Performance Analysis of Cache Memories

GURURAJ S. RAO

Stanford University, Stanford, California

ABSTRACT Using the Independent Reference assumption to model program behavior, the performance of different buffer organizations (Fully Associative, Direct Mapping, Set Associative, and Sector) are analyzed. (1) The expressions for their fault rate are derived To show more explicitly the dependence of the fault rate on the factors that affect it, distribution-free upper bounds on fault rates are computed for the Direct Mapping, Set Associative, and Sector buffers The use of such bounds is illustrated in the case of the Direct Mapping buffer (2) The performance of the buffers for FIFO and Random Replacement are shown to be identical (3) It is possible to restructure programs to take advantage of the basic organization of the buffers The effect of such restructuring is quantified for the Direct Mapping buffer It is shown that the performance of the Direct Mapping buffer under near-optimal restructuring is comparable to the performance of the Fully Associative buffer Further, the effect of this restructuring is shown to be potentially stronger than that of buffer replacement algorithms

KEY WORDS AND PHRASES buffer, cache, statistical analysis, performance analysis, program behavior models, paging, page replacement algorithms, program restructuring, fault rate, distribution-free bounds

CR CATEGORIES 4 35, 5 42, 6 34, 8 3

1. *Introduction*

In this paper we analyze the performance of various buffer (cache) organizations in a two-level demand paged memory system. These organizations are shown in Figures 1–4 and described more fully in [9]. Even though the analysis is in the context of two-level memory hierarchies with buffers, the analysis is applicable to any two-level memory systems with the mapping constraints described here. Since the previous work in the analysis of two-level paged memories is used to analyze memory systems with buffers in this paper, the same terminology is retained here [17, 12, 2]. For uniformity of reference, we assume that both the primary (buffer) level and the secondary (backing store) level are divided into equal sized units called pages in the backing store and page frames in the buffer. Associated with each page frame is a map that carries the identity of the page in that page frame. These buffers can be classified according to their mapping constraint.

In the Fully Associative buffer, any page in the backing store can be in any page frame. When a request for a page is presented to the buffer, all the map entries are compared in parallel (associatively) with the request to determine if the request is present in the buffer. If not, it is termed a page fault and the requested information is brought from the next level. In the Direct Mapping buffer, page i can be only in page frame (i mod m) if we have m page frames in the buffer (Figure 2). This buffer has the advantage of a trivial replacement rule. Of all the pages that map onto a page frame only one can actually be in

This research was supported by a fellowship from the Ministry of Education, Government of India, and by the Joint Service Electronics Project Contract N00014-75-C-0601 Computer time was provided by the Energy Research and Development Administration under Contract E(043)515

Author's present address IBM T J Watson Research Center, P.O Box 218, Yorktown Heights, NY 10598

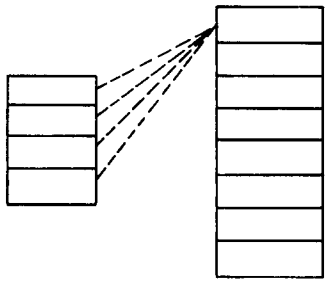

FIG 1 Fully Associative buffer

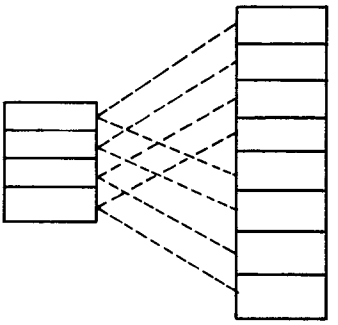

FIG 2 Direct Mapping buffer

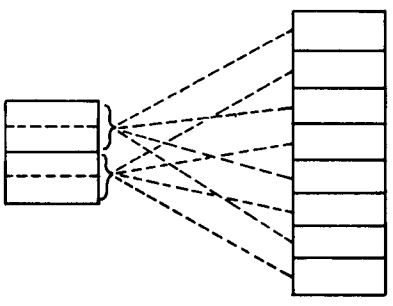

FIG 3 Set Associative buffer

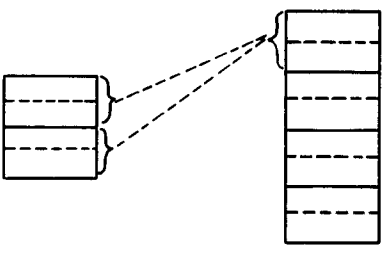

FIG 4 Sector buffer

the buffer at a time and hence if a page caused a fault, we would simply determine the page frame this page maps onto and replace the page in that page frame. This avoids the overhead of record keeping associated with a replacement rule In the Set Associative buffer, the buffer is divided into L sets with $s = m/L$ page frames/set. A page i in the backing store can be in any page frame belonging to the set (i mod L) (Figure 3). The cost of the associative search in a Fully Associative buffer depends on the number of entries to be simultaneously searched. The Set Associative buffer tries to cut this cost down and yet provide a performance close to that of the Fully Associative buffer. In the Sector buffer, the secondary store is divided into a number of sectors each composed of a number of blocks. The requests are for blocks and if a request is made for a block not in the buffer (i e on a fault), the sector to which this block belongs is brought into the buffer with the following constraints: A sector from backing store can be in any sector in the buffer, but the mapping of blocks in a sector is congruent. Also only the block that caused the fault is brought into the buffer, and the remaining blocks of this sector are marked invalid. This buffer tries to reduce the cost of the map. We now need only one map for each sector in the buffer with a bit (validity bit) for each block of a sector in the buffer.

Examples of computer systems using the buffers described above include the IBM System/360 Model 85, which has a Sector buffer organization, and the IBM System/370 Models 158 and 168, which use a Set Associative mapping with $s = 2$ and 4, respectively [20] The instruction buffer in the IBM System/370 model 158 uses Direct Mapping.

The performance of these buffers depends on (i) the organization of the buffer, (ii) the replacement rule used, and (iii) program behavior The purpose of this study is to analytically understand the interaction among these factors.

PERFORMANCE AND PROGRAM BEHAVIOR. From the point of view of the memory system, the number of faults caused in the buffer is to be minimized and hence we will use the limiting page fault rate (the ratio of the number of faults and the number of requests)

as our performance indicator. Using this indicator it is possible to estimate the delay seen by the processor because of the requested information not being available in the buffer [23]. An example of such an estimate, given a_i, the access time of level i, and the fault rate f, is $a_i(1-f) + a_i f$.

If the pages in the backing store are denoted by $X_1, X_2, \ldots, X_n$, then the string of references $r_1, r_2, \ldots, r_t$, where r_t is the page referenced at time t, completely characterizes the behavior of the program. Since such a characterization is analytically intractable, we need a model of program behavior. Several such models are discussed in the literature [7, 3, 22]. For the following reasons the Independent Reference Model (IRM) is used here:

(i) It is analytically tractable.
(ii) It gives a good indication of relative performances.
(iii) It predicts page fault rates reasonably well.

A recent study [3] has shown how the parameters of the IRM can be calculated to provide significant predictive power.

The following is a brief explanation of how this can be done. To distinguish this model from the conventional IRM, the new model is called the A_0-IRM. Modeling a system S (in this case the page reference string) can be viewed as obtaining a set of measurements on the system from which the parameters M of the model are computed, so that certain properties P of the system are captured in the model. If the analysis of the model yields properties Q (different from P) of the system that are close to the actual (observed) properties of the system, the model has predictive power. The property P captured in the A_0-IRM model is the requirement of good models that the fault rates of the model and the actual programs being modeled under optimal page replacement algorithms be very close to each other. This, coupled with property (ii) above, builds enough structure into the model for successful mimicry of the system.

It is known that A_0 and MIN are the optimal page replacement procedures for the IRM and for actual programs [7], respectively. It is also possible to express the fault rate of A_0 in terms of the parameters of the IRM, the page reference probabilities [7]. The procedure then to capture the above-mentioned property in the model is to "invert" the expression for the fault rate of A_0 so that the parameters of the IRM are expressed as a function of A_0 fault rates for a number of different memory sizes, and to use in these expressions MIN fault rates measured on actual programs being modeled instead of A_0 fault rates. For more details reference should be made to the paper by Baskett and Rafii [3]. The paper describes the empirical validation of the model by predicting other properties of page reference strings (LRU (Least Recently Used) and FIFO (First-In-First-Out) fault rates, working set sizes, etc.) and verifying these predictions. It also shows that this model adequately captures program referencing characteristics at cache levels. When the parameters are calculated in this fashion the IRM is said to have an "empirical" distribution.

Since we have chosen the page fault rate as our performance criterion, we can ignore factors that do not influence it. Thus it is not necessary to distinguish between the read and write natures of references. Any store policy not affecting page fault rate may be assumed.

In the rest of this paper we deal with three phases of the analysis of these buffers. Section 2 shows how to determine performance as a function of the buffer organization, the replacement rule, and the program behavior. Section 3 shows that the performance of all the buffers under FIFO and RR (Random Replacement) is identical. It also shows that the Direct Mapping buffer can be as capable as the Fully Associative LRU buffer Section 4 shows how distribution-free results can be found for the buffers and illustrates their use in the case of the Direct Mapping buffer. It also shows that the effect of buffer replacement algorithms on performance is secondary compared to the effect of restructuring based on buffer organization.

2. *Calculation of the Fault Rates for the Buffers*

BACKGROUND AND NOTATION. Assume that there are m page frames in the cache

and that the backing store has n (logical) pages. (The notation is slightly different for the Sector buffer and will be presented separately.) Denote the fault rate for a buffer with organization o, using replacement rule f by $F_f(o)$. The names Direct Mapping, Set Associative, Fully Associative, and Sector will be shortened to DM, SA, FA, and S, respectively.

Let the pages in the backing store be $X_1, \ldots, X_n$. Let the reference string be denoted by $r_1, \ldots, r_t, \ldots$. Let $[p_1, \ldots, p_n]$ be the distribution of the page reference probabilities, i.e.

$$\Pr[r_t = X_i] = p_i, \quad 1 \leq i \leq n, \quad \text{for all } t > 0.$$

The formal representation of replacement rules is covered in the literature (for example, [17]) and is not of much use here. Informally, any replacement algorithm f has to keep the contents of the buffer ordered in an "ordered list," which characterizes the state of the buffer at any time.

Let $S_t = (s_1, s_2, \ldots, s_m)$ denote this state at any time t, the indices referring to the position in the ordered list. Specification of the next state S_{t+1} for all possible s_t and r_t, the page referenced at time t, completely characterizes the replacement rule f. For example, for FIFO the ordered list is a queue with the head at s_1 and the tail at s_m, so that the page that has been in the cache longest is s_1. Hence, if there is a page fault on a reference to r_t, the new buffer state is $S_{t+1} = (s_2, s_3, \ldots, s_m, r_t)$.

Under the Independent Reference assumption, it has been shown [17, 7] that for the Fully Associative buffer, for the replacement rules considered here (LRU, FIFO, Random, A_0), the state sequence $\{s_t\}$ is a homogeneous Markov chain and that for $2 \leq m \leq n$, there exists a unique, equilibrium distribution for these states.

Let Q be the state space of this Markov chain, S a state belonging to Q, $P_f(S)$ the steady state probability of finding the chain in state S, and $p(S)$ the probability of a page fault in state S for replacement rule f.

Since $P_f(S)$ is the relative frequency of occurrence of state S in equilibrium, the page fault contribution by state S in steady state is $p(S)*P_f(S)$ and the limiting page fault rate for a given replacement algorithm f is given by

$$F_f(\mathrm{FA}) = \sum_{S \in Q} P_f(S)\ p(S). \tag{2.1}$$

King [17] has shown how to compute $P_f(S)$ for any state in the state space Q, for LRU, FIFO, and A_0. We will make use of this later.

But the method detailed above for computing the fault rate is not suitable for use in computing the fault rates of the other types of buffers. For this purpose, we will use the following method.

Franaszek and Wagner [12] have shown that, when $\{s_t\}$ is irreducible and aperiodic, the limiting fault rate can be written as

$$F_f = \sum_{i=1}^{n} p_i(1 - p_i(f)), \tag{2.2}$$

where $p_i(f) = \lim_{t \to \infty} \Pr[X_i \in S_t]$, $1 \leq i \leq n$, is a function only of the replacement rule f and $[p_1, \ldots, p_n]$ and is not dependent on the initial state S_0.

Equation (2.2) can be written as

$$F_f = \sum_{i=1}^{n} p_i q_i, \tag{2.3}$$

where q_i is the probability of not finding the page X_i in the cache in equilibrium.

This condition is satisfied for the algorithms considered here for the Fully Associative buffer when the number of pages with nonzero probability of reference is not less than $2m$ [1].

It is not difficult to show [21] that, when the distribution has nonzero probabilities, $\{s_t\}$ is irreducible and aperiodic. Hence eq. (2.3) holds for all these buffers. In the remainder of this section we show how the fault rates for the other buffers can be computed using

King's results where appropriate. The technique used is to group the pages in the backing store into disjoint subsets according to the mapping constraint and then use eq. (2.3).

DIRECT MAPPING BUFFER. Let G_i denote the set (group) of pages in the backing store that can be in page frame i of the buffer. Let $k = n/m$ be the cardinality of each of these sets. Refer to the pages in G_i by $X_1(i), X_2(i), \ldots, X_k(i)$ with probabilities of reference $p_1(i), \ldots, p_k(i)$, respectively Without loss of generality we can assume $p_1(i) \geq p_2(i) \geq \geq p_k(i)$, $i = 1, 2, \ldots, m$.

Let $D_i = \sum_{j=1}^{k} p_j(i)$, $i = 1, 2, \ldots, m$. Consider the two-state Markov chain with the state $S(1)$ being the state with page $X_j(i)$ in the buffer, while $S(0)$ is the state with page $X_j(i)$ missing from the buffer (Figure 5). When in state $S(1)$, a reference to any other page in G_i produces a fault causing removal of $X_j(i)$. Thus the transition probability from $S(1)$ to $S(0)$ is $D_i - p_j(i)$. Similarly, the transition probability from state $S(0)$ to $S(1)$ is $p_j(i)$.

Solution of this chain yields the equilibrium probability of state $S(0)$ (which is the probability that page $X_j(i)$ is absent from the buffer in steady state) to be

$$q_j(i) = (1 - p_j(i)/D_i). \tag{2.4}$$

Since the Direct Mapping buffer does not have a replacement rule, we can write eq. (2.3) as

$$F(\mathrm{DM}) = \sum_{i=1}^{n} p_i q_i,$$

which can be rewritten as

$$F(\mathrm{DM}) = \sum_{i=1}^{m} \sum_{j=1}^{k} p_j(i) q_j(i), \tag{2.5}$$

where $q_j(i) = \Pr[X_j(i)$ is not in the buffer in equilibrium].

Substituting for $q_j(i)$ from eq. (2.4) in eq. (2.5), we finally obtain

$$F(\mathrm{DM}) = \sum_{i=1}^{m} \left(D_i^2 - \sum_{j=1}^{k} p_j^2(i) \right) \Big/ D_i \tag{2.6}$$

as the expression for the limiting page fault rate of the Direct Mapping buffer.

SET ASSOCIATIVE BUFFER. Let the buffer have s page frames in each set and $L = m/s$ sets. Let G_i again denote the group of pages in the backing store that can be in set i. Let k be the cardinality of G_i. The page frames in set i and the pages in G_i form a Fully Associative cache-backing store combination C_i, the C_i being independent of each other This observation is central to the analysis of the Set Associative buffer.

Refer to the pages in G_i by $X_1(i), X_2(i), \ldots, X_k(i)$ with probabilities of reference $p_1(i), p_2(i), \ldots, p_k(i)$, respectively. Again without loss of generality, we can assume that $p_1(i) \geq p_2(i) \geq \cdots \geq p_k(i)$, $i = 1, 2, \ldots, L$.

Let $D_i = \sum_{j=1}^{k} p_j(i)$, $i = 1, 2, \ldots, L$. If we normalize $p_1(i), p_2(i), \ldots, p_k(i)$ by D_i to get $p_1^*(i), p_2^*(i), \ldots, p_k^*(i)$, then we can use King's formula to compute the fault rate for the

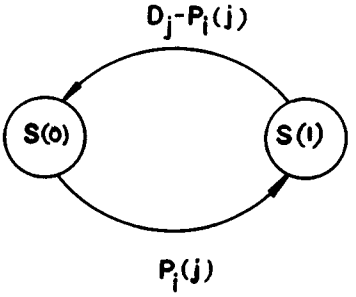

FIG 5 Markov chain for the Direct Mapping buffer

combination C_i for replacement rule f and distribution $[p_1^*(i), p_2^*(i), \ldots, p_k^*(i)]$. Denote this fault rate by $F_f(i)$.

Now we can rewrite eq. (2.3) as follows:

$$F_f(\mathrm{SA}) = \sum_{i=1}^{L} \sum_{j=1}^{k} p_j(i) q_j(i), \tag{2.7}$$

where $q_j(i)$ is the probability that page $X_j(i)$ is missing from the cache in equilibrium. The probability $q_j(i)$ is also the probability that page $X_j(i)$ is not in $S_t(i)$ as t tends to infinity. Equation (2.7) can now be written as

$$F_f(\mathrm{SA}) = \sum_{i=1}^{L} \sum_{j=1}^{k} D_i p_j^*(i) q_j(i). \tag{2.8}$$

Since $\sum_{j=1}^{k} p_j^*(i) q_j(i)$ is, from eq. (2.3), the fault rate $F_f(i)$ of C_i, we can write eq. (2.8) as

$$f_f(\mathrm{SA}) = \sum_{i=1}^{L} D_i F_f(i). \tag{2.9}$$

Thus to compute the fault rate for the Set Associative buffer, we would compute $F_f(i)$, the fault rate contribution by combination C_i, using King's results, weight it by D_i, and sum the product over all the combinations C_i. This result also shows that the Set Associative buffer is composed of Fully Associative buffers.

It is easy to verify that eq (2.9) reduces to the corresponding eq. (2.3) for the Fully Associative buffer when we let $s = m$.

SECTOR BUFFER. If we ignore the structure of sectors and treat them as units of transfer of information between the buffer and the backing store, there would be no difference between a Fully Associative buffer and a Sector buffer. The resulting organization is called the Equivalent Fully Associative buffer of the Sector buffer.

Suppose we have N blocks in the backing store organized into n sectors $X_1, X_2, \ldots, X_n$, with b blocks/sector, $b = N/n$. Denote the blocks in sector X_i by $Y_1(i), Y_2(i), \ldots, Y_b(i)$. We assume that the reference sequence $r_1, \ldots, r_t, \ldots$ consists of references to blocks and that $\Pr[r_t = Y_i(j)] = p_i^+(j)$. Let the buffer have m sectors with b blocks in each sector.

$$F_f(\mathrm{S}) = \sum_{i=1}^{n} \sum_{j=1}^{b} p_j^+(i) q_j(i), \tag{2.10}$$

where $q_j(i)$ is the probability of $X_j(i)$ being absent from the buffer in equilibrium.

The probability of reference to a sector X_i is given by

$$p_i = \sum_{j=1}^{b} p_j^+(i) \tag{2.11}$$

The event "block $X_j(i)$ is not in the buffer" can be seen to be the union of two disjoint events: (i) "The sector X_i is not in the buffer," and (ii) "the sector X_i is in the buffer and block $X_j(i)$ is not in the buffer." Denote the probabilities of these events by q_i and $q_j^+(i)$, respectively. So

$$q_j(i) = q_i + q_j^+(i). \tag{2.12}$$

Substituting eq. (2.12) in eq. (2.10) and using eq. (2.11), we obtain

$$F_f(\mathrm{S}) = \sum_{i=1}^{n} p_i q_i + \sum_{i=1}^{n} \sum_{j=1}^{b} p_j^+(i) q_j^+(i) \tag{2.13}$$

We can compute the Pr[the sector X_i is in the buffer and block $X_j(i)$ is not], i.e. $q_j^+(i)$, as follows:

$$q_j^+(i) = \Pr[\text{sector } X_i \text{ in buffer}]\ \Pr[X_j(i) \text{ not in buffer} | X_i \text{ in buffer}], \tag{2.14}$$

$$\Pr[X_t \text{ in buffer}] = 1 - q_t, \quad \Pr[X_j(i) \text{ not in buffer} | X_t \text{ in buffer}] = 1 - p_j^+(i)/p_t.$$

Using these in eq. (2.14), we obtain

$$q_j^+(i) = (1 - q_t)(1 - p_j^+(i)/p_t). \tag{2.15}$$

Equations (2.15) and (2.13) yield the fault rate for the Sector buffer as

$$F_f(\mathrm{S}) = \sum_{t=1}^{n} p_t q_t + \sum_{t=1}^{n} \sum_{j=1}^{b} p_j^+(i)(1 - q_t)(1 - p_j^+(i)/p_t). \tag{2.16}$$

By recognizing the first term as the fault rate of the Equivalent Fully Associative buffer (EFA) under replacement rule f, we can write eq. (2 16) as

$$F_f(\mathrm{S}) = F_f(\mathrm{EFA}) + \sum_{t=1}^{n} \sum_{j=1}^{b} p_j^+(i)(1 - q_t)(1 - p_j^+(i)/p_t). \tag{2 17}$$

Equation (2.16) relates the fault rate of the Sector buffer and the fault rate of the Equivalent Fully Associative buffer. We already know how to compute F_f(EFA) using King's results. To compute q_t for all the sectors, we resort to King's formulas for the equilibrium distribution for the states of the Equivalent Fully Associative buffer. By summing up the probabilities for the states not containing X_t, we obtain q_t.

3. *A Few General Comparisons*

We make two types of general comparisons here: (i) the behavior of two replacement algorithms across the whole range of the buffers, and (ii) the behavior of two specific types of buffers.

(i) The first general comparison concerns the behavior of the buffers under FIFO and RR: It is possible to show that for the IRM, the Fully Associative buffer has identical performances (in steady state) for these two rules [13]. Next we show that this result holds for all the other types of buffers. A brief intuitive explanation of this result for the Fully Associative buffer follows.

To show that this result holds for the Set Associative buffer, we make use of the result established earlier showing that it consists of independent Fully Associative buffers and backing store combinations C_t. In eq. (2.9) D_t, the sum of the reference probabilities for pages mapping on to set t, is independent of the replacement rule, while Gelenbe's result shows that the combination C_t has the same fault rate for RR and FIFO, i.e $F_{\mathrm{RR}}(i) = F_{\mathrm{FIFO}}(i)$. Thus $F_{\mathrm{FIFO}}(\mathrm{SA}) = F_{\mathrm{RR}}(\mathrm{SA})$.

To show that the result can be extended to the Sector buffer, we first note that it is possible to show that [13] $P_{\mathrm{FIFO}}(\mathrm{S}) = P_{\mathrm{RR}}(\mathrm{S})$ From Gelenbe's proof for the Fully Associative buffer, the Equivalent Fully Associative buffer has the same performance under RR and FIFO, i.e. $F_{\mathrm{FIFO}}(\mathrm{EFA}) = F_{\mathrm{RR}}(\mathrm{EFA})$ in eq. (2.17).

The probability of not finding sector X_t in the equilibrium state of the Equivalent Fully Associative buffer q_t is obtained by adding the equilibrium probabilities of all the states (of the Equivalent Fully Associative buffer) not containing X_t. Since each of these states has the same equilibrium probability under RR and FIFO, q_t is the same for RR and FIFO for a given sector X_t. Thus $F_{\mathrm{FIFO}}(\mathrm{S}) = F_{\mathrm{RR}}(\mathrm{S})$.

This result holds for the Direct Mapping buffer also because its fault rate is insensitive to the two algorithms. Thus all the buffers have the same fault rate for FIFO and RR.

Every time a replacement rule is used to replace a page, it is implicitly trying to assess the future demands of the program. Such an assessment is possible because programs exhibit a behavior called "locality" [7, 22, 8, 25, 19]. Intuitively we would expect a replacement rule that makes use of the knowledge of program behavior to choose a page for replacement, to be superior to one that does not. The only information about the locality of a program present in the IRM is in the distribution of page reference probabilities. So we would expect that a good replacement rule would appropriately treat a program's pages nonuniformly depending on their reference probabilities. LRU is an

example of such a replacement rule. For the Fully Associative buffer, it keeps the state information in a stack, with the top of the stack containing the most recently used page and the bottom containing the least recently referenced page When a page is referenced, it is brought to the top of the stack and therefore gets a "new lease on its life in the buffer." For the Independent Reference model, a good algorithm is one that tries to keep the most frequently used pages as long as possible in the buffer because that would tend to minimize the total number of faults. This is done by LRU because a page with a high probability of reference tends to be referenced more often when in the buffer and gets pulled to the top of the stack, thereby prolonging its stay in the buffer.

On the other hand, FIFO does not use any information about the page when making a decision to replace the page: FIFO uses a queue to maintain the state of the buffer with the page at the head of the queue to be replaced on a fault, while the page brought in on a fault is put in the tail position. At the end of m page faults (m is the size of the buffer), the page in the tail position would be at the head of the queue and would be replaced at the next fault irrespective of how often the page was referenced during the stay in the buffer.

This leads us to expect that FIFO performs as well as RR, because not using any information about the pages in the buffer for the selection of a page for replacement is equivalent to replacing a page at random. This equality in turn leads us to suspect that the changes in performance brought about by buffer replacement algorithms are small. Later it will be shown that this is indeed so

(ii) Capabilities of the Direct Mapping buffer. The Direct Mapping buffer is a simple and inexpensively organized buffer, and it does not have the overhead of the record keeping associated with a replacement rule. A previous simulation study [5] has shown that simple organizations using Direct Mapping buffers can be cost effective on minicomputers. The results derived here lend analytical support to that viewpoint.

Let $G_i = (X_1(i), X_2(i), \ldots, X_k(i))$ be the set of pages that can be in page frame i Since a reference to any page in G_i currently not in the buffer causes a fault, it is not difficult to see that the relative magnitudes of the probabilities of reference to these pages is important in determining the fault rate Thus we can see that the performance is dependent on the arrangement (mapping) of pages in the backing store. A way to quantify this arrangement is considered in Section 4 Here we are interested in showing that the Direct Mapping buffer, by taking advantage of this sensitivity, can perform as well as the Fully Associative LRU buffer. This is significant because the Fully Associative buffer is considered to have a good performance while the Direct Mapping buffer has no cost for page replacement.

First we prove the following

THEOREM 3.1 *For any distribution* $[p_1, p_2, \ldots, p_n]$, *there exist arrangements of pages in the backing store of a Direct Mapping buffer, for which* $F(DM) \leq 2B$, *where* $p_1 \geq p_2 \geq \geq p_n$ *and* $B = \sum_{i=m+1}^{n} p_i$

PROOF. Let $G_i = (X_1(i), X_2(i), \ldots, X_k(i))$, $i = 1, 2, \ldots, m$, be the set of pages which can be in page frame i. Let page $X_j(i)$ have probability of reference $p_j(i)$ for $j = 1, 2, \ldots, k$, and $i = 1, 2, \ldots, m$. Without loss of generality, we can assume that

$$p_1(i) \geq p_2(i) \geq \quad \geq p_k(i), \quad i = 1, 2, \ldots, m$$

Consider an incompletely specified arrangement for which $p_1(i) = p_i$ for $i = 1, 2, \ldots, m$, i.e. a mapping which maps the m most probable pages onto different page frames.

Let $E_i = \sum_{j=2}^{k} p_j(i)$ for $i = 1, 2, \ldots, m$. Then in the notation of Section 2,

$$D_i = p_i + E_i, \tag{3.1}$$

and from eq. (2.6), we have

$$F(\mathrm{DM}) \leq \sum_{i=1}^{m} \frac{(p_1(i) + E_i)^2 - p_1^2(i)}{p_1(i) + E_i} \tag{3.2}$$

$$= \sum_{i=1}^{m} \frac{2p_1(i)E_i + E_i^2}{p_1(i) + E_i}, \tag{3.3}$$

so that

$$F(\mathrm{DM}) - \sum_{i=1}^{m} E_i \leq \sum_{i=1}^{m} \frac{p_1(i)E_i}{p_1(i) + E_i}$$

$$\leq \sum_{i=1}^{m} E_i. \tag{3.4}$$

Since $\sum_{i=1}^{m} E_i = B$, we get the desired result. Q.E.D.

We will call the arrangements referred to above "near-optimal" arrangements.

Theorem 3.1 can be used in two ways to argue for the capabilities of the near-optimal Direct Mapping buffer:

(a) For the Fully Associative LRU cache, it can be shown that [12]

$$F_{\mathrm{LRU}}(\mathrm{FA}) \leq B\left[1 + \frac{m(1 - B)}{1 + (m - 1)B}\right], \tag{3.5}$$

with B as defined before.

There exist distributions (for example, $p_1 = p_2 = \quad = p_m$ and B small) for which the above LRU bound is very close to the actual LRU fault rate. That is, there exist distributions for which the LRU fault rate is very close to $B(1 + m)$. Comparing this with the upper bound on the fault rate for the near-optimal Direct Mapping buffer of 2B (which holds for any distribution), we can conclude that there are distributions and buffer sizes for which the Direct Mapping buffer will perform better than the Fully Associative LRU cache.

(b) The second argument concerns the "effectiveness" of an organization and its replacement rule. If the probabilities $[p_1, p_2, \ldots, p_n]$ are known in advance (again assuming that the pages are ranked in the decreasing order of their probabilities of reference), then the theoretically optimal procedure is to place the m most probable pages in the buffer always and service references to pages not in the buffer directly from the backing store (Since in practice all the requests are to be serviced from the cache, this is only a theoretical procedure.) This has a limiting fault rate of $F_0 = B$.

The worst performance of an organization relative to this optimal procedure is an indication of the "effectiveness" of the organization. From Theorem 3.1 and the inequality in (3.5), we see that the worst behavior of the near-optimal Direct Mapping buffer relative to the optimal procedure is bound by a constant (for any distribution and cache size), while for the Fully Associative LRU cache it is dependent on the size of the buffer.

From these two arguments, we can conclude the following:

There are distributions for which the performance of the near-optimal Direct Mapping buffer is comparable to that of the Fully Associative LRU cache.

Figures 6 and 7 show the performances of these two organizations for two distributions of the page reference probabilities. The mapping used for the Direct Mapping buffer is (for Figure 6) as follows:

1	2	3	m
$m + 1$	$m + 2$	$m + 3$	$2m$
$2m + 1$	$2m + 2$	$2m + 3$	$3m$

The numbers in column i indicate the indices of the pages which map onto page frame i. For Figure 7 it is

1	2	3	.	m
$2m$	$2m - 1$	$2m - 2$		$m + 1$
$2m + 1$	$2m + 2$	$2m + 3$		$3m$

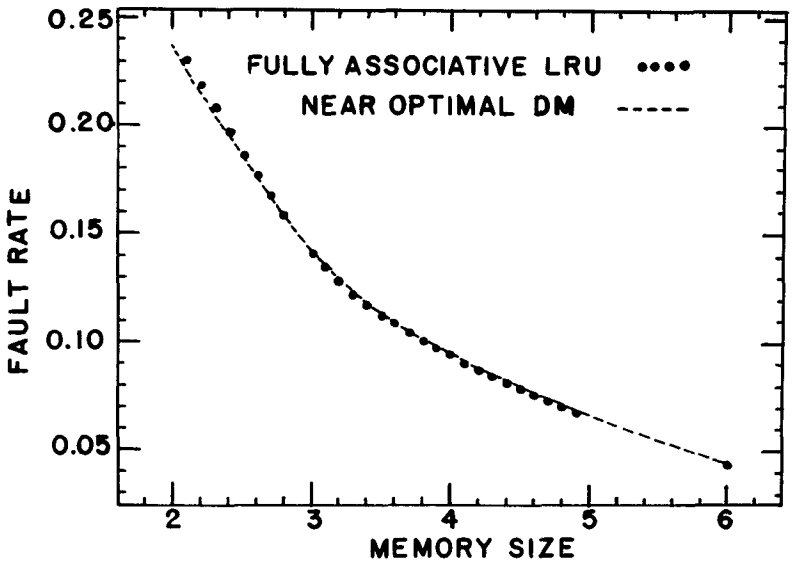

FIG 6 Near optimal Direct Mapping buffer versus Fully Associative buffer (empirical distribution)

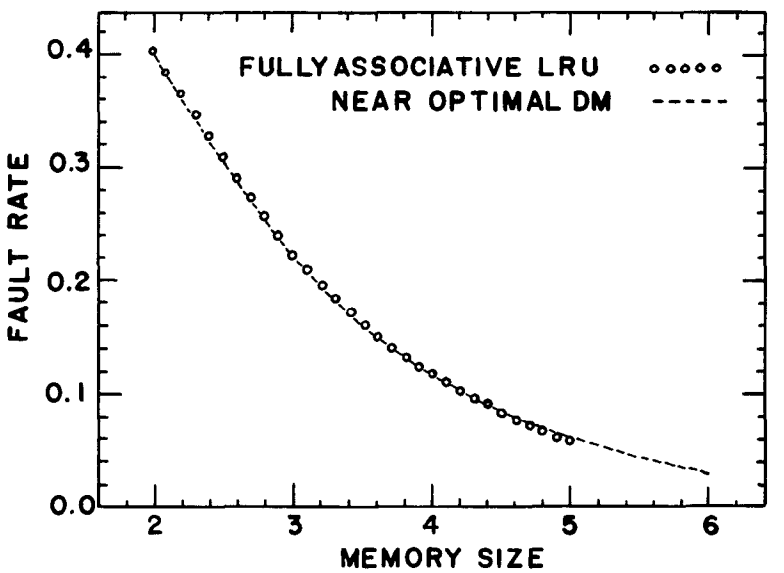

FIG 7 Near optimal Direct Mapping buffer versus Fully Associative buffer (geometric distribution)

4. *Distribution-Free Bounds on Fault Rate and Their Use*

IDEA OF A DISTRIBUTION-FREE RESULT. As indicated before, the purpose of an analysis such as this is to understand the interaction between the performance of a buffer and the factors which affect it. These parameters are: the size of the buffer, the type of mapping, the program behavior (i.e. the distribution of the page reference probabilities), and the replacement rule. The results in Section 2 showed how the limiting fault rate can be computed for a given type of mapping, buffer size, and probability distribution. But the dependence of the performance on a given parameter is not explicit enough. For example, the effect of changing the size of the buffer, while keeping the other parameters fixed, cannot be assessed immediately without computing the fault rates for both the sizes.

In addition to knowing the fault rates for the given parameters, often we would like to know the size of the buffer needed to satisfy a given performance level for a "given program behavior." Again the expressions for the fault rates do not permit us to compute this conveniently because of the implicit nature of the way the effect of these parameters is reflected in those expressions.

To explicitly see the effect of these parameters, a distribution-free result replaces the individual behavior of infrequently referenced pages by their collective behavior. Since the contribution of these pages to the fault rate is not significant, the resulting loss in accuracy is slight. The quantity B defined in Section 3 is an example of a characterization of such collective behavior. Given such collective behavior, we obtain upper bounds on the fault rate that are distribution-free in the other pages. Inequality (3.5) is an example of such a bound.

Later in this section we present two examples that illustrate how we can exploit the explicit interaction among the parameters. Besides this use, distribution-free upper bounds provide the following uses:

1. They give an idea of the worst performance of the combination of the organization and the replacement rule.

2. They are very easily computed. For the Fully Associative LRU buffer with n pages in the backing store and m page frames in the cache, the complexity of the expression for the fault rate is of the order of $\binom{n}{m}m!$ (this is the size of the state space). It is possible to reduce this complexity somewhat [6, 18], but still even for fairly small values of m and n, this computation is very expensive, while the distribution-free results can be computed on a hand calculator

DISTRIBUTION-FREE BOUNDS FOR THE BUFFERS. First we find the distribution-free upper bound for the Set Associative buffer and then derive the bound for the Direct Mapping buffer from that.

The characterization B of the collective behavior of the "tail" pages mentioned above is not suitable for the Set Associative buffer because it does not reflect the arrangement of pages in the backing store, while the fault rate depends on it (B is useful in representing the behavior of the tail pages of the Fully Associative buffer because the precise mapping of the pages onto page frames does not affect the latter's performance.) So we need a way of quantifying the effect of arranging pages in the backing store of the Set Associative buffer. To do this we proceed as follows.

Let $X_1(i), X_2(i), \ldots, X_k(i)$ be the pages in G_i (the group of pages which can be in the ith set in the buffer). Again, without loss of generality, we can assume that the pages in G_i are numbered in decreasing order of their probability of reference so that $p_1(i) \geq p_2(i) \geq \ldots \geq p_k(i)$, $i = 1, 2, \ldots, L$. Now define the "tail" probability E_i of G_i as

$$E_i = \sum_{j=s+1}^{k} p_j(i), \quad i = 1, 2, \ldots, L, \tag{4.1}$$

and the tail probability t of the distribution as

$$t = \sum_{i=1}^{L} E_i. \tag{4.2}$$

Then Theorem 4.1 gives the distribution-free upper bound on the fault rate for the Set Associative buffer using LRU.

THEOREM 4.1. *$F_{LRU}(SA) \leq st(1 - t)/(1 + (s - 1)t) + t$ for any distribution of page reference probabilities and any arrangement of pages in the backing store.*

PROOF. As shown in Section 2, we can write the expression for the fault rate as

$$F_{\mathrm{LRU}}(\mathrm{SA}) = \sum_{i=1}^{L} \sum_{j=1}^{k} p_j(i) q_j(i), \tag{4.3}$$

where $q_j(i)$ is the probability of not finding $X_j(i)$ in the buffer in the steady state. Next we

describe how to compute an upper bound on $q_j(\iota)$ and then show how the resulting expression can be maximized.

The method used in computing an upper bound on $q_j(\iota)$ is similar to the one used by Franaszek and Wagner [12] in arriving at the distribution-free upper bound on the fault rate of the Fully Associative LRU cache (inequality 3.5) The necessary condition for $X_j(\iota)$ to be absent from the buffer is that s distinct page references have occurred in C_ι since the last reference to $X_j(i)$. Since set ι has s pages including $X_j(i)$ this means that a page $X_r(i)$, where $r > s$, has been referenced since the last reference to page $X_j(i)$. This in turn means that in the steady state LRU stack for C_ι a page $X_r(i)$, $r > s$, is above $X_j(i)$.

The probability of this event can be found as follows [24]:

$$\begin{aligned} b(r, i) &= \Pr[\text{page } X_r(i) \text{ is ahead of page } X_j(\iota) \text{ in stack } i] \\ &= (p_r(i)/D_\iota)/(p_r(\iota)/D_\iota + p_j(\iota)/D_\iota) \\ &= p_r(\iota)/(p_r(i) + p_j(i)). \end{aligned} \tag{4.4}$$

From this we get

$$\Pr[\text{a page } X_r(i),\ r > s, \text{ is ahead of } X_j(\iota) \text{ in the LRU stack for combination } C_\iota] = E_\iota/(p_j(\iota) + E_\iota). \tag{4.5}$$

Thus

$$q_j(\iota) \le E_\iota/(p_j(\iota) + E_\iota). \tag{4.6}$$

From (4.3) and (4.6), we can write

$$F_{\mathrm{LRU}}(\mathrm{SA}) \le \sum_{\iota=1}^{L} \left[\sum_{j=1}^{s} p_j(\iota) E_\iota/(p_j(i) + E_\iota) + E_\iota \right] \tag{4.7}$$

$$= \sum_{\iota=1}^{L} \sum_{j=1}^{s} p_j(\iota) E_\iota/(p_j(i) + E_\iota) + t. \tag{4.8}$$

Now we maximize the right-hand side of (4.8) by writing:

$$\text{Maximize } F(p_1(1), \ldots, p_s(1), \ldots, p_1(L), \ldots, p_s(L)) = \sum_{\iota=1}^{L} \sum_{j=1}^{s} p_j(i) E_\iota/(p_j(i) + E_\iota) \tag{4.9}$$

such that $p_j(i) > 0$, $j = 1, 2, \ldots, s$ and $i = 1, 2, \ldots, L$, and

$$\sum_{\iota=1}^{L} \sum_{j=1}^{s} p_j(\iota) = 1 - t \tag{4.10}$$

We can write this as a nonlinear optimization problem.

$$\text{Maximize } F(p_1(1), \ldots, p_s(1), \ldots, p_1(L), \ldots, p_s(L)) = \sum_{\iota=1}^{L} \sum_{j=1}^{s} p_j(i) E_\iota/(p_j(i) + E_\iota)$$

such that $p_j(i) > 0$, $j = 1, 2, \ldots, s$ and $i = 1, 2, \ldots, L$, and

$$g_1 = \sum_{\iota=1}^{L} \sum_{j=1}^{s} p_j(i) \le 1 - t = b_1, \qquad g_2 = -\sum_{\iota=1}^{L} \sum_{j=1}^{s} p_j(\iota) \le t - 1 = b_2.$$

It is easy to show that [21] the objective function in (4.9) is concave and the constraint functions (which (4.10) gives rise to) are convex, being linear functions. Thus, Kuhn-Tucker conditions [15] are necessary and sufficient conditions for optimality and they state that

$$(p_1(1), \ldots, p_s(1), \ldots, p_1(L), \ldots, p_s(L)) \text{ maximizes } F$$

if and only if we can find numbers u_1 and u_2 such that

(1) if $\bar{p}_\iota^*(j) > 0$, then

$$\partial F/\partial p_i(j) - \sum_{k=1}^{2} u_k(\partial g_k/\partial p_i(j)) = 0$$

at $p_i(j) = \bar{p}_i^*(j)$, for $i = 1, 2, \ldots, s$, $j = 1, 2, \ldots, L$;
(2) if $\bar{p}_i^*(j) = 0$, then

$$\partial F/\partial p_i(j) - \sum_{k=1}^{2} u_k(\partial g_k/\partial p_i(j)) \leq 0$$

at $p_i(j) = \bar{p}_i^*(j)$, for $i = 1, 2, \ldots, s$, $j = 1, 2, \ldots, L$;
(3) if $u_i > 0$ then

$$g_i(p_1(1), \ldots, p_s(1), \ldots, p_1(L), \ldots, p_s(L)) - b_i = 0$$

at $p_k(j) = \bar{p}_k^*(j)$, $k = 1, 2, \ldots, s$, $j = 1, 2, \ldots, L$, $i = 1, 2$;
(4) if $u_i = 0$, then

$$g_i(p_1(1), \ldots, p_s(1), \ldots, p_1(L), \ldots, p_s(L)) - b_i \leq 0$$

at $p_k(j) = \bar{p}_k^*(j)$, $k = 1, 2, \ldots, s$, $j = 1, 2, \ldots, L$, $i = 1, 2$,
(5) $\bar{p}_i^*(j) \geq 0$, $i = 1, 2, \ldots, s$, $j = 1, 2, \ldots, L$,
(6) $u_i \geq 0$, $i = 1, 2$.

These conditions are satisfied by choosing $u_1 = u^2$ and $u_2 = 0$. Condition (1) requires u to satisfy

$$E_i/(p_j(i) + E_i) = u \quad \text{for} \quad j = 1, 2, \ldots, s \quad \text{and any } i \tag{4.11}$$

This requires

$$p_1(i) = p_2(i) = \cdots = p_s(i) = p(i) \quad \text{(say)}, \quad i = 1, 2, \ldots, L. \tag{4.12}$$

Thus

$$E_i/(p(i) + E_i) = u \quad \text{for} \quad i = 1, 2, \ldots, L. \tag{4.13}$$

From (4.13), we obtain

$$u = \sum_{i=1}^{L} E_i \Big/ \left(\sum_{i=1}^{L} p(i) + \sum_{i=1}^{L} E_i \right) = t \Big/ \left(\sum_{i=1}^{L} p(i) + t \right) \tag{4.14}$$

But from (4.12) and (4.10),

$$\sum_{i=1}^{L} sp(i) = 1 - t \tag{4.15}$$

so that

$$\sum_{i=1}^{L} p(i) = (1 - t)/s. \tag{4.16}$$

Thus from (4.16) and (4.14),

$$u = st/[1 + (s - 1)t]. \tag{4.17}$$

Finally, we substitute eqs. (4.11), (4.17), and (4.12) in eq. (4.8) to obtain

$$F_{\mathrm{LRU}}(\mathrm{SA}) \leq \sum_{i=1}^{L} \sum_{j=1}^{s} stp(i)/[1 + (s - 1)t] + t \tag{4.18}$$

$$= \sum_{i=1}^{L} s^2 tp(i)/[1 + (s - 1)t] + t. \tag{4.19}$$

Using eq. (4.15) in eq. (4.19), we obtain

$$F_{\mathrm{LRU}}(\mathrm{SA}) \leq st(1 - t)/[1 + (s - 1)t] + t, \tag{4.20}$$

which completes the proof. Q E D.

We have already seen that the Set Associative buffer is a general organization and that by setting the size of a set to one page frame, we get the Direct Mapping buffer, and setting $s = m$ yields the Fully Associative buffer. This is reflected in the above analysis. Setting $s = m$ in eq. (4.20) yields the distribution-free bound in (3.5) for the Fully Associative LRU cache, because when $s = m$, we have $t = B$.

By setting $s = 1$ in eq. (4.20), we obtain

THEOREM 4.2. *For the Direct Mapping buffer, $F(DM) \leq 2t - t^2$ for any distribution of the page reference probability distribution and any arrangement of pages in the backing store*

For a near-optimal arrangement, $p_1(i) = p_i$ for $i = 1, 2, \ldots, m$, and hence the tail probability

$$t = \sum_{i=1}^{m} E_i = \sum_{i=m+1}^{n} p_i = B.$$

So, for a near-optimal arrangement, Theorem 4.1 yields

$$F(\mathrm{DM}) \leq 2B - B^2 \tag{4.21}$$

This is a closer upper bound on F(DM) than the one given by Theorem 3.1.

Figures 8 and 9 illustrate the closeness of these upper bounds to the fault rates given by the expression in Section 2. These figures also show the sizable effect the arrangement can have on the fault rate. In both, the increase in performance gained by increasing the buffer size from 2 to 3 is more than offset by the drop in performance due to a change from a near-optimal arrangement to a nonoptimal arrangement

DISTRIBUTION-FREE RESULTS FOR THE SECTOR BUFFER. Next we obtain an upper bound on the difference between the performances of the Sector buffer and its Equivalent Fully Associative buffer

From eq. (2.17), we have

$$F_f(\mathrm{S}) = F_f(\mathrm{EFA}) + \sum_{i=1}^{n} (1 - q_i) \sum_{j=1}^{b} p_j^+(i)(1 - p_j^+(i)/p_i).$$

By calculating the second derivative of $p_j^+(i)(1 - p_j^+(i)/p_i)$ with respect to $p_j^+(i)$, we can

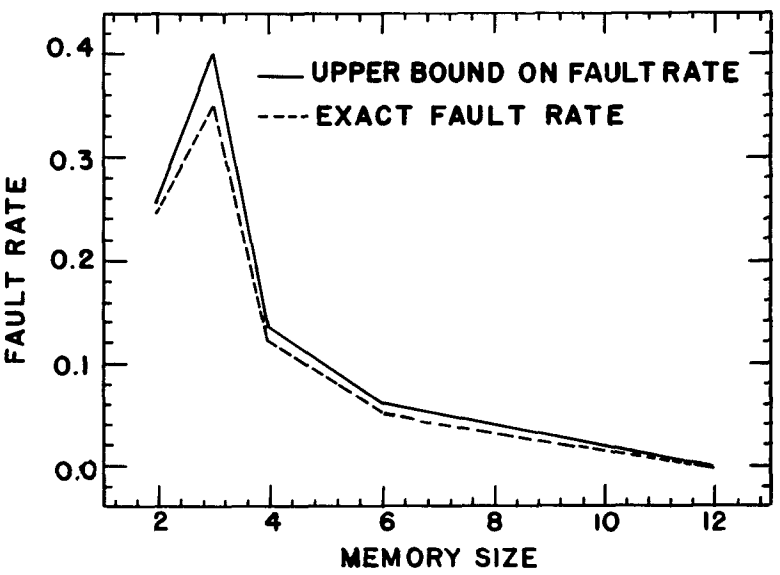

FIG 8 Upper bound on fault rate and actual fault rate for the Direct Mapping buffer (empirical distribution)

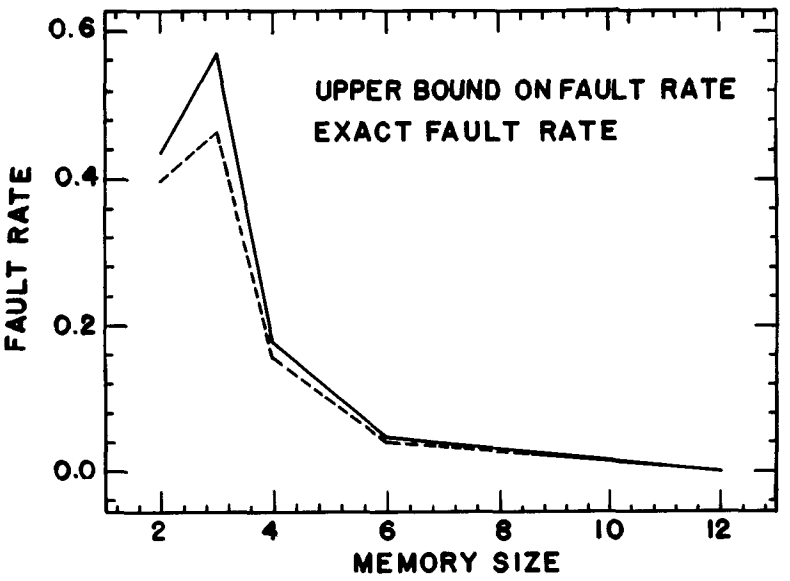

FIG 9 Upper bound on fault rate and actual fault rate for the Direct Mapping buffer (geometric distribution)

show that it is a concave function, so we maximize the right-hand side of the above equation by letting $p_1^+(i) = \quad = p_b^+(i) = p_i/b$, $i = 1, 2, \ldots, n$. This yields

$$F_f(\mathrm{S}) \leq F_f(\mathrm{EFA}) + \sum_{i=1}^{n} (1 - q_i)p_i(1 - 1/b).$$

This equation can be written as

$$F_f(\mathrm{S}) \leq 1 - (1/b)[1 - F_f(\mathrm{EFA})],$$

which can be used to obtain distribution-free upper bounds for the Sector buffer.

USE OF THESE BOUNDS FOR THE DIRECT MAPPING BUFFER. We have seen that the distribution-free upper bounds for the Direct Mapping buffer are very close to the actual fault rates for "practical" types of distributions (i.e geometric and empirical). Further, these bounds (which we can look upon here as approximations to actual fault rates) reflect the effect of the size of the buffer more explicitly than the expression for the actual fault rate. This property can be used to solve the "inverse" problem of computing the size of the buffer to achieve a given hit ratio (1 − fault rate) H for a given distribution.

From Theorem 4.2, we have $F(\mathrm{DM}) \leq 2t - t^2$. This can be written as $H = 1 - F(\mathrm{DM}) \geq t^2 - 2t + 1 = (1 - t)^2$, so that $\sqrt{H} \geq 1 - t$ or

$$t \geq 1 - \sqrt{H}. \tag{4.22}$$

This inequality can be interpreted as follows. With reference to Figure 10, if the tail probability for a given mapping and reference probability distribution is t then the hit rate for the corresponding parameters exceeds H. Alternatively, for the hit ratio to exceed H, the tail probability need not be smaller than t, i.e. to guarantee a hit ratio H, the tail probability need never be smaller than $1 - \sqrt{H}$.

For a near-optimal arrangement, it is very easy to obtain a close upper bound on the size of the cache memory required to obtain a hit ratio H, using this argument. For a near-optimal arrangement, $t = B$, as described in Section 3. So, all we need to do is to arrange the pages in decreasing order according to their probability of reference and starting at the tail end, accumulate the probabilities until the sum exceeds $1 - \sqrt{H}$. The number of pages m_0, whose probabilities are not included in the sum, is the upper bound on the size of the buffer required. For any other specified arrangement, we can obtain the memory size

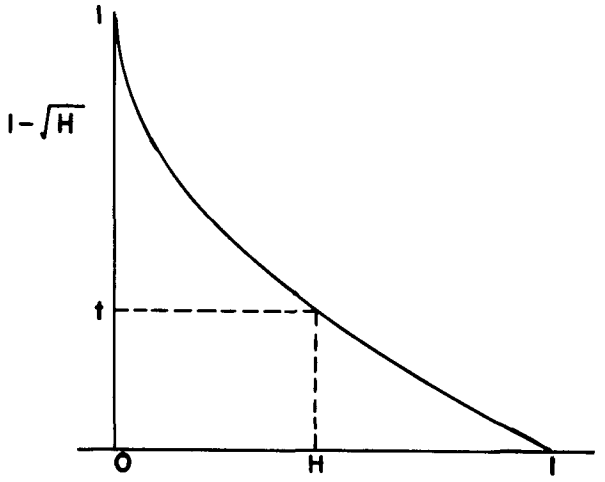

FIG 10 Illustration of $t \geq 1 - \sqrt{H}$

required m_a by first getting m_0 and then using this as the starting point of an iterative scheme which checks, for successively higher sizes of the buffer, whether the value of t exceeds $1 - \sqrt{H}$, for the specified arrangement.

RELATIVE EFFECTS OF RESTRUCTURING AND BUFFER REPLACEMENT ON PERFORMANCE. Next we wish to show that the effect of buffer replacement algorithms on performance is secondary when compared to the effect of the mapping on performance. We show this by comparing the maximum obtainable difference in performance between LRU—one of the best realizable algorithms—and RR—one of the worst realizable algorithms—with the maximum obtainable difference in performance for the best and worst mappings. Among the Fully Associative, Set Associative, and Direct Mapping buffers, the Fully Associative buffer is the most sensitive to the replacement algorithm (and the least sensitive to mapping) while the Direct Mapping buffer is the most sensitive to mapping (and the least sensitive to replacement algorithm). Thus we compare the difference in performance between LRU and RR on the Fully Associative buffer with that for the best and worst mapping on the Direct Mapping buffer.

Since the exact mappings for the Direct Mapping buffer that produce the lowest and highest fault rates are not known, we prove our point by using the distribution-free upper bound (which we again consider as an approximation to the fault rate) derived in Theorem 4.2 as follows The mapping that produces the smallest value for t is the near-optimal mapping as defined earlier. Correspondingly, we call the mapping that produces the highest value for t near-"worst." If the reference probabilities $p_1, p_2, \ldots, p_n$ are in decreasing order of magnitude, it is not difficult to see that the following mapping produces the largest value for t. Map $k = n/m$ pages starting at page X_j where $j = (i - 1)k + 1$ onto page-frame i.

Table I compares the difference in calculated, exact performance between LRU and RR for the Fully Associative buffer with that for near-optimal and near-worst mappings for the Direct Mapping buffer, for an empirical distribution It clearly shows that the potential of the mapping to cause a large change in performance is much greater than that due to buffer replacement. Table I also shows that the same conclusion would have been reached if we had considered the approximation to fault rate of the Direct Mapping buffer instead of the actual fault rate.

5. *Conclusions*

We have assumed a reasonable, tractable model of program behavior and tried to understand the relation between the factors affecting the performance of different buffer

TABLE I

(a) Calculated exact performance for the Fully Associative buffer for empirical distribution

m =	2	3	4	5	6
(i) RR	0 2808	0 1866	0 1315	0 0953	0 045
(ii) LRU	0 2408	0 1416	0.0931	0.0645	0 025

(b) Calculated exact performance for the Direct Mapping buffer for empirical distribution

m =	2	3	4	6
(i) Near-optimal	0 2348	0 1416	0 0941	0 0437
(ii) Near-worst	0.4558	0 4293	0 3962	0 3211

(c) The ratios of the change produced in (b) to the change produced in (a)

m =	2	3	4	6
Ratio [(b-ii) − (b-i)]/[(a-i) − (a-ii)]	4 4	5 8	7 4	53 1

(d) Calculated upper bound on performance for the Direct Mapping buffer for empirical distribution

m =	2	3	4	6
(i) Near-optimal	0 2621	0 1661	0 1120	0 05566
(ii) Near-worst	0 4933	0 4734	0 4459	0 3743

organizations. Toward this end we first showed how the fault rates for these buffers can be calculated and then showed how these expressions can be simplified in detail without losing much accuracy to obtain distribution-free upper bounds on their performance. We derived such bounds for the Direct Mapping buffer and the Set Associative LRU buffer. For the Sector buffer this bound was derived in terms of the upper bound on its Equivalent Fully Associative buffer's performance. We showed how these bounds for the Direct Mapping buffer can be used to compute a close upper bound on the amount of buffer memory needed to guarantee a given level of performance.

Further, we showed that buffer replacement has a secondary effect on performance compared to restructuring by first showing that FIFO and RR yield identical performances for all the buffers and then showing that the variation in performance between the near-best and near-worst mappings on the Direct Mapping buffer was much greater than that between the "worst" realizable and "best" realizable algorithms for the Fully Associative buffer.

These results are only as valid as the model is in capturing the programs' behavior. A feature of the Independent Reference model is that it predicts relative performance accurately [3]. The equivalence of the fault rates of the buffers for FIFO and RR, proved here, has been observed [4]. The result that predicts the near-optimal Direct Mapping buffer to be as capable as the Fully Associative LRU buffer shows that it is possible to restructure programs to take advantage of the hardware organization of the system. This is an extension of the notion introduced by Ferrari [11] to tailor restructuring algorithms to suit the memory management policy. This is in contrast to the method of restructuring to improve the locality of the program [16, 14, 10]. A few preliminary attempts to restructure programs to suit the hardware organization are reported in [21].

REFERENCES

1 ACEVEDO, M F A probabilistic study of two-level storage hierarchies M S Th , U of Texas, Austin, Tex , Dec 1972.

2 AVEN, O I , ET AL Some results on distribution-free analysis of paging algorithms *IEEE Trans Comptrs C-25*, 7 (July 1976), 737–745

3 BASKETT, F , AND RAFII, A The A0 inversion model of program paging behavior Tech Rep #STAN-CS-76-579, Dept Comptr Sci , Stanford U , Stanford, Calif . Oct 1976

4 BELADY, L A A study of replacement algorithms for virtual storage computers *IBM Syst J. 5*, 2 (1960), 78–101

5 BELL, J , CASASENT, D , AND BELL, C G An investigation of alternative cache organizations *IEEE Trans Comptrs. C-23*, 4 (April 1974), 346–351

6 BURVILLE, P J , AND KINGMAN, J F C On a model for storage and search *J Appl Probability 10* (1973), 697–701

7 COFFMAN, E G , AND DENNING, P J *Operating System Theory* Prentice-Hall, Englewood Cliffs, N J , 1973

8 COFFMAN, E G, AND RYAN, T A A study of storage partitioning using a mathematical model of locality *Comm ACM 15*, 3 (March 1972), 185–190
9 CONTI, C J Concepts for buffer storage Comptr Group News *2*, 8 (March 1969), 9–13
10 FERRARI, D Improving locality by critical working sets *Comm ACM 17*, 11 (Nov 1974), 614–620
11 FERRARI, D Improving program locality by strategy-oriented restructuring Information Processing, North-Holland Pub Co, Amsterdam, 1974, pp 266–270
12 FRANASZEK, P A, AND WAGNER, T J Some distribution-free aspects of paging algorithm performance. *J ACM 21*, 1 (Jan 1974), 31–39
13 GELENBE, E A unified approach to the evaluation of a class of replacement algorithms *IEEE Trans Comptrs C-22*, 6 (June 1973), 611–618
14 HATFIELD, D J, AND GERALD, J Program restructuring for virtual memory *IBM Syst J 10*, 3 (1971), 168–192
15 HILLIER, F S, AND LIEBERMAN, G J *Introduction to Operations Research* Holden-Day, San Francisco, 1972
16 JOHNSON, J W Program restructuring for virtual memory systems MAC TR-148, M I T, Cambridge, Mass, March 1975
17 KING, W F Analysis of paging algorithms Proc IFIP Congress, Ljublanjana, Yugoslavia, Aug 1971, 485–490
18 LENFANT, J The delay network model of program behaviour In *Computer Architecture and Networks*, E Gelenbe and R Mahl, Eds, North-Holland Pub Co, Amsterdam, 1974, pp 299–329
19 MADNICK, S E Storage hierarchy systems MAC TR-107, M I T, Cambridge, Mass, 1972
20 MADNICK, S E, AND DONOVAN, J J *Operating Systems* McGraw-Hill, New York, 1974
21 RAO, G S Performance analysis of cache memories Tech Rep No 110, SEL 76-019, Digital Systems Lab, Depts EE and Comptr Sci, Stanford U., Stanford, Calif, 1975
22 RAU, B R The stack working set A characterization of spatial locality Tech Rep. No 95, Digital Systems Lab, Stanford U, Stanford, Calif, July 1975
23 RAU, B R, AND ROSSMANN, G E Cache based computer systems A tutorial Tech Rep, Palyn Assocs, San Jose, Calif, March 1974
24 RIVEST, R L On self-organizing sequential search heuristics Res Rep 61, IRIA Laboria, Le Chesnay, France, March 1974
25 SHEMER, J E, AND SHIPPEY, G A Statistical analysis of paged and segmented computer systems *IEEE Trans Electron Comptrs EC-15*, 6 (Dec 1966), 855–863

RECEIVED JULY 1976, REVISED SEPTEMBER 1977