To appear in the proceedings of "ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems", May 1991.

A Model for Estimating Trace-Sample Miss Ratios*

David A. Wood Mark D. Hill R. E. Kessler

Computer Science Department
University of Wisconsin – Madison
1210 West Dayton Street
Madison, WI 53706
david@cs.wisc.edu

Abstract

Unknown references, also known as cold-start misses, arise during trace-driven simulation of uniprocessor caches because of the unknown initial conditions. Accurately estimating the miss ratio of unknown references, denoted by μ, is particularly important when simulating large caches with short trace samples, since many references may be unknown.

In this paper we make three contributions regarding μ. First, we provide empirical evidence that μ is much larger than the overall miss ratio (e.g., 0.40 vs. 0.02). Prior work suggests that they should be the same. Second, we develop a model that explains our empirical results for long trace samples. In our model, each block frame is either *live*, if its next reference will hit, or *dead*, if its next reference will miss. We model each block frame as an alternating renewal process, and use the renewal-reward theorem to show that μ is simply the fraction of time block frames are dead. Finally, we extend the model to handle short trace samples and use it to develop several estimators of μ. Trace-driven simulation results show these estimators lead to better estimates of overall miss ratios than do previous methods.

Index Terms – Cache memory, performance evaluation, cold-start behavior, trace-driven simulation, sampling techniques

*This work is supported in part by the National Science Foundation (MIPS-8957278 and CCR-8902536), A.T.& T. Bell Laboratories, Cray Research Foundation, Digital Equipment Corporation, and the graduate school at the University of Wisconsin–Madison. R. E. Kessler has been supported by a summer internship at Digital Equipment Corporation, and graduate fellowships from the National Science Foundation and the University of Wisconsin-Madison.

1 Introduction

A cache is a high-speed buffer that holds recently-used parts of main memory[Smit82]. When a reference is not found in the cache (a *miss*), the *block* of data that contains it is transferred from main memory and stored in a *block frame* in the cache. We say a block frame is *referenced* if it contains the requested block or obtains it as a result of a miss. Blocks and block frames are usually partitioned into S *sets* so that only the block frames in the reference's set must be searched. An A-*way set-associative* cache has A block frames per set (*associativity* A) and SA block frames overall.

Trace-driven simulation is the most-commonly-used technique for evaluating the performance of cache memories[Smit82]. Trace-sampling is a refinement that simulates multiple trace samples to estimate the miss ratio of a long base trace[Laha88]. Since most programs exhibit non-stationary miss ratios over very long intervals[Borg90], a single long trace sample may not be representative of the entire program. Instead, if we can accurately estimate the miss ratio of short samples, then the average over multiple short samples should more accurately predict the true miss ratio.

Any estimate of the true miss ratio should be unbiased (expected error of zero) and as accurate as possible (small mean squared error)[Bick77]. A key problem in finding an unbiased miss ratio estimator is that the simulator does not know which blocks reside in the cache at the beginning of a trace sample. Thus, when the simulator references a block frame for the first time, it cannot determine whether or not the reference actually misses. These potential misses have been called *cold-start misses*[East78] because they occur at the beginning of a simulation; we prefer to call them *unknown references* since they may or may not be actual misses.

Two techniques have traditionally been used to deal with this problem: (1) make trace samples long enough so that the cold-start bias is acceptably small, or (2) record metrics only after this bias is eliminated. The

first technique works because the number of unknown misses cannot exceed the number of block frames in the cache, limiting the absolute error. The second technique exploits the fact that the current cache state does not depend upon the initial contents once the trace has referenced every block frame. Thus the bias is eliminated by recording metrics only after the cache is full.

Recent trends toward larger caches render the above techniques undesirable by requiring extremely long trace samples. For example, consider a cache with 512 block frames and a 0.05 miss ratio estimate for a 100,000-reference trace. The bias in this estimate cannot exceed 10 percent ($512/(100,000 * 0.05) \approx 10\%$). Future caches, however, may have 128K block frames and 0.001 miss ratios, which require one billion references to achieve the same 10 percent maximum bias. The second technique fares no better; filling the large cache requires an absolute minimum of 128,000 references, and many times that in practice.

More recently, Laha, et al.[Laha88], and Stone [Ston90] have proposed similar techniques that warm-up each set independently. Rather than waiting for the entire cache to fill, they record metrics for each set that has filled. By not counting unknown references, these approaches implicitly assume that unknown references behave the same as randomly selected references.

In the central result of this paper we show that unknown references are special, and exhibit a much higher miss ratio than random references. We introduce a renewal-theoretic model to explain this surprising result. The model defines a block frame as *live* if its next reference hits, and as *dead* if its next reference misses. Using the renewal-reward theorem, we show that the miss ratio of unknown references is simply the fraction of time a block frame is dead. We use trace-driven simulation to validate this model for traces long enough to reference every block frame.

We then extend the model to handle shorter trace-samples. However, this second model relies upon the distribution of dead times, not just the ratio of means, and is much more difficult to compute. We introduce four approximations and show that one yields good results for sufficiently long samples. More work is required to determine when samples are long enough to provide acceptable error. For our data, if at least 60% of the block frames have been touched and there have been as many misses as there are block frames, then the error in the overall miss ratio is generally less than 10%. A second estimate produces slightly less accurate results, but is useful since it is trivial to compute.

In the next section, we review previously-proposed estimators, and show that the true miss ratio is much greater than intuition suggests. Section 3 introduces the renewal-theoretic model and explains this surprising observation. Section 4 applies the model to estimate the true miss ratio for long trace samples, and Section 5 generalizes it to support short trace samples. Both Section 4 and Section 5 present trace-driven simulation results to support our models. Finally, Section 6 summarizes our results and proposes future extensions to the model.

2 Estimating Miss Ratios of Unknown References

To obtain accurate miss ratio estimates using trace sampling, we need an accurate, unbiased estimator for the miss ratios of short trace samples. To do this, we focus on finding an accurate, unbiased estimator for the miss ratio of unknown references. Let the true miss ratio of a cache, denoted by m, be the fraction of references that would miss in a real cache during the execution of a particular trace sample. The true miss ratio depends upon the miss ratio of unknown references, which we denote by μ:

$$m = \frac{M + \mu U}{R} \tag{1}$$

where M is the number of references known to miss, U is the number of unknown references, and R is the total number of references in the sample.

The only two ways a simulation can exactly compute the true miss ratio is to either eliminate the unknown references or exactly compute μ. Laha, et al.[Laha88], have explored the first approach; in this paper, we focus on the problem of accurately predicting μ, the miss ratio of the unknown references.

While the miss ratio is not usually characterized in this way, researchers have commonly used several approximations to μ. The simplest and most common estimate assumes that all unknown references miss ($\hat{\mu}_{cold} = 1$).

$$\hat{m}_{cold} = \frac{M + 1 \times U}{R} = \frac{M + U}{R} \tag{2}$$

In this case, the unknown references are referred to as *cold-start* misses. This estimate has been widely used, and provides acceptable accuracy if M is much larger than U. However, this constraint is rarely met for large caches, resulting in overly pessimistic estimates of the miss ratio.

The other extreme is to assume that all unknown references hit ($\hat{\mu}_{hot} = 0$). Intuition suggests that this *hot-start* estimate is better for large caches, since they have low steady-state miss ratios.

$$\hat{m}_{hot} = \frac{M + 0 \times U}{R} = \frac{M}{R} \tag{3}$$

Trace	True m	True μ	Estimate of m (% Relative Error)					
			$\hat{m}_{cold}$		$\hat{m}_{hot}$		$\hat{m}_{ss}$	
mult1	0.024	0.353	0.039	(61.3%)	0.016	(-33.4%)	0.016	(-31.9%)
mult2	0.016	0.339	0.028	(76.5%)	0.010	(-39.3%)	0.010	(-38.1%)
tv	0.035	0.481	0.055	(55.4%)	0.017	(-51.4%)	0.018	(-49.5%)
sor	0.027	0.932	0.028	(6.0%)	0.005	(-81.6%)	0.005	(-81.2%)
tree	0.010	0.142	0.027	(167.6%)	0.007	(-27.8%)	0.007	(-26.4%)

Table 1: Comparison of Estimates of Miss Ratio m.

This table shows the measured and estimated miss ratio m, together with the percent relative error ($100\%*(\hat{m}-m)/m$) of the estimates. The true value of μ is much greater than m, indicating that the steady-state assumption is false. Each data point is the mean of several thousand samples, each containing 50,000 references. The cache configuration is 64K-bytes, direct-mapped, with 16-byte blocks.

However, $\hat{m}_{hot}$ has the unfortunate property of always under predicting the true miss ratio, which could lead to optimistic cache designs. Nevertheless, the hot-start and cold-start miss ratios are useful as bounds, since the true miss ratio must lie between them[Ston90]. For sufficiently large samples, i.e., large values of M, the relative difference between $\hat{m}_{cold}$ and $\hat{m}_{hot}$ becomes small, making it reasonable to ignore the difference.

A more intuitive estimate assumes that unknown references miss at the same rate as all other references. If we pick a single reference at random, it is no more or less likely to miss than any other, thus it misses with probability m ($\hat{\mu}_{ss} = \hat{m}_{ss}$):

$$\hat{m}_{ss} = \frac{M + \hat{m}_{ss}U}{R} = \frac{M}{R-U} \tag{4}$$

Stone originally proposed this *steady-state* estimate for direct-mapped caches[Ston90], which is equivalent to excluding the unknown references from the computation of the miss ratio. He used a different justification, arguing that references should only be counted if they access a *primed* set. As originally defined by Laha et al.[Laha88], a set containing A block frames is primed once it has received A unknown references. A simulation can only be sure that a reference hits or misses if it accesses a primed set. Since a set in a direct-mapped cache is primed after its first (unknown) reference, Stone ignores a total of U references.

Stone and Laha et al. generalize this estimate for set-associative caches, but neither model is very appealing since both exclude references that are known to hit. For example, suppose the first reference to a particular set touches block B. While the first reference is unknown, subsequent references to block B will hit (at least during the interval before the set becomes primed). Yet neither Stone's nor Laha et al.'s model includes these references in the miss ratio estimate.

Table 1 presents experimental data comparing the true value of m with the three estimates: $\hat{m}_{cold}$, $\hat{m}_{hot}$, and $\hat{m}_{ss}$. It contains data from five traces on a 64K-byte direct-mapped cache with 16-byte blocks. Each estimate is the arithmetic mean of between 1900 and 3500 contiguous trace contiguous samples, each 50,000 references long. Samples are drawn from a set of long traces, described more fully in Section 4.1. The initial references of each trace are used only to completely warm up the cache, allowing an exact computation of the true miss ratio for the remaining references. The remaining references, an average of 130 million per trace, are then broken into individual samples.

Despite the intuitive argument given above, Table 1 shows that the steady-state estimate predicts m as poorly as the cold-start and hot-start estimates. The assumption that μ equals m is clearly false: the data in columns two and three show that the true value of μ is much larger than the true miss ratio m. Furthermore, the inaccuracy of the estimates for μ introduce a significant error into the estimates for the total miss ratio m.

How is it possible that μ is much larger than m? Consider a cache with 1024 block frames and a synthetic trace-sample that references a block 20 times, then moves to the next block and references it 20 times, etc. The trace's true miss ratio, m, is 5%, because the probability that a random reference misses is 1 in 20. The miss ratio for unknown references, μ, however, is either 99.9% or 100%, depending upon whether the first reference of the sample misses. The first references to all other block frames miss every time. In the next section we develop a model that explains why μ and m are inherently different.

3 A Simple Renewal-Theoretic Model

In this section, we introduce a model for estimating μ, the fraction of unknown references that actually miss. This section is only concerned with the steady-state behavior of a single block frame in the cache. The next section applies the steady-state model to the behavior of an entire cache.

Cache performance is traditionally measured with the miss ratio: the fraction of references that miss in the cache. However, an equivalent alternative is to view the performance using the reciprocal of the miss ratio: the average number of references per miss. This view focuses our attention on the amount of time a block spends in the cache.

Consider a single block frame. We say that the jth *generation* G_j begins immediately after the jth miss occurs, and a new block is brought into the block frame. This generation ends, and a new one begins, when the block is replaced. The length of a generation is the total number of references in the trace between the misses that begin and end the generation. A generation includes the miss that ends it, but not the miss that begins it.

A generation G_j consists of two phases, a *live time* L_j and a *dead time* D_j, thus $G_j = L_j + D_j$. A block frame is said to be *live* if the block it contains will be referenced again before being replaced. Conversely, a block frame is said to be *dead* if the next reference to it will miss. We say a block frame is *referenced* only if it contains the requested block or obtains it as a result of a miss. Thus each address in the trace references exactly one block frame. A live time L_j begins a generation G_j, and a dead time D_j begins immediately after the last reference to the block. Both the generation and the dead time end when the block is replaced.

Now consider an arbitrary block frame at some random time t, as measured in references to all block frames. If the block frame is live, as at time t_1 in Figure 1, then the next reference to that block frame hits. Conversely, if the block frame is dead, as at time t_2, then the next reference to the block frame misses. Thus at a random time t, the probability that the next reference to a block frame misses is simply the probability that time t falls within a dead time.

Under this model, a block frame alternates between being live and dead. With a few simplifying assumptions, stated below, we can model the block frame as an alternating renewal process, and use standard renewal theory to compute the probability that time t occurs during a dead time. Let us assume that a miss to a block frame constitutes a renewal point, that is, that the process starts over independent of the past. Furthermore, assume that the live times L_j and dead times D_j are random variables drawn from two arbitrary distributions. Subsequent live times and dead times are independent, and identically distributed, although a dead time D_j may depend upon the preceding live time L_j. In practice, these assumptions are rarely strictly true for individual cache block frames. Nevertheless, they are close enough that the model yields good results, as shown in the next section.

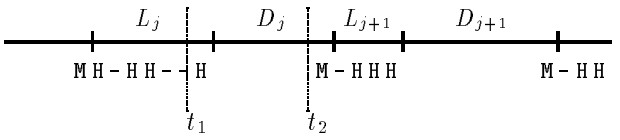

Figure 1: Live and Dead Times

This figure illustrates the live and dead times for an arbitrary block frame in the cache. An "M" denotes a miss, an "H" a hit, a "-" indicates a reference to another block frame during a live time, and a blank marks a dead time. Suppose we randomly pick time t_1. Since it falls within a live time, the next reference to the block frame must hit. Conversely, if we randomly pick time t_2, the next reference to the block frame will miss, since t_2 falls within a dead time.

The renewal-reward theorem states that random events see time-average behavior[Wolf89, Ross83]. In other words, the probability that a random time t falls within a block frame's dead time is simply the fraction of time that the block frame is dead. Mathematically, this is expressed as:

$$P\{\text{Random time } t \text{ lands during a dead time}\} = \frac{E[D_j]}{E[L_j] + E[D_j]} = \frac{E[D_j]}{E[G_j]} \quad (5)$$

where $E[X]$ denotes the expected value of a random variable X. Since the probability of landing in a dead time for some block frame i is simply the probability that the next reference to that block frame misses, we can extend our model to provide a theoretical basis for estimating μ. While this result may seem obvious, we need the formalism of renewal theory to extend the model in Section 5.

In related work, Puzak[Puza85] and Mendelson, Thiébaut, and Pradhan[Mend90] define the terms live and dead slightly differently. Their definitions apply to blocks, rather than block frames: they call a block live if it is in the cache and will be referenced again, and call it dead otherwise. Puzak analyzes the effect of replace-

ment policies on dead blocks. Mendelson, Thiébaut, and Pradhan use their definitions as part of an analytic model of cache performance. Neither examine the relationship between dead blocks and the miss ratio of unknown references.

4 A Model for Long Trace Samples

This section applies the renewal-theoretic model to estimate the miss ratio of long trace samples. We assume here that the trace samples are sufficiently long that they reference every block frame in a cache at least once. The next section examines the complications caused by short samples, which do not meet this requirement.

To calculate the miss ratio for a trace sample, we must know the fraction of unknown references that miss, μ. But this is just the fraction of block frames that are dead at the start of the sample. If we assume that the sample starts at a random time t and that live and dead times are identically distributed for all block frames, then we can apply our result from the previous section. Let $\hat{\mu}_{long}$ be our estimate of μ for long samples, and let V_i be a random variable that is 1 if the unknown reference to block frame i misses and 0 otherwise. Assume that the V_i are identically distributed, S is the number of sets, and A is the associativity. Then,

$$\begin{aligned}
\hat{\mu}_{long} &= E[\mu] \qquad (6)\\
&= E\left[\frac{\text{Unknown References that Miss}}{\text{Unknown References}}\right]\\
&= \frac{1}{SA}E\left[\sum_{i=1}^{SA} V_i\right] = \frac{1}{SA}\sum_{i=1}^{SA} E[V_i]\\
&= \frac{1}{SA}\sum_{i=1}^{SA} \frac{E[D_j]_i}{E[G_j]_i} = \frac{1}{SA}\sum_{i=1}^{SA} \frac{E[D_j]}{E[G_j]}\\
&= \frac{E[D_j]}{E[G_j]}
\end{aligned}$$

where $E[D_j]_i$ and $E[G_j]_i$ are the expected dead and generation times for block frame i, respectively. In words, if block frames have identical live and dead time distributions, then the fraction of unknown references that miss is simply the fraction of block frames that are dead at the start of the sample. But this is simply the fraction of time a single block frame is dead. Of course the live and dead times may not actually be identically distributed, but in the next section we show that this is not a first-order effect.

We illustrate the implications of this result by presenting an intuitive argument that the steady-state assumption is false, i.e., that $\mu \geq m$. An average generation includes one miss to a particular block frame i and $E[G_j]$ references scattered across all block frames. Since there are always SA active generations, one for each block frame, we expect SA misses for every $E[G_j]$ references. This yields a miss ratio $m = \frac{SA}{E[G_j]}$. Now consider a fully-associative cache ($S = 1$) with least-recently-used (LRU) replacement. After a block frame becomes dead, there must be A other misses before it is selected for replacement. Since there can be at most one miss per reference, the minimum dead time is simply A references.

$$\begin{aligned}
\hat{\mu}_{long} &= \frac{E[D_j]}{E[G_j]}\\
m &= \frac{SA}{E[G_j]}\\
\text{but } \min(D_j) &= A = SA\\
\text{Therefore } \hat{\mu}_{long} &\geq m
\end{aligned}$$

□

Note that $\hat{\mu}_{long}$ equals m if and only if all references miss in the cache. Since miss ratios are typically low, in practice $\hat{\mu}_{long}$ is much greater than m.

The argument for direct-mapped and set-associative caches is somewhat weaker. Consider a synthetic reference stream that accesses each set with equal probability. With such a stream the mean time between references to a set is simply S. Once a block frame in a particular set becomes dead, there must be A misses to that set before the block frame is replaced (assuming LRU). Thus, if the cache never hits the mean dead time is SA references. Since caches do hit, and the principles of locality state that references tend to cluster, $\hat{\mu}_{long}$ tends to be much larger than m. However, pathologic reference patterns, that allow μ less than m, are possible for set-associative caches. But as we show in the next section, most real programs exhibit the posited behavior.

This model provides a theoretical basis for the empirical observation that μ is greater than m. However, the simplifying assumptions cast doubt on the accuracy of the model. And without making much stronger assumptions about the reference distribution, we cannot say anything stronger about $\frac{E[D_j]}{E[G_j]}$. Instead, we use trace-driven simulation to validate the model and determine the relationship between $E[D_j]$, $E[G_j]$, and SA.

4.1 Empirical Validation

In this section, we present results from a trace-driven simulation experiment in which we computed the average live times and dead times for various cache sizes. We extended our cache simulator to maintain the time (reference number) of the last miss and last reference

Cache		Address Trace				
Size	Assoc	mult1	mult2	tv	sor	tree
4K	1	0.56	0.51	0.77	0.75	0.57
4K	2	0.57	0.50	0.81	0.72	0.54
4K	4	0.58	0.50	0.79	0.69	0.55
16K	1	0.54	0.66	0.80	0.92	0.50
16K	2	0.50	0.64	0.79	0.92	0.49
16K	4	0.50	0.63	0.77	0.92	0.51
64K	1	0.62	0.69	0.60	0.98	0.67
64K	2	0.62	0.74	0.55	0.98	0.63
64K	4	0.63	0.76	0.62	0.98	0.60

Table 2: True Miss Ratio of Unknown References μ

This table presents the measured values of μ for 5-million reference samples. Each data point represents the arithmetic mean of between 19 and 35 samples. The samples are long enough that 95% of the block frames are referenced, on average, for all but two of the trace/cache combinations. The two exceptions are outlined with boxes.

Cache		Address Trace				
Size	Assoc	mult1	mult2	tv	sor	tree
4K	1	9%	10%	1%	3%	2%
4K	2	7%	8%	1%	3%	-2%
4K	4	5%	10%	4%	3%	-2%
16K	1	13%	-5%	<1%	1%	<1%
16K	2	18%	-3%	1%	<1%	2%
16K	4	16%	-8%	4%	4%	<1%
64K	1	10%	-3%	12%	<1%	<1%
64K	2	11%	-5%	7%	<1%	11%
64K	4	11%	-5%	5%	1%	30%

Table 3: Relative Error in Long Trace Estimate $\hat{\mu}_{long}$.

This table presents the relative error in the long trace estimate $\hat{\mu}_{long} = \frac{E[D_j]}{E[G_j]}$. We compute $\hat{\mu}_{long}$ by calculating the mean dead and generation times of the entire trace and taking their ratio. The relative error exceeds 20% in only one case, and is less than 5% in over half the cases. Note that since both live and dead times vary with cache size and associativity, $\hat{\mu}_{long}$ does not necessarily change monotonically.

to each block frame. On each cache miss, the simulator computes the duration (in references) of the block frame's live and dead times, and updates the summary statistics. To eliminate cold-start effects from our validation experiments, the simulator does not count generations that begin with an unknown reference. The simulator also ignores generations terminated by the end of the trace. Because long generations are more likely to be excluded, as a result of the inspection paradox[Wolf89, Ross83] (also known as the residual life paradox[Klei75]), these simulation artifacts tend to bias the means downwards. We tried to minimize the effects of this bias by using long address traces.

We used five address traces gathered from a DEC Titan[Borg90], that contain multiprogramming behavior, but not operating system references. The traces *mult1* and *mult2* are multiprogramming workloads, consisting of both software development and CAD applications. *Tv* is a VLSI circuit timing verifier, *sor* is an implementation of the successive overrelaxation algorithm, and *tree* is a compiled Scheme program. The traces average 130 million references each (excluding initialization references), for a total of 650 million references. For the largest cache presented in this paper, the traces average over 500 generations (misses) per block frame. Thus the bias introduced by the simulation start-up and ending artifacts should be quite small.

We examine nine cache configurations in this paper, with sizes ranging from 4K-bytes to 64K-bytes and associativity varying from direct-mapped to four-way set-associative. All configurations have 16-byte blocks and use LRU replacement, writeback (main memory only updated on block replacement), and write-allocate (write misses bring blocks into the cache) policies. To exactly compute the true miss ratios, m and μ, we simulated enough references to completely warm-up the cache before calculating any metrics. Then we treated the remaining trace as a collection of contiguous 5-million reference samples. For most combinations of trace and cache configuration, 5 million references is sufficient to meet the long sample requirement of referencing every block frame. Each data point is the arithmetic mean of between 19 and 35 samples.

Table 2 presents the true miss ratio of unknown references for these samples. The values range from 0.50 to as high as 0.98. These data substantiate the observation in Section 2 that μ is much higher than m (which is never greater than 14%, and typically much smaller). Note that μ does not exhibit any clear trends as a function of either cache size or associativity.

Table 3 presents the relative error in our long sample estimate $\hat{\mu}_{long} = \frac{E[D_j]}{E[G_j]}$. We approximate $E[D_j]$ and $E[G_j]$ using the mean dead time and mean generation

time, computed over the entire trace. In over 80% of the cases, the relative error of $\hat{\mu}_{long}$ is within 10%, and in only one case does it exceed 20%. The corresponding error in m is much lower, since U is much less than M for these long samples.

This estimate is substantially better than the other estimates discussed in Section 2. The observed error exists in part because the model depends upon the potentially false independence and identical distribution assumptions. However, even with only a small number of samples, the error is acceptably low. We expect that increasing the number of samples should further reduce the error.

The results of these simulation experiments validate the accuracy of our model. This model provides a theoretical basis for the observation that unknown references miss at a rate much higher than the steady-state miss ratio m. However, the model assumes trace samples long enough to reference every block frame. But meeting this restriction for multi-megabyte caches requires samples with hundreds of millions of references, which may be impossible or impractical. Even more important, when samples do meet the restriction, the ratio of unknown references to misses, U/M, is small. This leads to a tight bound for m between $\hat{m}_{cold}$ and $\hat{m}_{hot}$. In other words, when U/M is small, the error in $\hat{\mu}$ has a negligible effect on the error in $\hat{m}$. Thus, while the long-trace model describes the behavior of unknown references, and is useful from a theoretical perspective, it requires generalization to be useful in practice.

5 A Model for Short Trace Samples

In this section, we extend the model to include the behavior of short trace samples, which do not reference every block frame in the cache. However, this new short-trace model depends upon the distribution of dead times, D_j, not just the ratio of means. Since we have not been able to characterize this distribution, we examine several approximations, including two that yield acceptable results.

The long-trace model predicts the fraction of dead block frames at a random time t, which is exactly the number of block frames that will miss on their first reference after time t. However, a simulation using a short trace sample may only reference a small fraction of the block frames. Temporal locality suggests that the first unknown reference is more likely to hit than the last unknown reference. Thus, block frames referenced during a short sample are more likely to be live than an "average" block frame. Therefore, the fraction of dead block frames referenced during a short sample beginning at time t is less than the fraction dead at time t. To correct for this effect, we need to predict the probability that a block frame is dead at time t *given* that it is referenced in the trace.

We do this by applying Bayes' Rule[Wolf89] to our alternating renewal process. Consider a trace sample containing N references beginning at some random time t. Let X_i be the event that an arbitrary block frame i is dead at time t. Let Y_i be the event that block frame i is referenced in the interval $[t, t+N)$, i.e., is referenced within the sample. If we assume that the X_i and Y_i are mutually independent and $P\{X_i\} = P\{X_j\}$ and $P\{Y_i\} = P\{Y_j\}$ for all i and j, then the probability that an unknown reference will miss is simply the conditional probability of X_i given Y_i:

$$\hat{\mu} = P\{X_i|Y_i\} \tag{7}$$

Using Bayes' Rule, we can express this as:

$$\hat{\mu} = \frac{P\{Y_i|X_i\}P\{X_i\}}{P\{Y_i\}} \tag{8}$$

But $P\{X_i\}$ is simply the fraction of time a block frame is dead, or $E[D_j]/E[G_j]$. And since we know the number of unknown references, U, for a particular sample, we can estimate $P\{Y_i\}$ as U/SA, where SA is the number of block frames. Finally, $P\{Y_i|X_i\}$ is the probability that when time t lands in a dead time D_j for block frame i, the time remaining in D_j, known as the *excess* $Y_d(t)$, is less than the sample size N, or $P\{Y_d(t) \leq N\}$. The renewal-reward theorem[Wolf89] states that:

$$P\{Y_d(t) \leq N\} = \frac{1}{E[D_j]} \sum_{x=1}^{N} P\{D_j \geq x\} \tag{9}$$

where the range on the sum begins at 1 because by definition $P\{D_j = 0\} = 0$.

Combining these equations yields our short sample estimate for μ:

$$\hat{\mu}_{short} = \frac{(P\{Y_d(t) \leq N\}) \left(\frac{E[D_j]}{E[G_j]}\right)}{\left(\frac{U}{SA}\right)} \tag{10}$$

$$= \frac{\frac{1}{E[D_j]} \sum_{x=1}^{N} P\{D_j \geq x\} \frac{E[D_j]}{E[G_j]}}{\frac{U}{SA}} \tag{11}$$

$$= \frac{SA}{U} \frac{1}{E[G_j]} \sum_{x=1}^{N} P\{D_j \geq x\} \tag{12}$$

To see why this estimate may be reasonable, we consider the two extremes. As $N \to \infty$, the sum converges to $E[D_j]$ and U converges to SA. Thus $\hat{\mu}_{short}$ converges to our long trace estimate $\frac{E[D_j]}{E[G_j]}$. At the other extreme, the sample contains a single reference ($N = 1$). In this

case, the sum is exactly 1, since $P\{D_j \geq 1\} = 1$, and U is exactly 1, yielding an estimate of $\frac{SA}{E[G_j]}$, which is just the miss ratio m. As we argued in Section 2, we expect an arbitrary reference to miss with probability m. Since the estimate is accurate at the two extremes, we would like to evaluate it for more typical values. However, the estimate relies upon the distribution of dead times, D_j, which we have not been able to accurately characterize. Further work is needed to determine whether this approach will yield useful estimators.

5.1 Estimates for μ

Rather than try to estimate the distributions of D_j, or equivalently $Y_d(t)$, we propose four simple estimates of μ for short traces. The first estimate is simply our long trace model

$$\hat{\mu}_{long} = \frac{E[D_j]}{E[G_j]} \tag{13}$$

which will give accurate estimates for sufficiently long traces.

The second estimator assumes that the long trace model accurately predicts the number of dead block frames at time t, but that all live block frames are referenced before the first dead one. Intuitively, the estimate assumes that all block frames not referenced by the sample are dead. This yields an estimate of:

$$\begin{aligned} \hat{\mu}_{last} &= \frac{\max(0, SA\frac{E[D_j]}{E[G_j]} - SA + U)}{U} \\ &= \max\left(0, \frac{\frac{U}{SA} + \frac{E[D_j]}{E[G_j]} - 1}{\frac{U}{SA}}\right) \end{aligned} \tag{14}$$

Taking the maximum with 0 accounts for the possibility that not all live block frames are referenced during a short sample.

The third estimate is the arithmetic mean of $\hat{\mu}_{long}$ and $\hat{\mu}_{last}$. The rationale for this estimate, called $\hat{\mu}_{split}$, comes from the observation that the first two estimates are generally larger than μ and smaller than μ, respectively. $\hat{\mu}_{long}$ is larger than μ because live block frames are referenced quickly, hence μ tends to be less than $\frac{E[D_j]}{E[G_j]}$. $\hat{\mu}_{last}$ is smaller than μ because it assumes that all unreferenced block frames are dead, which is not necessarily true. While $\hat{\mu}_{long}$ and $\hat{\mu}_{last}$ are not bounds in a strict sense, because the long-trace model is not precisely accurate, we shall see that the true value of μ tends to fall between them in practice.

Our final estimate, called $\hat{\mu}_{tepid}$, is the arithmetic mean of $\hat{\mu}_{hot}$ and $\hat{\mu}_{cold}$ (i.e., 0.5). This estimate has the advantage of requiring no computation, and, as we shall see in the next section, is also quite accurate.

5.2 Empirical Validation

Figure 2 plots the four short-sample estimates $\hat{\mu}_{long}$, $\hat{\mu}_{last}$, $\hat{\mu}_{split}$, and $\hat{\mu}_{tepid}$, as a function of U/SA. Each graph plots the true value of μ with the estimates for a particular trace simulating a 64K-byte direct-mapped cache, with 16-byte blocks. Each measured data point represents the average values of μ and U/SA for a set of samples. Each set contains 19-35 samples ranging in size from 10 thousand to 5 million references. The samples are evenly spaced throughout the base trace, and are non-contiguous (except for the largest sample size). Each sample size, except the largest, has multiple data points. As in the previous section, we estimate $\frac{E[D_j]}{E[G_j]}$ with the mean dead and generation times computed over the entire trace. Note that since $\frac{E[D_j]}{E[G_j]}$ is a function of both cache configuration and address trace, the estimates are not the same in each graph.

Figure 2 shows the results for the five traces; *mult1* and *mult2* are combined to save space and because they behave similarly. The measured data generally follow the hyperbolic curve predicted by $\hat{\mu}_{last}$ and $\hat{\mu}_{split}$. As U/SA approaches 1, μ approaches the long trace model, as predicted. For smaller values of U/SA, μ decreases rapidly. The traces *mult1* and *mult2* closely fit the prediction of $\hat{\mu}_{split}$. The traces *sor* and *tree* more closely follow the prediction of $\hat{\mu}_{last}$. And the trace *tv* most closely follows $\hat{\mu}_{tepid}$. Note that, as predicted, most of the data points fall between $\hat{\mu}_{long}$ and $\hat{\mu}_{last}$, which are much tighter than the bounds provided by $\hat{\mu}_{hot}$ and $\hat{\mu}_{cold}$.

The estimator $\hat{\mu}_{split}$ appears to be the best of the alternatives, although it clearly overpredicts most traces for small values of U/SA. While better estimators may exist, we believe that $\hat{\mu}_{split}$ is sufficiently accurate as long as U/SA is at least 0.6. We also consider $\hat{\mu}_{tepid}$ because it requires no computation. We focus on these estimates for the rest of this paper.

The results in Figure 2 demonstrate the accuracy of the model. However, the model depends upon $\frac{E[D_j]}{E[G_j]}$, which we estimated by calculating the mean dead and generation times over the entire trace. But this is impractical when using trace sampling, so we must estimate $\frac{E[D_j]}{E[G_j]}$ using only the short samples. This introduces an additional source of error. Accurately estimating this ratio can be difficult since the mean generation time may be very large. For example, the average generation time is 165,000 references for *mult1* on a 64 K-byte direct-mapped cache. Fortunately, we have observed that for most traces, especially the multiprogramming traces, the ratio of mean dead time to mean generation time remains roughly constant regardless of

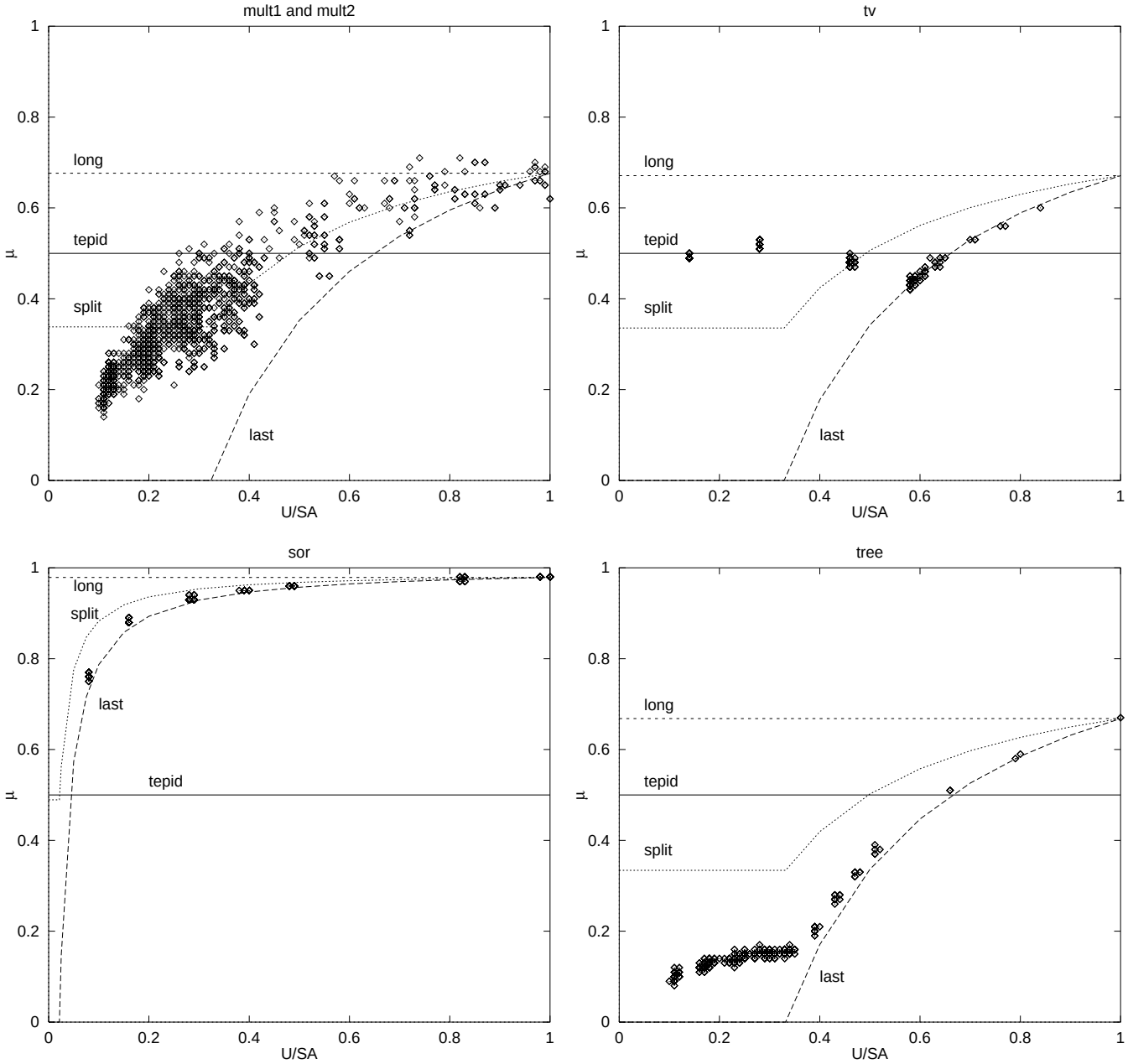

Figure 2: Estimates of μ as a Function of U/SA

This figure plots the estimates of μ as a function of U/SA, the fraction of block frames referenced during a sample. Each graph plots the four estimates, $\hat{\mu}_{long}$, $\hat{\mu}_{last}$, $\hat{\mu}_{split}$, and $\hat{\mu}_{tepid}$, as well as the measured data points. Each measured data point represents the average μ and U/SA of between 19 and 35 samples of a particular size. The samples sizes range from 10 thousand to 5 million. The cache size for all graphs is 64K-bytes, direct-mapped, with 16-byte blocks.

Configuration		Address Trace									
Cache Size	Assoc	mult1		mult2		tv		sor		tree	
50,000 Reference Samples											
4K	1	-17%	9%	-4%	19%	-36%	8%	-35%	-8%	-11%	14%
4K	2	-17%	11%	-2%	38%	-38%	13%	-33%	8%	-6%	14%
4K	4	-18%	13%	-4%	36%	-37%	17%	-30%	23%	-8%	20%
16K	1	0%	6%	12%	-2%	-37%	12%	-45%	-17%†	32%	28%
16K	2	-2%	22%	8%	24%†	-36%	18%	-45%	3%†	20%	6%
16K	4	-4%	37%†	6%	50%†	-35%	22%	-46%	4%†	9%	26%†
64K	1	42%	-15%†	47%	-19%†	4%	63%†	-46%	-33%†	251%	11%†
64K	2	45%	-3%†	45%	0%†	9%	106%†	-47%	-4%†	343%	132%†
64K	4	46%	133%†	47%	79%†	-11%	71%†	-47%	-2%†	537%	1138%†
500,000 Reference Samples											
64K	1	-18%	-7%	-20%	-31%	9%	95%	-49%	-2%	83%	45%†
64K	2	-21%	4%†	-25%	-6%†	16%	118%	-49%	1%	92%	62%†
64K	4	-24%	17%†	-27%	14%†	-7%	79%	-49%	2%	102%	149%†

Table 4: Relative Error of $\hat{\mu}_{tepid}$ (left) and $\hat{\mu}_{split}$ (right) Using Short Samples

This table plots the relative error in $\hat{\mu}_{tepid}$ and $\hat{\mu}_{split}$ where we approximate $E[D_j]/E[G_j]$ using only generations entirely contained within the short samples. Since the mean generation time may substantially exceed the sample length, this approximation introduces an additional source of error into $\hat{\mu}_{split}$. Data values in the table that do not meet the rule-of-thumb ($U/SA \geq 0.6$ and $M/SA \geq 1.0$) are marked by daggers (†).

sample size.

Table 4 compares $\hat{\mu}_{tepid}$ to $\hat{\mu}_{split}$ when we estimate $\frac{E[D_j]}{E[G_j]}$ using the mean dead and generation times from the short samples. The error in both estimates is much greater than observed for the long trace model, but is still acceptably low for the smaller caches. The error in $\hat{\mu}_{split}$ exceeds 20% in only 10 of the 30 cases. This translates to a much smaller error in the overall miss ratio, $\hat{m}_{split}$ (not shown): 28 of the 30 trace/cache combinations have relative error less than 10% and none are worse than 20%. The 64K-byte caches do not fare as well, with the error in $\hat{\mu}_{split}$ reaching 1000% for one data point. The problem, of course, is the small sample size. In the extreme example, the cache averages only 12 (known) misses per sample. Clearly a longer sample is necessary to predict $\frac{E[D_j]}{E[G_j]}$ with any accuracy.

Increasing the sample size to 500,000 references reduces the error substantially. Exactly how long the samples must be to achieve acceptable error is still an open question. The following experimental rule-of-thumb, however, holds for our data: the error in $\hat{m}_{split}$ is generally less than 10% if the samples are large enough to touch at least 60% of the cache ($U/SA \geq 0.6$) and average at least one miss per block frame ($M \geq SA$). Until we better understand the estimator's error, we suggest the conservative strategy of ensuring that $M \geq 2SA$.

Surprisingly, $\hat{\mu}_{tepid}$ also does quite well for small samples. This simple estimate has lower relative error than $\hat{\mu}_{split}$ in one-third of the cases examined. Since $\hat{\mu}_{tepid}$ requires no computation, some designers may choose to use it, rather than the more accurate $\hat{\mu}_{split}$.

Both estimates are much more accurate than the steady-state estimate $\hat{m}_{ss}$. As shown in Table 5, a continuation of Table 1, the error in $\hat{m}_{split}$ is much less than the error in $\hat{m}_{ss}$. The error in $\hat{m}_{tepid}$ is much less in four of the five cases, but is worse in the fifth. These results clearly show that using $\hat{m}_{split}$, or simply $\hat{m}_{tepid}$, is much better than using $\hat{m}_{ss}$.

6 Summary and Conclusions

In this paper, we showed that accurately estimating the miss ratio of unknown references is the key to obtaining accurate results from trace-sampling. Previous estimates generally assumed that unknown references behave like randomly chosen references, and miss at the steady-state miss ratio. We presented empirical results showing that unknown references miss at a much higher

Trace	True m	Relative Error		
		$\hat{m}_{ss}$	$\hat{m}_{tepid}$	$\hat{m}_{split}$
mult1	0.024	-31.9%	13.9%	-4.9%
mult2	0.016	-38.1%	18.6%	-7.4%
tv	0.035	-49.5%	2.0%	32.3%
sor	0.027	-81.2%	-37.8%	-27.0%
tree	0.010	-26.4%	69.9%	3.1%

Table 5: Comparison of $\hat{m}_{ss}$, $\hat{m}_{tepid}$, and $\hat{m}_{split}$.

This table shows the relative errors in the miss ratio estimates. Each data point is the mean of several thousand samples, each containing 50,000 references. The cache configuration is 64K-bytes, direct-mapped, with 16-byte blocks.

miss ratio that is independent of the steady-state miss ratio.

In the key result of this paper, we showed that for samples that reference every block frame, the miss ratio of unknown references is simply that fraction of time a block frame is dead.

$$\hat{\mu}_{long} = \frac{E[D_j]}{E[G_j]} \tag{15}$$

Trace-driven simulation data demonstrate the accuracy of this model.

We extended the model to handle short samples, which only reference a fraction of the block frames:

$$\hat{\mu}_{short} = \frac{SA}{U}\frac{1}{E[G_j]}\sum_{x=1}^{N} P\{D_j \geq x\} \tag{16}$$

However this estimate relies upon the distribution of D_j, which we have not yet characterized.

We examined several estimates for μ, and show that

$$\hat{\mu}_{split} = \max\left(\frac{E[D_j]}{2E[G_j]},\ \frac{E[D_j]}{2E[G_j]} + \frac{\frac{U}{SA} + \frac{E[D_j]}{E[G_j]} - 1}{\frac{2U}{SA}}\right) \tag{17}$$

produces good results for sufficiently long samples. We also showed that $\hat{\mu}_{tepid} = 0.5$ produces acceptable accuracy for many practical applications, yet requires no computation.

Several open problems remain. Most importantly, we must determine a robust criterion for estimating the error in $\hat{\mu}_{split}$. Preferably, this should be a function of the sample size or the fraction of referenced block frames. We plan to examine this issue fully in the near future, using larger caches and longer traces. We also plan to examine hierarchical caches and compare our results more fully with other techniques.

Better estimators of μ are also possible, perhaps by successfully characterizing the distributions of D_j or $Y_d(t)$. We would also like to explore extending this model to multiprocessor caches and stripped (filtered) traces.

7 Acknowledgements

We would like to thank S. Adve, G. Gibson, R. Nelson, H. Stone, M. Vernon, R. Wolff, the students in M. Vernon's Fall 1990 CS 838 class, and the anonymous referees for their comments and criticisms that helped improve this paper. And thanks to Anita Borg, David W. Wall, and DEC Western Research Lab for providing the use of their excellent, long, traces.

References

[Bick77] Peter J. Bickel and Kjell A. Doksum. *Mathematical Statistics*. Holden-Day, Inc., San Francisco, 1977.

[Borg90] Anita Borg, R. E. Kessler, and David W. Wall. Generation and analysis of very long address traces. In *Proceedings of the 17th Annual International Symposium on Computer Architecture*, pages 270–279, 1990.

[East78] Malcom C. Easton and Ronald Fagin. Cold-start vs. warm-start miss ratios. *Communications of the ACM*, 21(10), October 1978.

[Klei75] Leonard Kleinrock. *Queueing Systems Volume 1: Theory*. John Wiley and Sons, New York, 1975.

[Laha88] S. Laha, J. H. Patel, and R. K. Iyer. Accurate low-cost methods for performance evaluation of cache memory systems. *IEEE Transactions on Computers*, 37(11):1325–1336, November 1988.

[Mend90] Abraham Mendelson, Dominique Thiébaut, and Dhiraj Pradhan. Modeling live and dead lines in cache memory systems. Technical Report TR-90-CSE-14, Dept. of Electrical and Computer Engineering, University of Massachusetts, 1990.

[Puza85] T. R. Puzak. *Cache Memory Design*. PhD thesis, University of Massachusetts, Amherst, MA, 1985.

[Ross83] Sheldon M. Ross. *Stochastic Processes*. John Wiley and Sons, New York, 1983.

[Smit82] Alan Jay Smith. Cache memories. *ACM Computing Surveys*, 14(3):473–530, September 1982.

[Ston90] Harold S. Stone. *High-Performance Computer Architecture*. Addison Wesley, second edition, 1990.

[Wolf89] Ronald W. Wolff. *Stochastic Modeling and the Theory of Queues*. Prentice-Hall, 1989.