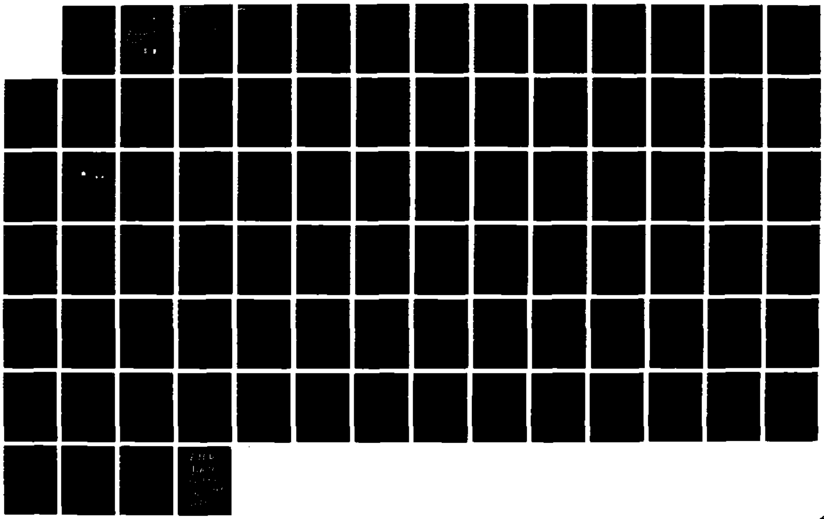
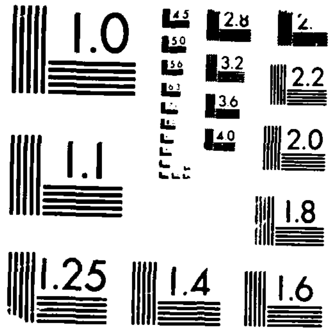
AD-A192 432 ACCURATE LOW-COST METHODS FOR PERFORMANCE EVALUATION OF CACHE MEMORY SYSTEMS(U) ILLINOIS UNIV AT URBANA COLL OF ENGINEERING L SUBHASIS FEB 88 CSG-83 N00014-84-C-0149 1/1

UNCLASSIFIED F/G 12/6 NL





MICROCOPY RESOLUTION TEST CHART

JREAU TANDARDS 196

February 1988

UILU-ENG-88-2212
CSG-83

COORDINATED SCIENCE LABORATORY
College of Engineering

2

DTIC FILE COPY

ACCURATE LOW-COST METHODS FOR PERFORMANCE EVALUATION OF CACHE MEMORY SYSTEMS

Subhasis Laha

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS None
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) (CSG-83) UILU-ENG-88-2212	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Coordinated Science Lab University of Illinois 6b. OFFICE SYMBOL (If applicable) N/A	7a. NAME OF MONITORING ORGANIZATION Office of Naval Research
6c. ADDRESS (City, State, and ZIP Code) 1101 W. Springfield Ave. Urbana, IL 61801	7b. ADDRESS (City, State, and ZIP Code) 800 N. Quincy St. Arlington, VA 22217
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Joint Services Electronics Program 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER N00014-84-C-0149
8c. ADDRESS (City, State, and ZIP Code) 800 N. Quincy St. Arlington, VA 22217	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)
Accurate Low-Cost Methods For Performance Evaluation of Cache Memory Systems

12. PERSONAL AUTHOR(S)
Subhasis Laha

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Technical	FROM ____ TO ____	1988 February	84

16. SUPPLEMENTARY NOTATION
None

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	cache memory systems, simulation, performance evaluation, split cache systems

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

Trace-driven simulation is a simple way of evaluating cache memory systems with varying hardware parameters. But to evaluate realistic workloads, simulating even a few million addresses is not adequate and such large scale simulation is impractical from the consideration of space and time requirements. In this work, new methods of simulation based on statistical techniques are proposed for decreasing the need for large trace measurements and for predicting true program behavior. In our method, sampling techniques are applied while collecting the address trace from a workload. This drastically reduces the

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT [✓ UNCLASSIFIED/UNLIMITED] ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE (Include Area Code) / 22c. OFFICE SYMBOL

DD FORM 1473, 84 MAR — 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

UNCLASSIFIED

space and time needed to collect the trace. New simulation techniques are developed to use the sampled data not only to predict the mean miss rate of the cache, but also to provide an empirical estimate of its actual distribution. A model is proposed to statistically project the results to different context-switch intervals from only one simulation of a small number of samples of a fixed size. A new concept of primed cache is introduced to simulate large caches by the sampling-based method. Finally, a cache model is developed to study the performance of different split caches.

ACCURATE LOW-COST METHODS FOR PERFORMANCE EVALUATION OF CACHE MEMORY SYSTEMS

BY

SUBHASIS LAHA

B.Tech., Indian Institute of Technology, Kharagpur, 1981
M.S., University of Illinois, 1984

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 1988

Urbana, Illinois

(1

ACCURATE LOW-COST METHODS FOR PERFORMANCE EVALUATION OF CACHE MEMORY SYSTEMS

Subhasis Laha, Ph.D.
Department of Electrical and Computer Engineering
University of Illinois at Urbana-Champaign, 1988

Trace-driven simulation is a simple way of evaluating cache memory systems with varying hardware parameters. But to evaluate realistic workloads, simulating even a few million addresses is not adequate and such large scale simulation is impractical from the consideration of space and time requirements. In this work, new methods of simulation based on statistical techniques are proposed for decreasing the need for large trace measurements and for predicting true program behavior. In our method, sampling techniques are applied while collecting the address trace from a workload. This drastically reduces the space and time needed to collect the trace. New simulation techniques are developed to use the sampled data not only to predict the mean miss rate of the cache, but also to provide an empirical estimate of its actual distribution. A model is proposed to statistically project the results to different context-switch intervals from only one simulation of a small number of samples of a fixed size. A new concept of primed cache is introduced to simulate large caches by the sampling-based method. Finally, a cache model is developed to study the performance of different split caches.

ACKNOWLEDGMENTS

I would like to express my sincere appreciation and gratitude to my thesis advisor Professor Janak H. Patel. Professor Patel's guidance and helpful suggestions were invaluable contributions to this work. I would like to thank Professors Edward S. Davidson and Ravishankar K. Iyer for their encouragement, suggestions and comments. I would also like to thank Professors Benjamin Wah and Daniel Reed for their comments. Special thanks to Professor Prithviraj Banerjee for his friendship and constant encouragement.

I would like to thank my colleagues at the Computer Systems Group of the Coordinated Science Laboratory for their assistance. Their friendship has been a wonderful experience during my graduate study. Finally, my thanks to all the secretaries, past and present, of the Computer Systems Group, who have always been extremely helpful and cordial.

TABLE OF CONTENTS

Chapter Page

LIST OF TABLES

LIST OF FIGURES

workload behavior: start-up effects, nonstationary behavior (change of program locality), intrinsic interference (different memory references competing for a cache set) and extrinsic interference (effect of multiprogramming). These analytical methods are quite inexpensive in most cases and provide insight into the cache behavior; but they involve many simplifying assumptions and the results offer only bounds and rough estimates of the performance of different cache organizations.

1.1.3. Trace driven simulation

This leaves us with simulation which is simple and quite easy to develop. A program address trace consists of a sequence of the virtual (usually) addresses, referenced by a program or programs, along with other relevant information. Trace driven simulation is the process of running the simulation model of a system with the trace. A simulator can also be driven by a random number generator; but because of the lack of existence of an accurate behavioral model of programs in the context of the cache, such simulation can never represent realistic program behavior as indicated earlier. On the other hand, a trace represents the behavior of at least one program; thus, trace driven simulation will characterize that program correctly. Trace driven simulation can be used to evaluate any existing or proposed architecture. The parameters of the architecture can be varied very easily simply by changing the input parameters of a simulator. For all these reasons, trace driven simulation has been chosen by most people [2, 3] for evaluation of memory systems.

1.1.3.1. Limitations In spite of all these nice features, there are several limitations of trace driven simulation which make it inadequate not only for our purpose, but also as a tool of performance evaluation in general.

(1) It simulates only a very small part of a program. For example, if a program runs for a thousand seconds of CPU time at the rate of one million memory references per

second, simulating a trace of one million addresses will represent only 0.1% of the whole program.

(2) Traces are usually collected from the initial portion of a program, rather than from the core. This practice of taking a single section of a program and projecting it for the entire program is highly questionable. Smith [2,3] has reported that an enormous degree of variability exists among traces in their measured miss ratios in cache. Our own experience with simulation of cache memories has shown that there is a large degree of variation in program behavior between one sample of a half million references and another sample of the same program at a different time. Even if the mean miss rate, measured this way, happens to be the same as that of the whole program, this mean value is not adequate to accurately reflect the performance of the system. We will show that the distribution of this measured miss ratio is far from the real one. Such experience suggests that simulation with a section of program chosen on an ad hoc basis with a few thousand references cannot accurately reflect the overall program behavior.

(3) Collection of large contiguous traces of a running system is either extremely costly or almost impossible. Most hardware monitors have a limited buffer memory because of the high cost of extremely fast memories which must match the speed of the cache it is monitoring.

(4) Any large scale simulation with a few million addresses is also very expensive from the consideration of computer time. Each such simulation, with one set of hardware parameters as input, typically takes up several minutes of CPU time on a minicomputer. Thus, running a number of simulation jobs with several combinations of architectural parameters for a single trace is computationally very expensive.

Most machines switch context every few thousand instructions, especially in a multiprogramming environment. The occurrence of context switch may be due to I/O interrupts or time-out of a fixed time slice. This effect of context switch on cache performance has been studied by others [8, 10, 11, 12]. Our objective is to statistically project the results of our sampling technique to predict the miss ratio of all different context-switch intervals with very little extra work.

1.1.3.2. Trace compression techniques To reduce the space and time required for trace driven simulation, some researchers have proposed trace-tape compression schemes [13, 14] for memory address trace data. Puzak's [14] techniques are particularly aimed at efficient cache evaluation. Trace stripping, the first of his two reduction schemes, requires one pass through the original trace tape to remove all the MRU (most recently used) hits from the top of the LRU stack. The second tape reduction scheme, congruence class sampling, achieves the compression by sampling the reference of only some of the congruence classes (i.e., sets) constituting the cache. When these two reduction schemes are used together, a compression of almost two orders of magnitude can be achieved with a reasonable degree of accuracy in determining the cache performance. But these methods require processing the whole trace at the beginning. Also, varying some of the cache parameters, such as the block size, becomes restrictive, once the original trace tape is reduced using some particular cache parameters.

1.2. Goal

The goal of this research is to establish a new method of trace driven simulation for cache memory systems, based on sampling techniques. Joint usage of statistical techniques and the knowledge of cache analysis is utilized to develop this technique. This will decrease the need for measurement and simulation of large traces and will overcome other associated limitations. The collection of sampled traces of running systems is compatible with the existing low-cost hardware monitors. Furthermore, this method will not only allow a prediction of the *mean miss rate*, but will also provide an empirical estimate of the *actual distribution* of the miss rate. Unlike trace compression techniques, in our method it suffices to collect only the sampled parts of the trace, thus, reducing the time and space overhead drastically. Trace compression can be performed on the sampled trace to gain even further reduction of time and space. The issues of simulating all different types of caches, large and small, direct-mapped and set-associative, unified and split will be addressed in this thesis. Also, the effect of different context-switch intervals is considered.

1.3. Outline

In Chapter 2, the basic methodology of sampling for small caches is presented. The validity of such a technique is established experimentally with the help of statistical analysis. A subsequent cache model extends this result for all different context-switch intervals.

Chapter 3 examines the problem of large cache simulation and presents a new priming technique. This involves sampling the cache in addition to sampling the trace.

Chapter 4 concentrates on split caches. Methods developed for unified caches are not applicable to split caches. A new cache model is, therefore, developed. This model is shown to handle unified caches also.

Finally, Chapter 5 summarizes this research and describes the future scope of this work.

CHAPTER 2

SAMPLING TECHNIQUE FOR SMALL CACHE

In this chapter, the basic sampling technique for small caches is proposed. A cache model is provided to extend the result of one simulation to other context-switch intervals. In conventional simulation, the effect of task switches on a program can be estimated by assuming the cache to be flushed totally immediately after a context switch [10, 15]. This is essentially true for smaller caches. For larger caches, the miss ratio obtained in this way provides only a lower limit on the cache performance. The preliminary aim of our sampling technique is to provide a cost-effective approach to estimate the distribution of this cold-start miss ratio for small caches. The issue of large caches is considered later in Chapter 3.

2.1. Sampling Technique

Instead of simulating all consecutive intervals of address references, as is done conventionally, we sample the address trace (Fig. 2.1) and run the simulation only on the sampled data.

2.1.1. Our methodology

(1) Choose a sample size which is typically a few thousand consecutive addresses. In the initial study, this corresponds to the task interval considered.

(2) Depending on the size of the whole trace, determine the average sampling interval to attain a certain number of samples.

(3) Collect only the sampled parts of the trace, so that the beginning of each sample coincides with a context switch.

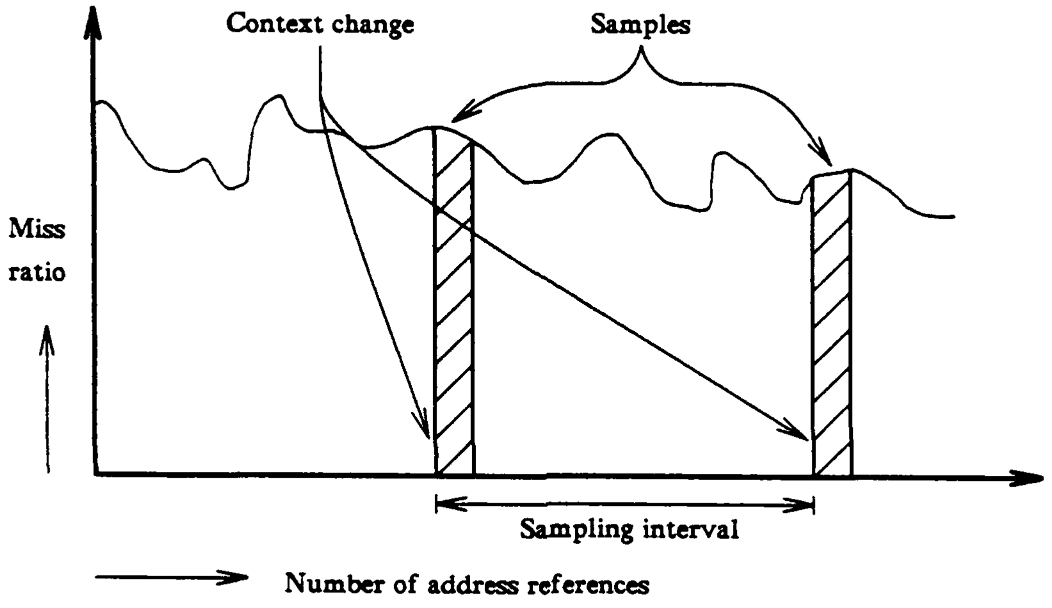


Fig. 2.1. Sampling technique

(4) Assuming the cache to be empty at the beginning of each sample, simulate the cache memory system for each sample.

We will determine the number of samples required for a wide variety of programs so that the distribution of the average miss ratio for the samples gives a reasonably accurate estimate of the cold-start miss ratio over the entire trace for *the same context switch interval as the sample size*.

2.1.2. Validation procedure

Simulating one sample of consecutive addresses represents the behavior of that particular context switch interval. Because of the cold-start, the miss rate is very high at the beginning of the interval and usually decreases as the cache fills. We consider the average value of the miss ratio at the end of the sample. To validate this methodology, it is essential to establish that the miss ratios of the samples truly represent the behavior of the continuous trace. Our objective is to determine the number of samples that is adequate to represent any program, irrespective of the nature of the program or the length of the trace. We will also show that the number of samples is the factor to be considered for sampling, rather than the sampling interval. Therefore, to validate this technique, different numbers of samples are considered for traces of different lengths from various programs, and statistical techniques are applied to compare the distribution of the sampled data with that of the actual distribution of the miss ratio.

2.2. Simulations Performed

In this section, specific details of the simulations performed are discussed.

2.2.1. Traces used

Since the cache measurements of LISP programs are few [3,16,17], LISP programs were employed in our case studies. The proposed methodology, however, is equally applicable to other programs. The traces of virtual addresses were collected by running the compiled code of *Franz LISP programs* as a child process in single-step mode on a VAX-11/780. These are continuous traces which were later sampled at various intervals and sizes and then simulated. The programs chosen are realistic workloads of varied nature. The nature of the programs and the lengths of the corresponding traces are listed below:

(1) FSIM : Fault Simulation program : Length $\simeq$ 2.9 million.

(2) DRC : Design Rule Check program for VLSI layout : Length $\simeq$ 5.7 million.

(3) ELI : English Language Interpreter program : Length $\simeq$ 1.8 million.

(4) RRL : Mathematical Rewrite Rule Lab program : Length $\simeq$ 2.0 million.

These traces are collected by running only part of these programs because of practical limitations. But they are of varied length and we will show that our methodology holds well, independent of this variation of trace length, so that this method can be applied to traces of entire programs also. Furthermore, the programs selected exhibit very different behavior; therefore, we contend that our methodology holds for a wide variety of programs.

2.2.2. Size and number of samples

Three different sample sizes are used: 5,000, 10,000 and 20,000 addresses. These sizes are chosen to be of the same order as the typical context switch interval on a VAX-11/780[4,5].

It is known that about 30 samples give a good estimate of any data that follows a Normal distribution [18]. At the beginning, it was assumed that the distribution of the cache miss ratio is normal, but later, it was found that this does not hold well in general.

Nonetheless, it will be shown that about 35 samples are always found to adequately predict the actual miss distribution with reasonable accuracy.

2.2.3. Architecture simulated

The architectural model has a three-level memory hierarchy: virtual, main and cache. We consider *real address cache*: the virtual address is first translated to a real address of the main memory which is subsequently mapped to the cache address space. The cache memory is *set-associative* and its blocks are replaced according to the *Least Recently Used* (LRU) policy. Cache is updated by *no-allocate* policy, i.e., during a write-miss, only the main memory is updated and the write is not counted as a cache miss [19].

The goal of this research, at this point, is to validate the new method of simulation which is applicable to all programs running on small caches. For that reason, typical small caches with fixed hardware parameters are simulated. The sizes chosen for the cache memory are 2 Kbytes, 4 Kbytes and 8 Kbytes (K = 1024). The results from caches of these different sizes are similar; hence, any one of them is representative of all the cases. In this thesis, the results of only the 4-Kbyte cache are included. The block size of the cache is kept fixed at 8 bytes and the set associativity is chosen as 2.

2.2.3.1. Approximation Both the instructions and the data in VAX-11 are of variable length. It has been shown in other studies [5] that most of the time, these lengths are within 4 bytes. Also, the VAX machine always fetches a block of 4 bytes together from the memory. Thus, it is quite reasonable to approximate the length of all the references to 4 bytes. This simplifies the process of gathering the trace. Since our aim is the validation of the methodology of simulation and not the evaluation of VAX, this approximation is of little consequence to us. During this simulation, if the address referenced fits exactly in an integral 4-byte boundary, only one block (of size at least 4 bytes) is fetched. Otherwise if the 4-byte field, starting from the current address in the trace, does not coincide with a 4-byte boundary and also surpasses the end of the current block of the cache, then the next block is also fetched. In the next section, the results of simulation are presented.

2.3. Results

Recall that the goal is to show that a sampling scheme can accurately estimate not only the mean value but also the distribution of the miss ratio. Note that it is assumed that a program is running with context switches. To establish the effectiveness of the sampling scheme, we compare the distribution of the sampled data with that of the continuous trace. As a first step, the mean and the standard deviations of the miss ratios of sampled data are compared with actual values of the cold-start miss ratio for the same context switch interval as the sample size (Table 2.1). The term "all" for the number of samples in the table signifies the results for the continuous trace without any sampling. It is found that the maximum error in mean miss ratio for approximately 35 samples is 5.5%; for about 20 samples this error can be as high as 13.8%. The standard deviation of miss ratio for about 35 samples is always very close to its actual value. Hence, it can be concluded that a good estimate of the mean and the standard deviations is achieved with approximately 35 samples.

The distribution from the continuous trace (not sampled) is compared with that obtained from the sampled trace in Figures 2.2 through 2.5. All these figures contain the histograms for the case using approximately 35 samples and the continuous trace, respectively. Figures 2.2 and 2.3 also include the case for approximately 20 samples. Of all the 12 cases considered, only four are given in these figures; the remainder appears in the Appendix. It is very apparent from these figures that, in general, the distribution profile is far from normal. It is found that the distributions of nearly 20 samples have significant

Table 2.1. Mean and Standard Deviations of Sampled Result

Program	Sample size	Mean miss ratio (for #samples)			%Error in mean (for #samples)		Std. dev. of miss ratio (for #samples)		
		≃ 20	≃ 35	All	≃ 20	≃ 35	≃ 20	≃ 35	All
FSIM	5K	.0743	.0767	.0781	-4.9	-1.8	.0158	.0177	.0160
	10K	.0576	.0609	.0600	-4.0	1.5	.0137	.0115	.0131
	20K	.0492	.0513	.0505	-2.6	1.6	.0125	.0098	.0107
DRC	5K	.0768	.0639	.0675	13.8	-5.3	.0539	.0443	.0430
	10K	.0630	.0550	.0582	8.2	-5.5	.0425	.0386	.0389
	20K	.0576	.0512	.0528	9.1	-1.9	.0389	.0349	.0353
ELI	5K	.0819	.0775	.0792	3.4	-2.1	.0452	.0379	.0406
	10K	.0680	.0691	.0673	1.0	2.7	.0359	.0351	.0343
	20K	.0606	.0616	.0604	0.3	2.0	.0295	.0311	.0307
RRL	5K	.0424	.0417	.0416	1.9	0.2	.0354	.0332	.0329
	10K	.0349	.0356	.0346	0.9	2.9	.0283	.0293	.0286
	20K	.0318	.0315	.0308	3.2	2.3	.0267	.0258	.0255

(a) Number of samples = 20

```
MIDPOINT
   Y                                                                      FREQ   CUM.    PERCENT    CUM.
                                                                                 FREQ               PERCENT
 0.028   |                                                                   0      0      0.00       0.00
 0.034   |*****                                                              1      1      5.00       5.00
 0.040   |***********************************                                7      8     35.00      40.00
 0.046   |********************                                               4     12     20.00      60.00
 0.052   |***************                                                    3     15     15.00      75.00
 0.058   |                                                                   0     15      0.00      75.00
 0.064   |**********                                                         2     17     10.00      85.00
 0.070   |*****                                                              1     18      5.00      90.00
 0.076   |**********                                                         2     20     10.00     100.00
          -----+----+----+----+----+----+----+
               1    2    3    4    5    6    7
                      FREQUENCY
```

(b) Number of samples = 36

```
MIDPOINT
   Y                                      FREQ   CUM.    PERCENT    CUM.
                                                 FREQ               PERCENT
 0.028   |                                   0      0      0.00       0.00
 0.034   |****                               2      2      5.56       5.56
 0.040   |********                           4      6     11.11      16.67
 0.046   |******************                 9     15     25.00      41.67
 0.052   |******************                 9     24     25.00      66.67
 0.058   |************                       6     30     16.67      83.33
 0.064   |******                             3     33      8.33      91.67
 0.070   |**                                 1     34      2.78      94.44
 0.076   |****                               2     36      5.56     100.00
          ----+---+---+---+--
              2   4   6   8
              FREQUENCY
```

(c) Continuous Trace

```
MIDPOINT
   Y                                           FREQ   CUM.    PERCENT    CUM.
                                                      FREQ               PERCENT
 0.028   |**                                      2      2      1.40       1.40
 0.034   |*******                                 7      9      4.90       6.29
 0.040   |************************               24     33     16.78      23.08
 0.046   |**********************************     34     67     23.78      46.85
 0.052   |*****************************          29     96     20.28      67.13
 0.058   |*********************                  21    117     14.69      81.82
 0.064   |****************                       16    133     11.19      93.01
 0.070   |*                                       1    134      0.70      93.71
 0.076   |*********                               9    143      6.29     100.00
          -----+----+----+----+----+----+----
               5   10   15   20   25   30
                      FREQUENCY
```

Fig. 2.2. Frequency bar chart of miss ratio for FSIM : Context-switch interval = 20K

(a) Number of samples = 20

```
MIDPOINT
    Y                                                        FREQ  CUM.  PERCENT     CUM.
                                                                   FREQ           PERCENT
 0.015  |***************                                       3     3    15.00     15.00
 0.033  |******************************                        6     9    30.00     45.00
 0.051  |                                                      0     9     0.00     45.00
 0.069  |*****                                                 1    10     5.00     50.00
 0.087  |**********                                            2    12    10.00     60.00
 0.105  |                                                      0    12     0.00     60.00
 0.123  |********************                                  4    16    20.00     80.00
 0.141  |**********                                            2    18    10.00     90.00
 0.159  |**********                                            2    20    10.00    100.00
        -----+----+----+----+----+----+
             1    2    3    4    5    6
                 FREQUENCY
```

(b) Number of samples = 36

```
MIDPOINT
    Y                                                  FREQ  CUM.  PERCENT     CUM.
                                                             FREQ           PERCENT
 0.015  |************                                    6     6    16.67     16.67
 0.033  |************************                       12    18    33.33     50.00
 0.051  |**                                              1    19     2.78     52.78
 0.069  |**                                              1    20     2.78     55.56
 0.087  |**********                                      5    25    13.89     69.44
 0.105  |********                                        4    29    11.11     80.56
 0.123  |**********                                      5    34    13.89     94.44
 0.141  |****                                            2    36     5.56    100.00
 0.159  |                                                0    36     0.00    100.00
        ----+---+---+---+---+---+
            2   4   6   8  10  12
               FREQUENCY
```

(c) Continuous Trace

```
MIDPOINT
    Y                                       FREQ     CUM.    PERCENT     CUM.
                                                     FREQ             PERCENT
 0.015  |*********                           174      174     15.34     15.34
 0.033  |*******************                 384      558     33.86     49.21
 0.051  |*                                    17      575      1.50     50.71
 0.069  |*                                    25      600      2.20     52.91
 0.087  |*******                             148      748     13.05     65.96
 0.105  |*******                             148      896     13.05     79.01
 0.123  |*******                             139     1035     12.26     91.27
 0.141  |***                                  67     1102      5.91     97.18
 0.159  |**                                   32     1134      2.82    100.00
        -----+----+----+----
            100  200  300
             FREQUENCY
```

Fig. 2.3. Frequency bar chart of miss ratio for DRC : Context-switch interval = 5K

(a) Number of samples = 35

```
MIDPOINT
    Y                                                     FREQ  CUM.   PERCENT    CUM.
                                                                FREQ             PERCENT
 0.007  |************                                      3     3      8.57      8.57
 0.021  |****                                              1     4      2.86     11.43
 0.035  |****************************                      7    11     20.00     31.43
 0.049  |****                                              1    12      2.86     34.29
 0.063  |********                                          2    14      5.71     40.00
 0.077  |****************                                  4    18     11.43     51.43
 0.091  |****************************                      7    25     20.00     71.43
 0.105  |********************************                  8    33     22.86     94.29
 0.119  |********                                          2    35      5.71    100.00
        |
        ---+---+---+---+---+---+---+---+
           1   2   3   4   5   6   7   8
                 FREQUENCY
```

(b) Continuous Trace

```
MIDPOINT
    Y                                        FREQ  CUM.   PERCENT    CUM.
                                                   FREQ             PERCENT
 0.007  |*********                            17    17      9.71      9.71
 0.021  |****                                  7    24      4.00     13.71
 0.035  |****************                     32    56     18.29     32.00
 0.049  |***                                   5    61      2.86     34.86
 0.063  |*****                                 9    70      5.14     40.00
 0.077  |***************                      30   100     17.14     57.14
 0.091  |*******************                  38   138     21.71     78.86
 0.105  |***************                      29   167     16.57     95.43
 0.119  |****                                  8   175      4.57    100.00
        |
        -----+----+----+----
            10   20   30
              FREQUENCY
```

Fig. 2.4. Frequency bar chart of miss ratio for ELI : Context-switch interval = 10K

(a) Number of samples = 33

```
 MIDPOINT
    Y                                                  FREQ   CUM.    PERCENT     CUM.
                                                              FREQ               PERCENT
0.0145   |******************                             9      9      27.27      27.27
0.0295   |************************                      12     21      36.36      63.64
0.0445   |**********                                     5     26      15.15      78.79
0.0595   |****                                           2     28       6.06      84.85
0.0745   |                                               0     28       0.00      84.85
0.0895   |****                                           2     30       6.06      90.91
0.1045   |**                                             1     31       3.03      93.94
0.1195   |**                                             1     32       3.03      96.97
0.1345   |**                                             1     33       3.03     100.00
          ----+---+---+---+---+---+
              2   4   6   8  10  12
                 FREQUENCY
```

(b) Continuous Trace

```
 MIDPOINT
    Y                                                  FREQ   CUM.    PERCENT     CUM.
                                                              FREQ               PERCENT
0.0145   |*********************                        107    107      27.23      27.23
0.0295   |**************************                   129    236      32.82      60.05
0.0445   |****************                              80    316      20.36      80.41
0.0595   |*                                              5    321       1.27      81.68
0.0745   |**                                            10    331       2.54      84.22
0.0895   |**                                            12    343       3.05      87.28
0.1045   |******                                        31    374       7.89      95.17
0.1195   |***                                           13    387       3.31      98.47
0.1345   |*                                              6    393       1.53     100.00
          ----+---+---+---+---+---+--
             20  40  60  80 100 120
                 FREQUENCY
```

Fig. 2.5. Frequency bar chart of miss ratio for RRL : Context-switch interval = 5K

difference from the true ones; but the profiles of about 35 samples closely resemble the actual shape in all cases. A Kolmogorov-Smirnov (K-S) two-sample test [20,21] was performed on these distributions to check the null hypothesis that the distributions of the samples are equivalent to the actual ones.

The K-S test compares two independent empirical distributions of sizes m and n. The observations are designated $X_1, X_2, \ldots, X_m$ for the sampled case and $Y_1, Y_2, \ldots, Y_n$ for the continuous case. Let $S_1(x)$ and $S_2(x)$ designate the distribution functions of the observed X's and the observed Y's, respectively.

$$S_1(x) = (\textit{number of observed X's} \leqslant x) \,/\, m$$

$$S_2(x) = (\textit{number of observed Y's} \leqslant x) \,/\, n$$

Then the test statistic for the two-sample test is as follows.

$$D = \textit{maximum} \; |S_1(x) - S_2(x)|$$

That is, D is the least upper bound of all pointwise differences $|S_1(x) - S_2(x)|$. We are interested in using the statistic D to test the hypothesis H_0: $S_1(x) = S_2(x)$ against all alternatives, H_1: $S_1(x) \neq S_2(x)$. The hypothesis H_0 is rejected at the α level of significance, if the observed value of D is greater than the critical value specified by the $(1-\alpha)$ quantile of the K-S test. Now in our case, the two miss ratio distributions that are compared are obtained from the sampled trace and the continuous trace respectively. The maximum value of D, thus obtained, was always found to be smaller than the theoretically calculated value of D (according to K-S test) at a 0.20 level of significance; thus, the alternate hypothesis cannot be accepted. In other words, the sampled data lie within 100 (1-0.20)% = 80% confidence band for the distribution from the continuous trace. Hence, it is more supportive of the null hypothesis that the two distributions have no measurable difference.

Clearly, 20 samples are not adequate to represent program behavior for trace lengths ranging from 1.8 million references to 5.7 million references. It is also clear that 35 samples are adequate for all these traces. This suggests that the number of samples used is a controlling factor for the validity of this sampling method, irrespective of the length of the trace. Finally, since the programs used for collecting these traces are of widely varying nature, it would appear that 35 samples are adequate for this method.

2.3.1. Other information

The distribution of the miss ratios provides a lot of information about the program behavior which is lost when only the mean value is considered. For example, the distribution of miss ratio is useful in the performance analysis of cache memories in multiprocessors and cache memories with a load-through feature or a finite write-back buffer. In load-through and write-back, overlapped with CPU operations, the penalty *per miss* will be higher when the intermiss distance is small compared to the case when the misses are placed far apart. These distributions for the four programs are quite different from one another; but, for a particular program, the distributions are similar for various context switch intervals (i.e., for different sample sizes in our method). The miss ratio distribution of the DRC program has two distinct modes while the distribution of the FSIM program is nearest to the normal distribution among the four programs. This indicates that the miss ratio of the DRC program is spread over two regions which are apart from each other; the region with smaller value corresponds to the tight loop structures in the program. The miss ratio versus time plot (Fig. 2.6) for this program corroborates this feature. The FSIM program on the other hand has its miss ratio randomly spread over a single band as found in its miss ratio versus time plot.

2.3.2. Comparison with conventional simulation

In conventional simulation, the measured distribution of the miss ratio over a small portion of the program execution can be quite different from the distribution of the miss

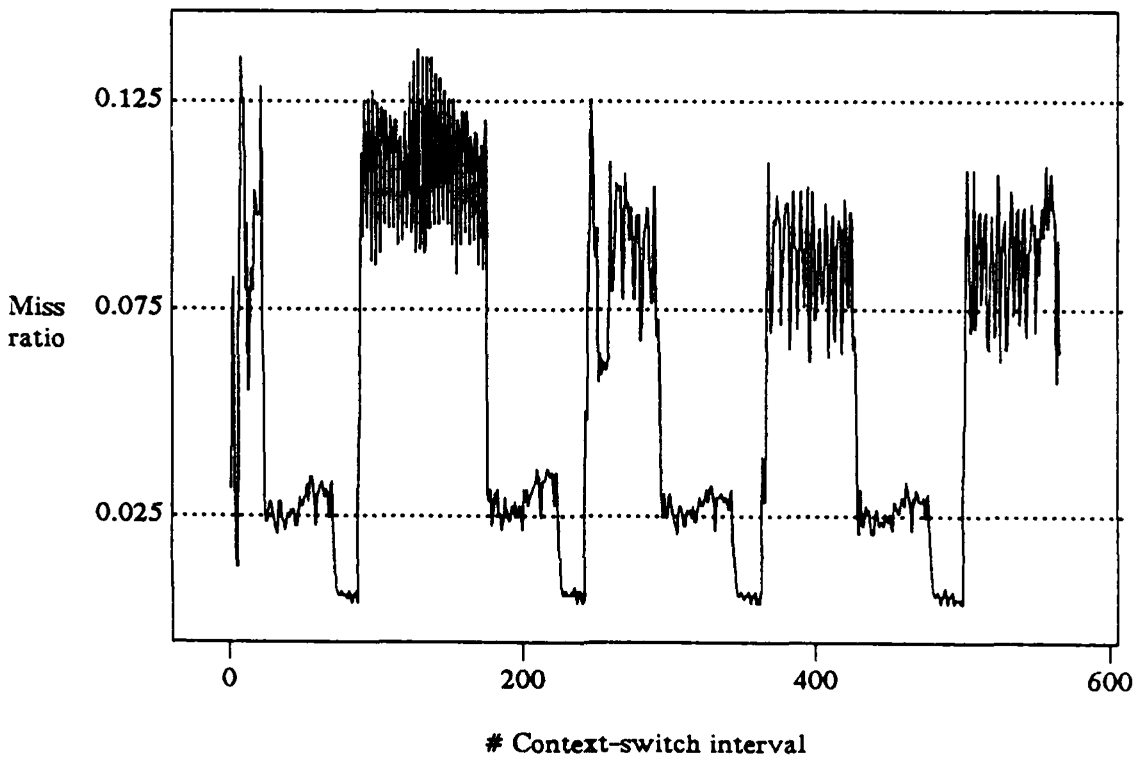


Fig. 2.6. Miss ratio versus time for DRC : Context-switch interval = 10K

ratio of the whole program. The conventional method for the cold-start miss ratio can be considered as simulating consecutive samples with no interval in between. To compare conventional simulation with our method, the distribution of the miss ratio of 35 consecutive context switch intervals of size 5K addresses is compared with the actual distribution of the miss ratio for the DRC program, because this simulation has the same space-time cost as our method. These two distributions are found to differ considerably in mean value, standard deviation and other statistical properties (Fig. 2.7). This is expected from Figure 2.6 which shows that 35 consecutive intervals, each of size 5K, cover only a small, initial part of the program which is very different from the rest. This shows the inadequacy of the conventional method to represent actual program behavior. Only recently, a study [22] provided simulations from different portions of the program trace (as opposed to just the initial part).

There is only one pathological case where the sampling technique may not prove satisfactory : (1) the miss ratio versus time plot of the program has a step function as in Figure 2.6, (2) this variation is periodic (constant time), (3) the sampling interval has exactly the same period and (4) all the samples are taken predominantly from the top or bottom level of the step function. This can be avoided by taking the samples at random intervals. However, as the behavior of any real workload is never strictly periodic (e.g., due to asynchronous interrupts), this problem would not arise in a real case. To compare uniform sampling with random sampling, samples are taken at random intervals for some of the cases and it is found that the accuracy of the results is the same, irrespective of the nature of the sampling interval being uniform or random. This is due to the fact that the miss rate behavior is itself somewhat random in time and, therefore, uniform sampling is sufficient. However, if in some case strong periodicity of the miss rate is suspected, the use of random sampling is recommended.

(a) Conventional method with 35 intervals

(Mean : 0.083; Standard deviation : 0.048)

```
MIDPOINT
   Y                                                          FREQ   CUM.    PERCENT     CUM.
                                                                     FREQ               PERCENT
 0.015  |*************************                               5      5     14.29      14.29
 0.033  |*****                                                   1      6      2.86      17.14
 0.051  |******************************                          6     12     17.14      34.29
 0.069  |***********************************                     7     19     20.00      54.29
 0.087  |***************                                         3     22      8.57      62.86
 0.105  |***************                                         3     25      8.57      71.43
 0.123  |*****                                                   1     26      2.86      74.29
 0.141  |********************                                    4     30     11.43      85.71
 0.159  |*************************                               5     35     14.29     100.00
        |
        ----+----+----+----+----+----+----+
            1    2    3    4    5    6    7
                  FREQUENCY
```

(b) Complete Trace

(Mean : 0.068; Standard deviation : 0.043)

```
MIDPOINT
   Y                          FREQ    CUM.   PERCENT     CUM.
                                      FREQ              PERCENT
 0.015  |*********             174     174    15.34      15.34
 0.033  |*******************   384     558    33.86      49.21
 0.051  |*                      17     575     1.50      50.71
 0.069  |*                      25     600     2.20      52.91
 0.087  |*******               148     748    13.05      65.96
 0.105  |*******               148     896    13.05      79.01
 0.123  |*******               139    1035    12.26      91.27
 0.141  |***                    67    1102     5.91      97.18
 0.159  |**                     32    1134     2.82     100.00
        |
        ----+----+----+----
           100  200  300
           FREQUENCY
```

Fig. 2.7. Comparison of conventional method for DRC : Context-switch interval = 5K

2.4. Evaluation of Other Context Switch Intervals

Thus far we have established a sampling based technique to predict the miss ratio of programs with context switch intervals, equal to the size of the sample used. In this section, we propose a model of cache behavior to extend this result for predicting the miss ratio for context switch intervals different from the sample size. Our aim is to perform only one simulation on samples of one size and statistically project the results for context switch intervals of other sizes. Once simulation is performed for samples of a particular size, information about task intervals of smaller sizes is already present in the results. Therefore, only those intervals which are greater than the sample size need to be predicted. To do that, we will first consider how the mean value of miss ratio over the samples varies within the duration of the sample.

Figure 2.8 shows the typical plot of the mean miss ratio (over different samples) with the number of address references during the period of the sample size. The value of the mean miss ratio at each point is calculated by considering the number of hits and misses from the end of the samples, rather than from the beginning. As a result, the miss ratio at any intermediate point of the sample includes the behavior of the cache only after this point, but not before it. Thus, the effect of the cold-start at the beginning of the sample is not reflected in the miss ratio of the later segments of the sample. This enables us to check if the instantaneous mean miss rate in an interval reaches a stable value as the cache gets more and more filled. It is observed in all our experiments that the miss ratio really attains a steady state for sufficiently long samples. In any case, the value of the miss ratio at the tail of the curve may differ only by a small amount from the value of the miss ratio obtained by a bigger interval.

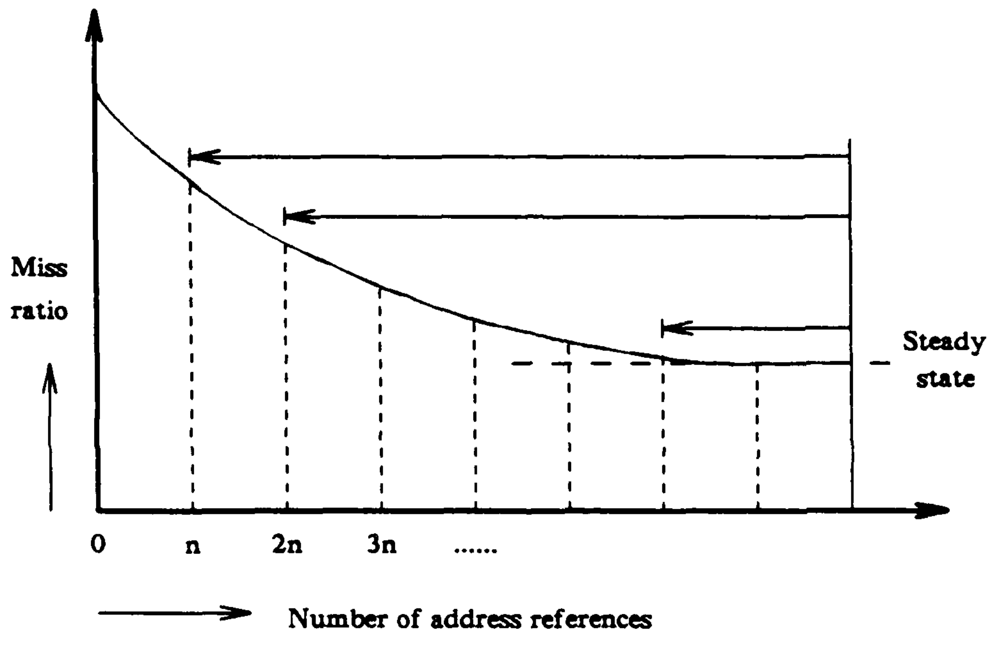


Fig. 2.8. Mean miss ratio versus number of addresses

2.4.1. Our model

Given a sample size n_0 which is long enough for the miss ratio to reach a stable value, this steady state will continue for the rest of a larger task interval n (Fig. 2.9).

A capacitor model for cache which has some similarity to this model was proposed in [23]. Based on our model, the value of the miss ratio for a larger context switch interval n can be easily calculated and is given by

$$m(n) = \frac{n_0}{n} m(n_0) + (1 - \frac{n_0}{n}) m_{st}$$

where $m(n)$: miss ratio for interval n

$m(n_0)$: miss ratio for interval n_0

m_{st} : steady-state value of miss ratio.

2.4.2. Results

Based on the formula above, the miss ratio for the 20K interval is predicted from that of the 5K and 10K samples for all the traces; also, the miss ratio for the 10K context switch interval is calculated from that of the 5K samples (Table 2.2). It is clear that the projected value is close to the actual value, measured from the continuous trace. The projected value is always larger than the measured value in Table 2.2. This can be seen easily from Figure 2.9. If the miss ratio continues to drop beyond n_0, i.e., it has not reached a steady state, then the projected value will always be greater than the steady-state value. The error can be reduced by increasing n_0. The standard deviations are close. In addition, K-S tests were performed and the corresponding distributions were found to match satisfactorily.

2.4.2.1. Continuous tasks This model is also extended to the case of continuous tasks when a program runs to completion without any context switching. According to this model, the continuous task can be considered as a single large interval and the estimated

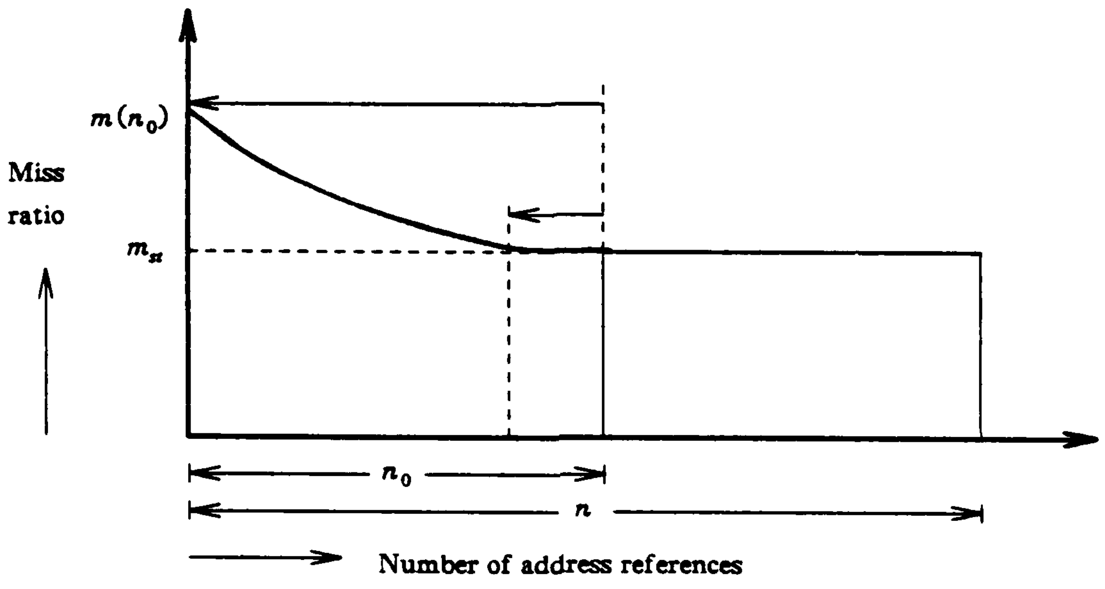


Fig. 2.9. Model for large context switch intervals

Table 2.2. Calculated Miss Ratios for Different Context Switch Intervals

Program	Context switch interval	Actual measured miss ratio	Miss ratio calculated from context switch interval	
			5K	10K
FSIM	10K	0.060	0.062	
	20K	0.051	0.053	0.054
DRC	10K	0.057	0.061	
	20K	0.052	0.056	0.0525
ELI	10K	0.067	0.067	
	20K	0.061	0.0615	0.0635
RRL	10K	0.034	0.035	
	20K	0.030	0.0315	0.032

value of the miss ratio simply becomes equal to m_{st}, the measured steady-state value of the miss ratio. Recall that the continuous task has only one cold-start while the sampled task has a cold-start for every sample. Therefore, the cold-start effect from the samples must be removed to match with the continuous task.

The miss ratio for the continuous task has been predicted from that of the 5K, 10K and 20K samples for the programs DRC, FSIM and RRL (Table 2.3). The predicted miss ratio for the continuous task is compared with the miss ratio from the complete trace for all the cases. The error in mean value of these projected results is shown in the first column of error in Table 2.3. To separate the error due to projection from the error due to sampling, the identically located samples from a simulation of the continuous task on the complete trace are also considered. Comparing this value with the results of complete simulation gives just the sampling error and is included in the second column of error in Table 2.3. For the FSIM trace, the error due to projection is high when the sample size used is less than 20K. This shows that the storage requirement of this program is higher than the other two. To further quantify the error due to this projection method (excluding sampling error), the distribution of the projected miss ratio and the distribution of the miss ratio of the identically located samples from continuous simulation are compared. The standard deviations of these distributions, shown in Table 2.3, are close. Finally, the K-S test performed on these distributions shows them to be satisfactorily identical to each other.

2.4.2.2. Variable intervals In practice, a program switches context at variable intervals due to asynchronous events, such as I/O requests and interrupts. For such varying context switch intervals, the resultant miss ratio can be derived by first estimating the miss ratio for each of these intervals by this method and then determining their weighted average to achieve the desired distribution of the corresponding context switch intervals.

Table 2.3. Projected Mean and Standard Deviations for Continuous Task

Program	Actual	Sample	Mean value		%Error (mean)		Std. dev.	
	mean	size	project	sample	project	sample	project	sample
DRC	0.049	5K	0.049	0.046	0.0	-6.1	0.043	0.042
		10K	0.045	0.046	-8.2	-6.1	0.037	0.037
		20K	0.047	0.049	-4.1	0.0	0.035	0.036
FSIM	0.039	5K	0.049	0.044	25.6	12.8	0.021	0.023
		10K	0.045	0.040	15.4	2.6	0.015	0.015
		20K	0.041	0.039	5.1	0.0	0.012	0.015
RRL	0.028	5K	0.028	0.030	0.0	7.1	0.029	0.028
		10K	0.028	0.028	0.0	0.0	0.027	0.028
		20K	0.026	0.029	-7.1	3.6	0.024	0.025

2.4.3. Comments on this model

Our objective is to predict the miss ratio of all different context switch intervals by some method which uses the sampled result, thus utilizing all the advantages of the basic sampling technique. On careful examination of all the existing analytical methods [8, 10, 11] to estimate the cold-start miss ratio for different task intervals, it has been found that these methods need to analyze the complete trace and, therefore, cannot be applied in a straightforward way to the sampled trace. Therefore, some new model was needed to be used with the sampled trace. Though our model is rather heuristic in nature, it is based on the results from the sampling technique, has almost no additional overhead and estimates the different miss ratios with a degree of accuracy no less than that of existing methods. The conventional methods require simulation of the whole trace and subsequent computation which is far more complex than the proposed method. However, it has been observed that this new model holds well only for small caches (16 Kbytes or less). For large caches, a different methodology is required. This is presented in the next chapter.

CHAPTER 3

SAMPLING TECHNIQUE FOR LARGE CACHE

For caches larger than 16 Kbytes, a significant amount of data is retained across the context switches [24]. Modeling this by purging the simulated cache or by assuming that context-switching has no effect on such caches is inaccurate [22]. For the same reason, these caches require very long traces to avoid the start-up effect of an empty cache. Furthermore, the miss ratio in large caches is usually very small and thus to obtain reliable measurements related to misses, one needs extremely long traces. It is these large caches which are becoming very common, those whose behavior can be studied only by using massively long trace. Therefore, an appropriate sampling method will be most useful for large caches. In this section, we present our methodology for large cache simulation along with the results.

3.1. Methodology for Large Cache Simulation

With the sampling method, when a large cache is simulated with a sample, there is no way of recreating the exact state of the cache which would have been attained if complete trace had been used. Therefore, our strategy is to simulate only those portions of the cache whose true state can be recreated from the sampled trace. The inaccuracy, caused by those addresses of the sample of the trace whose contribution to hits or misses cannot be determined from the simulation of only the samples, is thus eliminated. The portion of the cache whose exact state can be constructed is called the *primed* portion of the cache.

At the beginning of each sample, the cache is assumed to be empty. It starts getting filled as more and more references are simulated. A few definitions regarding our methodology are introduced here.

Primed Set: A set is considered primed when it is completely filled by the addresses simulated, starting from an empty cache at the beginning of the sample. For a direct-mapped cache, the term primed set will always refer to a filled block.

Significant Address (or Significant Reference): An address (or reference), used in sampling-based simulation, is considered significant only if it maps into a primed set. Only the significant references are considered towards the total count of hits and misses.

Significance Ratio: The ratio of number of significant references to the total number of references is known as the significance ratio.

The following illustration will further clarify these terms. The cache memory considered is 2-way set-associative and follows the LRU policy for replacing the blocks. All the caches discussed in this section are virtual address caches, i.e. the virtual address is directly mapped to cache address. To simplify the example, only those addresses of the reference string that map to a particular set are considered. Let this sub-string of the references be $R = \{A,A,A,B,B,A,C,C\}$. Starting with an empty cache, each reference in R may cause a *fill*, a *hit*, or a *miss*. Note that the term "miss" here refers to a replacement. At the beginning of the sample, a reference causes a "fill" which will actually cause a hit or a miss if a complete trace is simulated instead of only the samples. To identify the behavior of these references in the cache address space, the following notations are introduced with respect to the set under consideration:

f_i = i-th distinct address causing a fill

h_i = i-th distinct address causing a hit

m_i = i-th distinct address causing a miss

Table 3.1 shows the state of the set of the cache as the cache is simulated with the reference string R. Because LRU replacement policy is used, the cache block containing the most recent address is designated as *recent* and the other block is called *previous*. The set is

Table 3.1. Illustration of Primed Set and Significant Reference

Reference number	Reference	Cache event	Contents of set		Comments
			Recent	Previous	
1	A	f_1	A		Fill
2	A	h_1	A		Hit not counted
3	A	h_1	A		Hit not counted
4	B	f_2	B	A	Fill, set primed
5	B	h_2	B	A	Hit counted
6	A	h_1	A	B	Hit counted
7	C	m_3	C	A	Miss counted
8	C	h_3	C	A	Hit counted

empty at the beginning. Reference number 4 fills the set completely making it primed. Any later reference to the set is considered significant and adds to the total count of hits and misses. Any earlier reference to the empty or partially filled set is not counted. Clearly, all fills, including the last one to a set, cannot be counted as hits or misses with 100% certainty.

The term primed or nonprimed applies to a set only in case of sampling-based simulation. Two new terms are introduced here before further discussion of primed cache.

Sampled Trace State: If the cache memory system is simulated using a sampled trace up to any reference in a sample, then the state of a set is known as the sampled trace state of that set at that reference.

Continuous Trace State: If the cache memory system is simulated with a continuous trace up to any reference in the trace, then the state of a set is called the continuous trace state of that set at that reference.

Claim: The sampled trace state of a primed set is always identical to its continuous trace state, in an LRU cache.

Consider that a sample of the trace starts at the reference r_1. Then at r_1, the cache is empty in the sampling-based case; but, in general, the cache is at least partially filled at this point in the case of continuous trace simulation. In both these cases, the references, starting at r_1, are translated exactly to the same cache addresses. In the sampling method, these references will initially fill the cache while in the continuous case, they will result in filling or causing hits or misses. In the illustration given in Table 3.1, references 1 and 4 would cause fills, hits or misses in the case of continuous trace simulation while they cause only fills in the sampling-based method. But, at the end of reference 4, the set will have the address B in the recent position and address A in the other position regardless of the simulation method. Therefore, when a set becomes primed, its sampled trace state will be exactly the same as its continuous trace state. As a result, any later references to the same set by either method of simulation will have the same consequences. Hence, if we consider only the significant references in the trace-based simulation, the two simulation methods will yield identical results.

A more detailed analysis of the referencing pattern in the cache address space is required to implement this idea. Consider a string of cache addresses from a sampled trace to a particular set as shown in Figure 3.1. The cache is n-way set-associative where n is any integer greater than one. In the address string shown in the figure, let f_l be the last fill to that set and h_1's be the immediately following hits to the MRU (most recently used) position. The first non-MRU hit is designated as h_2 and the subsequent hits and misses (except the last miss) are labeled simply as h and m respectively. The intervening hits and misses are not particularly significant for this discussion. The first miss in the primed set is indicated as m_1 and the last miss as m_l. The reference string is demarcated with different time boundaries as follows:

T_0 : beginning of the sample

T_1 : just following the last fill f_l

T_2 : just preceding the first non-MRU hit h_2 (or first miss m_1 whichever comes first)

T_3 : just preceding first miss m_1

T_4 : just preceding last miss m_l

T_5 : end of the sample

The set is primed at time T_1. It is important to note that the sample ends at a random point T_5, but the other T_i's (i=1,2,3,4) are not totally random references in the address string. If the miss ratio is calculated by considering only the references in the interval bounded by different combinations of these time boundaries, the predicted miss ratio may overestimate or underestimate the actual value, depending on the choice of these intervals.

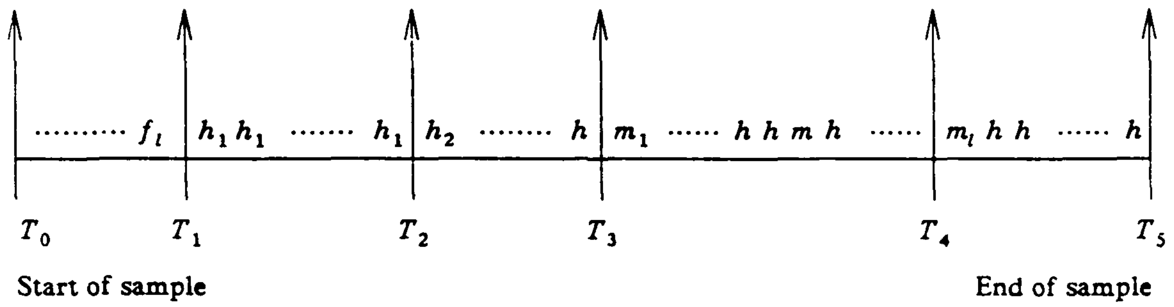


Fig. 3.1. Different intervals for significant references

The references lying in such an interval are considered towards the final count of hits and misses and are, therefore, called ***significant references***. The effect of choosing some of the more meaningful pairs of these time boundaries are considered below.

$T_0 - T_5$: This interval corresponds to the whole sample, without any priming. Needless to say, the miss ratio, thus computed, is always an overestimate because of the extra cold-start effect introduced at the beginning of each sample.

$T_1 - T_5$: Computation of the miss ratio based on this interval underestimates the miss ratio. Last fill f_l is actually a miss or a hit to the bottom of the LRU stack. In a large set-associative cache, the MRU hit ratio is higher than 90% (sometimes even higher than 95%), as noted by the authors and other researchers [14,22]. As a result, this last fill is usually followed by a long trail of MRU hits. The exclusion of the last fill (which is a good candidate for a miss) and the inclusion of these MRU hits tend to bias the miss ratio towards a lower value.

$T_2 - T_5$: By excluding the trail of MRU hits between T_1 and T_2, the bias in the estimation of the miss ratio of the type present in the previous case is reduced here. Because the occurrence of the non-MRU hits is somewhat random, this interval will neither consistently overestimate nor consistently underestimate the actual value. We can, therefore, expect a result much closer to the true value.

$T_3 - T_5$: In this case, the references always start from a miss. This will most likely overestimate the miss ratio since a large number of hits before T_3 has been ignored.

$T_3 - T_4$: This allows us to have an unbiased interval for the significant references because it starts with a miss and ends just short of a miss. Therefore, this seems to be the most desirable interval from the theoretical point of view. But, in practice, the miss ratio is so low for large caches that a considerable number of filled sets have less than two misses in a sample and are therefore not counted in this method and only the sets with more misses are included. Hence, this method results in overestimation of the result.

Other intervals are not meaningful enough to be discussed. Among the different pairs of boundaries considered here, $(T_2 - T_5)$ is expected to give the most reasonable estimate of the miss ratio and is, therefore, selected as the final method. However, this interval does not work for direct-mapped cache where the concept of non-MRU hit does not exist and T_2 always coincides with T_3. As a result, this method will always overestimate the miss ratio for a direct-mapped cache.

3.2. Simulations Performed

Since no long, single segment of a trace from a multiprogramming environment was available to us, we synthesized a trace with multiple processes, but no operating system, by interleaving segments of the four program traces, FSIM, DRC, ELI and RRL (mentioned in Section 2.2.1). We assume a randomized sequence of context switching intervals with a mean of 15000 references. The total length of the resultant trace is 4 million addresses.

As with small caches, the results from the sampling-based method are compared with those from the simulation of the continuous trace. To avoid the inaccuracy due to the start-up effect, the first half million references of the trace are ignored when counting the overall miss ratio. In the sampling-based method, these first half million references are simply skipped.

The architectural model used in this case is somewhat different. The main difference is the use of a virtual address cache. The virtual address, which has the process ID embedded in it, is directly converted to the cache address. Another difference is the write policy which is chosen to be write-allocate, i.e., during a write-miss, the cache memory is always updated. The replacement policy is LRU, as before. Both the direct-mapped and the set-associative caches are considered. Cache sizes from as small as 8 Kbytes to as large as 128 Kbytes are simulated. The block size is kept fixed at 16 bytes in one case and in another case, the block size is varied to keep the number of blocks constant.

3.3. Results

The suggested method has been applied to direct-mapped caches and set-associative caches of associativity 2 and 4. The sampling interval is 100,000 and the sample size is chosen as 60,000. Thus there are 40,000 references between the end of a sample and the start of the next sample. This sampling interval gives us 35 samples from the trace. A higher interval-to-size ratio of the samples is desirable, but could not be achieved because of the limited length of our continuous trace. However, a larger ratio is not so important here, because our emphasis is on the validity of the primed cache based simulation, and not simply on the sampling method.

The results for large caches are included in Tables 3.2 and 3.3. Table 3.2 lists the miss ratio for caches of fixed block size (= 16 bytes) while Table 3.3 presents results for caches of fixed number of blocks. As explained earlier, measurement on $(T_0 - T_5)$ always overestimates the miss ratio and that on $(T_1 - T_5)$ underestimates it. For direct-mapped caches, estimation based on $(T_2 - T_5)$ is found to be considerably higher than the actual value and has been excluded from the tables. The measured results for direct-mapped cache present only the range in which it lies - the range being quite small for small caches and increasing with an increase in cache size. Note that this range is rather small even for large caches when the block size is also large (as in Table 3.3). This is because of fewer numbers of blocks in such caches (compared to caches with small block size) causing a lesser number of fills which are the source of inaccuracy in the measurement. In all cases, this range can be

Table 3.2. Estimated Miss Ratio for Large Cache (Block size = 16, Sample size = 60000)

Set size	Cache size(KB)	Actual miss ratio	Measured miss ratio T_0-T_S	Measured miss ratio T_1-T_S	Measured miss ratio T_2-T_S	Error (%) for (T_2-T_S)	Sig. ratio for (T_2-T_S)
1	8	0.0458	0.0472	0.0418			
	16	0.0307	0.0340	0.0250			
	32	0.0235	0.0293	0.0170			
	64	0.0149	0.0225	0.0078			
	128	0.0115	0.0212	0.0049			
2	8	0.0253	0.0270	0.0243	0.0264	4.3	0.68
	16	0.0194	0.0229	0.0168	0.0204	5.2	0.42
	32	0.0140	0.0204	0.0101	0.0136	-2.9	0.21
	64	0.0104	0.0196	0.0069	0.0108	3.8	0.12
	128	0.0078	0.0193	0.0050	0.0078	0.0	0.06
4	8	0.0216	0.0235	0.0213	0.0220	1.9	0.71
	16	0.0175	0.0209	0.0151	0.0165	-5.7	0.44
	32	0.0125	0.0196	0.0104	0.0124	-0.8	0.15
	64	0.0084	0.0192	0.0056	0.0075	-10.7	0.04
	128	0.0064	0.0191	0.0034	0.0044	-31.3	0.01

Table 3.3. Estimated Miss Ratio for Large Cache (Sample size = 60000)

Set size	Cache size(KB)	Block size(byte)	Actual miss ratio	Measured miss ratio T_0-T_5	Measured miss ratio T_1-T_5	Measured miss ratio T_2-T_5	Error (%) for (T_2-T_5)	Sig. ratio for (T_2-T_5)
1	8	8	0.0611	0.0640	0.0537			
	16	16	0.0307	0.0340	0.0250			
	32	32	0.0203	0.0237	0.0164			
	64	64	0.0143	0.0171	0.0113			
	128	128	0.0107	0.0134	0.0088			
2	8	8	0.0361	0.0396	0.0330	0.0370	2.5	0.60
	16	16	0.0194	0.0229	0.0168	0.0204	5.2	0.42
	32	32	0.0109	0.0144	0.0087	0.0108	-0.9	0.30
	64	64	0.0067	0.0100	0.0063	0.0085	26.9	0.23
	128	128	0.0039	0.0071	0.0043	0.0058	48.7	0.17
4	8	8	0.0324	0.0358	0.0308	0.0323	-0.3	0.63
	16	16	0.0175	0.0209	0.0151	0.0165	-5.7	0.44
	32	32	0.0099	0.0136	0.0080	0.0095	-4.0	0.26
	64	64	0.0056	0.0092	0.0050	0.0059	5.4	0.14
	128	128	0.0031	0.0067	0.0038	0.0047	51.6	0.11

further lowered by increasing the sample size. A more accurate method for direct-mapped caches has been devised and can be found in Chapter 4.

For set-associative caches, the error for the measurement on $(T_2 - T_5)$ and the corresponding significance ratio are also included in the tables. The error of these estimated results for 2- and 4-way set-associative caches is less than 6%, as long as the measured miss ratio is greater than 0.01. To determine the error due only to priming, the sample size is increased to 100,000, keeping the sampling interval unchanged at 100,000, so that there is no unaccounted reference between successive samples. The results are shown in Tables 3.4 and 3.5.

3.3.1. Discussion

The results from Tables 3.4 and 3.5 are similar to those in Tables 3.2 and 3.3, and further confirm the observation that the error of the estimated results may increase beyond 10%, when the measured miss ratio is smaller than 0.01. It is found that for miss ratios higher than 0.01, a significance ratio of 0.1 or higher will yield reasonably accurate results, but for smaller miss ratios, a significance ratio as high as 0.3 is desirable. For such small miss ratios, the number of misses is so few that the characteristics of the primed sets are no longer representative of the behavior of the whole cache. Also, a small variation in the absolute value in this range will cause an appreciable percentile change in the miss ratio. For these low miss rates, we might need even longer sample sizes to attain a satisfactorily high significance ratio (as in the case of 128 Kbyte cache). If continuous trace is used instead of sampled trace with priming, a trace length on the order of hundreds of millions of addresses is required to determine such low miss rates with any reasonable confidence. Moreover, an inaccuracy of 20-30% in the measured miss ratio does not appreciably affect the overall evaluation of system performance (e.g., effective memory access time, pipeline delay etc.) for most existing machines when the miss ratio attains extremely low values.

Table 3.4. Estimated Miss Ratio for Large Cache
(Block size = 16, Sample size = 100000)

Set size	Cache size(KB)	Actual miss ratio	Measured miss ratio			Error (%) for (T_2-T_5)	Sig. ratio for (T_2-T_5)
			T_0-T_5	T_1-T_5	T_2-T_5		
1	8	0.0458	0.0463	0.0430			
	16	0.0307	0.0324	0.0263			
	32	0.0235	0.0271	0.0182			
	64	0.0149	0.0201	0.0088			
	128	0.0115	0.0185	0.0055			
2	8	0.0253	0.0261	0.0245	0.0258	2.0	0.80
	16	0.0194	0.0213	0.0176	0.0196	1.0	0.59
	32	0.0140	0.0181	0.0112	0.0134	-4.3	0.34
	64	0.0104	0.0169	0.0078	0.0096	-7.7	0.21
	128	0.0078	0.0162	0.0059	0.0075	-3.8	0.11
4	8	0.0216	0.0224	0.0214	0.0216	0.0	0.83
	16	0.0175	0.0193	0.0161	0.0169	-3.4	0.62
	32	0.0125	0.0171	0.0107	0.0116	-7.1	0.30
	64	0.0084	0.0160	0.0066	0.0072	-14.3	0.09
	128	0.0064	0.0157	0.0048	0.0054	-15.6	0.04

Table 3.5. Estimated Miss Ratio for Large Cache (Sample size = 100000)

Set size	Cache size(KB)	Block size(byte)	Actual miss ratio	Measured miss ratio T_0-T_s	Measured miss ratio T_1-T_s	Measured miss ratio T_2-T_s	Error (%) for (T_2-T_s)	Sig. ratio for (T_2-T_s)
1	8	8	0.0611	0.0625	0.0559			
	16	16	0.0307	0.0324	0.0263			
	32	32	0.0203	0.0224	0.0171			
	64	64	0.0143	0.0162	0.0120			
	128	128	0.0107	0.0126	0.0092			
2	8	8	0.0361	0.0378	0.0341	0.0363	0.6	0.74
	16	16	0.0194	0.0213	0.0176	0.0196	1.0	0.59
	32	32	0.0109	0.0130	0.0093	0.0107	-1.8	0.45
	64	64	0.0067	0.0089	0.0061	0.0071	6.0	0.37
	128	128	0.0039	0.0061	0.0045	0.0052	33.3	0.27
4	8	8	0.0324	0.0340	0.0315	0.0322	-0.6	0.77
	16	16	0.0175	0.0193	0.0161	0.0169	-3.4	0.62
	32	32	0.0099	0.0121	0.0087	0.0093	-6.0	0.44
	64	64	0.0056	0.0079	0.0051	0.0056	0.0	0.27
	128	128	0.0031	0.0055	0.0036	0.0039	25.8	0.20

In any case, if traces from real workloads are used instead of the workload synthesized from only user programs spanning a small portion of address space (as used here), the presence of supervisor code will result in a much higher miss ratio and this problem of estimating very low miss ratio will not arise. Therefore, we conclude that our method of using only the primed sets and the interval $(T_2 - T_5)$ gives accurate estimates of average miss ratios in large caches. Comparison of the distribution of the estimated miss ratio with the actual one needs further investigation. For some of the replacement policies other than LRU, it might be possible to come up with different strategies to prime the cache. However, this method of priming is not useful for direct-mapped or split caches. A different methodology is, therefore, presented in the next chapter.

CHAPTER 4

SAMPLING TECHNIQUE FOR SPLIT CACHE

The caches considered thus far are all unified in nature. In this chapter, the simulation of split caches is dealt with in detail. These split caches have different parts allotted to different kinds of references, such as instructions and data. The different parts of the split cache may have referencing patterns characteristic of that part of the cache. The referencing behavior in unified caches is much more random in nature. Therefore, the simulation techniques which implicitly utilize such randomness of addressing are not applicable to split caches. New simulation techniques, based on sampling, are proposed for split caches.

4.1. Description of Split Cache

In a single or unified cache system, the different types of references are stored in the same physical cache. There are several advantages of dividing the cache into more than one part, each complete with its own independent controller for the specific reference stream. The most common way of splitting the cache is to use one part for instruction and the other part for data.

Advantages of Split Cache Systems

(1) **Higher bandwidth:** Splitting the cache into instruction- and data-part doubles the bandwidth; the cache can now serve two requests in the time it formerly needed for one. In most CPUs, the requests for instruction and data are complementary in time and this fact can be specially utilized in pipelined CPUs. Typically, there are several stages in a pipeline - instruction fetch, instruction decode, operand address generation, operand fetch, execution and transmission of the results to their destination. Therefore, while one instruction is being fetched from the instruction (I) cache, another

instruction can fetch its operands from the data (D) cache. Also, the logic that arbitrates priority between instruction and data fetch can be simplified.

(2) **Optimal design of each part:** In split cache, it is possible to optimize the different cache parameters (e.g., block size, set size, fetch policy, replacement policy) for the different aspects of the program behavior. For instance, in the I-cache, it is beneficial to use longer blocks (or prefetching), compared with the block length of the data cache.

(3) **Efficient design:** These dedicated parts of the cache can be specifically designed to have a simpler, more efficient structure, tailored to the referencing pattern of that cache. I-cache design is simpler than D-cache design, because stores into the instruction cache are disallowed by most programming languages. An I-cache may contain the recently fetched instructions in decoded form, thereby reducing the time for decoding the instructions. With a single cache system, it is not always possible to place the cache immediately adjacent to all of the logic which will access it. A split cache, on the other hand, can have each of its parts placed properly near the control circuitry, thereby reducing the access time.

There are also some disadvantages introduced by the split cache organization.

Disadvantages of Split Cache Systems

(1) **Underutilization of different parts:** The size of the working set varies from workload to workload and, in particular, the fractions of the program space for instruction and data vary as well. As the instructions and data must be confined within their own parts in a split cache, they are unable to share the common resource available in a unified cache. As a result, the overall miss ratio in a split cache may be higher than its unified counterpart. A dynamically split design is suggested in [25], but it is too complex.

(2) **Problem of consistency and coexistence:** Two copies of the same information may exist in two different parts. Even if the program is not self-modifying, both instructions and data may coexist in the same block. In the case of modifying codes, any modification of an instruction must be reflected before the instruction is executed.

4.1.1. Practical systems

Many experimental and commercial computers use split caches. The RISC-based experimental minicomputer IBM801 [26,27] allows asynchronous fetching of instructions and data from its cache. Some of the earlier commercial computers which use split cache are Burroughs B7800 [28], Univac 1100/90 and Honeywell DPS88. Among the more recent commercial systems, there are microprocessor systems such as Intel iAPX386 and MIPS-X [29]; also, there are some minicomputer systems such as Pyramid-90X [30] and Gould 9080 [31]. Because instructions have better locality, the I-cache is often smaller in size than the D-cache. The Pyramid has a 32 Kbyte D-cache and only a 4 Kbyte I-cache. On the other hand, the MIPS M/500 has an I-cache (16K) larger than its D-cache (8K).

4.1.2. Our configuration

Dividing the cache equally into instruction and data parts is not the only configuration a split cache can have. These caches can be split in different ways. The data part can be further divided into stack data and nonstack data parts. One or the other of these parts could also be omitted from being cached [32]. In our configuration, half of the cache is allotted to instructions and the other half is equally divided between stack and nonstack data. It was observed that all of the traces we used have instructions, stack data and non-stack data references roughly in the same proportion as the corresponding cache size. All other cache parameters and the traces used are the same as chosen for large cache simulation in the previous chapter. The total size of the cache is varied from 32 Kbytes to 128 Kbytes.

In the next section, the problems of simulating split cache by means of priming will be discussed.

4.2. Problem of Simulation

These caches mentioned above were simulated using priming techniques, described in the chapter on large caches. It was found that this technique overestimates the miss ratio for the instruction cache and the stack cache. The result is satisfactory only for the non-stack cache. This happens because of the high sequentiality and spatial locality of the instructions and the stack references. The priming technique cannot handle the very low miss rate of these parts. The following illustration, though somewhat oversimplified, indicates the basic cause of this problem.

In this illustration, the cache memory considered is 2-way set-associative and follows the LRU replacement policy. Consider a reference stream spanning over one spatial locality (with high degree of sequentiality) and the cache to be empty to start with. If the first address in that locality maps to a particular set of the cache, then the subsequent virtual addresses will map to correspondingly subsequent sets in the cache, filling only one of the two blocks in these affected sets (Fig. 4.1). A virtual address which is away from this first virtual address by half the size of the cache will fill the first set and will make it primed. Before this, there will be no primed sets and all these references will not be counted as hits or misses. Consequently, if the locality does not span more than half the cache size (which may be large), in case of the highly sequential reference streams, there will be no primed sets resulting from this locality and the very few primed sets, otherwise obtained, will fail to represent the behavior of the whole cache. Actually, only those sets having a high replacement rate will become primed. As a result, it will cause overestimation of the miss ratio, as observed experimentally in the instruction cache and the stack cache.

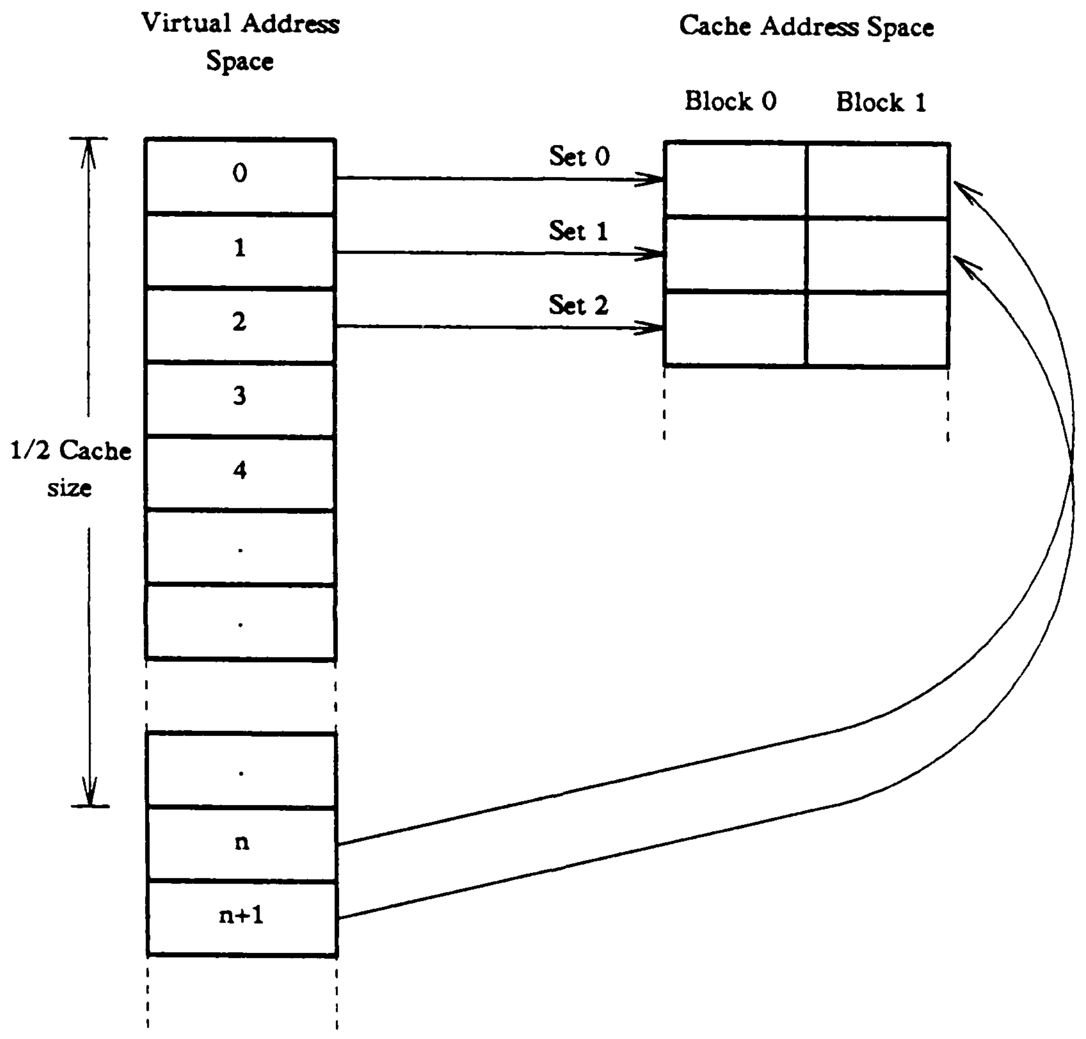


Fig. 4.1. Problem of split cache simulation

In the unified cache, the referencing pattern resulting from different types of addresses is somewhat random and, therefore, there are many more primed sets. To circumvent this problem of sequentiality in split cache, the following cache model is proposed.

4.3. Cache Model

Consider the *actual* warm-start miss ratio averaged over all the samples. Given two sufficiently wide (on the order of a quarter of the sample size) time windows T_1 and T_2 of the sample, the actual mean miss ratio over these two intervals will be (almost) the same as the mean miss ratio over the entire sample (Fig. 4.2), because of averaging them over all the samples. If the sizes of T_1 and T_2 are the same (i.e., $|T_1| = |T_2|$), the total number of misses in each of these intervals will also be the same. Let the number of such misses in each of these intervals be M.

Now consider the *measured* (as opposed to actual) cold-start mean miss ratio in these two intervals T_1 and T_2 of the samples, averaged over all the samples as before, but starting with an empty cache at the beginning of the sample. Because of forcing this extra cold-start effect at the beginning of the sample, there will be a number of fills (counted as misses) which are actually hits or misses as discussed in the previous chapter. Let f_n be the number of fills in some interval T_n $(|T_1| = |T_2| = |T_n|)$. Then, according to this model, the fraction of fills which are actually not misses and are counted as extra misses is *fixed* over any such window T_n and let this fraction be p. It has been experimentally observed that this fraction p is really almost constant as long as the initial (about a quarter or so) portion of the sample is not considered. Then, the measured number of misses m_n is given by

$$m_n = M + p.f_n \qquad (4.1)$$

Applying this relation on the intervals T_1 and T_2.

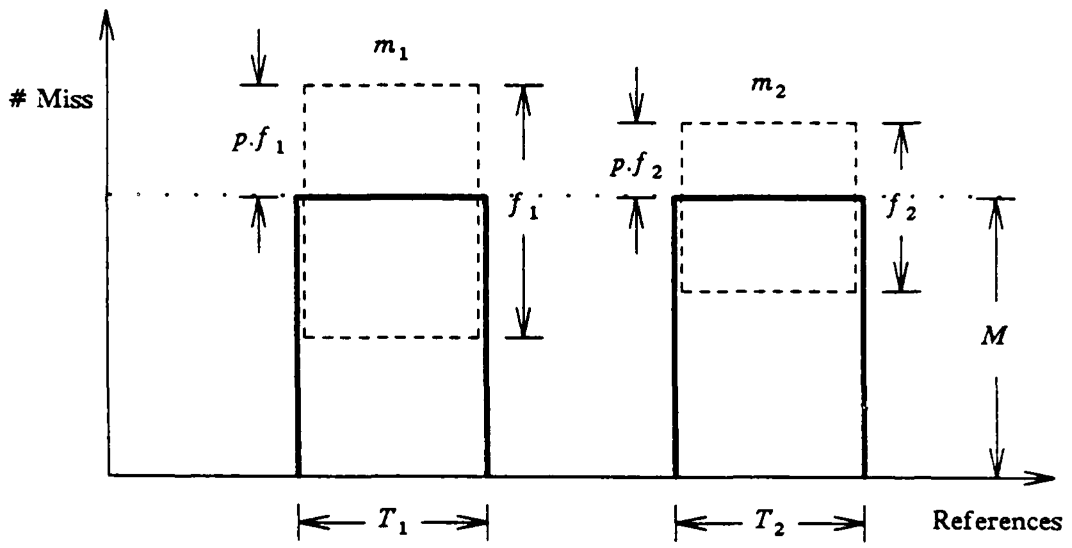


Fig. 4.2. Split cache model

$$m_1 = M + p.f_1 \tag{4.2}$$
$$m_2 = M + p.f_2 \tag{4.3}$$

Eliminating p from these two relations, the value of M is obtained as:

$$M = m_2 - \frac{m_1 - m_2}{f_1 - f_2} f_2 \tag{4.4}$$

Hence, the value of the miss ratio can be estimated.

4.4. Results

The above model is applied separately on each part of the split cache, thereby obtaining a separate value of the fraction p and the corresponding miss ratio for that part. The overall miss ratio of the split cache is calculated by taking the weighted mean (by the number of references in each part) of these miss ratios. The trace that is used here is the same as the one synthesized by interleaving the four program traces in the last chapter. The sample size is chosen as 80,000 references and the sampling interval is 100,000 references. The first 500,000 references out of a total of 4 million references are not counted as hits or misses both in the case of the sampling-based simulation and the continuous simulation (as in the last chapter). T_1 and T_2 are chosen as the second (20,001-40,000 references) and fourth (60,001-80,000 references) quarter of the samples (Fig. 4.2).

Direct-mapped and set-associative caches are considered. The simulations are performed with different block sizes for cache sizes of 32 Kbyte, 64 Kbyte and 128 Kbyte. The results are presented in the Tables 4.1 (direct-mapped cache), 4.2 (2-way associative cache) and 4.3 (4-way associative cache). The actual and estimated values of the miss ratio for the different parts and also the overall values are included in these tables. The last column of each of these tables indicates the percentile error in estimating the overall miss ratio. This error is always less than 11% in the case of a direct mapped cache. In the case of a 2-way set-associative cache, the error is within 9%. When the set size becomes 4, the

Table 4.1. Estimated Miss Ratio for Split Cache
(Set size = 1, Sample size = 80000)

Cache size(KB)	Block size(B)	Non-Stack actual	Non-Stack est.	Stack actual	Stack est.	Instruction actual	Instruction est.	Overall actual	Overall est.	Overall error(%)
32K	16	.0604	.0658	.0057	.0064	.0106	.0110	.0206	.0221	7.3
	32	.0590	.0634	.0048	.0059	.0077	.0081	.0186	.0200	7.5
64K	16	.0478	.0507	.0042	.0052	.0065	.0060	.0153	.0159	3.9
	32	.0443	.0476	.0028	.0035	.0046	.0046	.0132	.0140	6.1
	64	.0457	.0491	.0017	.0020	.0037	.0037	.0127	.0135	6.3
128K	16	.0410	.0421	.0042	.0052	.0042	.0033	.0126	.0126	0.0
	32	.0360	.0387	.0028	.0035	.0028	.0024	.0104	.0109	4.8
	64	.0348	.0376	.0017	.0020	.0020	.0017	.0094	.0099	5.3
	128	.0369	.0417	.0012	.0014	.0016	.0014	.0095	.0105	10.5

Table 4.2. Estimated Miss Ratio for Split Cache
(Set size = 2, Sample size = 80000)

Cache	Block	Non-Stack		Stack		Instruction		Overall		
size(KB)	size(B)	actual	est.	actual	est.	actual	est.	actual	est.	error(%)
32K	16	.0489	.0530	.0031	.0031	.0058	.0058	.0148	.0157	6.1
	32	.0412	.0439	.0023	.0022	.0043	.0045	.0122	.0128	4.9
64K	16	.0411	.0437	.0012	.0012	.0033	.0030	.0114	.0117	2.6
	32	.0344	.0366	.0009	.0008	.0025	.0024	.0093	.0097	4.1
	64	.0304	.0318	.0006	.0007	.0019	.0018	.0080	.0083	3.8
128K	16	.0340	.0323	.0010	.0010	.0018	.0011	.0089	.0081	-9.0
	32	.0277	.0277	.0007	.0007	.0013	.0009	.0072	.0069	-4.2
	64	.0242	.0245	.0005	.0005	.0009	.0007	.0061	.0060	-1.6
	128	.0223	.0224	.0005	.0004	.0008	.0008	.0056	.0056	0.0

Table 4.3. Estimated Miss Ratio for Split Cache
(Set size = 4, Sample size = 80000)

Cache	Block	Non-Stack		Stack		Instruction		Overall		
size(KB)	size(B)	actual	est.	actual	est.	actual	est.	actual	est.	error(%)
32K	16	.0472	.0517	.0027	.0022	.0037	.0030	.0133	.0138	3.6
	32	.0382	.0406	.0020	.0018	.0030	.0027	.0107	.0110	2.8
64K	16	.0392	.0417	.0009	.0008	.0016	.0008	.0100	.0101	1.0
	32	.0331	.0350	.0007	.0006	.0012	.0009	.0083	.0085	2.4
	64	.0286	.0298	.0005	.0004	.0010	.0009	.0072	.0073	1.4
128K	16	.0312	.0256	.0007	.0008	.0011	.0002	.0078	.0061	-21.8
	32	.0258	.0247	.0005	.0005	.0007	.0002	.0064	.0058	-9.4
	64	.0232	.0237	.0004	.0004	.0006	.0002	.0057	.0056	-1.8
	128	.0205	.0210	.0004	.0004	.0004	.0002	.0050	.0049	-2.0

error is always less than 10%, except for the 128-Kbyte cache with block size of 16 bytes. For such small block size and large cache size, the number of blocks in the cache is large and the miss ratio is very small. Therefore, even after the first quarter of the sample, the fraction p has not reached the fixed value described in our model. For such cases, even longer samples are required. A simple way to check this is to calculate the fraction p for two different sets of T_1 and T_2, shifted within the sample, to check if the two corresponding values of p are close enough. The estimated value of the miss ratio of each of the parts of the split cache is always found to be close to the actual value, except for the cases where the miss ratio attains very low value. For example, for a cache size of 128 Kbytes and a block size of 32 bytes, the miss ratio of the stack cache which is actually 0.28% is estimated with an error of 25%.

The same cache model is applied to the unified caches chosen in Chapter 3 using the same address trace. The sample size and the sampling interval are chosen as 100,000 references. The results for the caches with fixed block size (=16 bytes) are presented in Table 4.4, while the results for the caches with a fixed number of blocks are in Table 4.5. It is observed in both these cases that this cache model can always estimate the miss ratio of the direct-mapped caches with reasonable accuracy (error less than 10%). For set-associative caches, the results are very satisfactory for the fixed number of blocks in Table 4.5. But when the block size is small (16 bytes) and the cache size is large (64 Kbyte or larger), the number of blocks is also large and this model does not handle the unified cache well (Table 4.5). A larger sample size is suggested for such cases. In the case of split caches, the behavior of each part of the cache is more coherent than the unified cache. As a result, the assumption that the fraction p remains constant holds better in split caches compared to unified caches. Therefore, the cache model works better for split caches.

Table 4.4. Estimated Miss Ratio for Unified Cache (Block size = 16, Sample size = 100000)

Set size	Cache size(KB)	Actual miss ratio	Measured miss ratio	Error (%)
1	8	0.0458	0.0449	-2.0
	16	0.0307	0.0302	-1.6
	32	0.0235	0.0219	-6.8
	64	0.0149	0.0152	2.0
	128	0.0115	0.0111	-3.5
2	8	0.0253	0.0258	2.0
	16	0.0194	0.0193	-0.5
	32	0.0140	0.0136	-2.9
	64	0.0104	0.0095	-8.7
	128	0.0078	0.0067	-14.1
4	8	0.0216	0.0217	0.5
	16	0.0175	0.0174	-0.6
	32	0.0125	0.0118	-5.6
	64	0.0084	0.0067	-20.2
	128	0.0064	0.0048	-25.0

Table 4.5. Estimated Miss Ratio for Unified Cache (Sample size = 100000)

Set size	Cache size(KB)	Block size(byte)	Actual miss ratio	Measured miss ratio	Error (%)
1	8	8	0.0611	0.0603	-1.3
	16	16	0.0307	0.0302	-1.6
	32	32	0.0203	0.0186	-8.4
	64	64	0.0143	0.0154	7.7
	128	128	0.0107	0.0112	4.7
2	8	8	0.0361	0.0369	2.2
	16	16	0.0194	0.0193	-0.5
	32	32	0.0109	0.0106	-2.8
	64	64	0.0067	0.0064	-4.5
	128	128	0.0039	0.0039	0.0
4	8	8	0.0324	0.0325	0.3
	16	16	0.0175	0.0174	-0.6
	32	32	0.0099	0.0096	-3.0
	64	64	0.0056	0.0051	-8.9
	128	128	0.0031	0.0028	-9.6

4.5. Discussion

The priming technique, described in Chapter 3, can be applied only to set-associative, unified caches. This new cache model is much more versatile - it can be applied to both direct-mapped as well as set-associative caches and also to split as well as unified caches. Only when the cache is large and the block size is small, does this model for split caches needs larger samples. In the course of developing this model and measuring the miss ratio of split caches, the following design issues of split caches have been observed.

4.5.1. Design issues

The performances of a split cache and of a unified cache in terms of the miss ratio are compared by considering some typical cache parameters in Table 4.6. The split cache is divided into instruction, stack data and nonstack data parts as described earlier. It is found that the miss ratio of split cache is always significantly higher than its unified counterpart for set-associative caches. However, for direct-mapped caches, the split cache performs considerably better than the unified cache. In the split cache organization, the cache is inherently more associative than the corresponding unified cache, because the different types of blocks, mapping to the same cache address, do not replace each other as in the case of a unified cache [32]. This indicates that at least part of the improvement in the hit ratio due to increasing the set size from 1 (direct-mapping) to 2 or more, may be gained by splitting the cache into several direct-mapped caches.

It is important to note that the split cache organization offers some other advantages, especially higher bandwidth and access time advantages. The performance can be further improved by optimal design of the different parts of the cache. For example, when the set size is 2, split caches of total sizes 64 Kbytes and 128 Kbytes with a block size of 64 bytes and 1/2, 1/4, 1/4 split have miss ratios of 0.0080 and 0.0061 respectively while their unified counterparts have miss ratios of 0.0067 and 0.0046. By choosing optimal

Table 4.6. Comparison of Unified Cache and Split Cache

Set size	Cache size(KB)	Block size(byte)	Miss ratio		
			Unified cache	Split cache	Difference(%)
1	32	32	0.0203	0.0186	-8.4
	64	64	0.0143	0.0127	-11.2
	128	128	0.0107	0.0095	-11.2
2	32	32	0.0109	0.0122	11.9
	64	64	0.0067	0.0080	19.4
	128	128	0.0039	0.0056	43.6
4	32	32	0.0099	0.0107	8.1
	64	64	0.0056	0.0072	28.6
	128	128	0.0031	0.0050	61.3

parameters for the different parts, split cache can perform almost as well as, or in some cases, even better than the unified cache. A 72 Kbyte split cache (Instruction: size 32K, block 64 bytes; Nonstack: size 32 K, block 128 bytes; Stack: size 8K, block 64 bytes) has a miss ratio of 0.0064.

As noted earlier, in the case of the direct-mapped cache, the miss ratio of the split cache has been found to be lower than that of the unified cache. For a cache size of 128 Kbytes, the miss ratios for the split and unified caches are respectively 0.0095 and 0.0107, as seen in Table 4.6. This can be improved by an optimally split, even smaller (112 Kbytes) cache which has a miss ratio of 0.0090 (Instruction: size 64K, block 128 bytes; Nonstack: 32K, block 64 bytes; Stack: 16K, block 128 bytes).

Split cache systems are becoming increasingly common, especially in microcomputer and minicomputer systems [33]. But relatively few systematic studies have been made in this direction. Small split caches can sometimes perform better than their unified counterparts, even for set-associative caches. The advantages of much higher bandwidth, better access time, and optimal and efficient design make these split caches a very strong competitor of unified caches in many applications. Our method of simulation can very efficiently evaluate all types of split caches for further investigation.

CHAPTER 5

CONCLUSIONS

5.1. Summary and Discussion of Results

This thesis has established a sampling-based technique for cache memory simulation. The method was validated on four very different program traces. It provides insight into the actual program behavior by giving *accurate* estimates of both the *mean value* and the *distribution* of the miss ratio. The experimental results show that approximately 35 samples can characterize the program behavior well. A simple cache model is proposed to extend the result of one simulation with fixed sample size to predict the miss ratio for other context-switch intervals. This basic sampling technique is applicable to small caches. In the case of large caches, where the assumption of flushing of the cache across the context switches is not valid, a new method of priming is provided for the sampling technique. The methods for unified caches cannot be used for split caches. Therefore, another cache model is provided for split caches, one which can also be used very effectively for unified caches.

In comparison to this sampling-based method, conventional techniques merely measure the mean miss ratio for one small segment of a program which may not be representative of the actual behavior. By collecting the samples over different parts of program execution, our method can represent actual behavior of the program. The *cost* of the sampling-based method in terms of both memory space and computer time is *significantly lower* than that of conventional techniques. For a trace length of more than 5 million references, this method typically requires collecting only 7% or less of the trace for smaller caches. For large caches, much longer segments of traces are required by the conventional method. For example, if the continuous trace is 100 million addresses long and the sample size is 100,000 references, then only 3.5% of the whole trace is required for the sampling-based method. Thus our method estimates the value of miss ratio accurately with much lower overhead. Trace compression methods of Puzak [14] can be used on the sampled trace to reduce the size even further.

5.2. Scopes of Sampling Technique

Sampling techniques are used very often in evaluating the performance of computer systems. Because of the limited size of the storage to collect data, traces are usually collected in small samples at a time. Thus, a systematic study of the proper use of such samples would be very useful. Unfortunately, there are no such studies to validate these sampling techniques. This study offers a systematic approach towards using sampling for cache memory evaluation. These sampling techniques can be broadly classified into two categories.

(1) When the history of the past events (before the beginning of the sample) do not affect the current events (during the sample), the sampling method is rather straightforward in nature. But it still needs to be validated experimentally. For example, the basic simulation technique of small caches falls in this category.

(2) In other cases, when the current events are dependent on history, knowledge of the architecture along with detailed analysis of its behavioral pattern is needed to develop a specific sampling methodology for that case. Priming of large caches exemplifies such a situation.

There are many promising applications where sampling could be very effectively used to reduce the overhead and to increase the accuracy of these measurements. Some of the numerous applications which will gain from such systematic approaches are as follows.

(1) *Memory Hierarchies:* Different levels of the memory other than the cache memory systems can also be evaluated using such techniques. These methods can greatly

facilitate comparing different memory organizations, management policies, etc.

(2) *Interconnection Networks:* The entire trace length of multiprocessor systems can be enormously long, too long to be stored and simulated by conventional techniques. It has been found that some of these cache models can be easily modified to simulate the interconnection networks of such systems. Related parameters, such as queue length and waiting time of multiprocessor networks, can be evaluated with much lower overhead.

(3) *CPU Performance:* There are many parameters of CPU which cannot be properly evaluated without detailed analysis of the instruction trace, but the final count of some events is not adequate for such cases. Therefore, evaluating effects of pipelining, register window, etc. are good candidates for sampling-based techniques.

Finally, all these experimental validations are dependent on the particular organization of the system and the workloads concerned. As with any trace based analysis, caution should be exercised in extrapolating the results from one case to a radically different situation. More analysis and measurement are needed to set general guidelines for these sampling-based techniques.

REFERENCES

[1] C. Alexander, W. Keshlear, F. Cooper, and F. Briggs, "Cache memory performance in a UNIX environment," *Computer Architecture News*, vol. 14, no. 3, pp. 41-70, June 1986.

[2] A. J. Smith, "Cache memories," *ACM Computing Surveys*, vol. 14, no. 3, pp. 473-530, September 1982.

[3] A. J. Smith, "Cache evaluation and the impact of workload choice," *Proceedings of the 12th Annual International Symposium on Computer Architecture*, vol. 13, no. 3, pp. 64-73, June 1985.

[4] D. W. Clark, "Cache performance in the VAX-11/780," *ACM Transactions on Computer Systems*, vol. 1, no. 1, pp. 24-37, February 1983.

[5] J. S. Emer and D. W. Clark, "A characterization of processor performance in the VAX-11/780," *Proceedings of the 11th Annual International Symposium on Computer Architecture*, vol. 12, no. 3, pp. 301-310, June 1984.

[6] J. R. Spirn, *Program Behavior: Models and Measurements*. New York: Elsevier, Operating and Programming Systems Series, 1977.

[7] J. E. Smith and J. R. Goodman, "Instruction cache replacement policies and organizations," *IEEE Transactions on Computers*, vol. C-34, no. 3, pp. 234-241, March 1985.

[8] W. D. Strecker, "Transient behavior of cache memories," *ACM Transactions on Computer Systems*, vol. 1, no. 4, pp. 281-293, November 1983.

[9] A. Agarwal, M. Horowitz, and J. Hennessy, "An analytical cache model," Computer Systems Lab, Stanford University, Report CSL T. R. 86-304, September 1986.

[10] M. Easton, "Computation of cold start miss ratios," *IEEE Transactions on Computers*, vol. C-27, no. 5, pp. 404-408, May 1978.

[11] I. J. Haikala, "Cache hit ratios with geometric task switch intervals," *Proceedings of the 11th Annual International Symposium on Computer Architecture*, vol. 12, no. 3, pp. 364-371, June 1984.

[12] P. M. Fenwick, "Some aspects of the dynamic behavior of hierarchical memories," *IEEE Transactions on Computers*, vol. C-34, no. 6, pp. 570-573, June 1985.

[13] A. J. Smith, "Two methods for the efficient analysis of memory address trace data," *IEEE Transactions on Software Engineering*, vol. SE-3, no. 1, pp. 94-101, January 1977.

[14] T. R. Puzak, "Analysis of cache replacement-algorithms," Ph.D. dissertation, Univ. of Mass, Amherst, Mass, February 1985.

[15] W. D. Strecker, "Cache memories for PDP-11 family computers," *Proceedings of the Third Annual Conference on Computer Architecture*, pp. 155-158, 1976.

[16] D. W. Clark, "Measurements of dynamic list structure use in Lisp," *IEEE Transactions on Software Engineering*, vol. SE-5, no. 1, pp. 51-59, January 1979.

[17] A. J. Smith, "Line (block) size choice for CPU cache memories," *IEEE Transactions on Computers*, vol. C-36, no. 9, pp. 1063-1075, September 1987.

[18] R. E. Walpole and R. H. Myers, *Probability and Statistics for Engineers and Scientists*. New York: Macmillan Publishing Company, 1985.

[19] K. Hwang and F. A. Briggs. *Computer Architecture and Parallel Processing*. New York: McGraw-Hill, 1984.

[20] W. W. Daniel. *Applied Nonparametric Statistics*. Boston: Houghton Miffin Company, 1978.

[21] R. V. Hogg and E. A. Tanis. *Probability and Statistical Inference*. New York: Macmillan Publishing Company, 1983.

[22] A. Agarwal, R. L. Sites, and M. Horowitz, "ATUM: A new technique for capturing address traces using microcode," *Proceedings of the 13th Annual International Symposium on Computer Architecture*, vol. 14, no. 3, pp. 119-127, June 1986.

[23] J. Voldman and L. W. Hoevel, "The Fourier-cache connection," *Proceedings of the Compcon*, vol. IEEE Order Number 341, pp. 344-354, Spring 1981.

[24] M. Kobayashi, "An empirical study of task switching locality in MVS," *IEEE Transactions on Computers*, vol. C-35, no. 8, pp. 720-731, August 1986.

[25] P. Favre and R. Kuhne, "Fast memory organization," *IBM Technical Disclosure Bulletin*, vol. 21, no. 2, pp. 649-650, July 1978.

[26] G. Radin, "The 801 minicomputer," *IBM Journal of Research and Development*, vol. 27, no. 3, pp. 237-245, May 1983.

[27] M. E. Hopkins, "A perspective on the 801/reduced instruction set computer," *IBM Systems Journal*, vol. 26, no. 1, pp. 107-121, 1987.

[28] Burroughs Corporation, *B7800 Information Processing Systems Reference Manual*, 1979.

[29] MIPS Computer Systems, Inc., *Performance Brief 2.2: MIPS M/800 and M/500 Systems*, Sunnyvale, California, April 1987.

[30] C. Barney, "RISC supermini has stellar performance," *Electronics*, vol. 56, no. 16, pp. 149-150, August 1983.

[31] Gould Inc., *Gould PN9080 Technical Manual*, Rolling Hills, Illinois, 1985.

[32] I. J. Haikala and P. H. Kutvonen, "Split cache organizations," Dept. of Computer Science, Univ. of Helsinki, Finland, Report C-1984-40, 1984.

[33] A. Kabakibo, V. Milutinovic, A. Silbey, and B. Furht, "A survey of cache memory in modern microcomputer and minicomputer systems," *Tutorial on Computer Architecture*, vol. IEEE Computer Society order number 704, pp. 210-227, 1986.

APPENDIX

FREQUENCY BAR CHARTS OF MISS RATIO

(a) Number of samples = 36

```
MIDPOINT
   Y                                     FREQ   CUM.    PERCENT     CUM.
                                                FREQ               PERCENT
0.045  |                                  0      0       0.00        0.00
0.055  |************                      6      6      16.67       16.67
0.065  |************                      6     12      16.67       33.33
0.075  |********************             10     22      27.78       61.11
0.085  |**************                    7     29      19.44       80.56
0.095  |******                            3     32       8.33       88.89
0.105  |****                              2     34       5.56       94.44
0.115  |**                                1     35       2.78       97.22
0.125  |**                                1     36       2.78      100.00
       ----+---+---+---+---+
           2   4   6   8  10
            FREQUENCY
```

(b) Continuous Trace

```
MIDPOINT
   Y                                     FREQ   CUM.    PERCENT     CUM.
                                                FREQ               PERCENT
0.045  |**                                 12     12     2.09        2.09
0.055  |***********                        57     69     9.93       12.02
0.065  |*************************         125    194    21.78       33.80
0.075  |************************          118    312    20.56       54.36
0.085  |***********************           115    427    20.03       74.39
0.095  |******************                 88    515    15.33       89.72
0.105  |*********                          45    560     7.84       97.56
0.115  |**                                 12    572     2.09       99.65
0.125  |                                    2    574     0.35      100.00
       ----+---+---+---+---+---+-
          20  40  60  80 100 120
               FREQUENCY
```

Fig. A.1. Frequency bar chart of miss ratio for FSIM : Context-switch interval = 5K

(a) Number of samples = 36

```
MIDPOINT
   Y                                       FREQ   CUM.    PERCENT    CUM.
                                                  FREQ              PERCENT
0.0335    |                                  0      0       0.00       0.00
0.0405    |******                            3      3       8.33       8.33
0.0475    |******                            3      6       8.33      16.67
0.0545    |**************                    7     13      19.44      36.11
0.0615    |************************         12     25      33.33      69.44
0.0685    |**********                        5     30      13.89      83.33
0.0755    |****                              2     32       5.56      88.89
0.0825    |******                            3     35       8.33      97.22
0.0895    |**                                1     36       2.78     100.00
          |
          ----+---+---+---+---+---+
              2   4   6   8  10  12
                  FREQUENCY
```

(b) Continuous Trace

```
MIDPOINT
   Y                                          FREQ   CUM.    PERCENT    CUM.
                                                     FREQ              PERCENT
0.0335    |****                                  8      8       2.79       2.79
0.0405    |**************                       27     35       9.41      12.20
0.0475    |******************                   36     71      12.54      24.74
0.0545    |*****************************        57    128      19.86      44.60
0.0615    |*******************************      62    190      21.60      66.20
0.0685    |********************                 39    229      13.59      79.79
0.0755    |******************                   35    264      12.20      91.99
0.0825    |********                             15    279       5.23      97.21
0.0895    |****                                  8    287       2.79     100.00
          |
          -----+-----+-----+-----+-----+-----+-
              10    20    30    40    50    60
                       FREQUENCY
```

Fig. A.2. Frequency bar chart of miss ratio for FSIM : Context-switch interval = 10K

(a) Number of samples = 36

```
MIDPOINT
   Y                                                  FREQ   CUM.    PERCENT     CUM.
                                                             FREQ              PERCENT
 0.008   |••••••••••                                    5       5     14.71      14.71
 0.024   |••••••••••••••••••••••••                     12      17     35.29      50.00
 0.040   |                                              0      17      0.00      50.00
 0.056   |••                                            1      18      2.94      52.94
 0.072   |••••••                                        3      21      8.82      61.76
 0.088   |••••••••••••                                  6      27     17.65      79.41
 0.104   |••••••                                        3      30      8.82      88.24
 0.120   |••••                                          2      32      5.88      94.12
 0.136   |••••                                          2      34      5.88     100.00
         |
          ----+---+---+---+---+---+
              2   4   6   8  10  12
                FREQUENCY
```

(b) Continuous Trace

```
MIDPOINT
   Y                                                  FREQ   CUM.    PERCENT     CUM.
                                                             FREQ              PERCENT
 0.008   |••••••••••••••                               35      35     13.01      13.01
 0.024   |••••••••••••••••••••••••••••                 71     106     26.39      39.41
 0.040   |••••••••••••                                 30     136     11.15      50.56
 0.056   |••••                                         11     147      4.09      54.65
 0.072   |••••                                          9     156      3.35      57.99
 0.088   |••••••••••••••                               35     191     13.01      71.00
 0.104   |•••••••••••                                  28     219     10.41      81.41
 0.120   |•••••••••••••••                              38     257     14.13      95.54
 0.136   |•••••                                        12     269      4.46     100.00
         |
          ----+----+----+----+----+----+----+
             10   20   30   40   50   60   70
                   FREQUENCY
```

Fig. A.3. Frequency bar chart of miss ratio for DRC : Context-switch interval = 10K

(a) Number of samples = 36

```
MIDPOINT
   Y                                             FREQ   CUM.     PERCENT     CUM.
                                                        FREQ              PERCENT
0.007   |********                                   4      4     11.76      11.76
0.021   |************************                  12     16     35.29      47.06
0.035   |**                                         1     17      2.94      50.00
0.049   |**                                         1     18      2.94      52.94
0.063   |******                                     3     21      8.82      61.76
0.077   |****                                       2     23      5.88      67.65
0.091   |************                               6     29     17.65      85.29
0.105   |**********                                 5     34     14.71     100.00
0.119   |                                           0     34      0.00     100.00
        ----+---+---+---+---+---+
            2   4   6   8  10  12
              FREQUENCY
```

(b) Continuous Trace

```
MIDPOINT
   Y                                                     FREQ   CUM.     PERCENT     CUM.
                                                                FREQ              PERCENT
0.007   |******************                                18     18     13.43      13.43
0.021   |********************************                  32     50     23.88      37.31
0.035   |****************                                  16     66     11.94      49.25
0.049   |********                                           8     74      5.97      55.22
0.063   |*                                                  1     75      0.75      55.97
0.077   |*********                                          9     84      6.72      62.69
0.091   |********************                              20    104     14.93      77.61
0.105   |**************************                        26    130     19.40      97.01
0.119   |****                                               4    134      2.99     100.00
        ----+----+----+----+----+----+--
            5   10   15   20   25   30
                FREQUENCY
```

Fig. A.4. Frequency bar chart of miss ratio for DRC : Context-switch interval = 20K

(a) Number of samples = 35

```
 MIDPOINT
    Y                                                  FREQ   CUM.   PERCENT    CUM.
                                                              FREQ             PERCENT
0.0155   |****************                                4      4    11.43     11.43
0.0325   |****************************                    7     11    20.00     31.43
0.0495   |****                                            1     12     2.86     34.29
0.0665   |                                                0     12     0.00     34.29
0.0835   |************************                        6     18    17.14     51.43
0.1005   |********************************                8     26    22.86     74.29
0.1175   |****************************                    7     33    20.00     94.29
0.1345   |********                                        2     35     5.71    100.00
0.1515   |                                                0     35     0.00    100.00
         ----+---+---+---+---+---+---+---+
             1   2   3   4   5   6   7   8
                   FREQUENCY
```

(b) Continuous Trace

```
 MIDPOINT
    Y                                                  FREQ   CUM.   PERCENT    CUM.
                                                              FREQ             PERCENT
0.0155   |***************                                37     37    10.57     10.57
0.0325   |****************************                   69    106    19.71     30.29
0.0495   |******                                         15    121     4.29     34.57
0.0665   |******                                         15    136     4.29     38.86
0.0835   |**************                                 36    172    10.29     49.14
0.1005   |*******************************                77    249    22.00     71.14
0.1175   |***************************                    68    317    19.43     90.57
0.1345   |**********                                     26    343     7.43     98.00
0.1515   |***                                             7    350     2.00    100.00
         ----+---+---+---+---+---+---+---
            10  20  30  40  50  60  70
                  FREQUENCY
```

Fig. A.5. Frequency bar chart of miss ratio for ELI : Context-switch interval = 5K

(a) Number of samples = 35

```
MIDPOINT
   Y                                          FREQ   CUM.    PERCENT      CUM.
                                                     FREQ              PERCENT
 0.006   |******                                3       3       8.57       8.57
 0.018   |                                      0       3       0.00       8.57
 0.030   |****************                      8      11      22.86      31.43
 0.042   |**                                    1      12       2.86      34.29
 0.054   |**                                    1      13       2.86      37.14
 0.066   |****                                  2      15       5.71      42.86
 0.078   |********************                 10      25      28.57      71.43
 0.090   |****************                      8      33      22.86      94.29
 0.102   |****                                  2      35       5.71     100.00
         |
         ----+---+---+---+---+
             2   4   6   8  10
               FREQUENCY
```

(b) Continuous Trace

```
MIDPOINT
   Y                                          FREQ   CUM.    PERCENT      CUM.
                                                     FREQ              PERCENT
 0.006   |********                              8       8       9.20       9.20
 0.018   |**                                    2      10       2.30      11.49
 0.030   |*****************                    17      27      19.54      31.03
 0.042   |****                                  4      31       4.60      35.63
 0.054   |*                                     1      32       1.15      36.78
 0.066   |*******                               7      39       8.05      44.83
 0.078   |****************************         28      67      32.18      77.01
 0.090   |**************                       14      81      16.09      93.10
 0.102   |******                                6      87       6.90     100.00
         |
         ----+----+----+----+----+---
             5   10   15   20   25
               FREQUENCY
```

Fig. A.6. Frequency bar chart of miss ratio for ELI : Context-switch interval = 20K

(a) Number of samples = 33

```
MIDPOINT
   Y                                                              FREQ   CUM.    PERCENT    CUM.
                                                                         FREQ               PERCENT
 0.007   |******************                                        9       9     27.27      27.27
 0.021   |**                                                        1      10      3.03      30.30
 0.035   |********************************                         16      26     48.48      78.79
 0.049   |                                                          0      26      0.00      78.79
 0.063   |**                                                        1      27      3.03      81.82
 0.077   |**                                                        1      28      3.03      84.85
 0.091   |********                                                  4      32     12.12      96.97
 0.105   |**                                                        1      33      3.03     100.00
 0.119   |                                                          0      33      0.00     100.00
         ----+---+---+---+---+---+---+---+
             2   4   6   8  10  12  14  16
                  FREQUENCY
```

(b) Continuous Trace

```
MIDPOINT
   Y                                         FREQ   CUM.    PERCENT    CUM.
                                                    FREQ               PERCENT
 0.007   |***********                         52      52     26.53      26.53
 0.021   |**                                  12      64      6.12      32.65
 0.035   |*******************                 94     158     47.96      80.61
 0.049   |                                     1     159      0.51      81.12
 0.063   |                                     2     161      1.02      82.14
 0.077   |**                                   9     170      4.59      86.73
 0.091   |***                                 16     186      8.16      94.90
 0.105   |**                                   9     195      4.59      99.49
 0.119   |                                     1     196      0.51     100.00
         ----+----+----+----+---
            20   40   60   80
              FREQUENCY
```

Fig. A.7. Frequency bar chart of miss ratio for RRL : Context-switch interval = 10K

(a) Number of samples = 33

```
 MIDPOINT
    Y                                               FREQ   CUM.    PERCENT    CUM.
                                                           FREQ              PERCENT
 0.008    |******************                          9      9     27.27      27.27
 0.018    |                                            0      9      0.00      27.27
 0.028    |********************************           16     25     48.48      75.76
 0.038    |**                                          1     26      3.03      78.79
 0.048    |                                            0     26      0.00      78.79
 0.058    |**                                          1     27      3.03      81.82
 0.068    |****                                        2     29      6.06      87.88
 0.078    |****                                        2     31      6.06      93.94
 0.088    |****                                        2     33      6.06     100.00
          ----+---+---+---+---+---+---+---+
              2   4   6   8  10  12  14  16
                     FREQUENCY
```

(b) Continuous Trace

```
 MIDPOINT
    Y                                        FREQ   CUM.    PERCENT    CUM.
                                                    FREQ              PERCENT
 0.008    |*************                      26     26     26.53      26.53
 0.018    |*                                   1     27      1.02      27.55
 0.028    |*************************          49     76     50.00      77.55
 0.038    |**                                  3     79      3.06      80.61
 0.048    |**                                  3     82      3.06      83.67
 0.058    |*                                   1     83      1.02      84.69
 0.068    |**                                  3     86      3.06      87.76
 0.078    |***                                 5     91      5.10      92.86
 0.088    |****                                7     98      7.14     100.00
          -----+----+----+----+----+
              10   20   30   40   50
                   FREQUENCY
```

Fig. A.8. Frequency bar chart of miss ratio for RRL : Context-switch interval = 20K

VITA

Subhasis Laha was born in Calcutta, India, on November 6, 1958. He received his B.Tech. degree in Electrical Engineering from the Indian Institute of Technology, Kharagpur, India, in 1981. He obtained the M.S. degree in Electrical Engineering from the University of Illinois in 1984. At the University of Illinois, he was employed as a research assistant with the Computer Systems Group at the Coordinated Science Laboratory from 1982 to 1987. After completing his doctoral degree, he will join the AT&T Information Systems, Naperville, Illinois, as a member of the technical staff.