Operating Systems

C. Weissman Editor

A Note on the Calculation of Average Working Set Size

D.R. Slutz and I.L. Traiger
IBM Research Laboratory, San Jose

Finite-length reference string of arbitrary structure are considered, and an exact expression for average working set size in terms of "corrected" interreference interval statistics is derived. An example is discussed; upper and lower bounds are obtained; and the average working set size function is shown to be efficiently obtained for a set of page sizes, in a single pass of the reference string.

This work follows the developments of a paper by Denning and Schwartz, who consider infinite-length reference strings which satisfy certain statistical properties and who derive an expression relating the asymptotic average working set size to the asymptotic missing page rate function under working set replacement.

Key Words and Phrases: working-set model, paging, program behavior

CR Categories: 4.3

1. Introduction

In "Properties of the Working-Set Model" [1], Denning and Schwartz consider reference strings of infinite length, generated by a stationary stochastic process, and derive an expression relating the asymptotic (with string length) average working set size (for any window size T) to the asymptotic missing page rate function under working set replacement. Although their results can be used to obtain an estimate of the asymptotic average working set size for a finite realization of such a stationary process, their results do not yield the exact average working set size for finite-length realizations. A later effort [2] bounds the error for a related problem.

In this paper, we consider finite length reference strings of arbitrary structure. Following the developments in [1], we obtain an exact expression for average working set size in terms of "corrected" interreference interval statistics, and provide an efficient procedure for determining both the missing page rate function and the average working set size function. The error between the actual average working set size and the estimate obtained from the technique of [1] is bounded. It is seen that, although the estimate was derived for a class of probabilistically generated reference strings, the estimate is often quite good in practice. In fact, the bound on the error depends only on the number of pages in the program, T, and on the length of the string (not on the structure of the string), and tends to zero as the length increases. Finally, we show that stack processing techniques [3, 4] can be augmented to obtain both the average working set size and the missing page rate functions efficiently for a set of page sizes, in a single pass of the reference string.

2. Definitions

Let $N = \{1, 2, \ldots, n\}$ be a set of *pages* and let $\rho_k = r_1, r_2, \ldots, r_k$, denote a *reference string* of length $k < \infty$, where $r_t \in N$, $1 \leq t \leq k$. (We have adopted the notation of [1], but we have added a subscript k to denote the reference string length.) Referring to the index of the references as "time," the *working set* $W(t, T)$ at time t, for $T \geq 1$, is defined as the set of distinct pages referenced in the interval $[t - T + 1, t]$ for $T \leq t$, or $[1, t]$ for $T > t$. The number of pages in $W(t, T)$ is called the *working set size* $w(t, T)$. We assume throughout that $w(0, T) = 0$ and that all N pages are referenced in ρ_k, so that $w(k, k) = n$.

In a paging system using a working set replacement algorithm, just the pages in $W(t, T)$ are maintained in main memory at time t. If r_{t+1} is not in $W(t, T)$, then a page transfer from a secondary storage device is required and a *page fault* is said to occur. Two performance measures of interest are the *average working set size* $s_k(T)$, $s_k(T) = 1/k \sum_{t=1}^{k} w(t, T)$, which reflects the main memory requirements of the program, and the *missing page rate* $m_k(T)$, which is the fraction of all references that cause page faults. As in [1], define $\Delta(t, T)$ for $0 \leq t \leq k - 1$ as

$$\Delta(t, T) = \begin{cases} 1, \text{ if } r_{t+1} \text{ is not in } W(t, T) \\ 0, \text{ otherwise} \end{cases}$$

The missing page rate is then $m_k(T) = 1/k \sum_{t=0}^{k-1} \Delta(t, T)$.

The *interreference interval* x_t at time t denotes the

Authors' address: IBM Research Laboratory, Monterey and Cottle Roads, San Jose, CA 95136

Fig. 1. Determination of interreference interval distribution.

t	1	2	3	4	5	6	7	8
r_t	c	b	a	b	b	a	a	c
Stack, TIME	c,1	b,2 c,1	a,3 b,2 c,1	b,4 a,3 c,1	b,5 a,3 c,1	a,6 b,5 c,1	a,7 b,5 c,1	c,8 a,7 b,5
c(1)	0	0	0	0	1	1	2	2
c(2)	0	0	0	1	1	1	1	1
c(3)	0	0	0	0	0	1	1	1
c(4)	0	0	0	0	0	0	0	0
c(5)	0	0	0	0	0	0	0	0
c(6)	0	0	0	0	0	0	0	0
c(7)	0	0	0	0	0	0	0	1
c(∞)	1	2	3	3	3	3	3	3

elapsed time since the previous reference to page r_t. Thus if r_3 and r_8 denote successive reference to the same page, then $x_8 = 5$. If r_t is the first reference to a page we define $x_t = \infty$. Letting $c_k(x)$, for $1 \le x \le k$ and $x = \infty$, be the number of occurrences of $x_t = x$ over ρ_k, the *interreference frequency function* $f_k(x)$ is $c_k(x)/k$ and the *interreference distribution function* $F_k(x)$ is $F_k(x) = \sum_{y=1}^{x} f_k(y)$.

Since $\Delta(t, T) = 1$ if and only if $x_{t+1} > T$, it follows that

$$m_k(T) = 1 - F_k(T) = 1 - (1/k) \sum_{x=1}^{T} c_k(x). \qquad (1)$$

Note that this result (from [1]) is exact for any finite length reference string.

3. Calculation of Average Working Set Size

Following [1] we observe that $w(t+1, T+1) = w(t, T) + \Delta(t, T)$. Then

$$\begin{aligned} s_k(T+1) &= 1/k \sum_{t=1}^{k} w(t, T+1) \\ &= 1/k \sum_{t=1}^{k} [w(t-1, T) + \Delta(t-1, T)] \\ &= s_k(T) - (w(k, T)/k) + m_k(T). \end{aligned} \qquad (2)$$

Using (1) yields

$$\begin{aligned} s_k(T+1) &= s_k(T) + (1/k)[k(1-F_k(T)) - w(k, T)] \\ &= s_k(T) + (1/k)\left[k - \sum_{x=1}^{T} c_k(x) - w(k, T)\right]. \end{aligned} \qquad (3)$$

For each time t we define a function $h_t(z)$, $z \ge 1$, as follows: if there exists a page y which, if referenced at time $t+1$ would yield $x_{t+1} = z$, then $h_t(z) = 1$; otherwise $h_t(z) = 0$. It is seen that $h_t(z)$ reflects the times of last reference to pages up through time t. It follows that $w(k, t) = \sum_{x=1}^{T} h_k(x)$ and (3) becomes

$$\begin{aligned} s_k(T+1) &= s_k(T) + (1/k) \\ &\quad \cdot\left[k - \sum_{x=1}^{T} (c_k(x) + h_k(x))\right] \\ &= s_k(T) + 1 - (1/k) \sum_{x=1}^{T} \hat{c}_k(x) \end{aligned} \qquad (4)$$

where $\hat{c}_k(x) = c_k(x) + h_k(x)$, $1 \le x \le k$, can be viewed as a "corrected" form of $c_k(x)$. We have then from (4)

$$s_k(T) = \sum_{z=0}^{T-1} \left(1 - (1/k) \sum_{x=1}^{z} \hat{c}_k(x)\right) \qquad (5)$$

which provides the basis of a procedure to calculate $s_k(T)$.

In [1], a similar result is obtained for a class of infinite-length reference strings, namely

$$s(T) = \sum_{z=0}^{T-1} m(z) = \sum_{z=0}^{T-1} (1 - F(z)), \qquad (6)$$

where $s(T)$, $m(T)$ and $F(T)$ are asymptotic values of average working set size, missing page rate and interreference distribution function, respectively. If this result is applied to finite-length strings, an estimate $s_k'(T)$ is obtained, where

$$\begin{aligned} s_k'(T) &= \sum_{z=0}^{T-1} m_k(z) \\ &= \sum_{z=0}^{T-1} \left(1 - (1/k) \sum_{x=1}^{z} c_k(x)\right). \end{aligned} \qquad (7)$$

We observe that

$$\begin{aligned} s_k'(T) - s_k(T) &= \sum_{z=0}^{T-1} 1/k \sum_{x=1}^{z} (\hat{c}_k(x) - c_k(x)) \\ &= 1/k \sum_{z=0}^{T-1} \sum_{x=1}^{z} h_k(x), \end{aligned}$$

and since $0 \le \sum_{x=1}^{z} h_k(x) \le n$, we have

$$0 \le s_k'(T) - s_k(T) \le nT/k. \qquad (8)$$

Equation (8) indicates the maximum error possible if the estimate $s_k'(T)$ is used. We note that $s_k'(T)$ is an overestimate, and that even for long reference strings a significant error is possible if the number of referenced pages is large but the average working set size is small. Alternatively, if $k > nT$, the error will be less than one page, and as k increases the error tends to zero. Note that the bound depends only on n, T, and k and not on the structure of the reference string. Thus a choice between using the estimate and the technique described here can be made before a program is ever run.

4. An Example

To calculate $s_k(T)$ and $m_k(T)$ it is sufficient to determine $\{c_k(x)\}$ and $\{h_k(x)\}$ (and thus $\{\hat{c}(x)\}$). $\{c_k(x)\}$ can be determined as described in [1]: as ρ_k is scanned a list TIME(i), $1 \le i \le n$, is maintained where, at time t, TIME(i) contains the time page i was last referenced in the interval $[1, t]$. As the scan of ρ_k proceeds, a set of counters $c(x)$, $1 \le x \le k$ and $x = \infty$, initially zeroed, are incremented as follows. Suppose $r_t = i$. The interreference interval x_t is then $x_t = t -$ TIME(i) if page i was previously referenced, and $x_t = \infty$ otherwise. In either case counter $c(x_t)$ is incremented and TIME(i) is reset to t. Following the scan of ρ_k note that $c_k(x) = c(x)$, $1 \le x \le k$.

Figure 1 contains an example for $n = 3$ and $k = 8$. Instead of maintaining the array TIME, we maintain an LRU stack [3] with the corresponding TIME element appended to the stack entry. (Counter $c(k=8)$ is omitted, since an interreference interval of k cannot

Fig. 2. Measured, "corrected" interreference interval distributions.

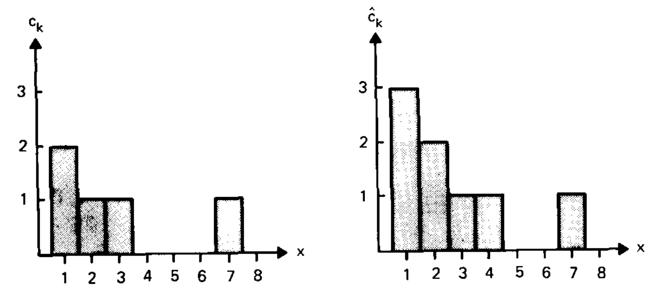
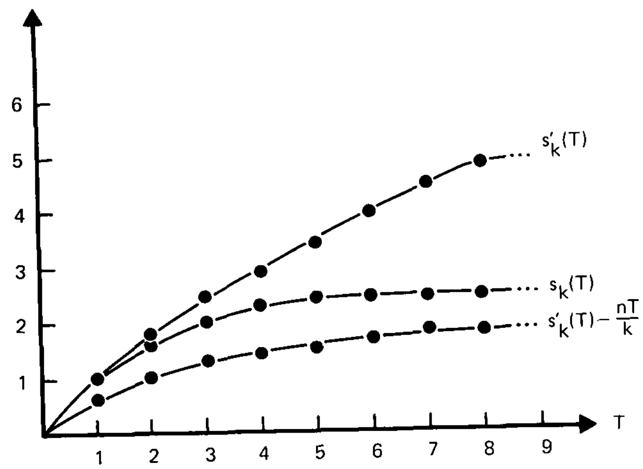


Fig. 3. Estimated and actual average working set sizes.



occur.) The counter values at time 8 are the desired $\{c_k(x)\}$ and are displayed in Figure 2. To determine $h_k(x)$ it is convenient to imagine pages a, b, and c all being referenced at time $t = 9$. The corresponding interreference times are determinable from the TIME entries at $t = 8$. For pages a, b, and c, x_9 is 2, 4, and 1, respectively, which gives $h_k(1) = h_k(2) = h_k(4) = 1$ and $h_k(x) = 0$ otherwise. The resulting distribution $\hat{c}_k(x)$ is included in Figure 2.

Figure 3 shows $s_k(T)$ and $s_k'(T)$ as calculated from eqs. (5) and (7). Also included is a lower bound on $s_k(T)$, obtained from $s_k'(T)$ using eq. (8). The curves are plotted only for $T \leq k$, since the average working set size is unchanged for all $T > k$. The apparent divergence of $s_k(T)$ and $s_k'(T)$ is primarily due to the short length of the reference string in this example. As k is increased the curves will converge.

5. Dependence on Page Size

If working set statistics are desired for a set of different page sizes, it appears that a separate reference string must be analyzed for each page size. We now show this is not necessary by developing a procedure analogous to that of [4], where the authors described a one-pass technique for determining hit ratio functions under LRU replacement for a range of page sizes.

Suppose that a program makes references to a linear address space of 2^v elements (words or bytes, for example), numbered 0 to $2^v - 1$. Page sizes are limited to 2^j, $0 \leq j \leq v$; and for a given page size 2^j, the address space is partitioned into 2^{v-j} pages, where each page consists of 2^j contiguous elements. The pages are numbered consecutively, from 0 to $2^{v-j} - 1$, such that, if r is the v-bit address of an element, the appropriate page address $r(j)$ consists of the $(v-j)$-bit prefix of r. For page size 2^j, we use $\rho_k(j) = r_1(j), r_2(j), \ldots, r_k(j)$ to denote the corresponding page reference string.

If a range of page sizes $2^j, j_L \leq j \leq j_H$, is of interest, we claim that an analysis of just the reference string $\rho_k(j_L)$ is sufficient to determine the set of missing page rate functions $\{m_k(T, j)\}$ and the set of average working set size functions $\{s_k(T, j)\}$. As $\rho_k(j_L)$ is scanned, a list of referenced pages and times of last reference are kept in LRU order, as described in the preceding example, and an array of interreference counters $c(1, j)$, $c(2, j), \ldots, c(k, j)$ (∞, j) is maintained for each value of j.

Consider the determination of the interreference time $x_t(j)$, for page reference $r_t(j)$, at time t. It suffices to find the most recent time of reference among all of the smaller pages of size 2^{j_L} which are contained in the parent page $r_t(j)$, since that time is the last reference to the parent. Because the referenced pages of size 2^{j_L} are listed in LRU order, it is sufficient to scan down the LRU stack and find the *first* entry whose $(v-j)$-bit prefix matches $r_t(j)$. The associated last reference time, say z, yields an interreference time $x_t(j) = t - z$ for page size 2^j, and counter $c(t-z, j)$ is incremented.

In general, $x_t(j)$, for all $j, j_L \leq j \leq j_H$, can be determined by a single scan of the LRU stack. As each stack entry is scanned all appropriate $c(x, j)$ counters are incremented. (A convenient method for determining the appropriate counters for each entry is given in [4].) The scan ceases when either $r_t(j_L)$ is found or the end of the stack is reached.

Reference [4] contains the details of maintaining the LRU stack, as well as a description of a one-pass procedure similar to the above for determining LRU hit ratios. It is easily shown that both procedures can be combined so that LRU and working set statistics can be obtained for a range of page sizes in a single scan of the reference string using a single LRU stack.

Received June 1973; revised April 1974

References

1. Denning, P.J., and Schwartz, S.C. Properties of the working-set model. *Comm. ACM 15*, 3 (Mar. 1972), 191–198.
2. Coffman, E.G., and Denning, P.J. *Operating Systems Theory*. Prentice-Hall, Englewood Cliffs, N.J., 1973, pp. 305–306.
3. Mattson, R.L., Gecsei, J., Slutz, D.R., and Traiger, I.L. Evaluation techniques for storage hierarchies. *IBM Syst. J. 9* (1970), 78–117.
4. Traiger, I.L., and Slutz, D.R. One-pass techniques for the evaluation of memory hierarchies. IBM Res. Rep. RJ 892, 1971.