IBM Research

ANALYSIS OF PAGING ALGORITHMS

W. F. King, III

March 17, 1971

RC 3288

C. 1
ref

FOR REFERENCE USE ONLY

DO NOT REMOVE FROM ASDD LIBRARY

Yorktown Heights, New York

San Jose, California

Zurich, Switzerland

ANALYSIS OF PAGING ALGORITHMS *

W. F. King, III
IBM Thomas J. Watson Research Center
Yorktown Heights, New York

ABSTRACT: From a simple stochastic model of a program, and a finite state model of a paging algorithm, we derive a Markov chain model of a program executing in a paged computer system. In order to compare the relative merits of paging algorithms, we define the long run expected rate of page faults. Markov chain analysis yields an expression for the long run expected page fault rate as a function of the stationary probability distribution of the Markov chain. This result is applied to compute the long run expected page fault rate expressions for three well-known paging algorithms; least recently used, first-in first-out, and A_0. Those expressions are compared for several distributional assumptions.

RC 3288 (#15047)
March 17, 1971
Systems evaluation
& measurement

* This paper is the unabridged version of RC-3141, November 10, 1970.

I. INTRODUCTION

Dynamic storage allocation in computer operating systems which support virtual machines is implemented via the well known concept of *paging* [1]. In a typical system a program (instructions and data) is partitioned into equal sized *pages* of information items, e.g., words or bytes. In addition, the main memory is divided into equal sized repositories for pages called *page frames*. At any time some of the pages of a given program may reside in certain page frames. If an executing program is to reference an information item, then the page containing that item must be resident in main storage. Hence one function of the dynamic storage allocation mechanism is to decide which pages are currently to be assigned to the page frames of main storage. The strategy employed to implement this mechanism is called a *paging algorithm*.

Several experimental studies of paging algorithms have been published [2,3]. Unlike these, the approach here is to develop a mathematical model of a program executing in a paged computer system employing a given paging algorithm. Informally, we will describe the *execution* of a program in a paged computer system as a sequence of references to pages containing information (instructions and data) needed by the program. The paging algorithm determines which pages are to be located in the page frames of the main storage. We say that a *page fault* occurs if the program attempts to reference a page not in main storage. Then for this model we will compute and compare the long run expected rate of page faults for the well-known LRU (least recently used) and FIFO (first in - first out) algorithms and for the A_0 algorithm of [4,5].

Section 2 introduces the program model and the model of a paging algorithm used herein. Also in that section, the expressions for the long run expected page fault rate for each of the above listed algorithms are presented. Finally in section 2 several comparisons of the three algorithms are made. Derivations of these expressions are included in Sections 3 and 4.

II. MODELS, RESULTS, AND COMPARISONS

2.1. Models

In this subsection we will develop a model of a program executing in a paged computer system. This model is based on a simple stochastic model of a program and on a paging algorithm model used in recent papers [4,5].

Consider a program consisting of a set $N = \{1,2, \ldots, n\}$ of n pages which is to be executed in a system with m, $1 \le m \le n$, page frames of main memory. As the program executes, it generates a sequence $r_1, r_2, \ldots, r_k, \ldots$ of references to pages, where if $r_k = i$, then the k^{th} reference was to page $i \in N$. We call the sequence a *reference string*. We define a stochastic structure on the reference string as described below. We assume that the sequence $\{r_k, k = 1,2, \ldots\}$ is a sequence of independent, identically distributed random variables where $\Pr[r_k = i] = \beta_i$, $1 \le i \le n$, and $\sum_{i=1}^{n} \beta_i = 1$.

Admittedly, this is a very simple and perhaps not realistic model of a program, however it is possible to extend the results of this paper to the Markovian model of a program introduced in [4]. Now with this program model, we can define a paging algorithm as follows:

Definition 1.[†]

A paging algorithm is a 6-tuple $A = (S_m, Q, N, q_0, s_0, \delta)$ where

$S_m = \{s \mid s \subseteq N, |s| = m\}$ is a finite set of memory states,

Q is a finite set of control states,

N is a finite set of n pages, $m \leq n$,

$q_0 \in Q$ is the initial control state ,

$s_0 \in S_m$ is the initial memory state , and

δ: $S_m \times Q \times N \rightarrow S_m \times Q$.

A pair $(s,q) \in S_m \times Q$ is a *configuration* of A. We require that $i \in s'$ if $\delta(s,q,i) = (s',q')$. This implies that if a program attempts to reference page i, then that page is assigned to a page frame in main storage.

If a program generates a reference string $r_1, r_2, \ldots, r_k, \ldots$ then we say that the paging algorithm *processes* this string by generating a sequence of configurations

$$\{(s_0,q_0)\ (s_1,q_1)\ (s_2,q_2)\ \ldots\ (s_k,q_k)\ \ldots\}$$

such that

$$(s_k,q_k) = \delta(s_{k-1},q_{k-1},r_k), \quad k \geq 1.$$

If $s_k \neq s_{k-1}$ then a page fault occurs. A paging algorithm is said to be a *demand* paging algorithm if $\delta(s,q,i) = (s,q')$ whenever $i \in S$. All algorithms investigated herein are demand paging algorithms.

Informally, an element of the set S_m is a list of the contents of the m page frames allocated to a program. Note that we are assuming that the program begins execution with certain pages, s_0, already in main memory. A control state $q \in Q$

[†]This definition is similar to that of [4].

is an encoding of any information which we wish to use to decide which pages are to be kept in main storage. The mapping δ selects the next memory and control state based on the present page referenced and the present memory and control states.

In order to study the paging behavior of a program executing under a given paging algorithm, it is only necessary to observe the sequence of configurations of the paging algorithm. If successive configurations are (s_k, q_k) and (s_{k+1}, q_{k+1}) where $s_k \neq s_{k+1}$, a page fault has occurred.

Example 1.

As an example, the least recently used algorithm (LRU) for $n=3$, $m=2$ is described below:

$$S_2 = \{(12), (13), (23)\},\ Q = \{q_{12}, q_{21}, q_{13}, q_{31}, q_{23}, q_{32}\},\ N = \{1,2,3\},$$

δ	1	2	3
$(12)q_{12}$	$(12)q_{21}$	$(12)q_{12}$	$(23)q_{23}$
$(12)q_{21}$	$(12)q_{21}$	$(12)q_{12}$	$(13)q_{13}$
$(13)q_{13}$	$(13)q_{31}$	$(23)q_{32}$	$(13)q_{13}$
$(13)q_{31}$	$(13)q_{31}$	$(12)q_{12}$	$(13)q_{13}$
$(23)q_{23}$	$(13)q_{31}$	$(23)q_{32}$	$(23)q_{23}$
$(23)q_{32}$	$(12)q_{21}$	$(23)q_{32}$	$(23)q_{23}$

$q_0 = q_{12}$, $s_0 = (12)$

The table defines δ in the following way: Arguments s,q are row labels, the present page referenced is the column label, and pair at the row-column intersection is the next configuration. The control state subscripting explicitly specifies the least recently referenced page, i.e., it is the left-most subscript.

This paging algorithm model or variants thereof is widely applicable to known paging algorithms. The Denning, Chen, and Shedler A_0 algorithm [5] and the well-known LRU and FIFO algorithms will be described herein using this model. In [4], Belady's MIN algorithm [2] is described in terms of this model. A least frequently used algorithm can be modeled by letting Q be an infinite set. Of course not all algorithms can be modeled in this structure. For example, suppose that

$$\delta(s,q,r_k) = \begin{cases} s',\ q' & \text{if } r_k \text{ is prime} \\ s'',\ q'' & \text{otherwise.} \end{cases}$$

Since primes cannot be recognized with the finite state mechanism of the model, this algorithm could not be modeled. However, practical paging algorithms do not usually have this property.

2.2. Results

The figure of merit that we use to compare paging algorithms is defined in this subsection. Then we formally specify the A_0, LRU, and FIFO algorithms. Finally for several choices for the distribution of r_k, we will compare the above algorithms using the figure of merit defined.

Definition 2.

Let Y_k, $k = 0,1, \ldots$ be the k^{th} configuration of a paging algorithm which is processing a reference string. If $Y_k = (s,q)$ and $Y_{k-1} = (s',q')$ where $s \neq s'$, then we say that the *transition* from (s',q') into (s,q) is a *page fault transition*. Let $N_{s,q}(\ell)$ be the number of *page fault transitions* into configuration (s,q) given that a total of ℓ transitions have occurred and that $Y_0 = (s_0,q_0)$. We denote the expected value of $N_{s,q}(\ell)$ by $\gamma_{s,q}(\ell)$. Hence $\frac{\gamma_{s,q}(\ell)}{\ell}$ is the expected rate of page fault transitions into (s,q).

Definition 3.

The long run expected page fault rate of algorithm A is defined to be

$$F(A) = \sum_{S_m \times Q} \left[\lim_{\ell \to \infty} \frac{\gamma_{s,q}(\ell)}{\ell} \right] .^{\dagger\dagger}$$

Next we will formally specify each of the previously mentioned algorithms and exhibit the corresponding F(A).

Let us assume that $\beta_1 \geq \beta_2 \geq \ldots \geq \beta_n$. Informally the A_0 algorithm replaces that page in memory with the smallest probability β_i.

Definition 4

Formally $A_0 = (S_m, q, N, q, s_0, \delta)$ where there is only one control state, $s_0 = (1,2,\ldots m)$ and

$$\delta(s,q,i) = \begin{cases} (s,q) & \text{if } i \in s \\ ((1,2, \ldots, m-1, i), q) & \text{if } i \notin s \end{cases} .$$

Note that in fact only one page is actually moved in and out of the memory.

Theorem 1.†

$$F(A_0) = \sum_{j=m}^{n} \beta_j - \frac{\sum_{j=m}^{n} \beta_j^2}{\sum_{j=m}^{n} \beta_j}$$

Next we define the LRU algorithm.

† The proofs of the next three theorems are presented in section 4. Theorem 1 was proven in [4] by a different method.

†† For the algorithms investigated herein, this limit always exists.

Definition 5.

LRU = $(S_m, Q, N, s_0, q_0, \delta)$ where

$$Q = \{q = (j_1, j_2, \ldots, j_m) \mid j_k \in N, \ 1 \le k \le m, \text{ and if } j_k = j_\ell, \ 1 \le \ell \le m, \text{ then } k = \ell\}$$

$s_0 = (1,2, \ldots, m)^\dagger$, $q_0 = (1,2, \ldots, m)^\dagger$; and

$$\delta(s,q,i) = \begin{cases} (s,q'') & \text{if } i \in s \\ (s',q') & \text{if } i \notin s \end{cases}$$

where $q'' = (j_1, \ldots, j_{k-1}, j_{k+1}, \ldots, j_m, i)$ if $i = j_k$ and $q' = (j_2, \ldots, j_m, i)$ and $s' = s \cup \{i\} - \{j_1\}$

Theorem 2. For $2 \le m \le n$ ††

$$F(LRU) = \sum_{q \in Q} \left[\frac{D_1(q) \prod_{i=1}^{m} \beta_{j_i}}{\prod_{i=1}^{m-1} (D_i(q) + \beta_{j_m})} \right]$$

where $D_i(q) = 1 - \sum_{k=i}^{m} \beta_{j_k}$.

The FIFO algorithm is defined below:

Definition 6.

FIFO = $(S_m, Q, N, s_0, q_0, \delta)$ where

$$Q = \{q = (j_1, \ldots, j_m) \mid j_k \in N \text{ and if } j_k = j_\ell \text{ then } k = \ell\},$$

$s_0 = (1,2, \ldots, m)^\dagger$, $q_0 = (1,2, \ldots, m)^\dagger$ and

† In section III we show that the choice of the initial configuration is immaterial.

†† Clearly when m = 1, all demand algorithms have identical page fault rates; hence for m = 1, $F(A) = F(A_0) = 1 - \sum_{j=1}^{m} \beta_j^2$.

$$\delta(s,q,i) = \begin{cases} (s,q) & \text{if } i \in s \\ (s',q') & \text{if } i \notin s \end{cases}$$

where $s' = s \cup \{i\} - \{j_1\}$ and $q' = (j_2, \ldots, j_m, i)$.

Theorem 3. For $2 \le m \le n$

$$F(\text{FIFO}) = \frac{\sum_{q \in Q} [D_1(q) \prod_{i=1}^{m} \beta_{j_i}]}{\sum_{q \in Q} [\prod_{i=1}^{m} \beta_{j_i}]} \quad \text{where } D_1(q) = 1 - \sum_{i=1}^{m} \beta_{j_i}$$

2.3. Comparisons of A_0, LRU, and FIFO

Because of the complicated form of the expressions given in theorems 1, 2, and 3, we are not able to make comparisons of these algorithms independent of the particular choices of the probabilities β_i, $1 \le i \le n$. However we will present several numerical studies in which the algorithms are compared for fixed choices of the β_i. First we begin with a corollary for a uniform distribution of r_k, i.e., $\beta_i = \frac{1}{n}$, $1 \le i \le n$.

Corollary 1. If $\beta_i = \frac{1}{n}$, $1 \le i \le n$, $F(A_0) = F(\text{LRU}) = F(\text{FIFO}) = \frac{n-m}{n}$.

Proof:

1. $F(A_0) = \frac{1}{n}(n-m+1) - \frac{[\frac{1}{n}]^2 (n-m+1)}{\frac{1}{n}(n-m+1)}$

$$= \frac{1}{n}(n-m+1-1) = \frac{n-m}{n}.$$

2. For LRU $D_i(q) = 1 - (\frac{m-i+1}{n}) = \frac{n-m+i-1}{n}$

and $D_i(q) + \beta_{i_m} = \frac{n-m+i}{n}$. Also there are $\binom{n}{m} m!$ elements $q \in Q$. Hence:

$$F(LRU) = \frac{\binom{n}{m} m! \left(\frac{n-m}{n}\right) \left(\frac{1}{n}\right)^m}{\left(\frac{1}{n}\right)^{m-1} (n-m+1)\ (n-m+2)\ \dots\ (n-1)}$$

$$= \frac{n!\ \ (n-m)}{(n-m)!\ n^2\ (n-m+1)\ \dots\ (n-1)} = \frac{n-m}{n}\ .$$

3. $$F(FIFO) = \frac{\binom{n}{m} m! \left(\frac{n-m}{n}\right) \left(\frac{1}{n}\right)^m}{\binom{n}{m} m! \ \left(\frac{1}{n}\right)^m} = \frac{n-m}{n}\ .$$ Q.E.D.

In Figure 1, we plot $F(A)$ versus m for $n=9$ assuming r_k is arithmetically distributed (solid line), geometrically distributed (dashed line), and uniformly distributed (dotted line). By an arithmetic distribution (geometric distribution) we mean that $\beta_i = ic$ ($\beta_i = c^i$) where c is a constant chosen so that $\sum_{i=1}^{n} \beta_i = 1$. In all cases, observe that $F(FIFO) \geq F(LRU) \geq F(A_0)$. We conjecture that this ordering is valid independent of the distribution on r_k.

The remainder of this paper is concerned with the proofs of theorems 1, 2 and 3.

III. MARKOV CHAINS AND SUBSET CHANGING TRANSITIONS

In section 2.1, we observed that the expected page fault rate can be derived by computing the rate of occurrence of certain types of transitions in the sequence $\{Y_k, k=0,1, \dots\}$. It is easy to show that this sequence of configurations is a Markov chain, i.e., states of the chain are configurations of the paging algorithm. We will describe the page fault transitions in the context of this Markov chain.

<u>Theorem 4</u>.

$$\Pr[Y_k = (s,q) \mid Y_{k-1}, \dots, Y_0] = \Pr[Y_k = (s,q) \mid Y_{k-1}]$$

$$= \sum_{\{i \mid \delta(Y_{k-1}, i) = Y_k\}} \beta_i .$$

This simple Markov chain is the mathematical description of a program executing under a given paging algorithm. In order to study the paging behavior of a program, we must observe those transitions in the Markov chain between configurations (s_k, q_k) and (s_{k+1}, q_{k+1}) where $s_k \neq s_{k+1}$.

Definition 7.

Given an irreducible, finite state Markov chain $\{Y_k,\ k = 0,1, \ldots\}$ with transition matrix $P = (p_{ij})$ and state set $\{0,1, \ldots, v\}$, let $\Pi = (b_1, b_2, \ldots, b_\mu)$ be a *partition* of the state set into μ disjoint subsets b_i, $1 \le i \le \mu$, so that $\bigcup_{i=1}^{\mu} b_i = \{0,1, \ldots, v\}$. By $\Pi(i) = b_j$ we mean that state $i \in b_j$. In a particular realization of a Markov chain, a transition from state i to j is a *subset changing* (SC) *transition* if $\Pi(i) \neq \Pi(j)$.

If we wish to compute the expected page fault rate (using the Markov chain describing the execution of a program) the subsets of the relevant partition contain all the configurations (s,q) with identical memory states s. Hence an SC transition corresponds directly to a page fault. In the remainder of this section we will develop a theorem which gives the expected rate of occurrence of SC transitions in a Markov chain.

Definition 8.

Let y_{ij} be the number of transitions in passing from state i to the first occurrence of j entered via a SC transition. Let the expected value of y_{ij}, $E(y_{ij}) = m_{ij}$.

Lemma 1.

$$m_{ij} = \sum_{k \neq j} p_{ik}(1 + m_{kj}) + p_{ij} \quad \text{for} \quad \Pi(i) \neq \Pi(j)$$

$$m_{ij} = \sum_{k=0}^{v} p_{ik}(1 + m_{kj}) \quad \text{for} \quad \Pi(j) = \Pi(j) \quad \text{and} \quad i = j$$

Proof:

1. $$m_{ij} = \sum_{k=0}^{v} E[y_{ij} \mid Y_1 = k]\, p_{ik}$$

$$= \sum_{k \neq j} E[y_{ij} \mid Y_1 = k]\, p_{ik} + E[y_{ij} \mid Y_1 = j]\, p_{ij}$$

Now if $\Pi(i) \neq \Pi(j)$ then $E[y_{ij} \mid Y_1 = j] = 1$

and $E(y_{ij} \mid Y_1 = k) = 1 + m_{kj}$. Hence $m_{ij} = \sum_{k \neq j} p_{ik}(1 + m_{kj}) + p_{ij}$

2. If $\Pi(i) = \Pi(j)$ then $E[y_{ij} \mid Y_1 = j] = 1 + m_{jj}$ since the first transition $(Y_0 = i, Y_1 = j)$ is not an SC transition.

Hence $m_{ij} = \sum_{k=0}^{v} p_{ik}(1 + m_{kj})$ Q.E.D.

Lemma 2.

If the Markov chain is irreducible with limiting vector $\underline{\alpha} = (\alpha_0, \alpha_1, \ldots, \alpha_v)$ then

$$m_{jj} = \frac{1}{\alpha_j - \sum_{\{k \mid \Pi(k)=\Pi(j)\}} \alpha_k \, p_{kj}}$$

Remark: Observe that if every subset of the partition is a singleton and if $p_{jj} = 0$ for all j, then every transition is an SC transition. In this case m_{ij} reduces to the usual mean first passage time.

Proof:

1. From Lemma 1 if $\Pi(i) \neq \Pi(j)$

$$m_{ij} = \sum_{k \neq j} p_{ik}(1 + m_{kj}) + p_{ij} = 1 + \sum_{k \neq j} p_{ik} \, m_{kj}.$$

If $\Pi(i) = \Pi(j)$, $m_{ij} = 1 + \sum_{k \neq j} p_{ik} \, m_{kj} + p_{ij} \, m_{jj}$

and $m_{ii} = 1 + \sum_{k \neq i} p_{ik} \, m_{ki} + p_{ii} \, m_{ii}$.

2. We can assume that the matrices $M = (m_{ij})$ and $P = (p_{ij})$ are given in the following standard form:

$$M = \begin{array}{c} \\ b_1 \\ b_2 \\ \vdots \\ b_\mu \end{array} \!\!\begin{array}{c} \begin{array}{cccc} b_1 & b_2 & \cdots & b_\mu \end{array} \\ \left[\begin{array}{cccc} \; & \; & \; & \; \\ & & & \\ & & & \\ & & & \end{array} \right] \end{array} \qquad P = \begin{array}{c} \\ b_1 \\ b_2 \\ \vdots \\ b_\mu \end{array} \!\!\begin{array}{c} \begin{array}{cccc} b_1 & b_2 & \cdots & b_\mu \end{array} \\ \left[\begin{array}{cccc} \; & \; & \; & \; \\ & & & \\ & & & \\ & & & \end{array} \right] \end{array}$$

Let E be the $(1 + v) \times (1 + v)$ matrix of ones, M_{dg} is the matrix which is identical to M on the diagonal and zero elsewhere and

$$\hat{P} = \begin{array}{c} b_1 \\ b_2 \\ \vdots \\ b_\mu \end{array}\left[\begin{array}{c|c|c|c} \multicolumn{1}{c}{b_1} & \multicolumn{1}{c}{b_2} & \multicolumn{1}{c}{\cdots} & \multicolumn{1}{c}{b_\mu} \\ & 0 & \cdots & 0 \\ \hline 0 & & \cdots & 0 \\ \hline \vdots & \vdots & \vdots & \vdots \\ \hline 0 & 0 & 0 & \end{array}\right]$$

is identical to P on the diagonal blocks and zero elsewhere.

3. With those matrix definitions we can write

$$M = E + P(M - M_{dg}) + \hat{P}\, M_{dg}.$$

Now $\underline{\alpha}P = \underline{\alpha}$ hence $\underline{\alpha}M = \underline{\alpha}\, E + \underline{\alpha}\, P(M - M_{dg}) + \underline{\alpha}\hat{P}\, M_{dg}$ which implies that $\underline{\alpha}(I - \hat{P})M_{dg} = (1,1, \ldots, 1)$. Q.E.D.

Definition 9.

Let $N_{ij}(n)$ be the number of SC transitions into state j given that the initial state was i and exactly n transitions have occurred. Let $E[N_{ij}(n)] = \gamma_{ij}(n)$. Hence if

$$\lim_{n\to\infty} \frac{\gamma_{ij}(n)}{n}$$

exists for each $0 \le j \le v$, then the expected long run rate of SC transitions, given that the initial state was i, is given by

$$\sum_{j=0}^{v} \lim_{n\to\infty} \frac{\gamma_{ij}(n)}{n}.$$

Observe that definition 3 is just the above definition cast into the terminology of paging algorithms.

Lemma 3.

$$\gamma_{ii}(n) = \frac{n}{m_{ii}} + \frac{1}{m_{ii}} \sum_{k=0}^{v} p_{ik}^{(n)}\, m_{ki} - 1.$$

Proof:

1. Let $T = W_1 + W_2 + \ldots + W_{N_{ii}(n)+1}$ where W_ℓ, $1 \le \ell \le N_{ii}(n)+1$, is the number of transitions between the $\ell-1^{st}$ and ℓ^{th} SC transition into state i. Assume that the initial state is also i. Hence $E(T) = E(W_1) + \ldots + E(W_{N_{ii}(n)+1})$, and from Lemma 2, $E(W_\ell) = m_{ii}$.

2. Now for some integer j, $N_{ii}(n) \ge j-1$ if and only if

$$\sum_{\ell=1}^{j-1} W_\ell \le n.$$

Hence the event $\{N_{ii}(n)+1 \ge j\}$ is dependent only on W_1, $W_2, \ldots, W_{j-1}$ and Wald's theorem on cumulative sums [6] applies. Therefore $E(T) = E(W)\ E(N_{ii}(n)+1) = m_{ii}\ (\gamma_{ii}(n) + 1)$.

3. Observe that the n^{th} transition must occur *between* the $N_{ii}(n)$ and $N_{ii}(n)+1$ SC transition into state i. Hence $T = n + h$ where h is the number of transitions between the n^{th} and the $N_{ii}(n)+1$ transition. Let $p_{ik}^{(n)}$ be the probability of being in state k after n transitions given that the initial state was i. Hence

$$E(h) = \sum_{k=0}^{v} p_{ik}^{(n)}\ m_{ki}.$$

Hence we conclude that

$$E(T) = n + \sum_{k=0}^{v} p_{ik}^{(n)}\ m_{ki} = m_{ii}\ (\gamma_{ii}(n)+1). \qquad \text{Q.E.D.}$$

Lemma 4.

$$\gamma_{ij}(n) = \frac{1}{m_{jj}}\ [n - m_{ij} + \sum_{k=0}^{v} p_{ik}^{(n)}\ m_{kj}].$$

Proof:

1. Assume the initial state to be i. Let $T = W_1 + W_2 + \ldots + W_{N_{ij}(n)+2}$ where W_ℓ, $1 \le \ell \le N_{ij}(n)+2$, is the number of transitions between the $\ell-1^{st}$ and ℓ^{th} SC transition into state j. Hence,

$$E(T) = E[W_1 + \ldots + W_{N_{ij}(n)+2}] \quad \text{where} \quad E(W_1) = m_{ij}$$

$$\text{and} \quad E(W_\ell) = m_{jj}, \quad 2 \le \ell \le N_{ij}(n) + 2.$$

2. Now for some integer k, $N_{ij}(n) \ge k-2$ if and only if

$$\sum_{\ell=1}^{k-2} W_\ell \le n.$$

Hence the event $\{N_{ij}(n)+2 \ge k\}$ depends only on $W_1, W_2, \ldots, W_{k-2}$, and Wald's Theorem applies.† Therefore $E(T) = m_{ij} + m_{jj}\ [\gamma_{ij}(n)+1]$.

3. Observe that the n^{th} transition must occur between the $N_{ij}(n)$ and $N_{ij}(n)+1$ transition into state i. Hence $T = n + h + W_{N_{ij}(n)+2}$ where h is as defined in the proof of the previous lemma. Therefore as before

$$E(h) = \sum_{k=0}^{v} p_{ik}^{(n)}\, m_{kj}.$$

Equating the two expressions for $E(T)$ gives the stated result.

Q.E.D.

The next theorem is the main result of this section, i.e., the formula for the long run expected rate of SC transitions for a given partition defined on an irreducible finite state Markov chain.

Theorem 5.

Given an irreducible, finite state Markov chain and a partition Π, then

$$\sum_{j=0}^{v} \lim_{n\to\infty} \frac{\gamma_{ij}(n)}{n} = 1 - \sum_{j=0}^{v} \left[\sum_{\{k|\Pi(k) = \Pi(j)\}} \alpha_k\, p_{kj} \right]$$

† Actually, this is a trivial extension to Wald's Theorem.

Proof:

1. $$\lim_{n\to\infty} \frac{\gamma_{ij}(n)}{n} = \lim_{n\to\infty} \frac{\gamma_{jj}(n)}{n} = \frac{1}{m_{jj}} \quad \text{from lemmas 3 and 4}$$

since

$$\lim_{n\to\infty} \frac{p_{ik}^{(n)} m_{kj}}{n\, m_{jj}} = 0.$$

This follows since $\lim_{n\to\infty} p_{ik}^{(n)}$ exists for an irreducible Markov chain.

2. Now using lemma 2

$$\sum_{j=0}^{v} \lim_{n\to\infty} \frac{\gamma_{ij}(n)}{n} = \sum_{j=0}^{v} \frac{1}{m_{jj}} = \sum_{j=0}^{v} \left[\alpha_j - \sum_{\{k|\Pi(k) = \Pi(j)\}} \alpha_k\, p_{kj} \right]$$

$$= 1 - \sum_{j=0}^{v} \left[\sum_{\{k|\Pi(k) = \Pi(j)\}} \alpha_k\, p_{kj} \right]$$

Q.E.D.

It is interesting to note that the long run expected rate of SC transitions is independent of initial state. In the context of the execution of programs in a paging environment, this says that the choice of initial configuration of the paging algorithm has no effect on the page fault rate. Of course the explicit assumption is that the Markov chain corresponding to the algorithm is irreducible.

IV. DERIVATIONS OF F(A)

Theorem 5 will be applied to the paging algorithm models of A_0, LRU, and FIFO to prove theorems 1, 2, and 3. In each case we apply the results of the previous section by proceeding with the following steps.

1. From the definition of the paging algorithm and theorem 4 we compute the transition matrix P associated with that algorithm.

2. We must show that the Markov chain described by P is irreducible.

3. The next step is to solve the eigenvector problem $\underline{\alpha} = \underline{\alpha}P$.

4. Next we must define the partition Π which specifies the SC transitions of the Markov chain. Since we want an SC transition to be a page fault transition, we will always place all configurations (s,q) with identical s in the same subset of the partition.

5. Finally we substitute into Theorem 5 to find $F(A)$.

For A_0, the states of the Markov chain are configurations (s,q) as in definition 4. However, since there is only one control state, q, we can represent a configuration by giving only s. Also note that the first $m-1$ elements of every memory state are identical. Hence we can denote configuration s where $s = (1,2, \ldots, m-1, i)$ as i for $m \le i \le n$. Therefore $Pr[Y_k = s \mid Y_{k-1} = s']$ is denoted by p_{ij} where $s' = (1,2, \ldots, m-1, i)$ and $s = (1,2, \ldots, m-1, j)$.

From Theorem 4 $p_{ij} = \sum_{\{k \mid \delta(i,k) = j\}} \beta_k$. Now from the definition of δ, if $k \in \{1,2, \ldots, m-1, i\}$, $\delta(i,k) = i$; hence $p_{ii} = \beta_i + \sum_{k=1}^{m-1} \beta_k$. Also if $k \notin \{1,2, \ldots, m-1, i\}$, $\delta(i,k) = k$; hence $p_{ik} = \beta_k$. If we let

$$\sum_{k=1}^{m-1} \beta_k = B$$

then the generic form of the transition matrix P for A_0 is

$$P = \begin{bmatrix} B+\beta_m & \beta_{m+1} & \beta_{m+2} & \cdots & \beta_n \\ \beta_m & B+\beta_{m+1} & \beta_{m+2} & \cdots & \beta_n \\ \\ \beta_m & \beta_{m+1} & \cdots & & B+\beta_n \end{bmatrix}$$

Since $p_{ij} > 0$ for all i and j, the Markov chain is irreducible. Therefore the equation $\underline{\alpha} = \underline{\alpha}P$ has a unique solution.

Lemma 5.

For A_0 let $\underline{\alpha} = (\alpha_m, \alpha_{m+1}, \ldots, \alpha_n)$ satisfy the equations $\underline{\alpha}P = \underline{\alpha}$ and $\sum_{i=m}^{n} \alpha_i = 1$. Then

$$\alpha_i = \frac{\beta_i}{1 - B} \quad \text{where} \quad B = \sum_{k=1}^{m-1} \beta_k .$$

Proof: Using the equation $\underline{\alpha}P = \underline{\alpha}$ for $0 \le i \le n-m$, we have that

$$\alpha_{m+i} = \alpha_m \beta_{m+i} + \alpha_{m+1}\beta_{m+i} + \ldots + \alpha_{m+i-1}\beta_{m+i} + \alpha_{m+i}(B + \beta_{m+i}) + \ldots + \alpha_n \beta_{m+i}$$

$$\alpha_{m+i} = \beta_{m+i} \sum_{j=m}^{n} \alpha_j + \alpha_{m+i} B = \beta_{m+i} + \alpha_{m+i} B$$

Hence $\alpha_{m+i} = \frac{\beta_{m+i}}{1-B}$. Also we must show that for this solution $\sum_{i=m}^{n} \alpha_i = 1$. This follows trivially.

Q.E.D.

Proof of Theorem 1:

1. From Theorem 5 $F(A_0) = 1 - \sum_{j=m}^{n} \left[\sum_{\{k | \Pi(k) = \Pi(j)\}} \alpha_k \, p_{kj} \right]$. Since no two configurations have identical memory states s, the partition Π contains $n-m+1$ subsets each containing exactly one state. Hence

$$F(A_0) = 1 - \sum_{j=m}^{n} \alpha_j \, p_{jj} = 1 - \sum_{j=m}^{n} \frac{\beta_j}{1 - B} (B+\beta_j) = \sum_{j=m}^{n} \beta_j - \frac{\sum_{j=m}^{n} \beta_j^{\,2}}{\sum_{j=m}^{n} \beta_j}$$

Q.E.D.

For LRU we first note that the control state q explicitly defines the memory state s; hence configurations are specified by giving the control state only. Let q and q' be two control states. Then by $p_{q',q}$ we mean $\Pr[Y_k = q \mid Y_{k-1} = q']$. From Theorem 4 $p_{q',q} = \sum_{\{k|\delta(q',k) = q\}} \beta_k$. From definition 5 if $q' = (i_1, \ldots, i_m)$, $k \neq i_\ell$, $1 \le \ell \le m$, and $q = (i_2, \ldots, i_m, k)$ then $p_{q',q} = \beta_k$. If $k = i_\ell$, $1 \le \ell \le m$, and $q = (i_1, \ldots, i_{\ell-1}, i_{\ell+1}, \ldots, i_m, k)$ then $p_{q',q} = \beta_k$. All other $p_{q,q'} = 0$.

Next we must show that P is irreducible, i.e., for any q,q' there is an integer k such that $p^{(k)}_{q',q} > 0$. Let $q = (j_1, \ldots, j_m)$ and $q' = (i_1, \ldots, i_m)$. By the previous discussion there is a control state q_1 such that $p_{q',q_1} = \beta_{j_1}$ where

$$q_1 = \begin{cases} (i_2, \ldots, i_m, j_1) & \text{if } j_1 \neq i_\ell \quad 1 \le \ell \le m \\ (i_1, \ldots, i_{\ell-1}, i_{\ell+1}, \ldots, i_m, j_1) & \text{if } j_1 = i_\ell \end{cases}.$$

Either case will be denoted by $q_1 = (xx \ldots x\, j_1)$. Now repeating the above argument, there is a state $q_2 = (xx \ldots x\, j_1\, j_2)$ such that $p_{q_1,q_2} = \beta_{j_2}$. In fact it is clear that $p_{q_i,q_{i+1}} = \beta_{j_{i+1}}$ where $q_i = (xx \ldots x\, j_1 \ldots j_i)$ and $q_{i+1} = (xx \ldots x\, j_1 \ldots j_{i+1})$. Since $q_n = (j_1 \ldots j_n)$, it follows that $p^{(n)}_{q',q} > 0$; hence P is irreducible.

<u>Lemma 6</u>.

For LRU let $\alpha_q = (j_1 \ldots j_m)$ be a typical element of the vector $\underline{\alpha}$ which satisfies $\underline{\alpha}P = \underline{\alpha}$ and $\sum_{q \in Q} \alpha_q = 1$. Then for $m \ge 2$

$$\alpha_q = \frac{\prod_{i=1}^{m} \beta_{j_i}}{\prod_{i=2}^{m} D_i(q)} \quad \text{where} \quad D_i(q) = 1 - \sum_{k=i}^{m} \beta_{j_k} .$$

<u>Proof</u>:

1. First we will show by substitution that $\underline{\alpha}$ solves $\underline{\alpha}P = \underline{\alpha}$. From the definition of the transition matrix P, if $\underline{\alpha}P = \underline{\alpha}$ then

$$\alpha_{j_1 \dots j_m} = \beta_{j_m} \left[\sum_{j \neq j_i} \alpha_{j,j_1 \dots j_{m-1}} + \sum_{1 \le k \le m} \alpha_{j_1 \dots j_m\, j_k \dots j_{m-1}} \right] .$$

After substituting into the above equation, the expressions for $\alpha_{j,j_1 \dots j_m}$ and $\alpha_{j_1 \dots j_{k-1}\, j_m\, j_k\, \dots\, j_{m-1}}$ we get

$$\alpha_{j_1 \dots j_m} = \beta_{j_m} \left[\sum_{j \neq j_i} \left[\frac{\beta_j \prod_{i=1}^{m-1} \beta_{j_i}}{\prod_{i=1}^{m-1} (D_i(q) + \beta_{j_m})} \right] + \sum_{1 \le k \le m} \left[\frac{\prod_{i=1}^{m} \beta_{j_i}}{\prod_{i=2}^{k} D_i(q) \prod_{i=k}^{m-1} (D_i(q) + \beta_{j_m})} \right] \right]$$

After tedious but trivial algebra, the right hand side of the above expression reduces to

$$\frac{\prod_{i=1}^{m} \beta_{j_i}}{\prod_{i=2}^{m} D_i(q)} .$$

2. Now we must show that $\sum_{q \in Q} \alpha_q = 1$. The proof is by induction on m. Let $m=2$, hence

$$\alpha_{j_1, j_2} = \frac{\beta_{j_1} \beta_{j_2}}{1 - \beta_{j_2}} .$$

Now

$$\sum_{j_1 \neq j_2} \alpha_{j_1,j_2} = \sum_{j_1 \neq j_2} \frac{\beta_{j_1} \beta_{j_2}}{1 - \beta_{j_2}} = \sum_{j_2} \left[\frac{\beta_{j_2}}{1 - \beta_{j_2}} \sum_{j_1 \neq j_2} \beta_{j_1} \right]$$

$$= \sum_{j_2} \beta_{j_2} = 1.$$

Hence the result is true for $m=2$. Now assume the result to be true for $m \leq m'$. Consider

$$q' = \sum_{\substack{j_1 \cdots j_{m'} \\ j \neq j_i}} \alpha_{j, j_1, \ldots, j_{m'}} = \sum \frac{\beta_j \prod_{j=1}^{m'} \beta_{j_i}}{\prod_{i=1}^{m'} D_i(q')} = \sum_{q'} \frac{\prod_{i=1}^{m'} \beta_{j_i}}{\prod_{i=1}^{m'} D_i(q')} \sum_{j \neq j_i} \beta_j$$

$$= \sum_{q'} \frac{\prod_{i=1}^{m'} \beta_{j_i}}{\prod_{i=2}^{m'} D_i(q')} = \sum_{q'} \alpha_{q'}.$$

But by the inductive hypothesis

$$\sum_{q'} \alpha_{q'} = 1.$$ Q.E.D.

<u>Proof of Theorem 2</u>:

1. All configurations (s,q) with identical memory states s are in a common subset of the partition Π. From Theorem 5 and the definition of a matrix P for LRU

$$F(\text{LRU}) = 1 - \sum_{\substack{q \in Q \\ q=(j_1 \cdots j_m)}} \left[\beta_{j_m} \sum_{1 \leq k \leq m} \alpha_{j_1 \cdots j_m \, j_k \cdots j_{m-1}} \right].$$

Now observe that the bracketed term above is equal to $\alpha_{j_1 \cdots j_m}$ - $\beta_{j_m} \sum_{j \neq j_i} \alpha_{j, j_1 \cdots j_{m-1}}$ from the proof of lemma 6. After substitution we get that

$$F(LRU) = 1 - \sum_{q \in Q} \left[\alpha_{j_1 \cdots j_m} - \beta_{j_m} \sum_{j \neq j_i} \left[\frac{\beta_j \prod_{i=1}^{m-1} \beta_{j_i}}{\prod_{i=1}^{m-1} (D_i(q) + \beta_{j_m})} \right] \right] .$$

$$= \sum_{q \in Q} \frac{D_1(q) \prod_{i=1}^{m} \beta_{j_i}}{\prod_{i=1}^{m-1} (D_i(q) + \beta_{j_m})} \qquad \text{Q.E.D.}$$

For FIFO we again note that the control state q explicitly defines the memory state s; hence configurations are specified by giving only q. Let q and q' be two control states. By $p_{q',q}$ we mean $Pr[Y_k = q \mid Y_{k-1} = q']$. From definition 6 and Theorem 4, if $q = q' = (j_1 \cdots j_m)$ then $p_{q,q'} = \sum_{i=1}^{m} \beta_{j_i}$. If $q = (j_2 \cdots j_m, j)$ and $q' = (j_1 \cdots j_m)$ then $p_{q',q} = \beta_j$; all other $p_{q,q'} = 0$.

In appendix A we show that of $m \leq m-2$, then the Markov chain described by P is irreducible.

<u>Lemma 7</u>.

For FIFO let α_q, $q = (j_1, \ldots, j_m)$, $q \in Q$ be a typical element of the vector $\underline{\alpha}$ which satisfies $\underline{\alpha}P = \underline{\alpha}$ and $\sum_{q \in Q} \alpha_q = 1$. Also assume $m \leq n-2$. Then

$$\alpha_q = \frac{\prod_{i=1}^{m} \beta_{j_i}}{\sum_{\substack{q \in Q \\ q = j_1 \cdots j_m}} \left[\prod_{i=1}^{m} \beta_{j_i} \right]} .$$

<u>Proof</u>:

1. From the definition of P and $\underline{\alpha}P = \underline{\alpha}$, if $q = (j_1, \ldots, j_m)$ then

$$\alpha_{j_1, \ldots, j_m} = \sum_{j \neq j_i} \beta_{j_m} \alpha_{j, j_1, \ldots, j_{m-1}} + \alpha_{j_1, \ldots, j_m} \sum_{i=1}^{m} \beta_{j_i} .$$

Now we substitute into the above equation the proposed solutions for

$$\alpha_{j, j_1, \ldots, j_m}$$

yielding

$$\alpha_{j_1 \ldots j_m} [1 - \sum_{i=1}^{m} \beta_{j_i}] = \frac{\sum_{j \neq j_i} \beta_{j_m} \beta_j \prod_{i=1}^{m-1} \beta_{j_i}}{\sum_{q \in Q} \left[\prod_{i=1}^{m} \beta_{j_i} \right]} \text{ . Now}$$

$$\alpha_{j_1 \ldots j_m} D_1(q) = \frac{D_1(q) \prod_{i=1}^{m} \beta_{j_i}}{\sum_{q \in Q} \left[\prod_{i=1}^{m} \beta_{j_i} \right]} .$$

2. We must also show that $\sum_{q \in Q} \alpha_q = 1$ but this follows trivially. Q.E.D.

<u>Proof of Theorem 3</u>:

1. As for LRU, all configurations (s,q) with identical s are in the same subset of the partition Π. From Theorem 5 and the definition of the matrix P for FIFO, if $m \leq n-2$

$$F(\text{FIFO}) = 1 - \sum_{q \in Q} \alpha_q \, p_{qq}$$

$$q=(j_1 \ldots j_m)$$

$$= 1 - \sum_{q \in Q} \frac{\left[\prod_{i=1}^{m} \beta_{j_i} \right] \left[\sum_{i=1}^{m} \beta_{j_i} \right]}{\sum_{q \in Q} \left[\prod_{i=1}^{m} \beta_{j_i} \right]}$$

$$= \frac{\sum_{q \in Q} \left[D_1(q) \prod_{i=1}^{m} \beta_{j_i} \right]}{\sum_{q \in Q} \left[\prod_{i=1}^{m} \beta_{j_i} \right]}$$

2. It is not difficult to show that this same equation holds if $m = n-1$ or $m = n$. Q.E.D.

V. EXTENSIONS

The analytical technique developed here for comparing paging algorithms is applicable to a more general class of program models. In [4] a Markov model of programs is introduced by defining a set Z of program states, an initial program state z_0 and a mapping $f: Z \times N \to Z$ which determines the next program state. Also the probability distribution for r_k depends on the present programs control state z_k. With this program model, we can still prove theorem 4; hence the techniques developed here can be applied.

ACKNOWLEDGEMENTS

The author greatfully acknowledges many stimulating discussions with R. Sethi and G. Shedler during the course of this work. Also appreciated was the critical reading of the manuscript by P. Yue. R. Tapscott wrote the program used in the numerical studies.

REFERENCES

[1] B. Randell and C.J. Kuehner. "Dynamic Storage Allocation Systems." C.A.C.M., Vol. 11, No. 5, May 1968, pp. 297-305.

[2] L.A. Belady. "A Study of Replacement Algorithms for a Virtual Storage Computer." IBM Systems Journal, Vol. 4, July 1966, pp. 78-101.

[3] M. Joseph. "An Analysis of Paging and Program Behavior." The Computer Journal, Vol. 13, No. 1, February 1970, pp. 48-54.

[4] A.V. Aho, P.J. Denning, and J.D. Ullman. "Principles of Optimal Page Replacement." Computer Sciences Laboratory Technical Report No. 82, Princeton University, January 1970.

[5] P.J. Denning, Y.C. Chen, and G.S. Shedler. "A Model of Program Behavior Under Demand Paging." IBM T.J. Watson Research Center, Report RC-2301, September 1968.

[6] N.L. Johnson. "A Proof of Wald's Theorem on Cumulative Sums." Ann. Math. Statistics, Vol. 30, No. 4, pp. 1245-1247.

[7] G. Birkhoff and T.C. Bartee. Modern Applied Algebra. McGraw-Hill, Inc., New York, 1970, pp. 221-222.

APPENDIX A

In this appendix we will show that if $m \leq n-2$, the Markov chain for FIFO with state set Q is irreducible. The approach is as follows:

1. First we show that if $q' = (j_1 \ldots j_k\ j_{k+1} \ldots j_m)$ and $q = (j_1 \ldots j_{k-1}\ j_{k+1}\ j_k \ldots j_m)$, there exists an integer ℓ so that $p^{(\ell)}_{q',q} > 0$. Note that if q' is considered as an ordered set, then q is the result of a transposition [7] on that set. Recall that any permutation on a set can be expressed as a sequence of transpositions [7]. Hence this result suffices to prove that there exists an integer ℓ so that $p^{(\ell)}_{q',q''} > 0$ for any q'' which is a permutation of q'.

2. Next we use this result to show that for any control states q, q' there exists an integer ℓ so that $p^{(\ell)}_{q',q} > 0$, hence the Markov chain is irreducible.

<u>Definition</u>.

For a given control state $q = (j_1 \ldots j_m)$ let $C(q) = \{q' \mid q'$ is any permutation of $q\}$.

<u>Definition</u>.

Consider two states $q = (j_1 \ldots j_m)$ and $q' = (i_1 \ldots i_m)$. By $(i_1 \ldots i_m) \to (j_1 \ldots j_m)$ we mean that $\exists\, j \in N$ so that $\delta_{FIFO}(q',j) = q$. By $\overset{*}{\to}$ we mean the transitive, reflexive closure of $\to$.

<u>Lemma A</u>:

If $m \leq n-2$ then $(j_1 \ldots j_{k-1}\ j_k\ j_{k+1} \ldots j_m) \overset{*}{\to} (j_1 \ldots j_{k-1}\ j_{k+1}\ j_k \ldots j_m)$.

Proof:

1. Let $x, y \in N$; $x, y \neq j_i$, $1 \le i \le m$. These exist because $m \le n-2$. We will exhibit the sequence of states $\{q_1, q_2, \ldots, q_\ell\}$ so that $q_1 = (j_1 \cdots j_{k-1}\ j_k\ j_{k+1} \cdots j_m)$, $q_\ell = (j_1 \cdots j_{k-1}\ j_{k+1}\ j_k \cdots j_m)$ and $q_i \to q_{i+1}$, $1 \le i \le \ell-1$.

$$(j_1 \cdots j_k\ j_{k+1} \cdots j_m) \to (j_2 \cdots j_k\ j_{k+1} \cdots j_m\ x) \to (j_3 \cdots j_k\ j_{k+1} \cdots j_m\ x\ j_1)$$

$$\overset{*}{\to} (j_k\ j_{k+1} \cdots j_m\ x\ j_1 \cdots j_{k-2}) \to (j_{k+1} \cdots j_m\ x\ j_1 \cdots j_{k-2}\ j_{k-1})$$

$$\to (j_{k+2} \cdots j_m\ x\ j_1 \cdots j_{k-1}\ y) \to (j_{k+3} \cdots j_m\ x\ j_1 \cdots j_{k-1}\ y\ j_k)$$

$$\to (j_{k+4} \cdots j_m\ x\ j_1 \cdots j_{k-1}\ y\ j_k\ j_{k+2}) \overset{*}{\to} (x\ j_1 \cdots j_{k-1}\ y\ j_k\ j_{k+2} \cdots j_{m-1})$$

$$\to (j_1 \cdots j_{k-1}\ y\ j_k\ j_{k+2} \cdots j_m) \to (j_2 \cdots j_{k-1}\ y\ j_k\ j_{k+2} \cdots j_m\ x)$$

$$\overset{*}{\to} (y\ j_k\ j_{k+2} \cdots j_m\ x\ j_1 \cdots j_{k-2}) \to (j_k\ j_{k+2} \cdots j_m\ x\ j_1 \cdots j_{k-2}\ j_{k-1})$$

$$\to (j_{k+2} \cdots j_m\ x\ j_1 \cdots j_{k-2}\ j_{k-1}\ j_{k+1}) \to (j_{k+3} \cdots j_m\ x\ j_1 \cdots j_{k-1}\ j_{k+1}\ j_k)$$

$$\to (j_1 \cdots j_{k-1}\ j_{k+1}\ j_k\ j_{k+2} \cdots j_m)$$

Q.E.D.

Corollary A:

If $m \le n-2$ then for any $q, q' \in C(q)$, $q' \overset{*}{\to} q$.

Theorem A.

If $m \le n-2$ then for any $q, q' \in Q$, $q' \overset{*}{\to} q$.

Proof:

1. Let $q' = (i_1, \ldots, i_m)$ and $q = (j_1, \ldots, j_m)$. Let $\{k_1, \ldots, k_\ell\} = \{i_1, \ldots, i_m\} \cap \{j_1, \ldots, j_m\}$ i.e., the set $\{k_1, \ldots, k_\ell\}$ consists of all page names which are both in q' and q. Also $\{k_{\ell+1}, \ldots, k_m\} =$

$\{i_1, \ldots, i_m\} - \{k_1, \ldots, k_\ell\}$, and $\{h_1, \ldots, h_{m-\ell}\} =$

$\{j_1, \ldots, j_m\} - \{k_1, \ldots, k_\ell\}$.

2. By lemma A $(i_1, \ldots, i_m) \overset{*}{\to} (k_{\ell+1}, \ldots, k_m\ k_1, \ldots, k_\ell)$.

   Now $(k_{\ell+1}, \ldots, k_m\ k_1, \ldots, k_\ell) \to (k_{\ell+2}, \ldots, k_m\ k_1, \ldots, k_\ell\ h_1)$

   $\overset{*}{\to} (k_1, \ldots, k_\ell\ h_1, \ldots, h_{m-\ell}) \in C(q)$. Hence by lemma A

   $(k_1, \ldots, k_\ell\ h_1, \ldots, h_{m-\ell}) \overset{*}{\to} q$.

Q.E.D.

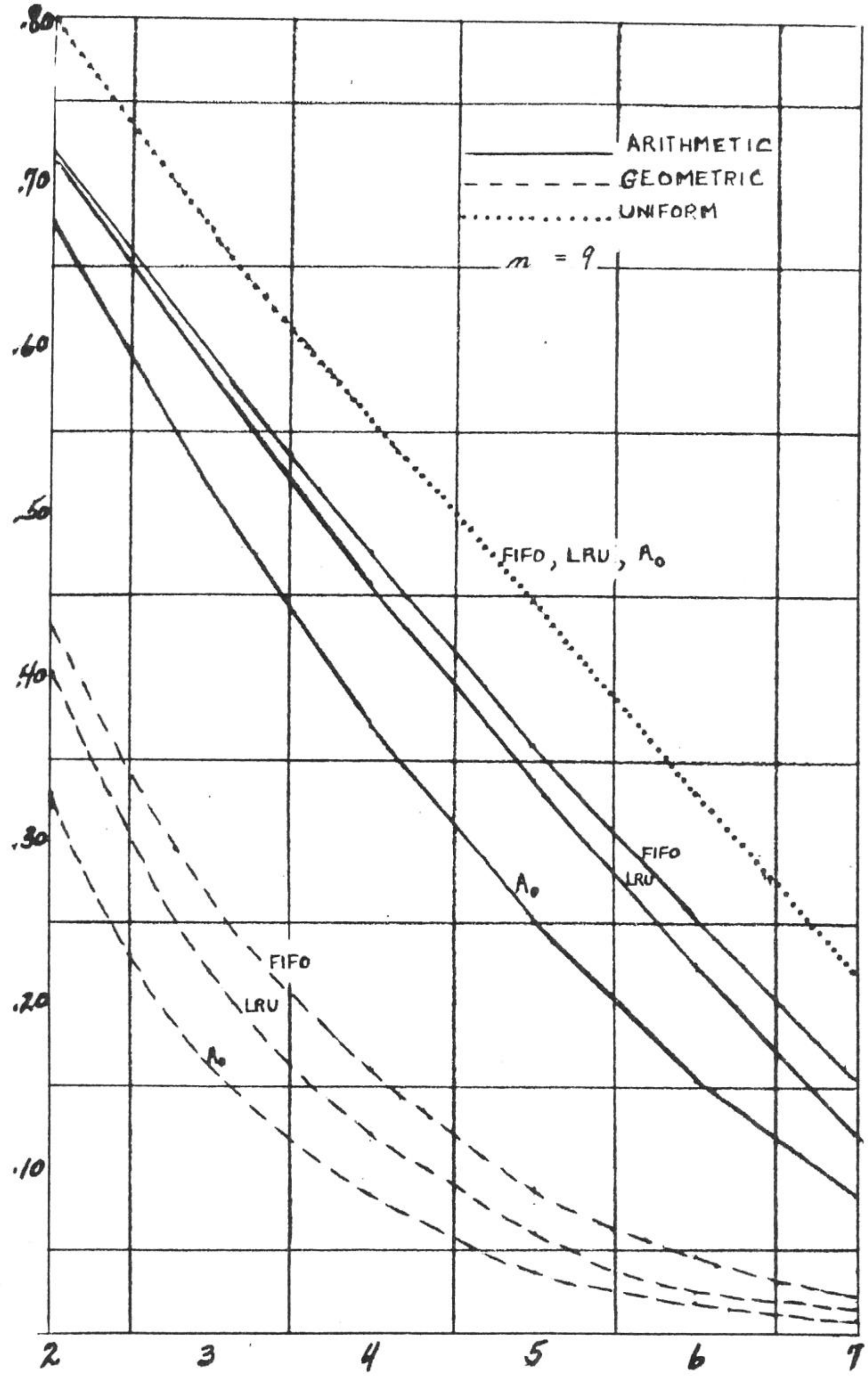

FIGURE 1 - F(A) VS. m FOR VARIOUS DISTRIBUTIONS OF n_k.