Computer Systems

G. Bell, D. Siewiorek, and S.H. Fuller, Editors

# Use of the LRU Stack Depth Distribution for Simulation of Paging Behavior

Rollins Turner and Bill Strecker
Digital Equipment Corporation

**Two families of probability distributions were needed for use by a virtual memory simulation model: headway between page fault distributions, and working set size distributions. All members of both families can be derived from the LRU stack depth distribution. Simple expressions for the computation of both kinds of distributions are given. Finally, examples are given of both families of distributions as computed from a published stack depth distribution.**

**Key Words and Phrases: virtual memory, paging, LRU stack, working set, headway between page faults, computer system simulation**

**CR Categories: 4.35, 4.6, 8.1**

## Introduction

A simulation model for virtual memory operating systems presented the need to describe the memory reference behavior of various programs. While detailed memory reference traces of the programs were available, it was desired to perform the simulation at a much higher level than that of the individual memory references. Events of interest were page faults and physical page allocations. The page allocation algorithm in the simulated system depended on knowledge of the number of pages referenced over a variable amount of time prior to the allocation decision. In the simulation, we did not wish to keep track of individual pages, only the number of pages allocated. Thus for each program two families of probability distributions were needed:

(1) Headway, in memory references, between page faults. This is a different probability distribution for each amount of main memory assigned to the program.

(2) Number of distinct pages touched on the last $N$ memory references (i.e. working set size for parameter $N$). This is a different probability distribution for each value of $N$.

Empirical values for the above distributions represented a large amount of data to determine for each program of interest and to include in the model. Therefore it was desired to find a compact representation for the essential characteristics of a program, and to derive specific distributions as required during a simulation run.

A set of statistics known as the *LRU stack depth distribution* [1, 2] appeared to be the best representation of the "essential characteristics" of a program's paging behavior. From this set of statistics, any of the probability distributions mentioned earlier can be derived.

## LRU Stack Depth Distribution

The LRU stack depth distribution is based on the concept of a least recently used page number stack. We examine a program's page reference stream, and at each memory reference we place the referenced page number on the top of the stack. If the page number appears lower in the stack, we delete it from that position. As we do this, we keep track of how often we find the referenced page number at each depth. From this information we can form an empirical distribution for the stack depth at which the page number was found. This will be a discrete distribution, with as many points as there are distinct pages in the reference stream. Using this distribution as the basis for our model of program behavior, we assume that the stack depth for each reference is a random variable. We assume that these random variables are independently and identically distributed throughout the run. Whether this assumption is justified is an empirical question that must be answered individually for each program of interest.

Through the rest of this paper we shall use the following notation for stack depth probabilities:

$L(I)$ = probability that the referenced page number will be at or above depth $I$; e.g. $L(2)$ is the probability that a page will be one of the two most recently referenced,

$G(I)$ = probability that it will be below depth $I$
$= 1 - L(I)$.



Authors' address: Digital Equipment Corporation, 146 Main Street, Maynard, Ma 01754.

### Distribution for Headway Between Faults

If a program is running in $K$ pages of main memory, using the least recently used (LRU) page replacement algorithm, a page fault will occur if and only if a reference is made with a stack depth greater than $K$. Hence the probability of no fault on any given reference is simply $L(K)$. The probability of making at least $N$ successful references without a page fault is therefore $L(K)^N$. The distribution function for headway between faults, when the program is running in $K$ pages of memory, is given by

$$H(K, N) = \text{prob}\,[\text{headway} < N] = 1 - L(K)^N. \quad (1)$$

For any given value of $K$ (the number of pages allocated) this is the probability distribution function required by the model in order to determine a randomly selected time for the next page fault. The expected headway between page faults, when the program is running in $K$ pages of memory is given by

$$E\,[\text{headway}] = \sum_{N=1}^{\infty} (1 - H(K, N)) = \frac{L(K)}{1 - L(K)}. \quad (2)$$

### Distribution for Working Set Size

Let us define $P(K, N)$ as the probability of touching $K$ distinct pages on $N$ successive references (i.e. the probability that the working set size with parameter $N$ will be $K$ at a randomly selected point in the reference stream). We note that $K$ and $N$ must be positive integers, and that $P(K, N) = 0$ whenever $K > N$. We can write the following recurrence relation for $P(K, N)$

$$P(K, N) = P(K - 1, N - 1)G(K - 1) + P(K, N - 1)L(K). \quad (3)$$

This relation is based on the fact that there are only two ways we could have reached the point of having $K$ unique pages on $N$ references. We could have had $K - 1$ after $N - 1$ references and gotten a stack depth greater than $K - 1$, or we could have $K$ after $N - 1$ and gotten a stack depth less than or equal to $K$.

If all of the $L(I)$'s are different, we can express $P(K, N)$ exactly in terms of their values

$$P(K, N) = \left[\prod_{i=1}^{K-1} G(i)\right] \cdot \left[\sum_{i=1}^{K} \frac{L(i)^{N-1}}{\prod_{\substack{j=1 \\ j \neq i}}^{K} (L(i) - L(j))}\right]. \quad (4)$$

A derivation of this expression is given in the Appendix. Unfortunately this expression presents numerical problems for many important cases. A more practical approach is to compute $P(K, N)$ directly from the basic recurrence relation.

It is convenient to express the values of $P(K, N)$ for all $K$ and a specific $N$ as a vector.

$$V(N) = \begin{bmatrix} P(1, N) \\ P(2, N) \\ P(3, N) \\ \vdots \\ P(M, N) \end{bmatrix}, \quad \text{where } M \text{ is the number of pages in the program.} \quad (5)$$

We can then write eq. (3) in matrix form:

$$V(N) = A\,V(N - 1), \quad (6)$$

$$\text{where } A = \begin{bmatrix} L(1) & 0 & 0 & 0 & \cdots \\ G(1) & L(2) & 0 & 0 & \cdots \\ 0 & G(2) & L(3) & 0 & \cdots \\ \vdots & \vdots & \vdots & & \\ & & & & L(M) \end{bmatrix}. \quad (7)$$

We then see that

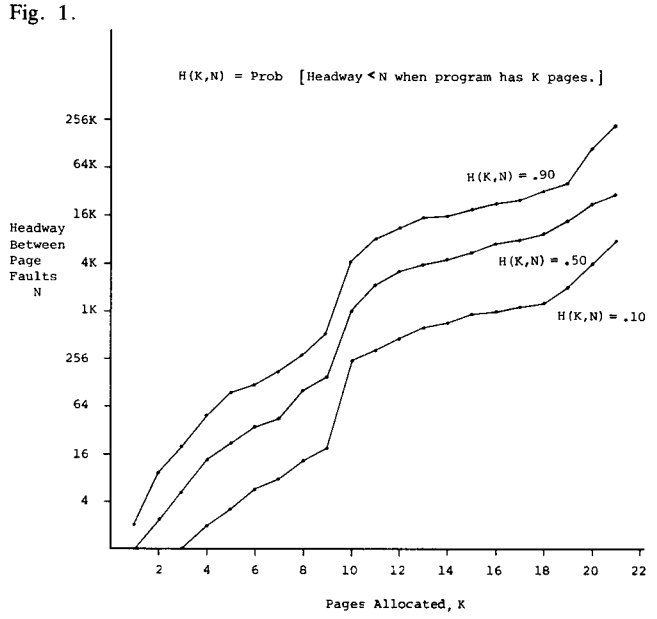
$$V(N) = A \cdot V(N - 1) = A^2 \cdot V(N - 2) = A^3 \cdot V(N - 3) = \cdots = A^{N-1} \cdot V(1). \quad (8)$$

We know the value of $V(1)$, since it is simply the set of probabilities for the number of distinct pages on one reference.

$$V(1) = \begin{bmatrix} P(1, 1) \\ P(2, 1) \\ P(3, 1) \\ \vdots \\ P(M, 1) \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \quad (9)$$

Hence we have a simple computational algorithm for finding values of $P(K, N)$ for large values of $N$. By repeatedly squaring matrix $A$ we get values of $P(K, N)$ for $N$ equal to corresponding powers of 2. The amount of computation to find values for $N$ equal to a power of 2 increases only as the base 2 logarithm of $N$. For example, to find $P(K, 32768)$ for all $K$ requires 15 matrix products. By summing appropriate powers of $A$, we can compute the values of $P(K, N)$ for any value of $N$.

Fig. 1.



## Examples

This section contains examples of computing specific probability distributions from both families—headway between page faults, and working set size. For these examples a stack depth distribution was taken from the literature [3]. The program consisted of 22 distinct pages, with a page size of 256 words. On a run consisting of 500,000 memory references, the stack depth frequencies given in Table I were observed. As before let $H(K, N)$ be the distribution function for headway between page faults when the program has $K$ pages of memory. Values of $H(K, N)$ computed from the stack depth distribution in Table I according to expression (8) are shown in Figure 1. It is interesting to note that if this program runs with just one page less than its full size, there is a fifty percent chance that it will fault in less than 50,000 references ($.99986^{50,000} \cong .5$).

### Working Set Size Distribution Example

We let $P(K, N)$ denote the probability that the working set size of parameter $N$ will be $K$. There is a different probability distribution for the working set size for each value of the parameter $N$. These probability distributions were computed from the stack depth distribution for various values of $N$ by the matrix product method, and the expected value of the working set size was computed for each. The expected value, and the points at which the cumulative probability reaches .01 and .99 are plotted as a function of $N$ in Figure 2. In Figure 3, the entire distribution is plotted for selected values of $N$. We see that for low values of $N$ the probability is concentrated in the low values of number of pages. As $N$ is increased, the distribution spreads out and moves from left to right. Finally, as $N$

Table I. Probability Distribution for Headway Between Page Faults.

| Depth | Relative frequency |
|---|---|
| 1 | .410146 |
| 2 | .398497 |
| 3 | .078842 |
| 4 | .065996 |
| 5 | .016778 |
| 6 | .010070 |
| 7 | .006547 |
| 8 | .004866 |
| 9 | .003692 |
| 10 | .004078 |
| 11 | .000218 |
| 12 | .000038 |
| 13 | .000046 |
| 14 | .000022 |
| 15 | .000032 |
| 16 | .000024 |
| 17 | .000020 |
| 18 | .000016 |
| 19 | .000020 |
| 20 | .000026 |
| 21 | .000012 |
| 22 | .000014 |

Fig. 2.

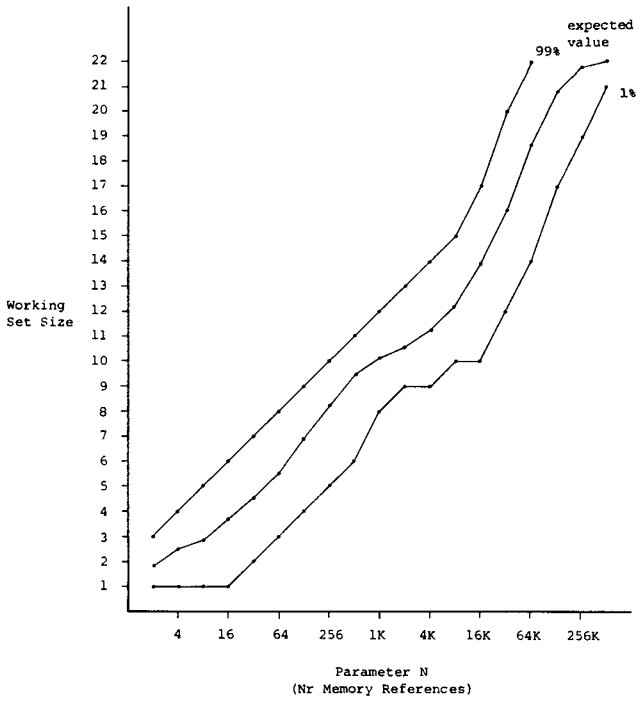
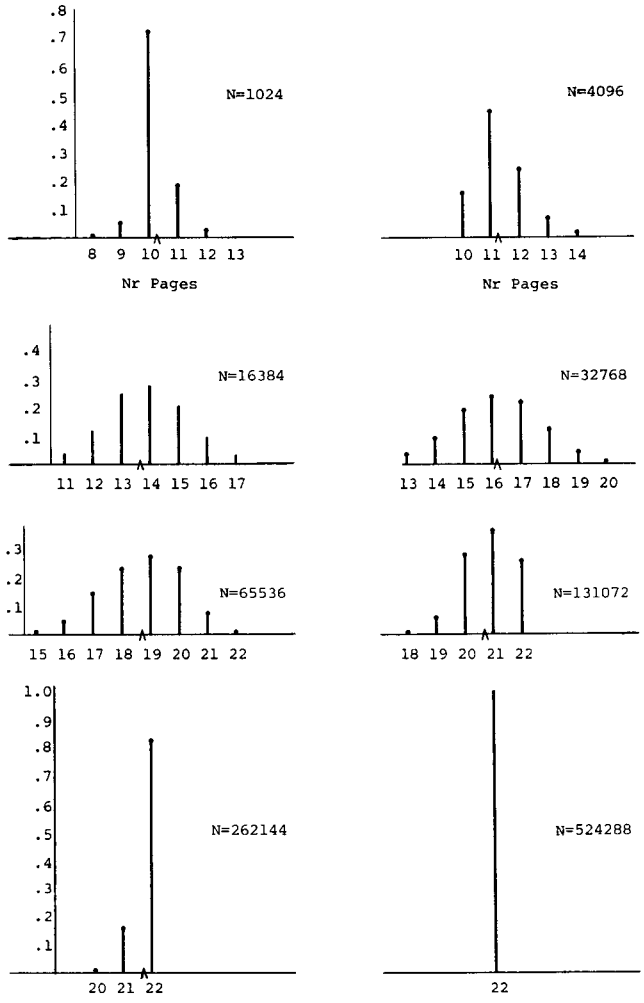



Fig. 3. Probability distributions for working set sizes for various values of parameter $N$ ($\Lambda$ is expected value).



approaches the total number of memory references, the probability concentrates entirely at the maximum number of pages. This example corresponds closely to the predictions for such distributions made by Denning and Schwartz [5]. It is interesting to compare the empirical values of this example, as shown in Figure 3, to the sketch given by Denning and Schwartz in their Figure 6. Tables of values from which Figures 1, 2, and 3 were drawn can be found in [4].

## Conclusion

The LRU stack depth distribution provides "the essential characteristics" of paging behavior that we need in order to simulate the behavior of a program running in a virtual memory environment with LRU page replacement. From this single set of statistics we can compute distributions for working set size and headway between faults for any desired values of their respective parameters.

## Appendix

We assume throughout that $L(i) \neq L(j)$ for $i \neq j$.

Let $K$ be a fixed, but arbitrary, positive integer and consider the sequence $P(K, N)$ for $N = 0, 1, 2, \ldots \infty$. Let us define $Q(K, Z)$ as the $z$ transform (or generating function) of this sequence

$$Q(K, z) = \sum_{N=0}^{\infty} P(K, N) z^N. \tag{1}$$

For all $K$ and $N > 0$ we have

$$P(K, N) = P(K-1, N-1)\, G(K-1) + P(K, N-1)\, L(K). \tag{2}$$

Multiplying both sides by $z^{N-1}$ and summing over $N = 1, 2, \ldots \infty$ gives

$$\frac{1}{z} \sum_{N=1}^{\infty} P(K, N)\, z^N = \sum_{N=0}^{\infty} P(K-1, N) z^N\, G(K-1) + \sum_{N=0}^{\infty} P(K, N) z^N\, L(K). \tag{3}$$

Or, substituting from (1),

$$(1/z)\,[Q(K, z) - P(K, 0)] = Q(K-1, z) G(K-1) + Q(K, z) L(K). \tag{4}$$

Solving for $Q(K, z)$, and substituting the value 0 for $P(K, 0)$ we get

$$Q(K, z) = \frac{z Q(K-1, z) G(K-1)}{1 - zL(K)}. \tag{5}$$

Applying (5) recursively $K$ times we get

$$Q(K, z) = \prod_{i=1}^{K} \frac{zG(i-1)}{1 - zL(i)}\, Q(0, z). \tag{6}$$

Since $P(0, N)$ is 1 for $N = 0$ and 0 for $N > 0$, $Q(0, z) = 1$, and (6) can be written

$$Q(K, z) = \prod_{i=1}^{K} \frac{zG(i-1)}{1 - zL(i)}. \tag{7}$$

Equation (7) is the transform of $P(K, N)$. We must now apply an inverse transformation to obtain an expression for $P(K, N)$. We write a partial fraction expansion of the product on the right of (7)

$$\sum_{i=1}^{K} \frac{a(i)}{1 - zL(i)} = \prod_{j=1}^{K} \frac{zG(j-1)}{1 - zL(j)}. \tag{8}$$

We can solve for an arbitrary $a(i)$, as follows. We multiply both sides by $1 - zL(i)$ to obtain

$$(1 - zL(i)) \sum_{j=1}^{K} \frac{a(j)}{1 - zL(j)} = (1 - zL(i)) \prod_{j=1}^{K} \frac{zG(j-1)}{(1 - zL(j))}. \tag{9}$$

Taking the limit as $z \to 1/L(i)$,

$$a(i) = \prod_{j=1}^{K} L(i)^{-1}\, G(j-1) \Big/ \prod_{\substack{j=1 \\ j \neq i}}^{K} \left(1 - \frac{L(j)}{L(i)}\right). \tag{10}$$

Or, multiplying numerator and denominator by $L(i)^K$,

$$a(i) = \prod_{j=1}^{K} G(j-1) \Big/ L(i) \prod_{\substack{j=1 \\ j \neq i}}^{K} (L(i) - L(j)). \tag{11}$$

Substituting (11) into (8) for each $a(i)$, we obtain

$$Q(K, z) = \left[\prod_{j=0}^{K-1} G(j)\right] \cdot \sum_{i=1}^{K} \left( 1 \Big/ L(i)(1 - zL(i)) \prod_{\substack{j=1 \\ j \neq i}}^{K} (L(i) - L(j)) \right). \tag{12}$$

Since $1/L(i)(1 - zL(i))$ is the $z$ transform of $L(i)^{N-1}$, we can invert (12) term by term to obtain

$$P(K, N) = \left[\prod_{i=0}^{K-1} G(i)\right] \sum_{i=1}^{K} \left( L(i)^{N-1} \Big/ \prod_{\substack{j=1 \\ j \neq i}}^{K} (L(i) - L(j)) \right) \tag{13}$$

*Acknowledgments.* Numerous discussions with colleagues in the DEC Research and Development Group—Jim Bell, Chuck Kaman, Mary Payne, and George Poonen—provided help toward the results given in this paper.

Received September 1975; revised September 1976

**References**

1. Mattson, R.L., Gecsei, J., Slutz, D.R., and Traiger, I.L. Evaluation techniques for storage hierarchies. *IBM Syst. J. 9*, 2 (1970), 78-117.
2. Spirn, F.R., and Denning, P.J. Experiments with program locality. Proc. AFIPS 1972 FJCC, AFIPS Press, Montvale, N.J., pp. 611-621.
3. Sager, G.R. Reproducing program memory reference behavior. Proc. Comptr. Sci. and Statist.: Seventh Annual Symp. on the Interface, Oct. 1973, pp. 41-47.
4. Turner, R. Simulation of paging behavior. Tech. Memo. 005-171-003, Digital Equipment Corp., Maynard Mass., Oct. 1975.
5. Denning, P.J., and Schwartz, S.C. Properties of the working set model. Proceedings of the Third Symposium on Operating Systems Principles, Stanford University, Stanford, Calif., Oct. 1971, pp. 130-140.