Discrete Characterization of Program Referencing Dynamics

Philippe De Smedt

Computer Science Division
Department of Electrical Engineering and Computer Sciences
University of California at Berkeley

ABSTRACT

A discrete generative model to describe the dynamics of program behavior in a virtual memory environment is presented. It is based upon the working set concept and models the changes in the working set from one time interval to another by comparing overlapping windows of references, to determine which pages are new, and which ones are no longer referenced. The statistics about arrivals and departures can be used in a stochastic string generation program. The behavior of the new strings is then compared to the one of the original string, as a test of the model's validity.

1. INTRODUCTION

Program behavior in a virtual memory environment can be characterized by analyzing the string of address references generated during a program's execution. Various models of referencing behavior have been proposed in the past. Crucial in most of these models are the notions of *locality* and *phase transition*, i.e., the shift in locality as a function of time. The *working set* concept [Denn68,Denn72] is intended to capture these properties of reference strings. This concept is pretty well known: the working set consists of the pages referenced during a moving virtual time interval (the *window*), time usually being measured in terms of references, rather than actual time units.

A continuous model characterizing a program's behavior based upon working sets has been proposed and implemented earlier [Ferr76,Ferr81a,Ferr81b, Dutt81, Lee82a]. In this model, the working set size is measured after each reference, for a given window size; the extracted data can then be used to stochastically generate an artificial string, whose properties are to be as close as possible to those of the real string.

The model described here is discrete, i.e., the working set size is measured after a fixed number of references. The advantage is that fewer measurements have to be made, and that fewer numbers are needed to characterize the string. If the interval's size is carefully chosen, the reduced number of observations should still allow for accurate artificial string generation . Another advantage is that fluctuations in the working set size become more visible, as opposed to the +1 or -1 fluctuations in the continuous case. A disadvantage is that for each interval two statistics have to be gathered: the number of pages in the previous working set not referenced during the present interval, hereafter called *departures*, and the number of pages not in the previous working set and referenced during the present time interval, called *arrivals*. This can, however, be viewed also as an advantage, since these quantities reveal information about the

The research reported here has been supported in part by the National Science Foundation under grant MCS80-12900 and by the State of California and the NCR Corporation under MICRO grant 1-532436-19900.

composition of the working set, i.e., "old" versus "new" pages.

In the continuous case we also have to deal with pairs of numbers: the times at which a change in the working set size occurs, and the size of the working set at that time. Flat-faults, defined as instances of a working set's change where an arrival coincides with a departure, thus leaving the working set size unchanged [Dutt81, Lee81b], also have to be accounted for. These changes do, however, occur rather infrequently. For the trace we are using in our experiments, containing 500,000 references, and a window size of 5000, the working set size changes only 1157 times [Lee81a] ; however, if we choose for instance a measurement interval of 1000 references in the discrete case, only 500 observations have to be made. The choice of the length of the measurement interval will be discussed in a subsequent section.

As outlined above, the data gathered during the discrete measurement phase are arrivals and departures. How this is done is described in more detail below. The actual observations are used to generate probability distributions, which in turn are used in a generative model to invoke page arrivals and cause pages to depart.

Section 2 describes the model and the definitions used in more detail. In section 3 a number of interesting theoretical aspects of the model and their practical applicability are discussed. The next section describes the extraction of arrival and departure statistics from the original string, which in our experiments contains 500,000 references made during the execution of an APL program on an IBM 360/91; this string is described in [Smit76]. We also show how these statistics can be used to generate probability distributions. During the same measurement phase, a program profile , i.e., a distribution of how often the various pages are referenced throughout the program, is extracted. Section 5 describes the artificial string generation. It shows in depth how the generation program operates, i.e., how it uses the above-mentioned probability distributions, how the generated string can be made "smooth", what boundary, beginning and ending conditions to watch out for, and so on. The next step is to compare the performance indices of the new traces with those of the original one, under the LRU and WS policies. This is done in section 6. The final section is devoted to conclusions .

2. MODEL DESCRIPTION

The purpose of our discrete model is to provide a viable characterization of a program's working set fluctuations over time. Two parameters play a crucial role in the model.

The first is the *window size*, i.e., the number of past references considered when determining the pages in the working set. This parameter is analogous to the one used in continuous characterizations. In this study, the window size is referred to as τ. We will assign different values to τ, usually multiples of 1000 references, for reasons to become clear further on.

The second parameter is the *interval size*, or the number of references between consecutive measurements. Note that the time we refer to is *virtual time*, and is measured in references, rather than in actual time units. The interval size will be denoted by T , and will for practical reasons also be a multiple of 1000.

Figure 1 shows how T and τ fit into the model. In this example, τ is chosen to be equal to 2000, and T is equal to 1000.

In general T will be less than τ, otherwise not every reference would be captured by the model, as can be seen in figure 2.

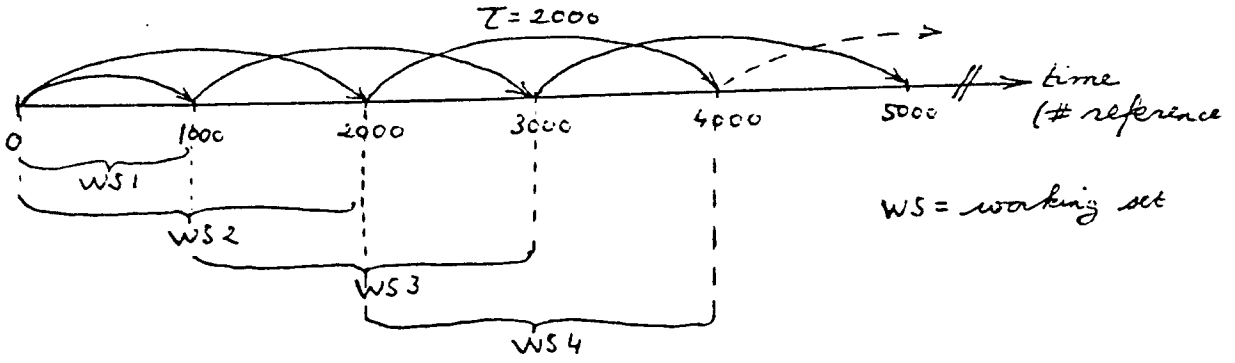

Figure 1. The parameters T and τ.

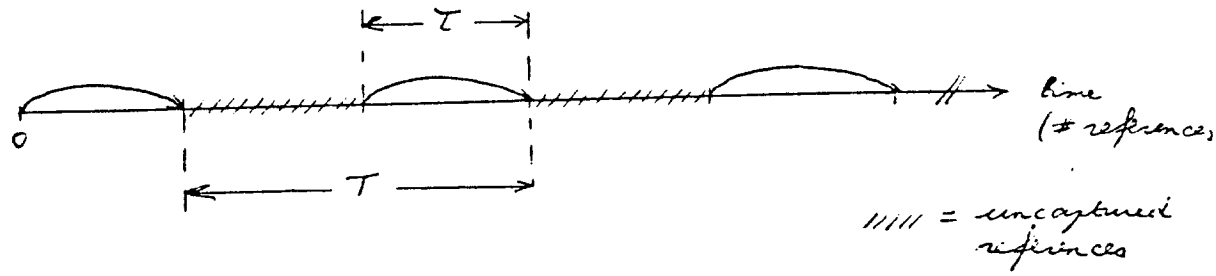

Figure 2. No overlap between measured reference sets when $T > \tau$.

T and τ will be input parameters to the extraction program used to determine the values of *arrivals* and *departures*, hereafter referred to as a and d. We repeat here the definitions of these two variables as given in the introduction.

Arrivals is the number of pages referenced during a certain interval, which are not contained in the previous working set. To denote a specific interval, a is subscripted: a_n means the number of arrivals during the n -th interval, which is the interval between references $(n-1)T$ and nT.

Departures denotes the number of pages in the previous working set which are not referenced during the present interval. Whereas arrivals can occur at any moment during the interval, departures take place at the end, for it is only

at that time that one is sure the page has not been referenced. Analogously to arrivals, the symbol d is subscripted: d_n means the number of departures at the end of the n-th interval, which coincides with reference nT.

The working set size at the end of the n-th interval is denoted by w_n.

A second subscript can be used to denote the window size, if necessary. This is shown further on in this section.

Figure 3 shows how the working set size fluctuates in time.

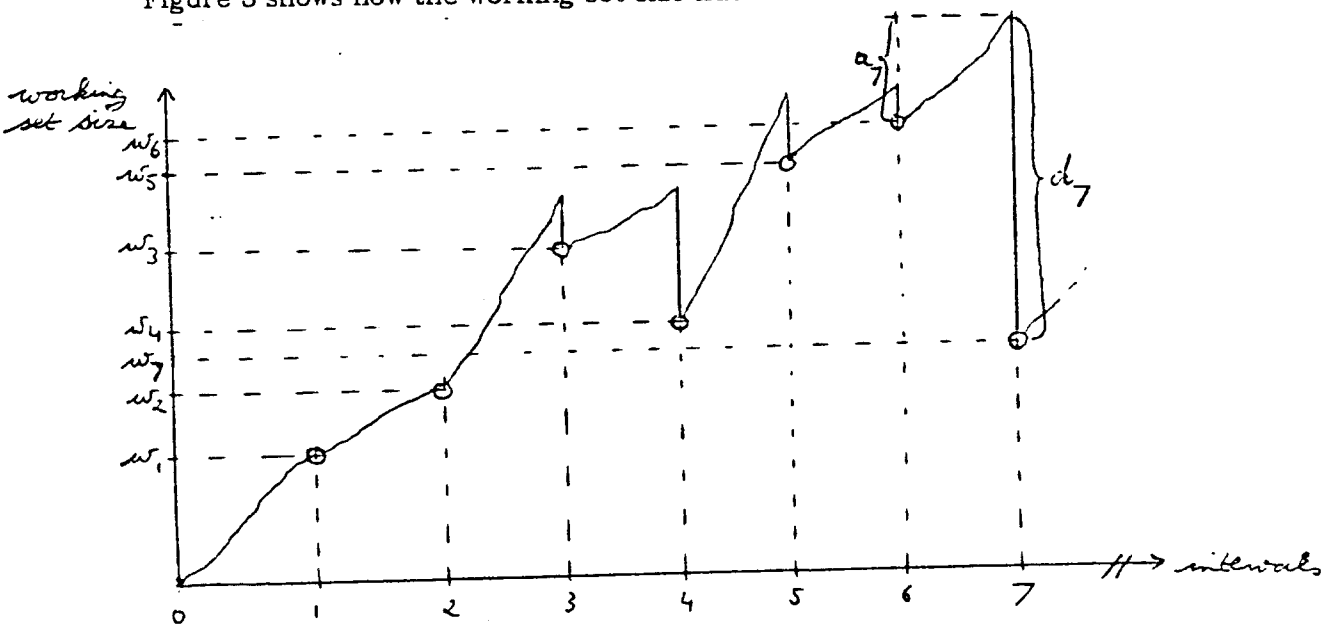

Figure 3. Working set size fluctuations

It should be kept in mind that only the encircled points of the curve are part of our model. These are the points that coincide with the end points of an interval.

Within an interval, the curve is monotonically non-decreasing, i.e., arrivals are added as they occur, while departures only take place at the end of an interval.

Assuming that the working set at time 0 is empty ($w_0=0$), the following observations can be made:

(a) there are no departures at the end of the first interval: all pages referenced are new, and no pages were referenced before, i.e., $d_1=0$.

(b) the drops in the curve, or the number of departures, can never exceed the size of the previous working set: if not, more pages would disappear than are available, leading to negative values; hence, $d_n \leq w_{n-1}$.

Figure 4 visualizes our definitions of the parameters T and τ and of the variables a and d.

Thus the following equation holds:

$$w_{n,\tau}=w_{n-1,\tau}+a_{n,\tau}-d_{n,\tau}$$

for all integer n, such that $0 \leq n \leq \frac{L}{T}$, with initial condition $w_{0,\tau}=0$. L denotes the total length of the string; in our experiments $L=500{,}000$.

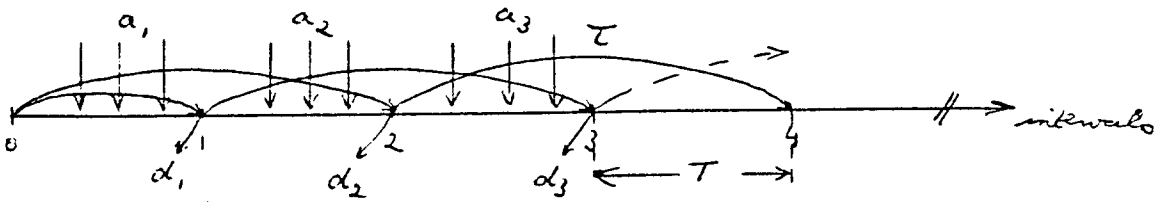

Figure 4. Visualization of arrivals and departures

The notation used here is as follows: the first subscript stands for the number of the interval, the second subscript denotes window size τ, i.e. the value of τ chosen in this case. Usually τ can be omitted, unless one wants to distinguish between various values of τ, as is the case in the next section.

Thus, the previous equation can be rewritten as:

$$w_n = w_{n-1} + a_n - d_n$$

with $w_0=0$.

This means that, whenever the values of any pair of variables are given, i.e., a and d, a and w, d and w, the values of the third variable, w, d and a respectively, can easily be derived.

We mentioned before that we favor multiples of 1000 as choices for τ and T. The reason is the following: for the discrete model to make sense, the minimum interval size chosen should not be too small; on the other hand, one has to allow for a fine enough granularity to capture the fluctuating behavior of the working set size. Hence the choice of 1000 (as a minimum).

An important advantage is that, by considering references in blocks of 1000 instead of one by one, we only have to store the identities of the pages referenced within that block, and, for profile extraction purposes, their frequency of occurrence, and *not* the *order* in which they occur. Thus, instead of having to keep on hand a sequence of 500,000 numbers, 500 records can be stored, which look as follows:

record # page-id1 frequency1 page-id2 frequency2 ... page-id*n* frequency*n*

As each interval contains 1000 references, the sum of the "frequency counts" within an interval will be equal to 1000.

After having converted the original string to this reduced or compressed string, we can get rid of the former, saving over 95 % in disk space.

As in our discrete measurement model we do not keep track of the order in which events occur *between* two measurement points, no information is lost, as long as we are satisfied with multiples of 1000 as values for T and τ. Of course, if one so desires, nothing prevents one from using the same compression technique for values other than 1000.

Compression methods are very helpful in trace driven simulations. In [Smit77] some of them are described in detail.

3. THEORY

The following theorems can be used to obtain the values of parameters w, a and d, for characterizations different from the original one.

The following equation was derived in the previous section:

$$w_{n,\tau}=w_{n-1,\tau}+a_{n,\tau}-d_{n,\tau} \tag{0}$$

for any $n>0$.

THEOREM 1

The working set size at time n with window size τ is equal to the working set size at time $n-1$ with window size $\tau-1$, plus the number of arrivals between times $n-1$ and n with window size $\tau-1$:

$$w_{n,\tau}=w_{n-1,\tau-1}+a_{n,\tau-1} \tag{1}$$

Proof

From figure 5 it is clear that $w_{n,\tau}$ covers all pages in $w_{n-1,\tau-1}$ plus the ones that arrive between $n-1$ and n, when the window size is $\tau-1$. Q.E.D.

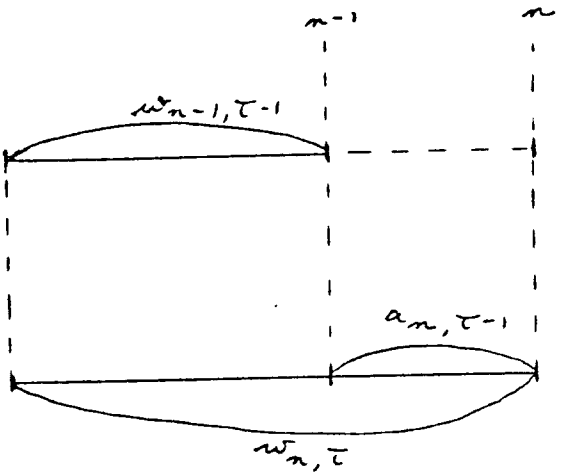

Figure 5. Proof of theorem 1

THEOREM 2

The working set size at time n with window size τ is equal to the working set size at time n with window size $\tau-1$, plus the number of departures at time n with window size $\tau-1$:

$$w_{n,\tau}=w_{n,\tau-1}+d_{n,\tau-1} \tag{2}$$

Proof

From (0) and (1) we obtain

$$w_{n,\tau}=w_{n-1,\tau}+a_{n,\tau}-d_{n,\tau}$$

$$w_{n,\tau}=w_{n,\tau+1}-d_{n,\tau} \quad (1)$$

$$w_{n,\tau+1}=w_{n,\tau}+d_{n,\tau}$$

$$w_{n,\tau}=w_{n,\tau-1}+d_{n,\tau-1} \quad Q.E.D.$$

THEOREM 3

$$a_{n,\tau}-d_{n,\tau}=a_{n,\tau-1}-d_{n-1,\tau-1} \tag{3}$$

Proof

From (0) we obtain

$$a_{n,\tau}-d_{n,\tau}=w_{n,\tau}-w_{n-1,\tau}$$

from (1) and (2)

$$a_{n,\tau}-d_{n,\tau}=w_{n-1,\tau-1}+a_{n,\tau-1}-w_{n-1,\tau-1}-d_{n-1,\tau-1}$$

$$a_{n,\tau}-d_{n,\tau}=a_{n,\tau-1}-d_{n-1,\tau-1} \quad Q.E.D.$$

THEOREM 4

The number of arrivals between $n-1$ and n with window size τ is less than or equal to the number of arrivals between $n-1$ and n with window size $\tau-1$:

$$a_{n,\tau}\leq a_{n,\tau-1} \tag{4}$$

Proof

The working set at time $n-1$ with window size τ includes all pages which are in the working set at the same time, with window size $\tau-1$. Therefore, all new pages at time n with window size τ will also be new pages for a window size equal to $\tau-1$. Hence the number of arrivals with window size $\tau-1$ is at least as large as the number of arrivals with window size τ. Q.E.D.

THEOREM 5

The number of departures at time n with window size τ is less than or equal to the number of departures at time $n-1$ with window size $\tau-1$:

$$d_{n,\tau}\leq d_{n-1,\tau-1} \tag{5}$$

Proof

From (3):

$$a_{n,\tau}-a_{n,\tau-1}=d_{n,\tau}-d_{n-1,\tau-1}$$

The left-hand side is less than or equal to 0 (from (4)), and so is the right-hand side. Q.E.D.

COROLLARY 1

If the number of arrivals between $n-1$ and n with window size $\tau-1$ is equal to 0, then the number of arrivals between $n-1$ and n with window size τ is equal to 0. Also, the number of departures at time n with window size τ is then equal to the number of departures at time $n-1$ with window size $\tau-1$:

$$if\ (a_{n,\tau-1}=0)\ then\ (a_{n,\tau}=0) \tag{6a}$$

$$and\ (d_{n,\tau}=d_{n-1,\tau-1}) \tag{6b}$$

Proof

The first part follows from (4); the second part is obtained by substituting (6a) into (3). Q.E.D.

COROLLARY 2

If the number of departures at time $n-1$ with window size $\tau-1$ is equal to 0, then the number of departures at time n with window size τ is equal to 0. Also, the number of arrivals between $n-1$ and n with window size τ is then equal to the number of arrivals between $n-1$ and n with window size $\tau-1$:

$$if\ (d_{n-1,\tau-1}=0)\ then\ (d_{n,\tau}=0) \tag{7a}$$

$$and\ (a_{n,\tau}=a_{n,\tau-1}) \tag{7b}$$

Proof

The first part follows from (5); the second part is obtained by substituting (7a) into (3). Q.E.D.

It is obvious that results (4) through (7) can be applied iteratively, i.e.:

$$a_{n,\tau+\Delta}\leq a_{n,\tau-1} \text{ for } \Delta\geq 0 \tag{8}$$

$$d_{n+\Delta,\tau+\Delta}\leq d_{n-1,\tau-1} \text{ for } \Delta\geq 0 \tag{9}$$

$$if\ (a_{n,\tau-1}=0)\ then\ (a_{n,\tau+\Delta}=0) \tag{10a}$$

$$and\ (d_{n+\Delta,\tau+\Delta}=d_{n-1,\tau-1}) \text{ for } \Delta\geq 0 \tag{10b}$$

$$if\ (d_{n-1,\tau-1}=0)\ then\ (d_{n+\Delta,\tau+\Delta}=0) \tag{11a}$$

$$and\ (a_{n,\tau+\Delta}=a_{n,\tau-1}) \text{ for } \Delta\geq 0 \tag{11b}$$

Applying these theorems can save us time when trying to extract statistics about arrivals and departures for characterizations different from the original one. Consider for example how one could derive the values of w, a and d values from a string with $T = 1$ and $\tau = 3$ when the string has been deleted and only the values of a and d are known for $T = 1$ and $\tau = 2$; in other words, the only information available is n pairs of arrival and departure values. Empirically, we have verified that most intervals contain either no arrivals or no departures, sometimes neither. The probability of $a = 0$, or $d = 0$ grows for larger values of τ, indicating less fluctuations in the working set when larger windows are compared to each other. By applying corollaries (6) and (7), we were able to reconstruct 73 % of the values of a and 73 % of the values of d. As shown later, this is sufficient to approximate very closely the real arrival and departure distributions for $T = 1$ and $\tau = 3$. It is clear that this procedure, which only takes a couple of seconds of VAX-11/780 CPU time, compares favorably to having to extract the same statistics from the real trace.

The following table shows the percentages of the values of a and d that can be derived from our string from the measured ($T = 1$, $\tau = 2$) characterization for ($T = 1$, $\tau = 3$);($T = 1$, $\tau = 4$) and ($T = 1$,$\tau = 5$) characterizations.

T=1,τ=2	T=1,τ=3	T=1,τ=4	T=1,τ=5
a=100%	73%	68%	61%
d=100%	73%	67%	59%

A similar table for ($T = 1, \tau = 11$);($T = 1, \tau = 12$) and ($T = 1, \tau = 13$), derived from ($T = 1, \tau = 10$):

T=1,τ=10	T=1,τ=11	T=1,τ=12	T=1,τ=13
a=100%	91%	86%	80%
d=100%	91%	84%	79%

A special case where these theorems can be applied is that of the continuous characterization of a string, where the time between two consecutive measurements is just one reference, i.e., $T = 1$. Of course, τ can still take on any positive value. In the continuous case, a large majority of arrivals and departures values are 0. As mentioned in [Lee82a], the working set size changes only 1157, 619 and 525 times for τ values of 5000, 10000 and 20000 respectively, and flat-faults, i.e., instances where a page arrival and departure coincide, leaving the working set size unchanged, occur very rarely.

The following example shows the derivation of arrivals and departures for τ = 1001, 1002 and 1003, from the ($T = 1, \tau = 1000$) characterization:

a = 1 0 0 1 1 0 0 1 0 0 1 0 0 0 1
d = 0 0 1 0 0 0 0 1 1 1 0 1 0 1 1 for τ=1000

a = 1 0 0 ? 1 0 0 1 0 0 ? 0 0 0 ?
d = 0 0 0 ? 0 0 0 0 1 1 ? 0 1 0 ? for τ=1001

a = 1 0 0 ? ? 0 0 1 0 0 ? 0 0 0 ?
d = 0 0 0 0 ? 0 0 0 0 1 ? ? 0 1 ? for τ=1002

a = 1 0 0 ? ? 0 0 1 0 0 ? 0 0 0 ?
d = 0 0 0 0 0 ? 0 0 0 0 ? ? ? 0 ? for τ=1003

In this table there are a few gaps, mainly as a result of ($a = 0, d = 1$),($a = 1, d = 0$) successions. It is easy to see why this pattern causes problems: a departure, immediately followed by an arrival, when the window size is τ, can lead to one of two alternatives for window size τ+1: if the departing page is different from the arriving page with window size τ, the result with window size $\tau + 1$ will be a flat-fault ($a = 1, d = 1$); if on the other hand both pages are the same, no arrival or departure will take place with window size $\tau + 1$ ($a = 0, d = 0$).

This example is unrealistic in that it contains a high percentage of working set size changes. It is, however, intended to show the applicability of the theorems introduced above. In practice, characterizations for τ = 1001, 1002 and 1003 can be derived with over 98 % accuracy (i.e., 98 % of the arrival and departure values can be extracted from the τ = 1000 characterization).

4. EXTRACTION OF ARRIVAL AND DEPARTURE STATISTICS FROM THE ORIGINAL STRING

4.1. Compression of the string

The original string with 500,000 references, generated during the execution of an APL program on an IBM 360/91, was compressed using the technique described above: for each group of 1000 references, only the identity of the pages and their frequency of occurrence were retained. This method saves a substantial amount of storage space. The disadvantage is the loss of the intra-interval order of reference, but, due to the nature of our discrete model, this information would be irrelevant anyway.

4.2. Determining arrivals and departures from one window to the next

The way the extraction program proceeds is as follows. The pages referenced in a certain interval are stored as linearly linked lists. A window, being a succession of intervals, is represented by a one-dimensional array, the elements of which are the aforementioned linked lists. To determine the working set transition, defined as the change in pages in the working set from one window to the next, one has to count the pages referenced during the second window but not during the first (arrivals) and the pages referenced during the first window but not during the second (departures). It is clear from figure 6 that only pages referenced during the non-overlapping parts of the windows have to be considered as potential arrivals or departures.

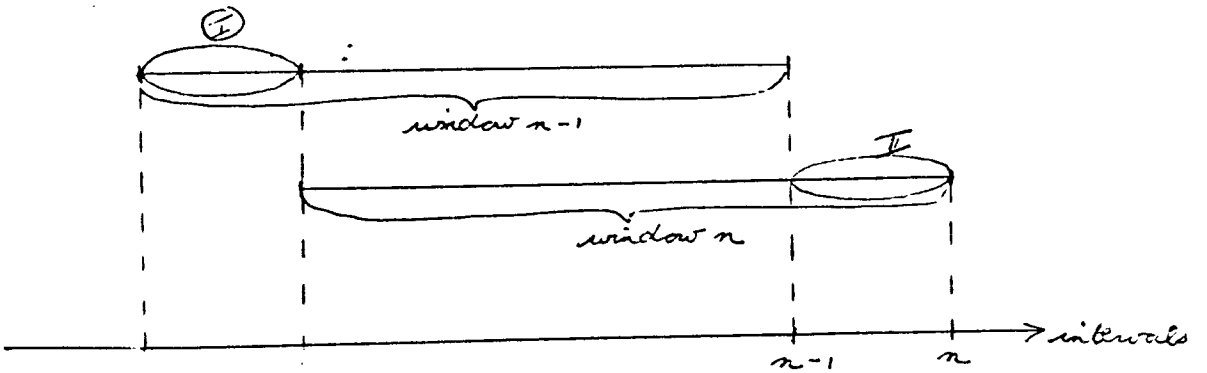

Figure 6. Determining arrivals and departures

Only pages in Ⓘ, i.e., pages referenced before the beginning of the second window, are eligible for departure at the end of the n -th interval, while only pages in Ⓘ i.e., pages referenced after the end of the first window, can be arrivals during the n -th interval. All pages referenced in between are common to both windows and cannot therefore be arrivals or departures.

In practice, the extraction program, when determining the number of arrivals, compares the linked lists in Ⓘ with the linked lists composing the window ending at the end of interval $n-1$. When counting the number of departures, the linked lists in Ⓘ are compared with the linked lists in the window ending at the end of interval n. In figure 7, an example is given for $T = 2$ and $\tau = 5$ (measurements are made every 2000 references, and the window size is 5000

references).

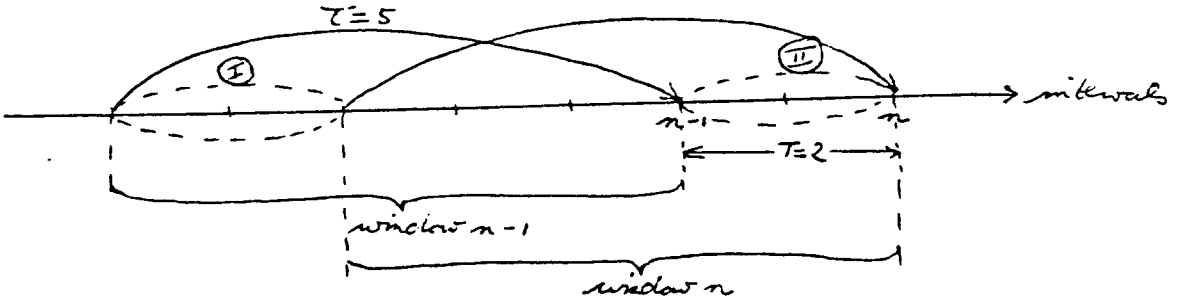

Figure 7. Comparison of two consecutive windows

The arrivals are determined by comparing the two linked lists composing the n -th interval with the five linked lists composing the window ending at the end of interval $n-1$. Departures are counted in a similar way.

4.3. Extracting the program profile

The program profile gives, for each page referenced, its relative frequency of occurrence. By keeping track of the number of occurrences within each interval (1000 references), it is easy to reconstruct the profile. One just has to sum up the numbers of a page's occurrences over all these intervals. These numbers are then used to obtain a cumulative frequency distribution, which is the basis of the artificial trace generation process described in the next section.

4.4. Arrivals distribution

After having obtained the number of arrivals for each interval, a frequency distribution is derived, which shows, for each number observed, how many intervals contain that number of arrivals. The observed numbers of arrivals can thus be used as inputs to the string generation program.

It should be noted that, unlike the program profile, this distribution is not unique, and depends both on the interval size T and on the window size τ.

4.5. Departures distribution

The extraction of departure distributions is essentially the same as for arrivals.

4.6. Working set size distribution

To obtain this distribution, we have to determine the working set size at the end of each interval. This is easily done by observing that the original working set is empty, and by just adding the number of arrivals to, and subtracting the number of departures from, the previous working set size.

4.7. Derivation of arrival and departure distributions for related characterizations

We stated above that under certain conditions our theorems can be used to derive arrival and departure statistics for related characterizations. The results obtained in this way can be used to build frequency distributions, which can be compared to the distributions obtained directly from the real trace.

Only a and d values which can be derived are incorporated in the distribution.

This leads to the conclusion that in some cases the time-consuming procedure of comparing linked lists to obtain arrivals and departures can be circumvented. Of course, the theorems can only be applied when their very strict conditions are satisfied.

5. GENERATING ARTIFICIAL STRINGS

The validity of a model can only be verified by using it and comparing the results with direct observations of the phenomenon we are trying to model.

In our particular case, this means generating artificial reference strings and comparing their behavior and characteristics to those of the original string.

The properties of a good model can be summarized as follows: its parameters should be easy to extract and as few as possible, it should accurately capture the phenomenon to be analyzed, and it should be capable of reproducing that phenomenon as closely as possible.

In the second section we have given a description of our model, which is essentially stochastic in nature.

The information needed to build the model, i.e., the program profile and the arrivals and departures distributions, is easily obtained, and does not require large amounts of storage space. As to the other two criteria above, the quality of the model can only be verified after artificial strings have been generated. This is done in the next section.

5.1. Distributions used in the string generation process

5.1.1. The program profile

It is necessary to take program profile information into account when generating new strings, as it will assist us in filling the intervals with the right "mix" of pages, resembling the original one. The program profile will play a role both in the determination of *which* pages to reference within an interval and *how often* these pages are to be referenced.

5.1.2. Arrivals, departures, and working set size distributions

We proved earlier that any of these distributions can be derived from the other two. While this could indicate that it does not make a difference which ones we choose, we observe that the generation algorithm's stability increases if the working set size distribution is used, together with either of the other two distributions. The reason is the following: if the arrivals and departures distributions are specified, the initial random choices for the numbers of arrivals and departures will have too great an effect on the working set sizes over the remaining intervals. For example, an unusually high value for a and a low value for d will bring the working set size in too high an "orbit".

The use of the working set size distribution as a given protects against this phenomenon by keeping the generated working set sizes closer to the ones

observed in the original string.

The generation algorithm described below uses the departures distribution and the working set size distribution. The reason for choosing the departures distribution is explained below.

5.2. The generation process

Using these distributions, we now have the basic tools to generate artificial strings. However, a number of precautions need to be taken to ensure that the generated strings resemble the original one closely enough.

5.2.1. Starting conditions

While in general each interval contains both arrivals and departures, the initial intervals, i.e., the ones numbered from 1 through $\frac{\tau}{T}$, only contain arrivals. Therefore, one has to make sure that in these intervals the working set size grows, and that no departures are allowed.

5.2.2. Smoothing

In the continuous case, the working set size fluctuates smoothly, i.e., only +1 and -1 changes are observed. In our discrete model, the measured working set sizes fluctuate with larger amplitudes. The strings to be generated, however, will be measured continuously.

If the discrete model is applied without special precautions, arriving pages suddenly emerge in a certain interval, possibly causing a drastic change in the working set, and similarly departing pages suddenly disappear after a certain interval. It is this suddenness which leads us to suggest the following measures:

(1) newly arriving pages are only referenced after the first half of each interval;

(2) pages to depart are only referenced during the first half of each interval.

One can of course imagine more sophisticated smoothing schemes, but in order to keep the model simple yet powerful, the above scheme was preferred.

Figure 8 shows what could happen if smoothing were not applied. Newly arriving pages will probably have their first reference near the beginning of the interval. The reason is that even with a relatively low reference probability, say 1 %, the first reference occurs on the average at the latest after 100 references. Hence, there is a concentration of arrivals at the beginning of the interval. Similarly, pages with a low reference probability are likely to have their last reference close to the end of the last interval before their disappearance. Hence the sudden drop at the end of the interval under consideration.

The above is of course only true when working set sizes are measured continuously.

It is clear from the picture that strings generated without the measures described above will exhibit drastically overstated working set sizes: the curve will rise from a value w_{n-1} to $w_{n-1}+a_n$, stay at that level until close to the end of the interval, and then drop to $w_{n-1}+a_n-d_n$, a value which is equal to w_n, the new working set size. We discovered this in earlier experiments when we were not yet aware of this phenomenon.

By applying the measures that we suggested above, the initial part of the working set size curve in a given interval will generally be flat: new pages are not yet allowed, and the pages which will have to disappear from the working set at the end of this interval are still in the working set. The reason for this was explained above. Near the middle of the interval departing pages, i.e., pages only

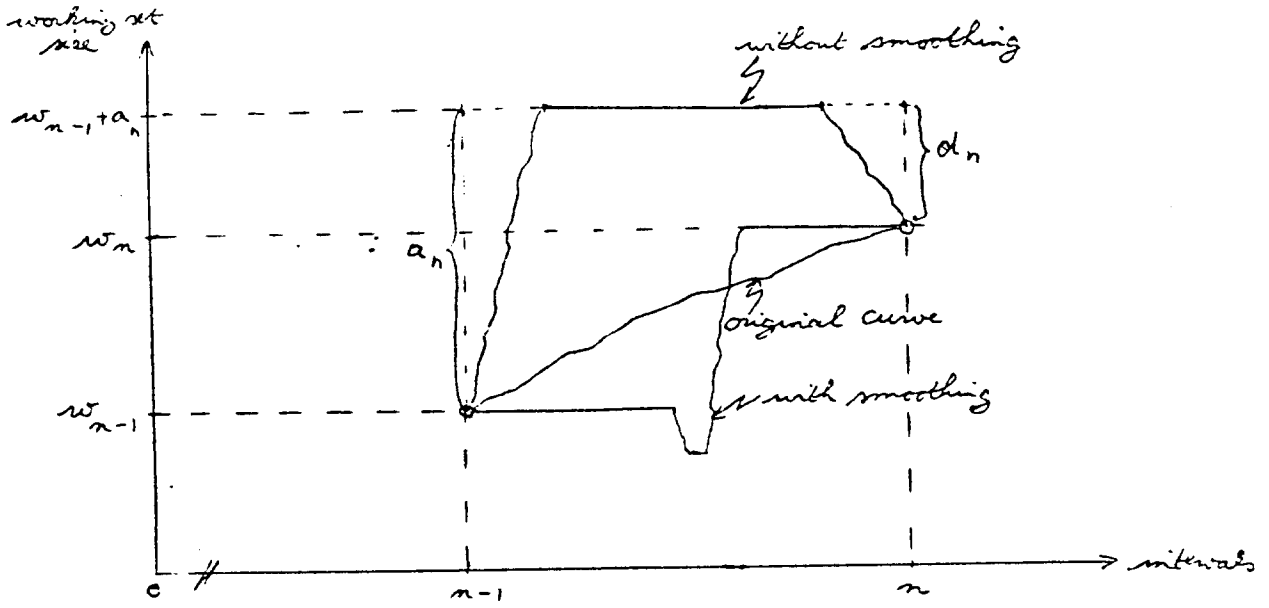

Figure 8. The effect of smoothing

referenced until the middle of the $n-\tau$-th interval, cause a drop in the curve. This drop is reversed by newly arriving pages which have their first reference after the first half of the interval. Overall, the average working set size over the interval is much closer to the one we observed for the original string.

5.2.3. Ensuring that each page is referenced

As can be observed from the program profile, some pages have an extremely low reference probability. To closely resemble the characteristics of the original string, even these pages have to be referenced. Therefore, the generation algorithm will "force" them to be referenced.

This is done in two ways:

(1) for each page, an interval is chosen at random in which the page has to be referenced if it has never been referenced before; and

(2) in each interval, it is made sure that all pages in the generated working set are referenced at least once.

5.3. The string generation algorithm

This section describes how the profile and the working set size and departures distributions are used to generate references.

For each interval, the basic procedure is as follows:

(1) determine how many of the pages referenced during the present interval n will not be referenced during the next interval; this value is equal to $d_{n,\tau}$. This step explains why we need the departures distribution: in order to cause pages to depart at the end of the n -th interval, we have to mark them for deletion intervals in advance; hence we need the departures distribution to look τ intervals ahead;

(2) determine the size $w_{n+1,\tau}$ of the next working set size, under the constraint $w_{n+1,\tau} \geq d_{n+2,\tau}$; note that the value of $d_{n+2,\tau}$ is obtained as described in (1);

(3) knowing $w_{n,\tau}$, $w_{n+1,\tau}$ and $d_{n+1,\tau}$, derive $a_{n+1,\tau}$;

(4) choose $a_{n+1,\tau}$ new pages, using the procedure described below;

(5) choose $d_{n+\tau,\tau}$ pages not to be referenced again during the next τ intervals, also as described below.

To choose pages to be added to the working set, we look at the set of eligible pages, i.e., the ones that are not in the present working set and that are not "banned" by being marked as pages to depart at the end of a future interval. Then, the program profile is invoked to select the new pages.

To choose the pages not to be referenced again, we first have to determine how many there will be by sampling from the departures distribution. To determine which pages to choose, the complement of the program profile distribution is applied; hence pages with a smaller reference probability are more likely to be chosen.

Once we have determined the set of pages for the new interval, the program profile is once again invoked to generate new references, of course under the above described constraints for arriving and departing pages.

Using the program profile during the generation process ensures that the generated strings will have profiles similar to the one of the original string.

6. RESULTS

Artificial strings were generated for the following (T,τ) pairs: (1,5), (1,10), (2,5), (5,10) and (10,20).

It took between 634 and 867 seconds of VAX-11/780 CPU time to generate these five strings using the program profile, the departures distribution , and the algorithm described above.

The strings for which $T=1$ took the most time to generate. The reason is the following: for each interval, a new set of pages needs to be generated; for $T=1$ this is done 500 times, while $T=10$ only requires 50 such computations.

It is guaranteed by our algorithm that every page is referenced. Therefore the number of pages referenced in each of the five cases is equal to 110.

Another characteristic which gives a good indication of how well the generated strings resemble the original one is the *coefficient of resilience*, which indicates the probability of two consecutive references to the same page. Its value is 0.544 for the original string. The following table gives the coefficient of resilience for the generated strings:

string(T,τ)	coefficient of resilience
(1,5)	0.540
(1,10)	0.539
(2,5)	0.540
(5,10)	0.542
(10,20)	0.543

Two paging policies were applied to the artificial strings: WS and LRU. The statistics gathered are: *mean working set size*, *maximum working set size*, *changes of slope*, *page fault rate* and *maximum interfault time*.

6.1. Working set policy

The following table shows the values of the above statistics for the original trace, executed under the working set policy with τ = 5000, 10000 and 20000. These numbers were obtained from [Lee82a]:

Statistics for the original string					
τ=	mean wss	max wss	slope changes	fault rate	max interfault time
5000	16.9	45	1157	0.00118	32092
10000	20.9	56	619	0.000648	111695
20000	26.17	78	525	0.000562	111695

The numbers for the artificial strings generated by our model are given below:

Statistics for generated strings					
string(T,τ)	mean wss	max wss	slope changes	fault rate	max interfault time
(1,5)	16.9	44	1172	0.00123	60033
(1,10)	19.7	56	632	0.000663	111695
(2,5)	17.0	44	1180	0.00120	63255
(5,10)	21.0	56	627	0.000661	111695
(10,20)	26.7	78	531	0.000570	111695

The results are close to those for the original string for the same τ, mainly because the working set size is the critical factor in our model, and also because of the precautions described in the previous section. It is intuitively clear that a model based upon working set size characteristics will perform best under the working set policy.

6.2. LRU policy

The values obtained from the real trace, and reported in [Lee82a], are given below. The number of page frames is LRU's counterpart of WS's window size τ, and is in our experiments equal to 31. The reason for choosing this value is the fact that, with 31 page frames available, the page fault rate is equal to the one observed under the WS policy for τ = 10000.

Statistics for the original string with 31 frames	
page fault rate	max interfault time
.00146	111416

The results for the artificial strings are given below:

Statistics for the artificial strings with 31 frames		
string(T,τ)	page fault rate	max interfault time
(1,5)	0.142	112314
(1,10)	0.072	112211
(2,5)	0.138	113212
(5,10)	0.071	112375
(10,20)	0.032	111843

As could be expected, the results are rather poor: the generated strings by no means exhibit the behavior observed in the original one. Hence, they cannot be used to perform experiments on the performance of the LRU policy. Similar results are reported in [Lee82a]. The reason is that the order of references, crucial in LRU, is not reproduced by the discrete model, or any model which does not deal with page identities. The page fault rate under the LRU policy is extremely sensitive to the order in which pages are referenced.

7. CONCLUSIONS

We have presented a model based upon a discrete characterization of the behavior of a program in a virtual memory environment.

First, we have defined the variables of the model, and shown how their values can be obtained from the program's reference string for specific values of T, the interval size and τ, the window size. The statistics of interest in our model are the *arrivals* and *departures* of pages, as measured with respect to two consecutive intervals.

We have then shown how certain theoretical results can be used to assist us in this measurement procedure.

The next step has been the description of the procedure that can be used to generate artificial strings, whose performance indices under paging policies were then compared to those of the original string.

From these comparisons, the following conclusions can be drawn: the generated strings approximate rather closely the original one when analyzed under the working set policy. However, the same strings perform poorly when executed under the LRU policy. The reason is that both policies are in sharp contrast to each other: the working set concept basically acknowledges phase transitions during the execution of a program, while the LRU policy is based on the assumption of stationary reference patterns.

Our model tries to combine the advantage of a working set based model, i.e., the fact that it can capture program dynamics, with the advantages of discrete models, i.e., their more compact characterization of the phenomenon to be modeled.

ACKNOWLEDGEMENT

The author wishes to thank his research advisor, Professor Ferrari, for his continuous guidance and support.

Discussions with various members of the PROGRES group were very enlightening.

Tzong-yu Paul Lee deserves special thanks for his encouragement and advice.

Professor Alan Smith provided valuable assistance by reviewing earlier drafts of this report.

BIBLIOGRAPHY

[Denn68]
P.J. Denning, "The Working Set Model of Program Behavior," Communications of the ACM, Vol. 11, No. 5, May 1968, pp. 323-333

[Denn72]
P.J. Denning and S.C. Schwartz, "Properties of the Working Set Model," Communications of the ACM, Vol. 15, No.3, March 1972, pp. 191-198

[Dutt81]
C.R. Dutt, "Dynamic Characterization and Reproduction of Memory Consumption with Working-Set-Based Generative Model," Master's Report, University of California, Berkeley, August 1981

[Ferr76]
D. Ferrari, "The Improvement of Program Behavior," Computer 9, November 1976, pp. 39-47

[Ferr81a]
D. Ferrari, "Characterization and Reproduction of the Referencing Dynamics of Programs," Performance '81, F. Kylstra, Ed., North-Holland, Amsterdam, November 1981, pp. 363-372

[Ferr81b]
D. Ferrari, "A Generative Model of Working Set Dynamics," Proceedings of the Sigmetrics Conference on Measurement and Modeling of Computer Systems, 1981, pp. 52-57

[Lee82a]
T.P. Lee, "Experimental Design of a Generative Model Based on Working Set Size Characterizations," Master's Report, University of California, Berkeley, June 1982

[Lee82b]
T.P. Lee, "Properties of Flat-Faults in Working Set Model," PROGRES Report, Computer Science Division, University of California, Berkeley, 1982

[Smit76]
A.J. Smith, "A Modified Working Set Paging Algorithm," IEEE Transactions on Computers, C-25, September 1976, pp. 907-914

[Smit77]
A.J. Smith, "Two simple methods for the Efficient Analysis of Memory Address Trace Data," IEEE Transactions on Software Engineering, SE-3, January 1977, pp. 94-101