ON TRANSPARENTLY MODIFYING USERS' QUERY DISTRIBUTIONS

B.J. Oommen* and D.T.H. Ng*

SCS-TR-147

November 22, 1988

This research was supported by the Natural Sciences and Engineering Research Council of Canada.

School of Computer Science, Carleton University

Ottawa, Canada K1S 5B6

ON TRANSPARENTLY MODIFYING USERS' QUERY DISTRIBUTIONS†

B. J. Oommen* and D. T. H. Ng*

ABSTRACT

In this paper we introduce a concept which is completely new to the areas of computer science and data manipulation. Let $\mathfrak{R} = \{R_1, R_2, \dots, R_N\}$ be a set of data elements. The elements of $\mathfrak{R}$ are accessed by the users of the system according to a fixed but unknown distribution $\mathfrak{S}=\{s_1, s_2, \dots, s_N\}$, referred to as the users' query distribution. The manager of the system organizes the data $\mathfrak{R}$ so as to minimize the cost of retrieving them. Thus, if $\mathfrak{R}$ is maintained as a linear list it is advantageous that the elements of $\mathfrak{R}$ are sorted in the descending order of $\mathfrak{S}$, the access probabilities. In this paper we consider the problem of transforming the users' query distribution $\mathfrak{S}$ into a new distribution $\mathfrak{S}'$. The latter transformation is done in a fashion that is transparent to the user. Furthermore, rather than having the manager organize the data according to the distribution $\mathfrak{S}$ he can maintain the data according to the distribution $\mathfrak{S}'$ to obtain superior data retrieval characteristics. After posing the problem in its mathematical generality we propose a particular Distribution Changing Technique (DCT) filter and show that it indeed transforms the original distribution expediently. The problem of cascading DCT filters has also been studied and some initial theoretical results have been presented. Numerous computational and simulation results which validify the theoretical results presented have also been included.

Keywords : Data Retrieval, Data Organization, Adaptive Data Structures, Query Statistics, Information Storage and Retrieval.

---

† Partially supported by the Natural Sciences and Engineering Research Council of Canada.

* School of Computer Science, Carleton University, Ottawa : K1S 5B6, CANADA.

I. INTRODUCTION

One of the most basic problems in computer science is that of organizing data in such a way that the cost of maintaining the data in an organized fashion and the cost of retrieving the data is minimized. Indeed, this problem is now of paramount importance because memory costs are declining and the amount of data to be stored seems to be ever increasing. Databases which were considered to be large a decade ago are comparatively dwarfs relative to the very large databases in existence today.

To pose the problem that we are studying in the right perspective we consider the following simplified model of data organization. Let $\mathfrak{R}=\{R_1, R_2, \ldots, R_N\}$ be a set of data elements. Each element of $\mathfrak{R}$ can be a file, a table (relation), or in general, be a segment of the database itself. The elements of $\mathfrak{R}$ are accessed by the users of the system according to a fixed distribution $\mathfrak{S}=\{s_1, s_2, \ldots, s_N\}$. $\mathfrak{S}$ is called the users' query distribution and is assumed unknown. The intention is that the data $\mathfrak{R}$ is organized by the system manager in such a way so as to minimize the cost of retrieving the elements. Thus, if $\mathfrak{R}$ is maintained as a linear list it is advantageous that the system manager order the elements of $\mathfrak{R}$ in the descending order of $\mathfrak{S}$, the access probabilities.

The problem of having a **linear list** organize itself has been studied extensively. Reviews on self-organizing strategies have been published over the years (see Gonnet *et. al.* [4], Rivest [13] and Hester *et. al.* [7]). McCabe [10, pp.398-399; 11] was the first to propose a solution to this problem, and his solution rendered the list dynamically self-organizing by moving the accessed element to the front of the list every time it was accessed. He also introduced a scheme which is called the transposition rule which dictates that an accessed element is interchanged with its preceding element in the list, unless, of course, it is at the front of the list. For an analysis of the transposition rule we refer the reader to [1, 2, 10, 11, 13]. Generalizations of the transposition rule are the Move-k-Ahead rule and the POS(k) rule. In the former, the accessed element is moved k-positions forward towards the front of the list unless it is found in the first k positions - in which case it is moved to the front. The POS(k) rule moves the accessed element to position k of the list if it is in positions k+1 through n. It transposes it with the preceding element if it is in positions 2 through k. If it is the first element it is left unchanged.

The literature reports two absorbing **stochastic** list organizing schemes [7, 12] both of which are due to Oommen and Hansen. The first of these algorithms is essentially a move-to-front algorithm with the exception that on the n^{th} access, the accessed element is moved to the front of the list with a probability f(n) which is systematically **decreased** whenever an element is accessed. The scheme was shown to have the property that if $s_i > s_j$, then the probability of absorption into an arrangement in which R_i precedes R_j is always greater than 0.5. The second algorithm in [12] is a **move-to-rear** scheme in which the accessed element R_j is moved to the **rear** of the list with a probability q_j which is progressively decremented every time **the element** is accessed. In this case, the probability of converging to the optimal arrangement can be made as close to **unity** as desired. The literature also reports of two absorbing **deterministic move-to-rear** list organizing schemes [7, 12] the first of which was shown to be optimal and the second conjectured to be optimal. The latter conjecture has now been proven.

As opposed to maintaining the data as a linear list, the data may also be maintained as a binary tree. If we are given $\mathfrak{R}$ and the set $\mathfrak{S}$ the problem of constructing an optimal binary search tree has been extensively studied. The most well known scheme, due to Knuth [7], uses dynamic programming techniques and produces an optimal binary search tree using $O(N^2)$ time and space. Alternatively, Walker and Gotlieb [14] have used dynamic programming and divide and conquer techniques to yield a nearly optimal binary search tree using O(N) space and O(N log N) time.

A binary search tree is not quite so simple to reorganize as a linked list for the following reasons. Firstly, the lexicographic ordering property of the binary search tree must be preserved throughout the reorganization of the tree. This can sometimes be in conflict with the request to move a record upwards in the tree. As well, a restructuring operation in a binary search tree may move more than just one record; it could also move an entire subtree either up or down in the tree. The above reasons preclude a direct application of any of the successful list organizing schemes because these schemes take no consideration for any ordering property that might have to be maintained among the elements, and they also take no consideration for the fact that the subtree representation of a sublist may also have to be reorganized.

In order to preserve the binary search property, **rotation** is the primitive tree restructuring operation. Allen and Munro presented the first memoryless scheme [2], which is analogous to the move-to-front rule. This scheme uses rotations to move the accessed record up to the root of the tree, and the rule is hence called the **move-to-root** scheme. They also developed another scheme called the **simple exchange** rule, which rotates the accessed element one level towards the root, similar to the transposition rule. Sleator and Tarjan [13] introduced a third scheme, which also moves the accessed record up to the root of the tree. This scheme, which uses a restructuring move called **splaying**, differs from that of Allen and Munro's in that it uses different variations of single and double rotations (see Bitner [5]), unlike the move-to-root scheme which applies only successive single rotations.

Among the schemes requiring extra memory is the **monotonic tree** scheme, first proposed by Knuth [7] as a possible method to structure a tree to obtain a nearly optimal tree. In a monotonic tree the root of every subtree has the property that it is the most probable node among the nodes of the subtree. More recently, Cheetham, Oommen and Ng [6] have presented a binary tree reorganizing strategy which requires linear memory but which performs a rotation **only** if the cost of the tree is reduced as a result of performing the operation. The theoretical properties of the scheme are derived in [6] and experimentally, it has been shown to be far superior to all other published dynamic restructuring methods. Furthermore, this scheme produces a tree which is very nearly optimal.

Similar schemes are available when the data is accessed in pairs or in clusters [8, 16, 22, 23, 26, 28, 30-33]. The classical example of this is the Object Partitioning Problem in which a set of W objects is partitioned into R classes in such a way that the objects that are accessed (used) more frequently together lie in the same class. Again, to render the problem non-trivial, we assume that their joint access probabilities are unknown. Solutions for this problem have been proposed for the case when the objects are attributes of a relation [30-33], and when they are records [32]. Solutions which tackle the problem in its generality have been proposed by Yu *et. al.* [32, 33]. More recently, learning automata solutions have been proposed by Oommen and Ma [16, 22, 23] for the case when the clusters are to be of equal sizes [22] and for the general Object Partitioning Problem [23]. These automata solutions converge an order of magnitude faster than the fastest schemes that were previously reported in the literature.

I.1 Problem Statement and Outlined Solution

In all the problems mentioned above, the efficiency of the data reorganization strategy is constrained by $\mathfrak{S}$, the users' query distribution. If $\mathfrak{S}$ is relatively "flat", and the elements of $\mathfrak{R}$ are all chosen with almost equal probability the advantage gained by adaptively organizing the data is minimal. Indeed, this corresponds to the case when the information content of the distribution is small, and thus the users' queries do not provide the necessary information required to reorganize the data in such a way as to get a remarkable increase in the efficiency of a retrieval. The heart of the matter is thus the fact that the users' query distribution can either make or break the strength of a data reorganizing strategy.

In this paper we introduce a concept which is completely novel to the field of computer science. Whereas the users' queries are distributed as per the distribution $\mathfrak{S}$, we shall present a theory by which the distribution $\mathfrak{S}$ can be transformed into a new distribution $\mathfrak{S}'$. This transformation is done in a fashion that is transparent to the user. Furthermore, the system manager now adaptively reorganizes the data (for example, as a list or as a binary tree using **any** of the techniques available) based on the new distribution $\mathfrak{S}'$. We shall show that the information content in the distribution $\mathfrak{S}'$ can be increased expediently. Informally speaking, this implies that if the data is reorganized due to $\mathfrak{S}'$, the net result is a structure that is superior to that which would have been obtained if the the data reorganization was achieved due to the distribution $\mathfrak{S}$.

The strategy by which we achieve the transformation is as follows. Let us suppose that the user requests an element R_i. Observe that that this can occur with a probability s_i. The element R_i is reported to the user. Apart from this, the request R_i is also fed into a module which we refer to as the Filter of the Distribution Changing Technique (DCT). This filter is completely defined by two quantities, namely, a list operator Ω and a function, called the Report Function, τ. The filter manipulates a dummy list using the list operator Ω. It also emits a command to the system manager to reorganize the data element R_j (where R_j may be the same as R_i) based on the Report Function, τ. However, the request to reorganize R_j is emitted with a probability s_j'. Notice that the system manager now reorganizes the data based on the distribution $\mathfrak{S}'=\{s_1', s_2', \dots , s_N'\}$. The intention is that the reorganization due to the distribution $\mathfrak{S}'$ is superior to the reorganization due to the distribution $\mathfrak{S}$.

Rather than merely discuss a particular DCT Filter, we shall present the theory of DCT filters in their mathematical generality. The properties of various filters $F = (\Omega , \tau)$ are catalogued in the body of the paper, and in particular, the properties of a particular filter referred to as F^* are discussed. We shall show that F^* indeed polarizes the users' query distribution by presenting to the system manager a magnified the value of the ratio (s_i'/ s_j') whenever $s_i > s_j$. A filter of this type is called an expediently polarizing DCT filter.

The details of how the list operator, Ω, is chosen and the technique for deciding on the element R_j to be reported to the system manager will be discussed in the subsequent sections. We however would emphasize the most interesting property of this mechanism which is that the entire DCT is done transparent to the user. Thus, although his data preferences may be almost void of information (i.e., the s_i's are almost equal), the DCT modifies the data so that the overall retrieval cost is still reduced.

Apart from studying the properties of a single DCT filter, we have also considered the problem of cascading a sequence of DCT filters. Some initial theoretical results have been presented which involve sequences of so-called Identity filters and sequences of expediently polarizing filters. In particular, a strong result has been proven which shows that even an infinite number of expediently polarizing DCT filters may not polarize the distribution into one which has almost all its probability mass concentrated on a single data element.

The paper also contains numerous computational and simulation results which clarify the theoretical results presented.

II. FUNDAMENTALS AND NOTATION

II.1 Lists and List Operations

$\mathfrak{R} = \{R_1, R_2, ..., R_N\}$ is a set of data elements accessed according to an unknown user's query distribution $\mathfrak{S} = \{s_1, s_2, ..., s_N\}$, where,

$$\sum_{i=1}^{N} s_i = 1. \qquad (1)$$

A list $\lambda(n)$ defined at time instant 'n' as $\lambda(n) = L_1(n)\ L_2(n)\ ...\ L_N(n)$, is a linear ordering of the data elements where for all i, j = 1,...,N, and $L_i(n) \neq L_j(n)$. When no ambiguity exists, we omit the reference to the time instant 'n'.

A list transforming operator Ω operating on a list $\lambda(n)$ at time 'n' transforms it into a new list $\lambda(n+1)$ at that time instant. We shall explicitly define two list operators which we shall use extensively in the future.

II.1.1 Ω_{MTF} : The Move-To-Front Operator

Ω_{MTF} is defined as the Move-To-Front operator. Given the list $\lambda(n)$ and a data element R_u as the input, Ω_{MTF} transforms $\lambda(n)$ into $\lambda(n+1)$ as follows. Let,

$$\lambda(n)=L_1 L_2 ... L_N, \text{ with } L_i, L_j \in \mathfrak{R}\ ;\ L_i \neq L_j \text{ for all } i, j. \qquad (2)$$

Then, if $R_u = L_k$, $\qquad \lambda(n+1) \equiv \Omega_{MTF}(\lambda(n), R_u)$, where,

$$\lambda(n+1) = \lambda(n) \qquad \text{if } k = 1.$$
$$\lambda(n+1) = L_k L_1 L_2 ... L_{k-1} L_{k+1} ... L_N \qquad \text{if } k>1. \qquad (3)$$

Note that Ω_{MTF} is the well known Move-To-Front rue which moves the accessed element to the front of the list.

II.1.2 Ω_{Tr} : The Transposition Operator

Ω_{Tr} is defined as the Transposition operator, which explicitly describes the rule which moves the accessed element R_u and interchanges it with its preceding element in the list unless R_u is at the front of the list. More explicitly, if $\lambda(n)$ is defined as in (2), and $R_u = L_k$, $\lambda(n+1) \equiv \Omega_{Tr}(\lambda(n), R_u)$, where

$$\lambda(n+1) = \lambda(n) \qquad \text{if } k = 1$$
$$\lambda(n+1) = L_1 L_2 ... L_k L_{k-1} L_{k+1} ... L_N \qquad \text{if } k > 1 \qquad (4)$$

II.1.3 A Generalized List Operator

Apart from Ω_{MTF} and Ω_{Tr}, a list operator can be defined in all generality

by an ordered list $\pi = (\pi_1, \pi_2, \ldots, \pi_N)$, where each π_i is a permutation of the set $\{1, 2, \ldots, N\}$. Then, if $R_u = L_k$, $\Omega(\lambda(n), R_u)$ is defined as $\lambda(n+1)$, where, if $\pi_k = \pi_{k(1)}\pi_{k(2)}\cdots\pi_{k(N)}$,
then,

$$\lambda(n+1) = L_{\pi_{k(1)}} L_{\pi_{k(2)}} \ldots L_{\pi_{k(N)}}.$$

Thus, the above operators Ω_{MTF} and Ω_{Tr} are alternatively defined by the permutations π_{MTF} and π_{Tr}, defined as below :

$$\pi_{MTF} = (123\ldots N;\ 213\ldots N;\ 312\ldots N;\ \ldots;\ k123\ldots(k\text{-}1)(k+1)\ldots N;\ \ldots;\ N123\ldots(N\text{-}1))$$
$$\pi_{Tr} = (123\ldots N;\ 213\ldots N;\ 132\ldots N;\ \ldots;\ 123\ldots k(k\text{-}1)(k+1)\ldots N;\ \ldots;\ 123\ldots N(N\text{-}1)).$$

Since we will be using the properties of Ω_{MTF} and Ω_{Tr} extensively, the relevant properties of these operators are stated below. For the sake of brevity, the proofs of these properties are omitted, except where the inclusion of the proof clarifies the explicit mathematical properties of the operators.

Lemma I

If Ω_{MTF} is the list reorganizing operator defined in II.1.1, then for all distinct indices i and j, the asymptotic probability of R_i preceding R_j is :

$$\Pr\,[R_i \text{ ultimately precedes } R_j] = s_i / (s_i + s_j) \qquad (5)$$

Proof : The proof of the lemma is found in [24] •••

To understand the properties of Ω_{Tr}, the transposition operator, it is advantageous to observe that the records migrate more slowly and thus one expects a slower convergence but a steady state cost which is closer to the optimal static ordering. Indeed, much of the analysis of the scheme can be comprehanded if one perceives the time reversibility of the underlying markov chain. The transition map of the chain is shown below in Figure I for the case when $\mathfrak{R} = \{R_1, R_2, R_3\}$.

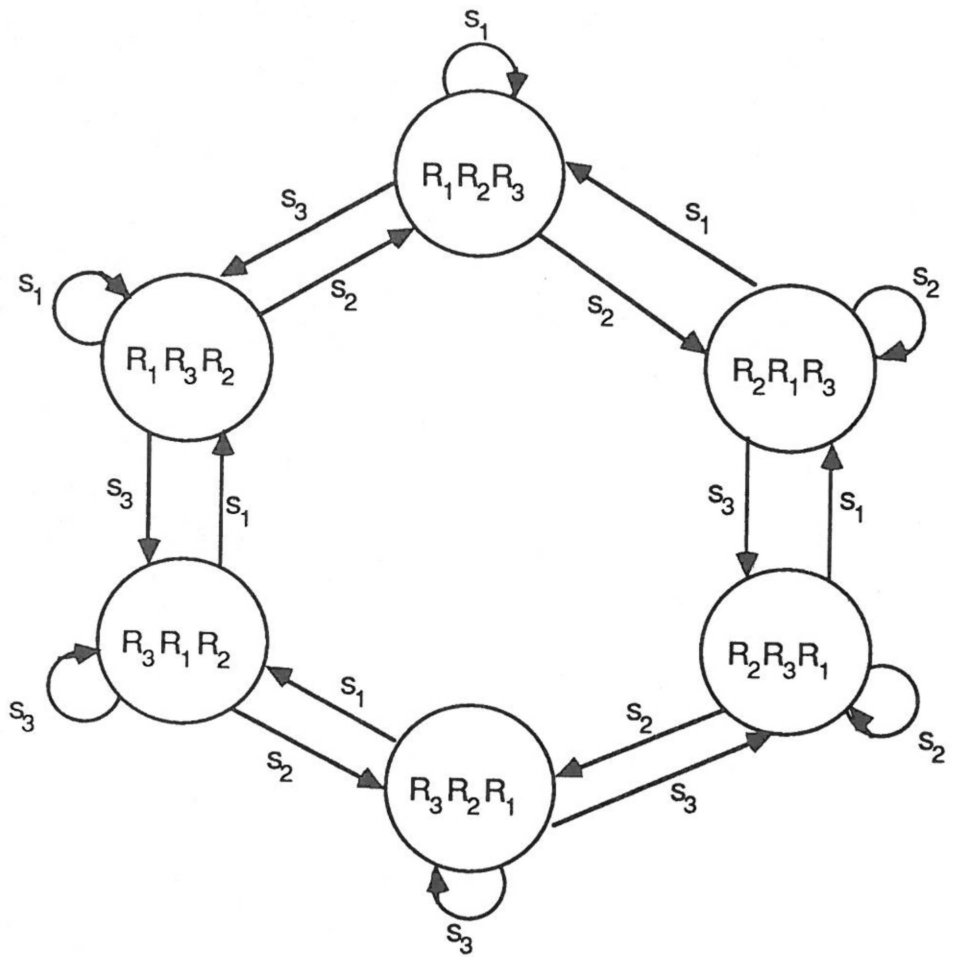

Figure I. Transition map of the Transposition Scheme, when $\mathfrak{R} = \{R_1, R_2, R_3\}$. Note the time reversibility of the chain.

It is easy to see the time reversibility of the chain since, if we consider the following path from state $(R_1R_2R_3)$ to itself in a clockwise fashion as $(R_1R_2R_3)\rightarrow(R_2R_1R_3)\rightarrow(R_2R_3R_1)\rightarrow(R_3R_2R_1)\rightarrow(R_3R_1R_2)\rightarrow(R_1R_3R_2)\rightarrow(R_1R_2R_3)$ the product of the transition probabilities in $s_1{}^2s_2{}^2s_3{}^2$. Indeed, this is exactly the same value if the paths were traversed in a counter clockwise fashion.

The main theorem which we use now follows. The theorem was originally proved by Rivest [24], but since the proof was sketchy, some of the critical details were omitted. A more complete proof of this result is included here.

Theorem I

Let the asymptotic stationary probability of the list being $(R_{i_1}R_{i_2}...R_{i_N})$ be expressed as Pr $(R_{i_1}R_{i_2}...R_{i_N})$. Then using Ω_{Tr}, the Transposition Operator, the stationary probabilities of the list obey :

$$\frac{\Pr(R_{i_1} R_{i_2} \ldots R_{i_j} R_{i_{j+1}} \ldots R_{i_N})}{\Pr(R_{i_1} R_{i_2} \ldots R_{i_{j+1}} R_{i_j} \ldots R_{i_N})} = \frac{s_{i_j}}{s_{i_{j+1}}}.$$

for $i \le j < N$ if $s_k \ne 0$ for $1 \le k \le N$. (6)

Proof.

Observe that if a set of probabilities obeys (6), it must be a stationary distribution, since $\mathcal{S}$ is time invariant, and the stationary distribution is the unique eigenvector of the underlying markov chain for the eigenvalue unity. But the asymptotic state probabilities obey,

$$\Pr(R_{i_1} \ldots R_{i_N}) = s_{i_1} . \Pr(R_{i_1} \ldots R_{i_N}) + \sum_{j=1}^{N-1} s_{i_j} . \Pr(R_{i_1} \ldots R_{i_{j+1}} R_{i_j} \ldots R_{i_N}) \qquad (7)$$

since the underlying Markov chain is irreducible and ergodic.

Consider equation (6). Rearranging terms, we have,

$$s_{i_j} \Pr(R_{i_1} \ldots R_{i_{j+1}} R_{i_j} \ldots R_{i_N}) = s_{i_{j+1}} \Pr(R_{i_1} \ldots R_{i_N}) \qquad (8)$$

To prove that (6) satisfies (7) we substitute (8) into (7) to yield the identity

$$\begin{aligned}\Pr(R_{i_1} \ldots R_{i_N}) &= s_{i_1} \Pr(R_{i_1} \ldots R_{i_N}) + \sum_{j=1}^{N-1} s_{i_{j+1}} \Pr(R_{i_1} \ldots R_{i_N}) \\ &= (s_{i_1} + \sum_{j=2}^{N} s_{i_j}) \Pr(R_{i_1} \ldots R_{i_N}) \\ &= \Pr(R_{i_1} \ldots R_{i_N}).\end{aligned}$$

Hence the theorem. •••

Apart from the above theorem, there are two other powerful results concerning Ω_{Tr}. To the best of our knowledge, these results have not been reported in the literature (although they have been alluded to in [24]), but are crucial for the later sections of the paper.

Theorem II

Let the asymptotic stationary probability of the list being $(R_{i_1}, R_{i_2}, \ldots, R_{i_N})$

be expressed as $\Pr(R_{i_1}, R_{i_2}, \ldots, R_{i_N})$. Then,

$$\frac{\Pr(R_{i_1} \ldots R_{i_{j-1}} R_{i_j} R_{i_{j+1}} R_{i_{j+2}} R_{i_{j+3}} \ldots R_{i_N})}{\Pr(R_{i_1} \ldots R_{i_{j-1}} R_{i_{j+2}} R_{i_{j+1}} R_{i_j} R_{i_{j+3}} \ldots R_{i_N})} = \left(\frac{s_{i_j}}{s_{i_{j+2}}}\right)^2$$

Proof.

Using the chain rule, we expand the left hand side of the above equation as follows :

$$\frac{\Pr(R_{i_1} \ldots R_{i_{j-1}} R_{i_j} R_{i_{j+1}} R_{i_{j+2}} R_{i_{j+3}} \ldots R_{i_N})}{\Pr(R_{i_1} \ldots R_{i_{j-1}} R_{i_{j+2}} R_{i_{j+1}} R_{i_j} R_{i_{j+3}} \ldots R_{i_N})} = \frac{\Pr(R_{i_1} \ldots R_{i_{j-1}} R_{i_j} R_{i_{j+1}} R_{i_{j+2}} R_{i_{j+3}} \ldots R_{i_N})}{\Pr(R_{i_1} \ldots R_{i_{j-1}} R_{i_{j+1}} R_{i_j} R_{i_{j+2}} R_{i_{j+3}} \ldots R_{i_N})} \times$$

$$\frac{\Pr(R_{i_1} \ldots R_{i_{j-1}} R_{i_{j+1}} R_{i_j} R_{i_{j+2}} R_{i_{j+3}} \ldots R_{i_N})}{\Pr(R_{i_1} \ldots R_{i_{j-1}} R_{i_{j+1}} R_{i_{j+2}} R_{i_j} R_{i_{j+3}} \ldots R_{i_N})} \times$$

$$\frac{\Pr(R_{i_1} \ldots R_{i_{j-1}} R_{i_{j+1}} R_{i_{j+2}} R_{i_j} R_{i_{j+3}} \ldots R_{i_N})}{\Pr(R_{i_1} \ldots R_{i_{j-1}} R_{i_{j+2}} R_{i_{j+1}} R_{i_j} R_{i_{j+3}} \ldots R_{i_N})}$$

$$= \frac{s_{i_j}}{s_{i_{j+1}}} \cdot \frac{s_{i_j}}{s_{i_{j+2}}} \cdot \frac{s_{i_{j+1}}}{s_{i_{j+2}}} = \left(\frac{s_{i_j}}{s_{i_{j+2}}}\right)^2 .$$

the last expression being obtained by applying (6) in a straight forward way. Hence the result. •••

A generalization of Theorem II is given be the following equation :

$$\frac{\Pr(R_{i_1} \ldots R_{i_N})}{\Pr(R_{i_1} \ldots R_{i_{j-1}} R_{i_{j+m}} R_{i_{j+1}} \ldots R_{i_{j+m-1}} R_{i_j} R_{i_{j+m+1}} \ldots R_{i_N})} = \left(\frac{s_{i_j}}{s_{i_{j+m}}}\right)^m \tag{10}$$

The result is proved in an analogous way by using the chain rule in a repeated fashion.

We are now in a position to present our Distribution Changing Technique.

III. DISTRIBUTION CHANGING FILTERS

III.1 The Report Function, τ

Central to the theory of the DCT is a function τ referred to as the Report Function. In general, it operates on a list and a data element to yield another data element. More explicitly, if λ is defined as in (2) and $R_u \in \mathfrak{R}$, then

$$\tau(\lambda, R_u) \equiv L_j$$

implies that the Report Function τ operates on λ and reports the value $R_v=L_j$ when the input is the element R_u. Note that R_v may or may not be equal to R_u. In general τ is a stochastic function. In particular, if $\tau(\lambda, R_u) = R_u$ for all u, τ is called the Trivial Report Function.

III.2 The DCT Filter

A DCT Filter is a pair $F = (\Omega, \tau)$, where Ω is a list operator and τ is a report function. Essentially the operation of the filter is as follows.

The DCT maintains a (hypothetical) list $\lambda = L_1 L_2 \ldots L_N$ where each $L_i \in \mathfrak{R}$ and $L_i \neq L_j$ for all i and j. When the user requests an element R_u, the element is provided by the system manager to the user. However, the system does not reorganize the physical data based on the accessed element R_u. Rather, the DCT filter uses its internal function τ and reports an element $L_j = \tau(\lambda, R_u)$ to the system manager. Physical data reorganization is done by the manager using the element L_j. The DCT filter then uses its Ω and R_u to update the (hypothetical) list $\lambda(n)$. Observe that the system manager reorganizes the data based on the distribution reported by $\tau(.,.)$ and not based on $\mathfrak{S}$.

In the above, we have repeatedly emphasized that for the DCT Filter $\lambda(n)$ is a hypothetical list. This is because the internal list for the filter need not contain any data elements, but needs only to be a list of indices. However, for the sake of simplicity of notation, we shall refer to $\lambda(n)$ as a list.

The actual process used by the DCT Filter is algorithmically given below. However, for the sake of conceptual simplicity, the schematic of a DCT filter is also given diagramatically.

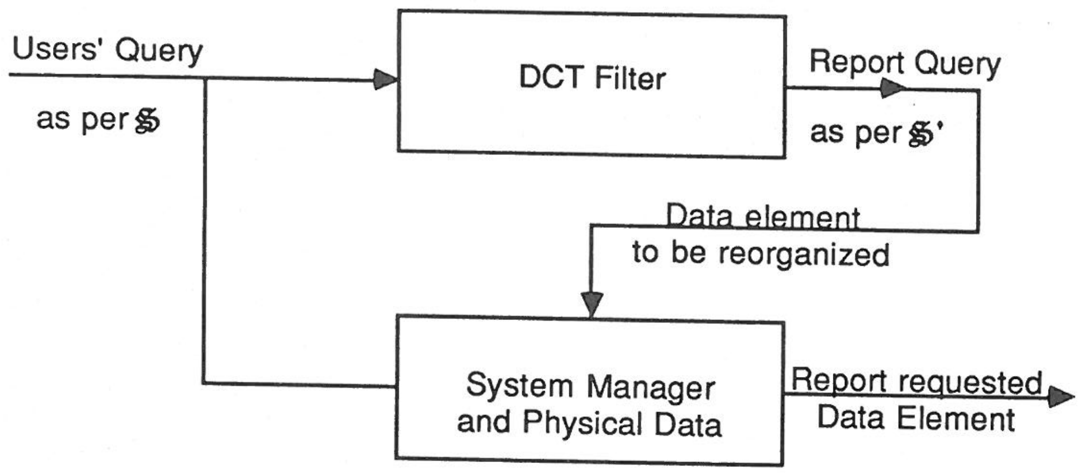

Figure II : A Schematic of the DCT Filter.

ALGORITHM DCT_FILTER (R_u, L_j)

Input : A data element requested $R_u \in \mathfrak{R}$.

Output : A data element $L_j \in \mathfrak{R}$ to be manipulated by the System Manager.

Filter Definitions : The filter is fully defined by the pair (Ω, τ) and maintains an internal list λ. λ is initially assigned as a random list.

Method :

```
    ReadInput ($R_u$)
    $L \leftarrow \tau(\lambda, R_u)$
    $\lambda \leftarrow \Omega(\lambda, R_u)$
    Output (L)
End ALGORITHM DCT_FILTER
```

We shall now present the properties of various physical DCT Filters.

Definition :

Let the users' query distribution be $\mathfrak{S}$ and let the **asymptotic** distribution of the elements reported by $\tau(.,.)$ be $\mathfrak{S}'$. A DCT Filter is called an Identity Filter if $\mathfrak{S} = \mathfrak{S}'$.

This implies that if $\mathfrak{S} = \{s_1, s_2, ..., s_N\}$ and $\mathfrak{S}' = \{s_1', s_2', ..., s_N'\}$, then for all i, $s_i = s_i'$. Note that independent of Ω, if for all u, $\tau(\lambda, R_u)$ is R_u, the DCT filter is an identity filter, since the element reported to the system manager is exactly the element requested by the user.

Theorem III

The DCT filter F_1 defined for $\Omega = \Omega_{MTF}$ and with $\tau(z) = L_1$ is an identity filter.

Proof.

The list maintained by the DCT Filter $\lambda = L_1 L_2 \ldots L_N$. Thus, given an input of R_u, R_u becomes the head of the list when Ω_{MTF} is applied, and hence the output sequence of the DCT is exactly the same as the input sequence except that the former is delayed by a single time unit. If R_i is the input request at time n, R_i will be the output presented to the system manager at time (n+1). Hence the theorem. •••

Theorem IV

Let $\Omega = \Omega_{Tr}$ and $\lambda = L_1 L_2 \ldots L_N$. Further let τ_T be the report function defined as follows :

$$\begin{aligned} \tau_T(\lambda, R_u) &= L_{j-1} & &\text{if } L_j = R_u \\ &= L_1 & &\text{if } L_1 = R_u \end{aligned} \tag{11}$$

Then the DCT filter $F_2 = (\Omega_{Tr}, \tau_T)$ is an Identity Filter.

Proof.

Let $\lambda = L_1 L_2 \ldots L_N$ be equal to the list $R_{i_1} R_{i_2} \ldots R_{i_N}$. We shall first derive an expression for the asymptotic probability of the element R_{i_j} being immediately in front of element $R_{i_{j+1}}$. Let this asymptotic probability be written as $B(R_{i_j}, R_{i_{j+1}})$. We derive an expression for this as follows. Using Theorem I, and considering all the possible locations for the sequence $R_{i_j} R_{i_{j+1}}$,

$$\frac{\Pr(R_{i_j} R_{i_{j+1}} R_{i_1} \ldots R_{i_{j-1}} R_{i_{j+2}} \ldots R_{i_N})}{\Pr(R_{i_{j+1}} R_{i_j} R_{i_1} \ldots R_{i_{j-1}} R_{i_{j+2}} \ldots R_{i_N})} = \frac{\Pr(R_{i_1} R_{i_j} R_{i_{j+1}} R_{i_2} \ldots R_{i_{j-1}} R_{i_{j+2}} \ldots R_{i_N})}{\Pr(R_{i_1} R_{i_{j+1}} R_{i_j} R_{i_2} \ldots R_{i_{j-1}} R_{i_{j+2}} \ldots R_{i_N})}$$

$$= \ldots = \frac{\Pr(R_{i_1} \ldots R_{i_{j-1}} R_{i_{j+2}} \ldots R_{i_N} R_{i_j} R_{i_{j+1}})}{\Pr(R_{i_1} \ldots R_{i_{j-1}} R_{i_{j+2}} \ldots R_{i_N} R_{i_{j+1}} R_{i_j})} = \frac{s_{i_j}}{s_{i_{j+1}}}$$

Since all of the ratios are equal, they are each equal to the sum of the numerators to the sum of the denominators due to *componendo ed dividendo*. Thus, summing the ratios we get,

$$\frac{\Pr(R_{i_j} R_{i_{j+1}} R_{i_1} \ldots R_{i_{j-1}} R_{i_{j+2}} \ldots R_{i_N}) + \ldots + \Pr(R_{i_1} \ldots R_{i_{j-1}} R_{i_{j+2}} \ldots R_{i_N} R_{i_j} R_{i_{j+1}})}{\Pr(R_{i_{j+1}} R_{i_j} R_{i_1} \ldots R_{i_{j-1}} R_{i_{j+2}} \ldots R_{i_N}) + \ldots + \Pr(R_{i_1} \ldots R_{i_{j-1}} R_{i_{j+2}} \ldots R_{i_N} R_{i_{j+1}} R_{i_j})} = \frac{s_{i_j}}{s_{i_{j+1}}}$$

(12)

Notice that $R_{i_1}, R_{i_2}, \ldots, R_{i_N}$ can be used to represent **any** permutation.

If we sum **all** the probabilities in (12) with all the possible values for the elements $R_{i_1}, R_{i_2}, \ldots, R_{i_{j-1}}, R_{i_{j+2}}, \ldots, R_{i_N}$, we indeed have the quantity $B(R_{i_j}, R_{i_{j+1}})$. Thus, we have,

$$\frac{B(R_{i_j}, R_{i_{j+1}})}{B(R_{i_{j+1}}, R_{i_j})} = \frac{s_{i_j}}{s_{i_{j+1}}}$$

(13)

From the asymptotic behaviour of Ω_{Tr} we compute the probability of R_i being the output as follows. Using the laws of total probability,

$$\Pr[R_i \text{ is the output}] = \sum_{k=1}^{N} \Pr[R_i \text{ is the output} \mid R_k \text{ is the input}] \, . \, s_k$$

$$= \sum_{k \neq i} (\Pr[R_i \text{ is the output} \mid R_k \text{ is the input}] \, . \, s_k) + \Pr[R_i \text{ is the output} \mid R_i \text{ is the input}].s_i$$

But for the DCT filter (Ω_{Tr}, τ_T) the probability $\Pr[R_i$ is the output $\mid R_k$ is the input$]$ is exactly $B(R_i, R_k)$. Furthermore for (Ω_{Tr}, τ_T) the probability

$\Pr[R_i$ is the output $\mid R_i$ is the input$] = \Pr[R_i$ is the first element in $\lambda]$.

But (12) states that

$$\frac{B(R_i, R_k)}{B(R_k, R_i)} = \frac{s_i}{s_k}.$$

Thus,

$$\Pr[R_i \text{ is the output}] = \sum_{k \neq i} B(R_k, R_i).s_i + \Pr[R_i \text{ is the first element in } \lambda].s_i$$

$$= s_i \left(\sum_{k \neq i} B(R_k, R_i) + \Pr[R_i \text{ is first element in } \lambda]\right)$$

However,

$$\sum_{k \neq i} B(R_k, R_i) + \Pr[R_i \text{ is the first element in } \lambda] = 1$$

since the sum of the probability of any element occurring before R_i and the probability of no element occurring before R_i is obviously 1. Hence,

$$\Pr[R_i \text{ is the output}] = s_i \,.\, 1 = s_i$$

and the result is proved. •••

For any general DCT Filter $F = (\Omega, \tau)$ where $\Omega \neq \Omega_{MTF}$ and $\Omega \neq \Omega_{Tr}$, the problem of finding the properties of τ so as to yield a non-trivial identity DCT filter remain open. Observe that a trivial identity filter is obtained by using the function $\tau(\lambda, R_u) \equiv R_u$.

III.3 Polarizing DCT Filters

Let the user's query distribution be $\mathfrak{S}$ and let the asymptotic distribution of the elements reported by the DCT Filter be $\mathfrak{S}' = \{s_1', s_2', \ldots, s_N'\}$. A DCT Filter is said to polarize expediently if for all i and j,

$$s_i' / s_j' > s_i / s_j \qquad \text{wherever } s_i > s_j.$$

The DCT Filter is said to polarize the distribution absolutely if the ratio $(s_i'/s_j') \rightarrow \infty$ for all $s_i > s_j$.

Observe that if we can design a polarizing DCT Filter, it will indeed increase the information content of the users' query distribution.

The whole problem of designing such polarizing filters is extremely fascinating. We shall present one such filter and refer to it as F*.

Theorem V

Let $F^* = (\Omega_{Tr}, \tau^*)$, where if $\lambda = L_1 L_2 \ldots L_N$,

$$\tau^*(\lambda, R_u) = L_1. \qquad (14)$$

Then F* is an expediently polarizing DCT Filter.

Proof.

For **any** pair of elements R_i and R_j, if $s_i > s_j$, we intend to prove that if the asymptotic probability of the DCT reporting R_i and R_j are s_i' and s_j', then,

$$s_i' / s_j' > s_i / s_j.$$

This means that the filter accentuates the ratio of the user's query probabilities. Also we note that since $\tau^*(.,.) = L_1$, the asymptotic probability $s_i' \equiv \Pr[\, R_i$ is output] is exactly equal to the asymptotic probability $\Pr[L_1 = R_i]$. We shall derive an expression for the ratio s_i' / s_j'.

Let α_p be the asymptotic probability that $L_1 = R_i$ and $L_{p+1} = R_j$. Furthermore, let β_p be the asymptotic probability that $L_1 = R_j$ and $L_{p+1} = R_i$.

We know from Theorem II that for any R_k,

$$\frac{\Pr(R_i\, R_k\, R_j \ldots)}{\Pr(R_j\, R_k\, R_i \ldots)} = \frac{s_i^2}{s_j^2}$$

(15)

Since (15) is true for all k, the ratio α^2 / β^2 can be obtained by summing the numerators and the denominators as per *componendo ed dividendo.* Thus,

$$\frac{\alpha_2}{\beta_2} = \frac{\sum_{k \neq i, j} \Pr(R_i\, R_k\, R_j \ldots)}{\sum_{k \neq i, j} \Pr(R_j\, R_k\, R_i \ldots)} = \frac{s_i^2}{s_j^2}$$

In an analogous way,

$$\frac{\alpha_3}{\beta_3} = \frac{s_i^3}{s_j^3}.$$

In general, for all $2 \leq p \leq N\text{-}1$, we have

$$\frac{\alpha_p}{\beta_p} = \frac{s_i^p}{s_j^p}.$$

(16)

Consider now the ratio $\Pr[L_1 = R_i] / \Pr[L_1 = R_j]$. Since the events are mutually exclusive the ratio of the probabilities can be added to yield :

$$\frac{s_i'}{s_j'} = \frac{\Pr[L_1 = R_i]}{\Pr[L_1 = R_j]} = \frac{\sum_{p=1}^{N-1} \alpha_p}{\sum_{p=1}^{N-1} \beta_p}$$

(17)

Let $\rho = s_i / s_j$. Note that by the hypothesis $\rho > 1$. Using (16), (17) is simplified as follows,

$$\frac{s_i'}{s_j'} = \frac{\sum_{p=1}^{N-1} \alpha_p}{\sum_{p=1}^{N-1} \beta_p} = \frac{\sum_{p=1}^{N-1} \rho^p \beta_p}{\sum_{p=1}^{N-1} \beta_p}.$$

(18)

But using the generalized theory of mean values [9] we know that given a set of elements $\{a_i\}$ and a set of weights $\{x_i\}$,

$$a_1^{x_1} a_2^{x_2} \dots a_r^{x_r} < \left(\frac{x_1 a_1 + x_2 a_2 + \dots + x_r a_r}{x_1 + x_2 + \dots + x_r}\right)^{x_1 + x_2 + \dots + x_r}$$

(19)

Identifying the $\{x_i\}$ with $\{\alpha_p\}$ in (18) and the $\{a_i\}$ with $\{\rho^p\}$ in (18), we get after some rather lengthy algebraic manipulations,

$$\frac{s_i'}{s_j'} > \rho^{\frac{\beta_1 + 2\beta_2 + \dots + (N-1)\beta_{N-1}}{\beta_1 + \beta_2 + \dots + \beta_{N-1}}}$$

(20)

Since both ρ and the exponent of ρ on the right hand side of (20) are strictly greater than unity, the ratio of s_i' to s_j' is strictly greater than ρ. Hence,

$s_i' / s_j' > s_i / s_j$

and the theorem is proved. ●●●

Remarks

1. The expression (20) is quite revealing. Observe that the exponent of ρ is the ratio of $\sum p.\beta_p$ to $\sum \beta_p$. But β_p is the asymptotic probability that $L_1 = R_j$ and $L_{p+1} = R_i$. Thus, if we consider the **conditional** asymptotic distribution of the location of R_i given that $L_1 = R_j$, we observe that the exponent of ρ is merely the mean (or expected) value of the position of R_i. This gives us a good "rule-of-thumb" estimate for R_j being reported to the system manager.

2. If we apply the general theory of mean values [9] using the set of $\{\alpha_p\}$ (instead of the set $\{\beta_p\}$), we can derive a **lower** bound on the ratios s_i' / s_j'. Indeed, applying (19) into (18) using the $\{\alpha_p\}$ we can simplify the expression to show that

$$\frac{s_i'}{s_j'} < \rho^{\frac{\alpha_1 + 2\alpha_2 + \dots + (N-1)\alpha_{N-1}}{\alpha_1 + \alpha_2 + \dots + \alpha_{N-1}}}$$

(21)

Combining (20) and (21) tells us that if we quantify the polarization of the DCT Filter F* as the ratio of (s_i'/s_j') to (s_i/s_j), this quantity will have a lower bound of unity but an upper bound specified by (21). These bounds are the tightest bounds that can be derived, since the generalized geometric means equal the generalized arithmetic means when the $\{s_i\}$ are all equal. This itself shows that F* polarizes a distribution expediently but not absolutely.

3. Viewed in its generality, a DCT Filter itself can be seen to be an automaton. Indeed, the state transition function of the automaton is the operator Ω and the output function is the report function $\tau(.,.)$. In this case, the automaton is a generalized Mealey machine in which the output is a function of the state of the automaton (a list ordering) and the input, R_u. Apart from this conceptual perspective, it is still unclear whether such a model of the DCT Filter adds to the generality of the filter itself, except for the fact that the transition function of this automaton can be rendered stochastic. An example of a single Mealey transition for F* is given below when $\mathfrak{R} = \{R_1, R_2, R_3, R_4\}$.

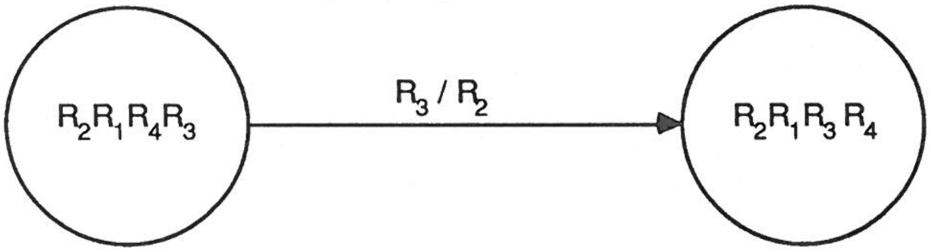

Figure III : A typical Mealey transition of F* when R= $\{R_1\ R_2\ R_3\ R_4\}$. With an input of R_3, the state changes as per Ω_{Tr} from $R_2\ R_1\ R_4\ R_3$ to $R_2\ R_1\ R_3\ R_4$. The output to the system manager as per τ^* is R_2.

4. In out study we have not been able to design an **absolutely** polarizing DCT filter. Indeed, we conjecture that such a filter does not exist. However, we shall now study the problem of cascading DCT filters in tandem. Our hope is that although single filters are at their best expediently (and not absolutely) polarizing, a sequence of filters could yield a new filter which is absolutely polarizing.

5. Although we have presented only one expediently polarizing DCT Filter, it is an easy task to extend the results of Theorem V to design families of expediently polarizing DCT filters. To show how this is done, consider the filter F^+ defined as follows :

$F^+ = (\Omega_{Tr}, \tau^+)$, where Ω_{Tr} is the Transposition operator, and,

$$\begin{aligned} \tau^+ &= \tau^* && \text{with probability } 0.5 \\ &= \tau_T && \text{with probability } 0.5 \end{aligned} \qquad (22)$$

with τ^* and τ_T being defined as per (11) and (14) respectively.

The fact that F^+ is polarizing is obvious since $F^* = (\Omega_{Tr}, \tau^*)$ is polarizing, and $F_2 = (\Omega_{Tr}, \tau_T)$ is an identity filter. Thus, the ratio s_i'/ s_j' for F^+ will satisfy :

$$\begin{array}{lll} s_i' / s_j' & > s_i / s_j & \text{with probability 0.5} \\ & = s_i / s_j & \text{with probability 0.5.} \end{array}$$

If we consider the ratio of the total probabilities, it is an elementary task to see that s_i' / s_j' will still be greater than s_i / s_j.

6. An interesting DCT filter which we have not analysed is the filter F_3, where, $F_3 = (\Omega_{Tr}, \tau_3)$, τ_3 defined as below for $\lambda = L_1L_2...L_N$.

$$\tau_3 = L_{Rand(1, j-1)} \quad \text{, where } L_j = R_u.$$

Observe that this filter operates the list using the transposition operator, but reports any random preceding element. Our experience indicates that this filter polarizes the distribution, but this is yet unproven.

IV. CASCADING DCT FILTERS

Till now we have studied the scenario of having a single DCT filter operate on a set of data elements which are accessed according to a distribution $\mathfrak{S}$. The intention was to use the filter to modify the distribution so that the system manager which re-organized the data elements organized them according to a distribution $\mathfrak{S}'$. The aim of the whole exercise was to polarize the distribution $\mathfrak{S}$ in such a way that the reorganization of the data according to $\mathfrak{S}'$ would be superior to the reorganization of the data as per $\mathfrak{S}$.

In the previous section, we presented a filter F* which operated on the input data distribution and expediently polarized it such a way that if $s_i > s_j$, then, for all $s_i, s_j \in \mathfrak{S}$ whose corresponding access probabilities after the filter are s_i', $s_j' \in \mathfrak{S}'$,

$$s_i' / s_j' > s_i / s_j.$$

However, we were unable to absolutely polarize the distribution such that the new access probabilities obeyed :

$$\frac{s_i'}{s_j'} \to \infty \qquad \text{whenever } s_i > s_j.$$

We shall now consider the problem of cascading DCT filters. We do this with the hope that although a single DCT filter may not polarize a distribution absolutely, a sequence of the filters arranged in tandem may achieve such a polarization. We shall however prove the counter-intuitive fact that even an infinite sequence of expediently polarizing filters may not yield an absolutely polarizing filter. However, before we do this, we shall state some elementary theorems about DCT filters in tandem.

Theorem VII

Let F and G be two identity DCT filters. Then if H is the DCT filter obtained by sequencing F and G, H is also an identity DCT filter.

Proof.

Let $\mathfrak{S}$ be the input distribution, and let $\mathfrak{S}^F = \{s_1^F, s_2^F, \ldots, s_N^F\}$ be the distribution after the filter F has operated on the data. Then, by definition,

$$\frac{s_i^F}{s_j^F} = \frac{s_i}{s_j} \qquad \text{for all i and j, } s_i > s_j .$$

<div align="right">(23)</div>

If this distribution $\mathbb{S}^F$ is the input to the filter G, then, by definition, the output of G is $\mathbb{S}^H = \{s_1^H, s_2^H, \ldots, s_N^H\}$ satisfying,

$$\frac{s_i^H}{s_j^H} = \frac{s_i^F}{s_j^F} \qquad \text{for all i and j, } s_i > s_j.$$

Combining (23) with the above proves the theorem! •••

Theorem VIII

Let F and G be two expediently polarizing DCT filters. Then, if H is the DCT filter obtained by sequencing F and G, H is also expediently polarizing.

Proof.

The theorem is proved analogously to the above theorem and uses the transitivity of the operation ">". •••

The next theorem is certainly the most interesting result which we can prove regarding sequencing DCT filters.

Theorem IX

Let G be any expediently polarizing filter. Then, the DCT filter obtained by sequencing an **infinite** number of such filters need not be absolutely polarizing.

Proof.

Let $\mathbb{S}^{(k)} = \{s_1^{(k)}, s_2^{(k)}, \ldots, s_N^{(k)}\}$ be the output distribution after the sequence of k filters has operated on the user's distribution $\mathbb{S}$. By definition, if $s_i > s_j$,

$$\frac{s_i^{(k+1)}}{s_j^{(k+1)}} > \frac{s_i^{(k)}}{s_j^{(k)}} \qquad \text{for } k \geq 1.$$

We need to show that there exists an expediently polarizing filter and a set of probabilities $\{s_i\}$ such that

$$\lim_{m \to \infty} \frac{s_i^{(m)}}{s_j^{(m)}}$$

tends towards a **finite** value for some i and j where $s_i > s_j$.

Consider the filter F* defined by Theorem V as F* = (Ω_{Tr}, τ^*), where τ^* reports the first element of the list. Indeed, we shall show that for **all** non-trivial distributions in which N=3, the

$$\lim_{m\to\infty} \frac{s_i^{(m)}}{s_j^{(m)}}$$

is finite for i and j being the two largest access probabilities.

Consider the case when N=3. Let the output distribution of the kth filter be $\{s_i^{(k)} \mid 1 \le i \le 3\}$. Then, using the properties of Ω_{Tr} and τ^*, it is seen after considerable algebra that :

$$s_1^{(k+1)} = \frac{(s_1^{(k)})^2 . s_2^{(k)} + (s_1^{(k)})^2 . s_3^{(k)}}{\text{Den}} \tag{24}$$

$$s_2^{(k+1)} = \frac{(s_2^{(k)})^2 . s_3^{(k)} + (s_2^{(k)})^2 . s_1^{(k)}}{\text{Den}} \tag{25}$$

$$s_3^{(k+1)} = \frac{(s_3^{(k)})^2 . s_1^{(k)} + (s_3^{(k)})^2 . s_2^{(k)}}{\text{Den}} \tag{26}$$

where, Den is the normalizing denominator enforcing $s_1^{(k+1)}+s_2^{(k+1)}+s_3^{(k+1)}=1$, and is :

$$\text{Den} = (s_1^{(k)})^2[s_2^{(k)} + s_3^{(k)}] + (s_2^{(k)})^2[s_3^{(k)} + s_1^{(k)}] + (s_3^{(k)})^2[s_2^{(k)} + s_3^{(k)}].$$

Since the s_i's are bounded and non-negative, the martingale convergence theorem assures their convergence with respect to k. In the equilibrium, we can solve for the terminal value of $s_i^{(\infty)}$ by solving (24)-(26). Let the terminal values be $\{s_i^*\}$. Then, by dividing (24) by (25) we get,

$$\frac{s_1^*}{s_2^*} = \frac{(s_1^*)^2 (s_2^* + s_3^*)}{(s_2^*)^2 (s_3^* + s_1^*)}, \text{ and,}$$

$$\frac{s_1^*}{s_3^*} = \frac{(s_1^*)^2 (s_2^* + s_3^*)}{(s_3^*)^2 (s_1^* + s_2^*)} . \tag{27}$$

(28)

Furthermore, $s_1^* + s_2^* + s_3^* = 1$. (29)

Cross multiplying (27) we get that (27) - (29) can be solved to yield an equilibrium solution whenever all three $\{s_i^* \mid 1 \leq i \leq 3\}$ are non-zero. This solution is $s_1^* = s_2^* = s_3^*$, and is, of course, the trivial solution.

The non-trivial solution is obtained if any one of the three, say, s_3^* converges to zero. Then, (27) suggests that every s_1^* / s_2^* **itself** is a fixed point solution implying that **every** vector value $[s_1^*, s_2^*, 0]$ is a potential solution. Thus, the ratio of s_1^* to s_2^* need not converge to infinity and the theorem is proved.

•••

Remark.

At the very outset, it may appear as if Theorems VIII and IX contradict each other. Indeed, they do not, since Theorem VIII merely states that the sequencing of expediently polarizing filters is always expediently polarizing. But since the product of a finite sequence of numbers which are greater than or equal to unity need not be infinite, the sequence of DCT filters need not be absolutely polarizing. Indeed, this is true in this case, since the ratio

$$\frac{\left(\frac{s_i^{(k+1)}}{s_j^{(k+1)}}\right)}{\left(\frac{s_i^{(k)}}{s_j^{(k)}}\right)}$$

tends to unity as $k \to \infty$.

V. EXPERIMENTAL RESULTS

To demonstrate the properties of the theory of DCT filters which we have introduced, we have simulated various filters so as to study the characteristics of their relative input and output distributions. To get a measure of the polarizing property of a particular filter, we have used the following index, χ, where,

$$\chi = \sum_{i=1}^{N} (a_i - s_i')^2,$$

where $a_i = 1$, if $s_i = \max_j s_j$;

$a_i = 0$, otherwise.

This is the chi-square measure quantifying the similarity between any distribution $\{s_i'\}$ and the distribution obtained for an absolutely polarizing scheme. Observe that if $s_i > s_j$ where $i < j$, the latter distribution would asymptotically be equal to the unit vector $[1\ 0\ 0\ \ldots\ 0]^T$.

Notice that χ measures the information content of the distribution. If all the s_i's are equal and $N>1$, ($\mathcal{S}$ is the trivial distribution) χ has a positive value. As the ratio of the s_i's increases, the value of χ decreases. χ eventually approaches zero as the ratio approaches infinity. To demonstrate the effect of the DCT filter we have computed the value of χ prior to the filter and the value of χ subsequent to the operation of the filter. These two quantities are called χ_i and χ_0 respectively.

Since N! is very large, it is impractical to theoritically measure χ_0 for large values of N. So, for values of N between 3 and 6, exact probability computation of the stationary distribution of Ω_{Tr} have been used to compute the output distribution of the DCT filter F*. This has been done for both the Wedge and Exponential distributionas follows. In the wedge distribution, s_i has the value :

$s_i = a - (i - 1)\Delta$, where $a = [n(n-1)\Delta + 2] / 2N$.

Note that the parameter Δ increases the slope of the wedge. In the case of the exponential distribution, s_i has the form :

$s_i = (1 - \Delta)^{(i-1)} / k$, where $k = [1 - (1-\Delta)^N] / \Delta$.

The results of the computation are remarkable and are tabulated in Tables 1 to 3. For example, when N = 4, if Δ=0.4 and the distribution is an exponential distribution, the input distribution is the probability vector $[0.459559\ 0.275735\ 0.165441\ 0.099265]^T$. The corresponding output distribution is $[0.587939\ 0.282816\ 0.101814\ 0.027431]^T$. Note that χ_i in this case is 0.256863, and the corresponding value of χ_0 is 0.405331.

Note that this represents a multiplicative decrease of 63%.

N	Wedge			Exponential		
	Δ	χ before	χ after	Δ	χ before	χ after
3	0.05	0.571667	0.528872	0.1	0.597786	0.565543
	0.1	0.486667	0.418007	0.2	0.524590	0.464640
	0.15	0.411667	0.335084	0.4	0.367347	0.279896
4	0.05	0.6125	0.506019	0.1	0.671891	0.601006
	0.1	0.5	0.379891	0.2	0.587775	0.464222
	0.15	0.4125	0.325673	0.4	0.405331	0.260897
5	0.01	0.761	0.705581	0.1	0.716026	0.606599
	0.03	0.689	0.560667	0.2	0.624465	0.450270
	0.05	0.625	0.473814	0.4	0.424705	0.256863
6	0.01	0.785083	0.697483	0.1	0.745181	0.600942
	0.03	0.874033	0.534782	0.2	0.647951	0.440753
	0.05	0.627083	0.465641	0.4	0.435318	0.257593

Table I. Expected values of χ before and after going through the DCT filter F*.

For the case of N>7, exact close form expressions for the output probabilities are not easily derived. To obtain the corresponding values in these cases, we have resorted to simulations. 100 experiment were conducted using 20,000 queries based on the input distribution $\mathfrak{S}$, where $\mathfrak{S}$ obeyed either a wedge or an exponential distribution. The output query pattern of the DCT filter was statistically analysed and the distribution $\mathfrak{S}'$ estimated. Using these estimates, the value of χ_i and χ_0 are tabulated for various values of N.

Again the results are remarkable. For example, when N = 10, if $\Delta = 0.1$, the input distribution is the probability vector [0.153533 0.138180 0.124362 0.111926 0.100733 0.090660 0.081594 0.073434 0.066091 0.059482]^T. The corresponding output distirbution is [0.311725 0.234812 0.069186

0.116155 0.075564 0.043810 0.025167 0.013415 0.006611 0.003555 $]^T$. Note that χ_i in this case is 0.583924, and the corresponding value of χ_0 is 0.801915.

N	Wedge			Exponential		
	Δ	χ before	χ after	Δ	χ before	χ after
10	0.001	0.891082	0.861311	0.05	0.852947	0.691010
	0.005	0.857062	0.718414	0.1	0.801915	0.583924
	0.01	0.818250	0.647274	0.15	0.747292	0.507129
15	0.001	0.919613	0.860642	0.05	0.883587	0.690875
	0.002	0.906453	0.796012	0.1	0.828068	0.579839
	0.005	0.870333	0.681501	0.15	0.767887	0.502577

With regard to the cascading filters, we have also performed computations to verify the validity of Theorem IX. Using the filter F*, the input distribution $[0.6\ 0.3\ 0.1]^T$ yields an output distribution $[0.6667\ 0.2917\ 0.04167]^T$. If this distribution is an input to a suceeding filter F*, the output distribution at the second level is $[0.7052\ 0.2858\ 0.007920]^T$. At the 3th level, the distribution has the value $[0.7139\ 0.2857\ 3.031\times10^{-4}]^T$ which is remarkably close to its terminal value $[0.7143\ 0.2857\ 0]^T$. Notice that the terminal value of the ratio of $s_1{}^k$ to $s_2{}^k$ is finite just as Theorem IX dictates.

VI. CONCLUSIONS

In this paper we have introduced concepts which are novel in the areas of computer science, data manipulation and data retrieval and have proposed a theory for the transparent modification of the distribution of the data elements of a user. To be more specific, if $\mathfrak{R} = \{R_1, R_2, \ldots, R_N\}$ is a set of data elements accessed according to a fixed but unknown distribution $\mathfrak{S}=\{s_1, s_2, \ldots, s_N\}$, we have considered the problem of transforming $\mathfrak{S}$ into a new distribution $\mathfrak{S}'$. This transformation is done transparent to the user. Also, rather than having the system manager organize the data according to the distribution $\mathfrak{S}$ he is advised to maintain the data according to the distribution $\mathfrak{S}'$ to obtain superior data retrieval characteristics.

The theory of a Distribution Changing Technique (DCT) has been presented in its mathematical generality. Various DCT filters have been catalogued, and in particular, we propose a particular filter F^* has been proposed. It has been shown that this filter transforms the original distribution expediently.

In this paper we have also studied the problem of cascading DCT filters. Some initial theoretical results have been presented. The strongest result in this connection is one which states that even an infinite number of expediently polarizing DCT filters may not be absolutely polarizing.

The paper also contains extensive computational and simulation results which illustrate the theoretical results which have been presented.

REFERENCES

1. Allen, B. and Munro, I., "Self-organizing binary search trees", J.ACM 25, 1978, pp.526-535.
2. Arnow, D.M. and Tenebaum, A.M., "An Investigation of the Move-Ahead-k Rules", Congressus Numerantium, Proc. of the Thirteenth Southeastern Conference on Combinatorics, Graph Theory and Computing, Florida, February 1982, pp. 47-65.
3. Bayer, P. J., "Improved bounds on the costs of optimal and balanced binary search trees", MAC Technical Memo-69, Nov. 1975.
4. Bitner, J.R., "Heuristics That Dynamically Organize Data Structures", SIAM J. Comput., Vol.8, 1979, pp. 82-110.
5. Burville, P.J. and Kingman, J.F.C., "On a Model for Storage and Search", J. Appl. Probability, Vol.10, 1973, pp. 697-701.
6. Cheetham, R.P., Oommen, B.J. and Ng, D.T.H., " On Using Conditional Rotation Operations to Adaptively Structure Binary Search Trees". To appear in the Proceeding of the 1988 International Conference on Database Theory, Bruges, Belgium, August/September 1988.
7. Gonnet, G.H., Munro, J.I. and Suwanda, H. 1981. Exegesis of self-organizing linear search. SIAM J. Comput., Vol.10, 613-623.
8. Hammer, M., and Niamir, B., "A Heuristic Approach to Attribute Partitioning", Proc. of the ACM SIGMOD Conference, 1979, pp.93-101.
9. Hardy, G.H., Littlewood, J.E., and Polya, G., Inequalities, Cambridge University Press, 1983.
10. Hendricks, W. J.,"An Account of Self-Organizing Systems", SIAM J. Computing, Vol. 5, 1976, pp. 715-723.
11. Hester, J.H. and Hirschberg, D.S., "Self-Organizing Linear Search", ACM Computing Surveys, Vol. 17, 1985, pp.295-311.
12. Kan, Y. C., and Ross, S.M., "Optimal List Order Under Partial Memory Constraints", J. App. Probability, Vol. 17, 1980, pp. 1004-1015.
13. Karlin, S. and Taylor, H.M., " A First Course in Stochastic Processes", Academic Press, 1975.
14. Knuth, D.E., "The Art of Computer Programming, Vol.3, Sorting and Searching", Addison-Wesley, Reading, MA., 1973.
15. Lakshmivarahan, S., "Learning Algorithms Theory and Applications", Springer-Verlag, New York, 1981.
16. Ma, D.C.Y., "Object Partitioning by using Learning Automata", M.C.S. Thesis, School of Computer Science, Carleton University, Ottawa, Canada, April 1986.
17. McCabe, J., "On Serial Files with Relocatable Records", Operations Research, Vol. 12, 1965, pp.609-618.
18. Mehlhorn, K., "Nearly optimal binary search trees", Acta Informatica, 5(1975), pp.287-295.
19. Narendra, K.S., and Thathachar, M.A.L., "Learning Automata -- A Survey",IEEE Trans. Syst. Man and Cybern., Vol.SMC-4, 1974, pp. 323-334.
20. Oommen, B.J., and Hansen E.R., "List Organizing Strategies using Stochastic Move-to-Front and Stochastic Move-to-Rear Operations", SIAM Journal of Computing, Vol. 16, No. 4, August 1987, pp.705-716.

21. Oommen, B.J., Hansen, E.R., and Munro, J.I., "Deterministic Optimal and Expedient Move-to-Rear List Organizing Strategies" To appear in Theoretical Computer Science.
22. Oommen, B. J., and Ma, D. C. Y., "Deterministic Learning Automata Solutions to the Equi-Partitioning Problem", IEEE Transactions on Computers, Vol. 37, January 1988, pp.2-14.
23. Oommen, B. J., and Ma, D. C. Y., "Stochastic Automata Solutions to the Object Partitioning Problem". To appear in The Computer Journal.
24. Rivest, R.L.,"On Self Organizing Sequential Search Heuristics", Comm. ACM, Vol. 19, 1976, pp. 63-67.
25. Sleator, D. D. and Tarjan, R. E., "Self-adjusting binary search trees", J.ACM, 32(1985), pp.652-686.
26. Schkolnick, M., " A Clustering Algorithm for Hierarchical Structures", ACM Trans. on Database Systems, 1977, pp.27-44.
27. Tenenbaum, A.M. and Nemes, R.M., "Two Spectra of Self-Organizing Sequential Search Algorithms", SIAM J. Comput., Vol.11, 1982, pp-557-566.
28. Van Rijsbergen, C.J., "A Theoretical Basis for the Use of Co-occurrence Data in Information Retrieval", J. Documentation, 1977, pp.106-119.
29. Walker, W. A. and Gotlieb, C. C., "A top-down algorithm for constructing nearly optimal lexicographical trees", in Graph Theory and Computing, Academic Press, New York, 1972.
30. Yu, C.T., and Salton, G., "Precision Weighting - An Effective Automatic Indexing Method", J. ACM, 1976, pp.76-78.
31. Yu, C.T., Siu, M.K., Lam, K., and Ozsoyoglu, M., "Performance Analysis of Three Related Assignment Problems", Proc. of the ACM SIGMOD Conference, May 1979.
32. Yu, C.T., Suen, C.M., Lam, K. and Siu, M.K.,, "Adaptive Record Clustering ", ACM Trans. on Database Systems, 1985, pp.180-204.
33. Yu, C.T., Siu, M.K., Lam K., and Tai, F., "Adaptive Clustering Schemes : General Framework", Proc. of the IEEE COMPSAC Conference, 1981, pp.81-89.

Carleton University, School of Computer Science
Bibliography of Technical Reports

SCS-TR-120	**Searching on Alphanumeric Keys Using Local Balanced Tree Hashing** E.J. Otoo, August 1987.
SCS-TR-121	**An O(√n) Algorithm for the ECDF Searching Problem for Arbitrary Dimension on a Mesh-of-Processors** Frank Dehne and Ivan Stojmenovic, October 1987.
SCS-TR-122	**An Optimal Algorithm for Computing the Voronoi Diagram on a Cone** Frank Dehne and Rolf Klein, November 1987.
SCS-TR-123	**Solving Visibility and Separability Problems on a Mesh-of-Processors** Frank Dehne, November 1987.
SCS-TR-124	**Deterministic Optimal and Expedient Move-to-Rear List Organizing Strategies** B.J. Oommen, E.R. Hansen and J.I. Munro, October 1987.
SCS-TR-125	**Trajectory Planning of Robot Manipulators in Noisy Workspaces Using Stochastic Automata** B.J. Oommen, S. Sitharam Iyengar and Nicte Andrade, October 1987.
SCS-TR-126	**Adaptive Structuring of Binary Search Trees Using Conditional Rotations** R.P. Cheetham, B.J. Oommen and D.T.H. Ng, October 1987.
SCS-TR-127	**On the Packet Complexity of Distributed Selection** A. Negro, N. Santoro and J. Urrutia, November 1987.
SCS-TR-128	**Efficient Support for Object Mutation and Transparent Forwarding** D.A. Thomas, W.R. LaLonde and J. Duimovich, November 1987.
SCS-TR-129	**Eva: An Event Driven Framework for Building User Interfaces in Smalltalk** Jeff McAffer and Dave Thomas, November 1987.
SCS-TR-130	**Application Frameworks: Experience with MacApp** John R. Pugh and Cefee Leung, December 1987.
SCS-TR-131	**An Efficient Window Based System Based on Constraints** Danny Epstein and Wilf R. LaLonde, March 1988.
SCS-TR-132	**Building a Backtracking Facility in Smalltalk Without Kernel Support** Daryl H. Graf and Wilf R. LaLonde , April 1988.
SCS-TR-133	**NARM: The Design of a Neural Robot Arm Controller** Wilf R. LaLonde and Mark Van Gulik, March 1988.
SCS-TR-134	**Separating a Polyhedron by One Translation from a Set of Obstacles** Otto Nurmi and Jörg-R. Sack, December 1987.

Carleton University, School of Computer Science
Bibliography of Technical Reports

SCS-TR-135	**An Optimal VLSI Dictionary Machine for Hypercube Architectures** Frank Dehne and Nicola Santoro, April 1988.
SCS-TR-136	**Optimal Visibility Algorithms for Binary Images on the Hypercube** Frank Dehne, Quoc T. Pham and Ivan Stojmenovic, April 1988.
SCS-TR-137	**An Efficient Computational Geometry Method for Detecting Dotted Lines in Noisy Images** F. Dehne and L. Ficocelli, May 1988.
SCS-TR-138	**On Generating Random Permutations with Arbitrary Distributions** B. J. Oommen and D.T.H. Ng, June 1988.
SCS-TR-139	**The Theory and Application of Uni-Dimensional Random Races With Probabilistic Handicaps** D.T.H. Ng, B.J. Oommen and E.R. Hansen, June 1988.
SCS-TR-140	**Computing the Configuration Space of a Robot on a Mesh-of-Processors** F. Dehne, A.-L. Hassenklover and J.-R. Sack, June 1988.
SCS-TR-141	**Graphically Defining Simulation Models of Concurrent Systems** H. Glenn Brauen and John Neilson, September 1988
SCS-TR-142	**An Algorithm for Distributed Mutual Exclusion on Arbitrary Networks** H. Glenn Brauen and John E. Neilson
SCS-TR-143	**Time is Not a Healer: Impossibility of Synchronous Agreement in Presence of Transmission Faults** N. Santoro, November 1988
SCS-TR-144	**Distributed Function Evaluation in Presence of Transmission Faults** N. Santoro and P. Widmayer, November 1988
SCS-TR-145	**Fault Intolerance of Synchronous Networks** N. Santoro and P. Widmayer, November 1988
SCS-TR-147	**On Transparently Modifying Users' Query Distributions** B.J. Oommen and D.T.H. Ng, November 1988
SCS-TR-148	**Smallscript: A User Programmable Framework Based on Smalltalk and Postscript** Kevin Haaland and Dave Thomas, November 1988