A New Technique for Enhancing Linked-List Data Retrieval: Reorganize Data Using Artificially synthesized Queries

B. J. OOMMEN AND D. T. H. NG

School of Computer Science, Carleton University, Ottawa K1S 5B6, Canada

Let $\mathbf{R} = \{R_1, R_2, \ldots, R_N\}$ be a set of data elements. The elements of R are accessed by the users of the system according to a fixed but unknown distribution $S = \{s_1, s_2, \ldots, s_N\}$, referred to as the users' query distribution. In this paper we consider the problem of organizing data so as to optimize its retrieval. However, rather than organize the data according to Ω, the stream of queries presented by the user, we suggest a scheme by which the data is organized based on a synthesized query stream Ω'. This synthesized stream possesses an underlying distribution, S'. Thus, in effect, the data organization is achieved according to the distribution S' and so, in one sense, the user's query distribution is modified without his knowing it. Furthermore, we show how this transformation can be done in such a way that the data storage achieved according to S' will be superior to that achieved if the data was stored according to the distribution S. The module which achieves this transformation is called a Distribution Changing Technique (DCT) Filter. In this paper we shall present the theory of DCT filters in its mathematical generality. We shall show that a DCT filter can be represented as Stochastic Mealy Automaton. Various DCT filters will be catalogued and, in particular, a filter F* will be presented. It has been shown that this filter transforms the original distribution expediently, and thus accentuates the information contained in the user's distribution. The problem of cascading DCT filtes has also been studied, and extensive computational and simulation results have been included which justify the theoretical results which have been presented.

Received August 1993; revised June, 1994

1. 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 it is minimized. Also, when data is to be stored, one of the primary considerations is the data structure used in the implementation and this depends on operations permitted on the data. Indeed, one of the most fundamental operations done on stored data is that of data retrieval because once the data has been retrieved, it can be modified or used for subsequent processing. Thus a vast majority of the research that has been done, both in applied and theoretical computer science, involves various storage techniques and the algorithms required to maximize the retrieval efficiency. Also, much of this research involves the study of how the data to be stored can be adaptively reorganized so as to increase the retrieval characteristics. This is the primary focus of this paper.

To pose the problem that we are studying in the right perspective we consider the following simplified model of data organization. Let $\mathbf{R} = \{R_1, R_2, \ldots, R_N\}$ be a set of data elements, where the elements of **R** are accessed by the users of the system according to a stationary distribution $S = \{s_1, s_2, \ldots, s_N\}$. S is called the users' query distribution. If S is known *a priori*, then the elements of **R** can be statically arranged so as to maximize the data retrieval efficiency. Indeed, in this case if **R** is arranged as a linear list, the retrieval is optimized when the elements of **R** are arranged in the descending order of their access probabilities (Allen and Munro, 1978; Arnow and Tenebaum, 1982; Bitner, 1979; Gonnet *et al.*, 1981; Hendricks, 1976; Hester and Hirschberg, 1985). Alternatively, if **R** is arranged as a binary search tree, the optimal tree can be computed by using the techniques developed by Knuth (1973) and 'almost optimal' trees can be generated by resorting to schemes such as those suggested by Walker and Gotlieb (1972).

For the sake of simplicity, let us assume that **R** is arranged as a linear list. Then for every distribution in which $s_1 > s_2 > s_3 \ldots > s_N$, the best list ordering is the one in which R_i precedes R_j if $i < j$. Thus we observe that whole families of distributions support the same optimal ordering. To clarify the point, consider the case when $N = 4$, and S_1 and S_2 are the distinct distributions:

$$S_1 = [0.252, 0.251, 0.249, 0.248]^T$$

$$S_2 = [0.8, 0.1, 0.07, 0.03]^T.$$

The optimal arrangement for both these query distributions is clearly the list $[R_1, R_2, R_3, R_4]$. However, whereas S_1 is almost flat in that the access probabilities are hardly distinguishable from each other, S_2 distinguishes between these probabilities in a more pronounced manner.

The above argument is very easy to follow and would have been sufficient if we were not faced with the unfortunate scenario that the user's query distribution is always unknown. Clearly, some process which directly or indirectly estimates this would have to be used before the above static organizing strategy can be employed. Observe, however, that although both S_1 and S_2 recommend the same optimal ordering, it is easier to learn this ordering if the user's query stream was generated from S_2 as opposed to it being generated from S_1. The reason for this is because the distribution S_1 is almost flat and the information contrast which leads to the conclusion that R_1 should be before R_2, etc., is practically concealed. On the other hand, this information is much more prominent in S_2, primarily because the contrast between the s_i's is more marked.

In this paper, we endeavor to achieve a goal which (to our knowledge) has not been studied in the areas of information storage and retrieval. Since the user's query distribution is unknown, data organization is typically achieved by studying a stream of user's queries. Let Ω be this stream of queries generated by the user based on his query distribution S. In this paper we shall present a methodology by which a superior stream of queries can be synthesized by gleaning the more important information in Ω and concealing the less important information resident in Ω. Thus, given the stream of queries Ω based on the user's query distribution S, we shall endeavor to synthesize an auxiliary stream Ω', based on which the data structure would be maintained. The new query stream Ω' has an underlying synthesized distribution S' in which the relative ratios of the original access probabilities are accentuated. Thus in the original distribution S, if $s_i > s_j$, the new distribution S' will satisfy $s_i' > s_j'$, and furthermore the ratio s_i'/s_j' will be greater than the previous corresponding ratio s_i/s_j. Thus if data organization is achieved on the basis of S' (as opposed to S), the probability of arriving at the optimal ordering can be increased, and this will, generally speaking, be true independent of the technique by which the data is organized.

If the s_i's are not known *a priori*, an estimation of these probabilities can be used to infer the optimal order. In such a case, the estimation procedure which uses the synthesized queries, Ω', will be much more accurate (in terms of converging to the optimal ordering) than the corresponding one which merely uses the less informative query stream Ω.

Although estimation of the access probabilities is a feasible solution, it is very seldom resorted to. Researchers have discouraged such an estimation procedure primarily to save on the additional time and storage requirements. Thus a vast body of research has gone into adaptively having the data structure organize itself, using memoryless heuristics. In the case of linear lists, various list organizing strategies have been presented and these include the move-to-front (MTF), the transposition rule (TR), the POS(K) rule, etc. (Allen and Munro, 1978; Arnow and Tenebaum, 1982; Bitner, 1979; Gonnet *et al.*, 1981; Hendricks, 1976; Hester and Hirschberg, 1985). In the MTF rule, the list is dynamically reorganized by moving the accessed element to the front of the list everytime it is accessed. In this case, it can be shown that the ultimate probability of a record R_i preceding another record R_j is given by $s_i/(s_i + s_j)$ (Hendricks, 1976; Hester and Hirschberg, 1985). Furthermore, if $s_1 > s_2 > s_3 \ldots > s_N$, (i.e. the optimal ordering is the list $[R_1, R_2, \ldots, R_N]$), the asymptotic probability for the MTF rule being in the optimal configuration is (Hendricks, 1976):

$$\Pr([R_1, R_2, \ldots, R_N]|S) = \prod_{i=1}^{N} \frac{s_i}{\sum_{j=1}^{i} s_j} = \prod_{i=1}^{N} \frac{1}{1 + \sum_{j>i} \frac{s_j}{s_i}} \quad (1)$$

Consider now the case when the stream of queries is generated by the distribution S_1, where, $S_1 = [0.252, 0.251, 0.249, 0.248]^T$. In this case the above probability has the pitiably small value of 0.0443. However if the query stream was more informative and was generated by the distribution $S_2 = [0.8, 0.1, 0.07, 0.03]^T$, the MTF rule would converge to the optimal ordering with probability 0.28. Thus notice that although both S_1 and S_2 recommend the same optimal ordering, MTF would converge to this ordering with a probability which is an order of magnitude higher in the case of S_2. The reason quite simply is that S_1 is almost 'flat' and the information content which distinguishes between the various permutations is negligible. As opposed to this, the distribution S_2 displays the difference between the access probabilities much more significantly.

In this paper our endeavor is to accentuate the relative differences between the access probabilities. Thus by generating the synthesised stream of queries, Ω', this stream will obey the distribution S' such that $s_i'/s_j' > s_i/s_j$ whenever $s_i > s_j$. A straightforward study of (1) reveals that using the MTF rule,

$$\Pr([R_1, R_2, \ldots, R_N]|S') > \Pr([R_1, R_2, \ldots, R_N]|S).$$

Thus, if we can achieve our goal, the synthesized stream of queries, Ω', will cause a convergence to the optimal ordering with a much higher probability than the original user's query stream Ω.

Various other list organizing strategies have also been proposed in the literature. A review of the ergodic schemes is found in Bitner (1979), Gronnet *et al.* (1981) and Hester and Hirschberg (1985). Linear and constant memory absorbing schemes with expedient and optimal properties are found in Oommen and Hansen (1987) and Oommen *et al.* (1990). In all the cases reported in the literature, if the reorganization is done based on synthesized stream Ω' (obeying S'), we believe that the probability of being in the optimal configuration is greater than that obtained when the users' query stream is used.

Accentuating (or polarizing) the user's query distribution has definitely proven advantages for linear lists. Although, in this paper, we have only shown the advantage of polarizing when the linear list is the physical data structure, we believe that such advantages can also be gained if the data structures used are binary trees or their variants (Bayer, 1975; Chetham *et al.*, 1988; Gonnet *et al.*, 1981; Knuth, 1973) or heaps typically used in large database applications. However, the theoretical advantage gained by polarizing in the context of these data structures is yet to be analyzed. We also believe that such advantages can be gleaned if the data is accessed in pairs (or clusters) (Hammer and Niamir, 1979; Ma, 1986; Oommen and Ma, 1988; 1992; Schkolnick, 1977; Van Rijsbergen, 1977; Yu and Salton, 1976; Yu *et al.*, 1981, 1985). In these cases the query synthesizer will have to manipulate sets of query elements and yield a synthesized *set* stream.

To present our problem in a more formal setting, we assume that S is unknown and that Ω is a query stream generated based on S. Let us suppose that the user requests the element R_i. This event occurs with probability s_i. This query is processed by presenting the user with the record that he has requested, i.e. R_i. However, the user's query is not the one which is primarily used to reorganize the data. The request R_i is fed into a filter referred to as the Distribution Changing Technique (DCT) Filter, and the output of this filter is used to achieve the reorganization. A brief synoptic description as to how this filter works is not out of place.

The DCT Filter is completely defined by two quantities, i.e. a list operator, Ω, and a function, called the Report Function, τ. The filter manipulates a dummy list using the list operator Ω. It then emits the synthesized request for the data element R_j (where R_j may be the same as R_i) based on the Report Function, τ. This element is now reorganized according to the rule used by the data reorganization strategy. However, as a result of the DCT Filter, the request to re-position the element R_j is emitted with a probability s'_j. Thus the data reorganization is achieved on the basis of the distribution $S' = \{s'_1, s'_2, \ldots, s'_N\}$. Indeed, if $s'_i/s'_j > s_i/s_j$ whenever $s_i > s_j$, the reorganization obtained due to the distribution S' is superior to the reorganization that would have resulted as a consequence of using the distribution S.

Rather than merely discuss a particular DCT Filter, in this paper, we shall present the theory of DCT filters in its mathematical generality. A general DCT filter will be shown to be a stochastic Mealy automaton (Narendra and Thathachar, 1989). Additionally, 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* expediently polarizes (accentuates) the user's query distribution by presenting a query stream generated according to S' in which $s'_i/s'_j > s_i/s_j$ whenever $s_i > s_j$.

The details of how the list operator, Ω, is chosen and the technique for deciding on the element R_j to be reported in the synthesized query will be discussed in the subsequent sections. The most interesting property of this mechanism is that the DCT is 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. Informally speaking, the DCT thus represents a computer scientist's 'magic hat' into which a user puts in a particular stream of queries but from which he is able to pull out a far more informative stream of queries.

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 optimally.

Although the paper is primarily theoretical, it also contains numerous computational and simulation results which clarify and justify the analytic results which have been derived. We are currently investigating how these principles can be utilized in a real-life local area network which has a single file server with the hope of enhancing the network's data retrieval capabilities.

2. FUNDAMENTALS AND NOTATION

2.1. Lists and list operations

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

$$\sum_{j=1}^{N} s_j = 1.$$

A list $\lambda(n)$ defined at time instant 'n' as $\lambda(n) = L_1(n)L_2(n)\ldots L_N(n)$, is a linear ordering of the data elements where for all i and j, $L_i(n) \neq L_j(n)$. 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 the next time instant. Two list operators which we shall use extensively are now defined.

2.1.1. 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,

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

Then, if $R_u = L_k$,

$\lambda(n+1) \equiv \Omega_{MTF}(\lambda(n), R_u)$, where,

$$\lambda(n+1) = \lambda(n) \quad \text{if } k = 1, \text{ and,}$$
$$\lambda(n+1) = L_k L_1 L_2 \ldots L_{k-1} L_{k+1} \ldots L_N \quad \text{if } k > 1. \tag{3}$$

Ω_{MTF} is the studied Move-To-Front rule which moves the accessed element to the front of the list.

2.1.2 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,

$$\lambda(n+1) = \lambda(n) \quad \text{if } k = 1, \text{ and,}$$
$$\lambda(n+1) = L_1 L_2 \ldots L_k L_{k-1} L_{k+1} \ldots L_N \quad \text{if } k > 1. \tag{4}$$

2.1.3. A generalized list operator

Apart from Ω_{MTF} and Ω_{Tr}, a list operator can be defined in all generality by an ordered list π, where, $\pi = (\pi_1, \pi_2, \ldots, \pi_N)$, and 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 $\pi_k = \pi_{k(1)}\pi_{k(2)} \ldots \pi_{k(N)}$, 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:

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

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 properties of the operators.

LEMMA 1. *If Ω_{MTF} is the list reorganizing operator defined in 2.1.1, then for all distinct indices i and j, the asymptotic probability of R_i preceding R_j is:*

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

Proof. The proof of the lemma is found in Rivest (1976).

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. Much of the analysis of the scheme can be comprehended if one perceives the time reversibility (Ross, 1981) of the underlying Markov chain. The transition map of the chain is shown below in Figure 1 for the case when the list $\mathbf{R} = \{R_1, R_2, R_3\}$. To see the time reversibility of the chain we consider the following path from state $(R_1R_2R_3)$ to itself in a clockwise fashion as:

$$(R_1R_2R_3) \to (R_2R_1R_3) \to (R_2R_3R_1) \to (R_3R_2R_1)$$
$$\to (R_3R_1R_2) \to (R_1R_3R_2) \to (R_1R_2R_3).$$

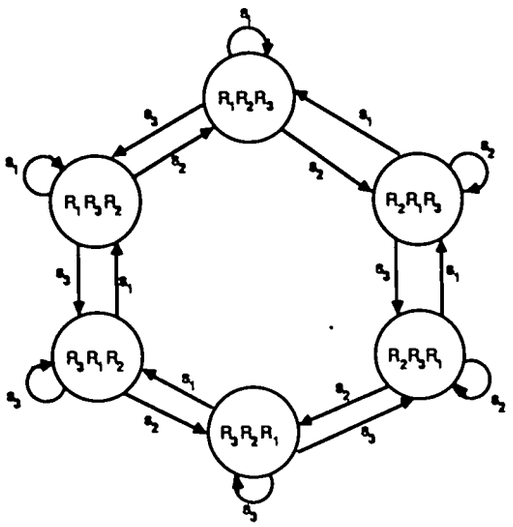

FIGURE 1 Transition map of the tranposition scheme, when $\mathbf{R} = \{R_1, R_2, R_3\}$. Note the time reversibility of the chain.

Clearly, the product of the transition probabilities is $(s_1s_2s_3)^2$. Indeed, this is exactly the same value if the paths were traversed in a counter-clockwise fashion.

The theorem about the asymptotic properties of Ω_{Tr} now follows. The theorem was originally proved by Rivest (1976) but the critical details were omitted. A detailed proof of this result is found in Ross (1981). Although it is not included here in the interest of brevity, we encourage the reader to go through the salient details of the proof because the principles used there will be used extensively later in subsequent proofs, and in the description of DCT filters.

THEOREM 1. *For Ω_{Tr}, the Transposition Operator, let the asymptotic stationary probability of the list being in $(R_{i_1}R_{i_2}\ldots R_{i_N})$ be expressed as $Pr(R_{i_1}R_{i_2}R_{i_N})$. Then, if all the s_j's are positive, the stationary probabilities of the list, obey:*

$$\frac{\Pr(R_{i_1}R_{i_2}\ldots R_{i_j}R_{i_{j+1}}\ldots R_{i_N})}{\Pr(R_{i_1}R_{i_2}\ldots R_{i_{j+1}}R_{i_j}\ldots R_{i_N})} = \frac{s_{ij}}{s_{ij+1}} \tag{6}$$

Proof The proof of the result is found in Rivest (1976) and Ross (1981).

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 formally proved in the literature—although they have been alluded to in Rivest (1976). Since they are crucial for the later sections of the paper we will now state and sketch their proofs.

THEOREM 2. *For Ω_{Tr}, the Transposition operator, let the asymptotic stationary probability of the list being in $(R_{i_1} R_{i_2} \ldots R_{i_N})$ be expressed as Pr $(R_{i_1} R_{i_2} \ldots R_{I_N})$. Then,*

$$\frac{\Pr(R_{i_1} R_{i_2} \ldots R_{i_{j-1}} R_{i_j} R_{i_{j+1}} R_{i_{j+2}} \ldots R_{i_N})}{\Pr(R_{i_1} R_{i_2} \ldots R_{i_{j-1}} R_{i_{j+2}} R_{i_{j+1}} R_{i_j} \ldots R_{i_N})} = \left(\frac{s_{i_j}}{s_{i_j+2}}\right)^2 \quad (7)$$

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

$$\begin{aligned}
&\frac{\Pr(R_{i_1} R_{i_2} \ldots R_{i_{j-1}} R_{i_j} R_{i_{j+1}} R_{i_{j+2}} \ldots R_{i_N})}{\Pr(R_{i_1} R_{i_2} \ldots R_{i_{j-1}} R_{i_{j+2}} R_{i_{j+1}} R_{i_j} \ldots R_{i_N})} \\
&= \frac{\Pr(R_{i_1} R_{i_2} \ldots R_{i_{j-1}} R_{i_j} R_{i_{j+1}} R_{i_{j+2}} \ldots R_{i_N})}{\Pr(R_{i_1} R_{i_2} \ldots R_{i_{j-1}} R_{i_{j+1}} R_{i_j} R_{i_{j+2}} \ldots R_{i_N})} \\
&\quad \times \frac{\Pr(R_{i_1} R_{i_2} \ldots R_{i_{j-1}} R_{i_{j+1}} R_{i_j} R_{i_{j+2}} \ldots R_{i_N})}{\Pr(R_{i_1} R_{i_2} \ldots R_{i_{j-1}} R_{i_{j+1}} R_{i_{j+2}} R_{i_j} \ldots R_{i_N})} \\
&\quad \times \frac{\Pr(R_{i_1} R_{i_2} \ldots R_{i_{j-1}} R_{i_{j+1}} R_{i_{j+2}} R_{i_j} \ldots R_{i_N})}{\Pr(R_{i_1} R_{i_2} \ldots R_{i_{j-1}} R_{i_{j+2}} R_{i_{j+1}} R_{i_j} \ldots R_{i_N})}. \\
&= \frac{s_{i_j}}{s_{i_{j+1}}} \frac{s_{i_j}}{s_{i_{j+2}}} \frac{s_{i_{j+1}}}{s_{i_{j+2}}} = \left(\frac{s_{i_j}}{s_{i_{j+2}}}\right)^2, \quad (8)
\end{aligned}$$

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

Remark. A generalization of Theorem 2 is given by the following equation:

$$\frac{\Pr(R_{i_1} R_{i_2} \ldots R_{i_{j-1}} R_{i_j} R_{i_{j+1}} \ldots R_{i_{j+M-1}} R_{i_{j+M}} R_{i_{j+M+1}} \ldots R_{i_N})}{\Pr(R_{i_1} R_{i_2} \ldots R_{i_{j-1}} R_{i_{j+M}} R_{i_{j+1}} \ldots R_{i_{j+M-1}} R_{i_j} R_{i_{j+M+1}} \ldots R_{i_N})} = \left(\frac{s_{i_j}}{s_{i_{j+M}}}\right)^M \quad (9)$$

The proof of the result is analogous to the above proof and is omitted.

We are now in a position to present our theory on DCT filters.

3. DCT FILTERS

3.1. The report function, τ

Central to the theory of DCT filters 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 $R_u \in R$, then

$$\tau(\lambda, R_u) \equiv L_j \quad (10)$$

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.

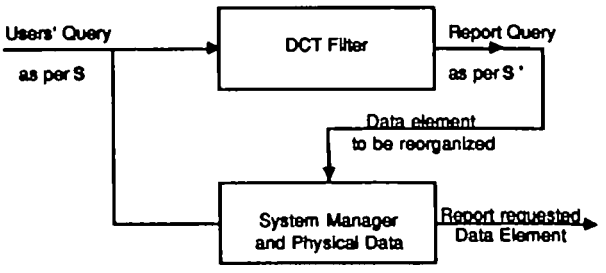

FIGURE 2. Schematic of the DCT filter.

3.2. 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 $\lambda(n) = L_1 L_2 \ldots L_N$ where each $L_i \in R$ and $L_i \neq L_j$ for all i and j. When the user requests an element R_u, the query is satisfied by providing *this* element to the user. However, the system does not reorganize the physical data based on the accessed element R_u. Rather, the DCT fitler uses its internal function τ and reports an element $L_j = \tau(\lambda, R_u)$ as a synthesized query which is used in the data reorganization strategy. The DCT filter then uses its list operator Ω and R_u to update the (hypothetical) list $\lambda(n+1)$.

In terms of nomenclature, let the users' query distribution be S and let the asymptotic distribution of Ω', the reported stream be S'. A DCT filter is called an identity filter if $S = S'$. This implies that if $S = \{s_1, s_2, \ldots, s_N\}$ and $S' = \{s'_1, s'_2, \ldots, s'_N\}$, 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.

Observe that the physical data is reorganized based on the distribution of the synthesized queries as reported by $\tau(.,.)$ and not based on the user's query distribution, S. Also note that we have emphasized that for a DCT filter $\lambda(n)$ is a hypothetical list. This is because this list 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. Typically, $\lambda(0)$ is randomly assigned.

The formal DCT Filter is algorithmically given below and schematically in Figure 2. After presenting them we proceed to state and prove the properties of various physical DCT Filters.

ALGORITHM DCT_FILTER (R_u, L_j)
Input: A data element requested $R_u \in \mathbf{R}$.
Output: A data element $L_j \in \mathbf{R}$ 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.
Method:
  ReadInput (R_u)
  $L \leftarrow \tau(\lambda, R_u)$
  $\lambda \leftarrow \Omega(\lambda, R_u)$
  Output (L)
End ALGORITHM DCT_FILTER

THEOREM 3. *The DCT filter F_1 defined for $\Omega = \Omega_{MTF}$ and with $\tau(.,.) = L_I$ is an identity filter.*

Proof. The list maintained by the DCT Filter is $\lambda = L_1 L_2 \ldots L_N$. 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 as the synthesized query element to the data reorganization scheme at time (n + 1). Hence the theorem.

THEOREM 4. *Let $\Omega = \Omega_{Tr}$ and $\lambda = L_1 L_2 \ldots L_N$. Further let τ_T be the report function defined as follows:*

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

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 $\Gamma(R_{i_j}, R_{i_{j+1}})$.

Using Theorem 1 and considering all the possible locations for the sequence $R_{i_j}, R_{i_{j+1}}$,

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

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

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

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

$$\frac{\Pr(R_{i_j} R_{i_{j+1}} R_{i_1} \ldots R_{i_{j-1}} R_{i_{j+2}} \ldots R_{i_N}) + \ldots + \Pr(R_{i_1} \ldots R_{i_{j-1}} F_{i_{j+2}} \ldots R_{i_N} R_{i_j} R_{i_{j+1}})}{\Pr(R_{i_{j+1}} R_{i_j} R_{i_1} \ldots R_{i_{j-1}} R_{i_{j+2}} \ldots R_{i_N}) + \ldots + \Pr(R_{i_1} \ldots R_{i_{j-1}} R_{i_{j+2}} \ldots R_{i_N} R_{i_{j+1}} R_{i_j})} = \frac{s_{i_j}}{s_{i_{j+1}}}. \tag{12}$$

At this point it is important to notice that $(R_{i_1} R_{i_2} \ldots R_{i_N})$ can represent any permutation. If we sum *all* the probabilities in (12) for all the possible values for $R_{i_1} R_{i_2} \ldots R_{i_{j-1}} R_{i_{j+2}} \ldots R_{i_N}$, we indeed have the quantity $\Gamma(R_{i_j}, R_{i_{j+1}})$. Thus, we have,

$$\frac{\Gamma(R_{i_j}, R_{i_{j+1}})}{\Gamma(R_{i_{j+1}}, R_{i_j})} = \frac{s_{i_j}}{s_{i_{j+1}}}. \tag{13}$$

We now have to compute the asymptotic probability of the filter yielding R_u as its output. We shall utilize the asymptotic properties of Ω_{Tr} to evaluate the latter. Using the laws of total probability we know that:

$$\Pr[\text{Output} = R_i] = \sum_{k=1}^{N} \Pr[\text{Output} = R_i | \text{Input} = R_k] \cdot s_k$$

$$= \sum_{k \neq i} \Pr[\text{Output} = R_i | \text{Input} = R_k] \cdot s_k$$

$$+ \Pr[\text{Output} = R_i | \text{Input} = R_i] \cdot s_i$$

However, for the DCT filter (Ω_{Tr}, τ_T) the quantity Pr [Output = R_i|Input = R_k] is exactly $\Gamma(R_i, R_k)$. Furthermore for (Ω_{Tr}, τ_T) the probability

$$\Pr[\text{Output} = R_i | \text{Input} = R_i] = \Pr[R_i \text{is the first element in } \lambda].$$

However, (12) states that:

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

Thus,

$$\Pr[R_i \text{is the output}]$$

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

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

However, the quantity

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

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 unity. The theorem follows.

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

3.3. Polarizing DCT filters

Let the user's query distribution be S and let the asymptotic distribution of the elements reported by the DCT filter be $S' = \{s'_1, s'_2, \ldots, s'_N\}$. A DCT filter is said to polarize expediently if for all i and j.

$$s'_1/s'_j > s_i/s_j \quad \text{whenever} \quad s_i > s_j.$$

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

In the setting of the paper the reader should observe that if we can design a polarizing DCT filter, it will indeed accentuate the information content of the users' query distribution and thus lead to superior data retrieval. The whole problem of designing such polarizing filters is extremely fascinating. We shall present one such filter and refer to it as F*.

THEOREM 5. *Let* $F^* = (\Omega_{Tr}, \tau^*)$, *where if* $\lambda = L_1 L_2 \ldots L_N$,

$$\tau^*(\lambda, R_u) = L_1. \tag{14}$$

Then F is an expediently polarizing DCT filter.*

Proof. First of all, to get an intuitive feeling for the filter, it is advantageous to observe that the filter maintains a hypothetical list λ constantly reorganized using the Transposition rule. The synthesized query element presented in the synthesized query stream is merely the first element of *this* list.

More formally, 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,

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

Also we note that since $\tau^*(.,.) = L_1$, the asymptotic probability $s_i' \equiv \Pr[\text{Output} = R_i]$ 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 that from Theorem II that for any R_k,

$$\frac{\Pr(R_i R_k R_j \ldots)}{\Pr(R_j R_k R_i \ldots)} = \left(\frac{s_i}{s_j}\right)^2 \tag{15}$$

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,

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

In an analogous way, we can show that:

$$\frac{\alpha_3}{\beta_3} = \left(\frac{s_i}{s_j}\right)^3$$

Indeed, in general, for all $2 \leqslant p \leqslant N-1$, we have:

$$\frac{\alpha_p}{\beta_p} = \left(\frac{s_i}{s_j}\right)^p \tag{16}$$

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:

$$\frac{s_i'}{s_j'} = \frac{\Pr[L_1 = R_i]}{\Pr[L_1 = R_j]} = \frac{\sum_{p=1}^{N-1} \alpha_p}{\sum_{p=1}^{N-1} \beta_p} \tag{17}$$

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

$$\frac{s_i'}{s_j'} = \frac{\sum_{p=1}^{N-1} \alpha_p}{\sum_{p=1}^{N-1} \beta_p} = \frac{\sum_{p=1}^{N-1} \rho^p \beta_p}{\sum_{p=1}^{N-1} \beta_p} \tag{18}$$

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

$$a_1^{x_1} a_2^{x_2} \ldots a_R^{x_R} < \left(\frac{x_1 a_1 + x_2 a_2 + \ldots + x_R a_R}{x_1 + x_2 + \ldots + x_R}\right)^{x_1+x_2+\ldots+x_R} \tag{19}$$

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,

$$\frac{s_i'}{s_j'} > \rho^{\xi}$$

where

$$\xi = \frac{\beta_1 + 2\beta_2 + \ldots + (N-1)\beta_{N-1}}{\beta_1 + \beta_2 + \ldots + \beta_{N-1}} \tag{20}$$

Since both ρ and ξ are strictly greater than unity, the ratio s_i'/s_j' is strictly greater than ρ. Hence, $s_i'/s_j' > s_i/s_j$, and the theorem is proved.

Remarks

(i) As mentioned earlier, this filter maintains the hypothetical list λ which is constantly reorganized using the transposition rule. The synthesized query element presented in the synthesized query stream is merely the first element of *this* list. We have yet been unable to get an intuitive reasoning for why this scheme works so efficiently. Informally, we have been provided (we have presented these results at various seminars and are grateful to many of those who participated in these seminars for their lively discussions) with various arguments that seem to support the various intuitive perspectives, but these arguments collapse when we realize that the exact same report function operates detrimentally when used in conjunction with alternative list operators.

(ii) The expression (20) is quite revealing. Observe that the exponent of ρ is the ratio of the summation $\Sigma_{p=1}^{N-1} p\beta_p$ to the summation $\Sigma_{p=1}^{N-1} \beta_p$. However, 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

FIGURE 3. A typical Mealy transition of F* when $\mathbf{R} = \{R_1, R_2, R_3, R_4\}$. With an input of R_3, the state changes as per Ω_{Tr} from $R_2R_1R_4R_3$ to $R_2R_1R_3R_4$. The output generated as the synthesized query element is R_2 as per τ^*.

location of R_i given that $L_l = R_j$, we observe that the exponent of ρ is merely the **expected** value of the position of R_i. This gives a good 'rule-of-thumb' estimate for R_j being reported as the synthesized query element to be used in the physical data reorganization process.

(iii) If we apply the general theory of mean values [Hardy *et al.* 1983] using the set of $\{\alpha_p\}$ (instead of using it as above to the set $\{\beta_p\}$), we can derive an **upper** bound on the ratio s_i'/s_j'. Indeed, applying (19) into (18) using the $\{\alpha_p\}$ we can simplify the expression to show that:

$$\frac{s_i'}{s_j'} < \rho^{\eta}$$

where

$$\eta = \frac{\alpha_1 + 2\alpha_2 + \ldots + (N-1)\alpha_{N-1}}{\alpha_1 + \alpha_2 + \ldots + \alpha_{N-1}} \qquad (21)$$

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). Furthermore, 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.

(iv) Viewed in its generality, a DCT Filter itself can be seen to be an automaton or a finite state machine. 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 Mealy 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 Mealy transition for F* is in Figure 3 when $\mathbf{R} = \{R_1, R_2, R_3, R_4\}$.

(v) Throughout the course of our research endeavours 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.

(vi) Although we have presented only one expediently polarizing DCT Filter, it is an easy task to extend the results of Theorem 5 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

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

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

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

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

Considering total probabilities, it is easy to see that s_i'/s_j' will still be greater than s_i/s_j.

(vii) An interesting DCT filter which we have not analysed is the filter F_3, where,

$F_3 = (\Omega_{Tr}, \tau_3)$, and, τ_3 is defined as below for $\lambda = L_1L_2 \ldots L_N$.

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

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

4. 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 S. The intention was to present a synthesized stream of queries obeying a distribution S'. The aim of the whole exercise was to accentuate the distribution S in such a way that the data reorganization according to S' would be superior to that done as per S. Indeed, in the previous section, we presented a filter F* which operated on the input data distribution and expediently polarized it in such a way that if $s_i > s_j$, then, for all $s_i, s_j \in S$ whose corresponding access probabilities after the filter are $s_i', s_j' \in S'$,

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

However, we were unable to obtain a filter which absolutely polarized the distribution such that:

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

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

accomplish it. However, in the strictest sense, the analysis of cascading DCT Filters does not follow from the serial analysis of the individual filters. This difference, although not so obvious, is actually true, and is a consequence of the fact that when two Markov chains are cascaded, the new chain operates in the composite space. However, this computation is extremely well approximated if the following Equilibrium Assumption, is made.

Equilibrium Assumption. If DCT filters are cascaded, the input distribution of any filter is independent and equals the equilibrium output distribution of the preceding DCT filter.

The above assumption is an approximation as long as the corresponding composite Markov chain is in transition. When the chain has converged, it is valid to machine accuracy.

In this section we shall prove some counter-intuitive facts. We shall show that subject to the Equilibrium Assumption, 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 7. *Let F and G be two Identity DCT filters. Then, subject to the Equilibrium Assumption, if H is the DCT filter obtained by sequencing F and G, H is also an identity DCT filter.*

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

$$\frac{s_i^F}{s_j^F} = \frac{s_i}{s_j}. \tag{23}$$

If this distribution S^F is the input to the filter G, then, by definition, the output of G is $S^H = \{s_1^H, s_2^H, \ldots, s_N^H\}$ which for all i and j satisfy:

$$\frac{s_i^H}{s_j^H} = \frac{s_i^F}{s_j^F}.$$

Combining (23) with the above proves the theorem!

THEOREM 8. *Let F and G be two expediently polarizing DCT filters. Then, subject to the Equilibrium Assumption, 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 real number comparison operator '>'.

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

THEOREM 9. *Let G be any expediently polarizing filter. Then, subject to the Equilibrium Assumption, the DCT filter obtained by sequencing an infinite number of such filters need not be absolutely polarizing.*

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

$$\frac{s_i^{(K+1)}}{s_j^{(K+1)}} > \frac{s_i^{(K)}}{s_j^{(K)}} \quad \text{for all } k \geqslant 1.$$

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 $s_i \neq s_j$.

Consider the filter F^* defined by Theorem 5 as $F^* = (\Omega_{Tr}, \tau^*)$, where τ^* reports the first element of the list. Indeed, we shall show that for all non-trivial distributions in which N = 3,

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

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 $\{s_i^{(K)} | 1 \leqslant i \leqslant 3\}$. Then, using the properties of Ω_{Tr} and τ^*, it is seen after considerable algebra that:

$$s_1^{(K+1)} = \frac{(s_1^{(K)})^2 \cdot (s_2^{(K)}) + (s_1^{(K)})^2 \cdot (s_3^{(K)})}{\Delta} \tag{24}$$

$$s_2^{(K+1)} = \frac{(s_2^{(K)})^2 \cdot (s_3^{(K)}) + (s_2^{(K)})^2 \cdot (s_1^{(K)})}{\Delta} \tag{25}$$

$$s_3^{(K+1)} = \frac{(s_3^{(K)})^2 \cdot (s_1^{(K)}) + (s_3^{(K)})^2 \cdot (s_2^{(K)})}{\Delta} \tag{26}$$

where Δ is the normalizing denominator enforcing:

$$s_1^{(K+1)} + s_2^{(K+1)} + s_3^{(K+1)} = 1,$$

and is

$$\Delta = (s_1^{(K)})^2[s_2^{(K)} + s_3^{(K)}] + (s_2^{(K)})^2[s_3^{(K)} + s_1^{(K)}] + (s_3^{(K)})^2[s_1^{(K)} + s_2^{(K)}]$$

Since the s_i's are bounded and non-negative it is easy to see that this sequence converges 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:

$$\frac{s_1^*}{s_2^*} = \frac{(s_1^*)^2 \cdot (s_2^* + s_3^*)}{(s_2^*)^2 \cdot (s_3^* + s_1^*)}, \text{ and,} \tag{27}$$

$$\frac{s_1^*}{s_3^*} = \frac{(s_1^*)^2 \cdot (s_2^* + s_3^*)}{(s_3^*)^2 \cdot (s_1^* + s_2^*)} \tag{28}$$

Furthermore,

$$s_1^* + s_2^* + s_3^* = 1. \qquad (29)$$

Cross-multiplying (27) we get that (27)–(29) can be solved to yield an equilibrium solution whenever all three $\{s_i^*\}$ 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^* can be finite, and the theorem is proved.

Remark

(i) At the very outset, it may appear as if Theorems 8 and 9 contradict each other. Indeed, they do not, since Theorem 8 merely states that subject to the Equilibrium Assumption, the sequencing of expediently polarizing filters is always expediently polarizing. But since the product of an infinite sequence of numbers which are greater than or equal to unity need not be infinite, under the same assumption, the sequence of DCT filters need not be absolutely polarizing. Indeed, this is true in this case, since the ratio

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

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

(ii) Cascading DCT filters can yield quite fascinating results. Indeed, cascading filters which are not even identity filters (and which may even be 'anti-polarizing') can result in a polarizing filter. One example of such a filter is that which is obtained by cascading two filters each of which have Ω_{Tr} as their list operator, and which reports the **last** element of the list in the report function. Thus, if $\Omega = \Omega_{Tr}$ and $\lambda = L_1 L_2 \ldots L_N$, let τ_4 be the function defined as follows:

$$\tau_4(\lambda, R_u) = L_N.$$

Then it can be shown that subject to the Equilibrium Assumption, the filter obtained by cascading two F_4 DCT filters is expediently polarizing!!

5. 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,

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

and

$$a_i = 1 \quad \text{if} \quad s_i = \max_j s_j$$

$$= 0 \quad \text{otherwise.}$$

TABLE 1. Expected values of χ before and after going through the DCT filter F*

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

This is the chi-square measure quantifying the similarity between any distribution $\{s_i'\}$ and the distribution obtained for an absolute 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 $N > 1$, (S is the trivial distribution) χ has a positive value. As the ratio of the s_i's increases, the value of χ decreases and 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 and subsequent to the operation of the filter. These quantities are called χ_i and χ_o respectively.

Since N! is very large, it is impractical to theoretically measure χ_o 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 F*. This has been done for both the wedge and exponential distributions defined as follows. In the Wedge distribution, s_i has the value:

$$s_i = a - (i-1)\delta, \qquad \text{where} \quad a = [n(n-1)\delta + 2]/2N.$$

Note that the parameter δ increases the slope of the wedge.

In the case of the exponential distribution, s_i has the form:

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

The results of the computation are remarkable and are tabulated in Tables 1 and 2. For example, when $N = 4$, if $\delta = 0.4$ 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. This represents a multiplicative decrease of 63% in χ_i!

TABLE 2. Expected values of χ before and after going through the DCT filter F* for N = 10 and 15

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

For the case of $N > 7$, exact closed form expressions for the output probabilities are not easily derived. To obtain the corresponding values in these cases, we have resorted to simulations. 100 experiments were conducted using 20 000 queries based on the input distribution S, where S obeyed either a wedge or an exponential distribution. The output pattern of the stream of synthesized queries generated by the DCT filter was statistically analysed and the distribution S′ estimated. Using these estimates, the value of χ_i and χ_o are tabulated for various values of N in Table 2. Again the results are remarkable. For example, when N = 10, if $\delta = 0.1$, S is:

$$[0.153533 \quad 0.138180 \quad 0.124362 \quad 0.111926 \quad 0.100733$$
$$0.090660 \quad 0.081594 \quad 0.073434 \quad 0.066091 \quad 0.059482]^T.$$

The corresponding output distribution S′ is:

$$[0.311725 \quad 0.234812 \quad 0.069186 \quad 0.116155 \quad 0.075564$$
$$0.043810 \quad 0.025167 \quad 0.013415 \quad 0.006611 \quad 0.003555]^T$$

In this case χ_i is 0.583924, and χ_o is 0.801915 implying a multiplicative decrease of 72.8% in χ_i.

We have also performed computations to verify the validity of cascading, Theorem 9 and the Equilibrium Assumption. Using the filter F*, the input distribution:

$$S = [0.6 \quad 0.3 \quad 0.1]^T$$

yields an output distribution

$$S' = [0.6667 \quad 0.2917 \quad 0.04167]^T.$$

If this is the input to a suceeding filter F*, the output distribution at the second level is:

$$S^{(2)} = [0.7052 \quad 0.2858 \quad 0.009]^T.$$

At the third level, the distribution has the value:

$$S^{(3)} = [0.7139 \quad 0.2857 \quad 0.0004]^T,$$

which is remarkably close to its terminal value

$$S^{(\infty)} = [0.7143 \quad 0.2857 \quad 0.00]^T.$$

Notice that the terminal value of the ratio of $s_1^{(k)}$ to $s_i^{(k)}$ is finite just as Theorem 9 dictates.

6. CONCLUSIONS

In this paper we have presented a new technique for enhancing data retrieval by suggesting that the data be organized using artificially synthesized queries as opposed to the queries generated by the users. The intention of this exercise is to optimize the data retrieval characteristics of the system. Thus, if the data is accessed according to unknown distribution S, rather than organize the data according to Ω, the stream of queries presented by the user, the data is organized based on a synthesized stream Ω'. This synthesized stream obeys a distribution S′. As a consequence of this exercise, the user's query distribution is modified without his knowing it. We have shown that if this operation is done appropriately the data storage achieved according to S′ can be superior to that achieved if the data was stored according to the distribution S. The module which achieves this transformation is called a DCT filter.

In this paper we have considered the theory of DCT filters in its mathematical generality. It is shown that a DCT filter can be represented as stochastic Mealy automaton. Various DCT filters have been catalogued and, in particular, a particular filter F*, which expediently transforms the original distribution, has been proposed. F* thus accentuates the information contained in Ω.

We have also studied the problem of cascading DCT filters. The strongest result in this regard states that even an infinite sequence of expediently polarizing filters may not polarize absolutely.

The paper also contains extensive computational and simulation results which justify the theoretical results which have been presented. We are currently investigating how these principles can be utilized to enhance the data retrieval characteristics of a real-life local area network with a single file server.

ACKNOWLEDGEMENTS

We are very grateful to Dr Valiveti who proofread the original manuscript for us, provided us with many helpful comments and helped us in the preparation of the final manuscript. Remark (ii) mentioned in Section 4 is due to him. We are also grateful to Professors Reichstein and Dehne who proofread sections of the manuscript. This work was partially supported by the National Sciences and Engineering Research Council (NSERC) of Canada. The work of D.T.H.Ng. was supported by an Ontario graduate scholarship. A preliminary version of this paper was presented at the *1992 ORSA-CSTS Conf. on Computer Science and Operations Research: New Developments in Their Interfaces, Williamsburg*, VA, January 1992.

REFERENCES

Allen, B. and Munro, I. (1978) Self-organizing binary search trees. *JACM*, **25**, 526–535.

Arnow, D. M. and Tenebaum, A. M. (1982) An Investigation of the Move-Ahead-k Rules, Congressus Numerantium. In *Proc 13th Southeastern Conf. on Combinatorics, Graph Theory and Computing*, pp. 47–65, Florida.

Bayer, P. J. (1975) Improved bounds on the costs of optimal and balanced binary search trees. *MAC Tech. Memo-69*.

Bitner, J. R. (1979) Heuristics that dynamically organize data structures. *SIAM J. Comput.*, **8**, 82–110.

Burville, P. J. and Kingman, J. F. C. (1973) On a model for storage and search. *J. Appl. Prob.* **10**, 697–701.

Cheetham, R. P., Oommen, B. J. and Ng, D. T. H. (1988) On using conditional rotation operations to adaptively structure binary search trees. *Proc. 1988 Int. Conf. on Database Theory*, pp. 161–175, Bruges, Belgium.

Gonnet, G. H., Munro, J. I. and Suwanda, H. (1981) Exegesis of self-organizing linear search. *SIAM J. Comput.* **10**, 613–623.

Hammer, M., and Niamir, B. (1979) A heuristic approach to attribute partitioning. *Proc. ACM SIGMOD Conf.* pp. 93–101.

Hardy, G. H., Littlewood, J. E., and Polya, G. (1983) *Inequalities*. Cambridge University Press, Cambridge.

Hendricks, W. J. (1976) An account of self-organizing systems. *SIAM J. Comput.*, **5**, 715–723.

Hester, J. H. and Hirschberg, D. S. (1985) Self-organizing linear search. *ACM Comput. Surv.* **17**, 295–311.

Kan, Y. C., and Ross, S. M. (1980) Optimal list order under partial memory contraints. *J. App. Prob.* **17**, 1004–1015.

Karlin, S. and Taylor, H. M., *A First Course in Stochastic Processes*. Academic Press, New York.

Knuth, D. E. (1973) *The Art of Computer Programming Vol. 3: Sorting and Searching*. Addision-Wesley, Reading, MA.

Lakshmivarahan, S. (1981) *Learning Algorithms Theory and Applications*, Springer-Verlag, New York.

Ma, D. C. Y. (1986) *Object Partitioning by using Learning Automata*. MCS Thesis, School of Computer Science, Carleton University, Ottawa, Canada.

McCabe, J., (1965) On serial files with relocatable records. *Operations Res.* **12**, 609–618.

Mehlhorn, K. (1975) Nearly optimal binary search trees. *Acta Informatica*, **5**, 287–295.

Narendra, K. S. and Thathachar, M. A. L. (1989) *Learning Automata*. Prentice-Hall, Englewood Cliffs, NJ.

Oommen, B. J. and Hansen E. R. (1987) List organizing strategies using stochastic move-to-front and stochastic move-to-rear operations. *SIAM J. Comput.*, **16**, 705–716.

Oommen, B. J., Hansen, E. R. and Munro, J. I. (1990) Deterministic optimal and expedient move-to-rear list organizing strategies. *Theor. Comp. Sci.* **74**, 183–197.

Oommen, B. J. and Ma, D. C. Y. (1988) Deterministic learning automata solutions to the equi-partitioning problem. *IEEE Trans. Comp.* **37**, 2–14.

Oommen, B. J., and Ma, D. C. Y. (1992) Stochastic Automata solutions to the object partitioning problem. *Comp. J.* **35**, A105–A120.

Rivest, R. L. (1976) On self organizing sequential search heuristics. *Commun. ACM*, **19**, 63–67.

Sleator, D. D. and Tarjan, R. E. (1985) Self-adjusting binary search trees. *JACM*, **32**, 652–686.

Schkolnick, M. (1977) A clustering algorithm for hierarchical structures. *ACM Trans. Database Syst.* **2**, 27–44.

Tenenbaum, A. M. and Nemes, R. M. (1982) Two spectra of self-organizing sequential search algorithms. *SIAM J. Comput.* **11**, 557–566.

Van Rijsbergen, C. J. (1977) A theoretical basis for the use of co-occurrence data in information retrieval. *J. Documentation*, 106–119.

Walker, W. A. and Gotlieb, C. C. (1972) A top-down algorithm for constructing nearly optimal lexicographical trees. *Graph Theory and Computing*. Academic Press, New York.

Yu, C. T. and Salton, G. (1976) Precision weighting—an effective automatic indexing method. *J. ACM*, **23**, 76–78.

Yu, C. T., Suen, C. M., Lam, K. and Siu, M. K. (1988) Adaptive record clustering. *ACM Trans. Database Syst.* 180–204.

Yu, C. T., Siu, M. K. Lam, K., and Tai, F. (1981) Adaptive clustering schemes: general framework. *Proc. IEEE COMPSAC Conf.* 81–89.

Ross, S. M. (1981) *Introduction to Probability Models*. Academic Press, New York.