more common in the real world than unicorns.

An issue we have hardly touched upon is the cost of whatever reorganization scheme one chooses. This seems acceptable if we limit the discussion to schemes that use roughly the same operations. The three methods we discussed fill this condition, except the time-limited CS. A specific reason to avoid the issue was that when the reorganization cost is included in the reference cost, comparing different schemes becomes technology-dependent.

Do we actually propose the CS to be used in real-life applications? The answer depends on the extent to which real life satisfies our model assumptions. The tritest of these is the independence of successive references. For various reasons, of which we believe the reader could supply several examples, our computational processes do not normally behave in this manner. The assumption may be reasonably if approximately true when the reference string is the result of superposing numerous sources – it often happens then that the serial correlation in the resulting sequence is negligible. When the independence premise fails, the best policy clearly depends on the failure mode: when the references exhibit a periodic nature, over the entire list, Move To the Rear (deterministically!) suggests itself; a strong locality property promotes MTF, with more complicated patterns producing more elaborate optimal permutation rules.
Next comes the assumption of fixed probabilities. Presumably, even when the values wobble, but without changing their sorted order, the CS remains the method of choice. When this is not the case, the issue is open.

Indeed, the field swarms with open questions. To name a few:
- Devising an efficient adaptive scheme for an early termination of the CS;
- Detection of a possible dependence structure in the reference model;
- For a given structure – obtaining the optimal strategy;
- To combine the last two: fashioning an adaptive control mechanism to fit a reorganization rule as knowledge of the reference model evolves;
- For a time-varying reference model – determining and detecting critical points.

Under any model, how should dynamic lists, with insertions and deletions, be handled?
At what stage are other data structures, more expensive to maintain but with shorter search sequences competitive?
How are they to be reorganized? etc.

For some of those, such as dynamic lists, even a proper definition of the problem remains elusive. Answers to any of these questions promise to be interesting both from the practical and theoretical points of view.

REFERENCES


APPENDIX: Proof of Theorem 2.2

We first show, that for all \( m \geq 1, i \neq j \) and \( 1 > p_i \geq p_j > 0 \)

\[
E^{(m-1)}(p_i, p_j) - E^{(m)}(p_i, p_j)
\]

Let \( q = p_i + p_j \). We consider separately two cases:

(a): \( q = 1 \). Then we write \( p_i \equiv p \) and \( p_j = 1 - p \) where \( \frac{1}{2} \leq p \leq 1 \). We further separate the cases of even and odd number of references:

(i) Let \( k = 2l + 1 \) for an integer \( l \geq 0 \), then

\[
E(k; p_i, p_j) = E(2l+1; p, 1-p) = p^r + (1-2p) \sum_{r=l+1}^{2l+1} \binom{2l+1}{r} p^r (1-p)^{2l+1-r}, \tag{A.1}
\]

\[
E(k-1; p_i, p_j) = E(2l; p, 1-p) = p^r + (1-2p) \sum_{r=l+1}^{2l} \binom{2l}{r} p^r (1-p)^{2l-r} + \frac{1}{2} \binom{2l}{l} p^l (1-p)^l. \tag{A.2}
\]

Taking the difference and expanding the coefficient \( \binom{2l+1}{r} \), the sums cancel out to leave merely

\[
E(k-1; p_i, p_j) - E(k; p_i, p_j) = \frac{1}{2} \binom{2l}{l} p^l (1-p)^l (1-2p)^2, \tag{A.3}
\]

which is non-negative.

(ii) When \( k = 2l, l \geq 1 \), the expressions equivalent to (A.1) and (A.2) can be manipulated in a similar way to show that in this case

\[
E(k-1; p_i, p_j) - E(k; p_i, p_j) = 0.
\]

Hence, for all \( k \geq 1, 1 > p_i \geq p_j > 0, p_i + p_j = 1 \), the \( E(k; p_i, p_j) \) are monotonic decreasing in \( k \):

\[
E(k-1; p_i, p_j) - E(k; p_i, p_j) \geq 0. \quad p_i + p_j = 1 \tag{A.4}
\]

The sum in the definition of \( E^{(m)}(p_i, p_j) \), equation (2.8), leaves only the term \( k=m \) when \( p_i + p_j = 1 \). This implies

\[
E^{(m-1)}(p_i, p_j) - E^{(m)}(p_i, p_j) = E(m-1; p_i, p_j) - E(m; p_i, p_j) \geq 0. \quad p_i + p_j = 1 \tag{A.5}
\]

(b): \( q < 1 \). We introduce the notation \( p_i = qp \), and \( p_j = q(1-p) \). Since \( p_i \geq p_j, \frac{1}{2} \leq p \leq 1 \). From the defining equation (2.7)

\[
E(k; p_i, p_j) = q^{k+1} E(k; p, 1-p). \tag{A.6}
\]

We now have to consider the difference
\[ E^{(m-1)}(p_i, p_j) - E^{(m)}(p_i, p_j) = \sum_{k=1}^{m-1} \binom{m-1}{k} (1-p_i - p_j)^{m-1-k} E(k; p_i, p_j) + \frac{1}{2} (p_i + p_j)(1-p_i - p_j)^{m-1} \]

\[ - \sum_{k=1}^{m} \binom{m}{k} (1-p_i - p_j)^{m-k} E(k; p_i, p_j) - \frac{1}{2} (p_i + p_j)(1-p_i - p_j)^{m}. \]

Using the notation of (A.6), a reorganization like the one that produced equation (A.3) brings the difference to

\[ \sum_{k=1}^{m-1} \binom{m-1}{k} (1-q)^{m-1-k} q^{k+2} [E(k; p, 1-p) - E(k+1; p, 1-p)] + \frac{1}{2} (1-q)^{m-1}(p_i - p_j)^2 \geq 0. \]

Finally, from (a) and (b), we have

\[ E^{(m-1)}_{CS}(C) - E^{(m)}_{CS}(C) = \sum_{1 \leq i < j \leq n} [E^{(m-1)}(p_i, p_j) - E^{(m)}(p_i, p_j)] \geq 0, \]

which completes the proof. \qed