Microarchitecture-Independent Cache Modeling for Statistical Simulation

Davy Genbrugge, Lieven Eeckhout, Koen De Bosschere

* *ELIS, Ghent University, Sint-Pietersnieuwstraat 41, 9000 Gent, Belgium*

ABSTRACT

Computer architects heavily rely on simulation tools during the design process of their next generation processor. Statistical simulation allows for making quick performance estimates early in the design cycle. However, the current state-of-the-art in statistical simulation still uses microarchitecture-dependent cache models which is impractical when exploring cache hierarchy design spaces.

This paper introduces a microarchitecture-independent cache model based on the distributions of the LRU-stack distances and the memory address distances. Preliminary results show that our model is capable of making accurate miss rate predictions.

KEYWORDS: Statistical Simulation, Cache Modeling

1 Introduction

Previous work has shown that statistical simulation is capable of making fast and accurate performance estimates [Genb06]. The idea behind statistical simulation is to collect a number of microarchitecture-dependent and microarchitecture-independent program characteristics, called a statistical profile. This profile serves as input for the synthetic trace generator. The synthetic trace is then simulated on a trace-driven statistical simulator yielding performance estimates such as IPC. The main advantage is that the synthetic trace is very small. Genbrugge et al. [Genb06] show that a synthetic trace of 1M instructions suffices to accurately estimate performance of superscalar processors – IPC errors vary between -2% and 5.5%. This leads to a huge simulation speedup compared to detailed processor simulation.

A limitation of current statistical simulation frameworks though is that the statistical profile still depends on some microarchitectural parameters such as the branch predictor and the caches. We must collect a new profile each time these microarchitectural parameters are changed during design space exploration. Ideally the statistical profile should be independent of the microarchitecture. In this paper we introduce a microarchitecture-independent cache model, which allows us to profile the caches only once. Our model focuses on set-associative caches with an LRU replacement policy and bit selection as the set mapping

[1] E-mail: {dgenbrug,leeckhou,kdb}@elis.UGent.be

scheme. Our model is based on the LRU-stack distances [Smit78, Hill89] and the distances between the referenced address and the addresses above on the stack.

This paper is organized as follows. First we will take a closer look at previous work on cache modeling. Then we will describe our model, followed by the experimental setup and the evaluation. Finally we conclude.

2 Related Work

Cache modeling has been addressed by many researchers over the past decades.

Smith [Smit78] has studied the relation between the LRU-stack distances, set-associativity and fully-associativity. He concluded that the miss rate of a set-associative cache can be estimated from the LRU-stack distance distribution of a fully-associative cache. The miss rate equals then $1 - \sum_{i=1}^{associativity} P_i$, with P_i the probability of a hit at depth i of one of the LRU-stacks of a set-associative cache with S sets. P_i can be derived from Q_j, the probability of a hit at depth j of the fully-associative LRU-stack, as follows.

$$P_i = \sum_{j=i}^{\infty} \binom{j-1}{i-1} \cdot \left(\frac{1}{S}\right)^{i-1} \cdot \left(\frac{S-1}{S}\right)^{j-i} \cdot Q_j$$

This formula was further evaluated by Hill and Smith [Hill89]. Note that they model spatial locality only by collecting the LRU-stack distances on a per cache line size basis. Furthermore, they assume that each set is equally likely to be touched on a memory access.

Berg et al. [Berg04] introduced StatCache, a tool which allows for estimating the miss ratio of fully associative caches with random replacement policy. They model temporal locality by collecting the reuse distance on a per cache line size basis. Note there is a subtle difference between the LRU-stack distance and the reuse distance. The LRU-stack distance is the number of unique memory references, whereas the reuse distance is the number of all memory references between two references to the same cache line.

None of the above approaches explicitly model spatial locality. In section 3 we show how we can model both temporal and spatial locality.

3 Microarchitecture-Independent Cache Model

The behavior of caches is completely determined by the spatial and temporal locality patterns inherent to a sequence of memory references. The LRU-stack distance is a metric that relates to temporal locality. The distance between two addresses is a metric that relates to spatial locality. Since we focus on caches with bit selection as set mapping algorithm, we do not express this distance as the arithmetic difference but as the XOR between two addresses.

For each instruction we collect the LRU-stack distance histogram for a fully-associative cache. For each LRU-stack distance j ($j > 1$) we also collect an array of length $j - 1$ with the XOR-distances between the addresses at depth $1 \ldots j - 1$ and the address at depth j of the LRU-stack.

Again the miss rate per instruction is computed as $1 - \sum_{i=1}^{associativity} P_i$. The P_i is now derived from the LRU-stack distance distribution and the distribution of the arrays with the address distances. We no longer assume that each set is equally likely to be touched on a memory access but we use the distribution of the arrays with the address distances for

cache size	32 KB								64 KB							
cache line size	32 B				64 B				32 B				64 B			
sets	1024	512	256	128	512	256	128	64	2048	1024	512	256	1024	512	256	128
ways	1	2	4	8	1	2	4	8	1	2	4	8	1	2	4	8

Table 1: Cache configurations used in this paper.

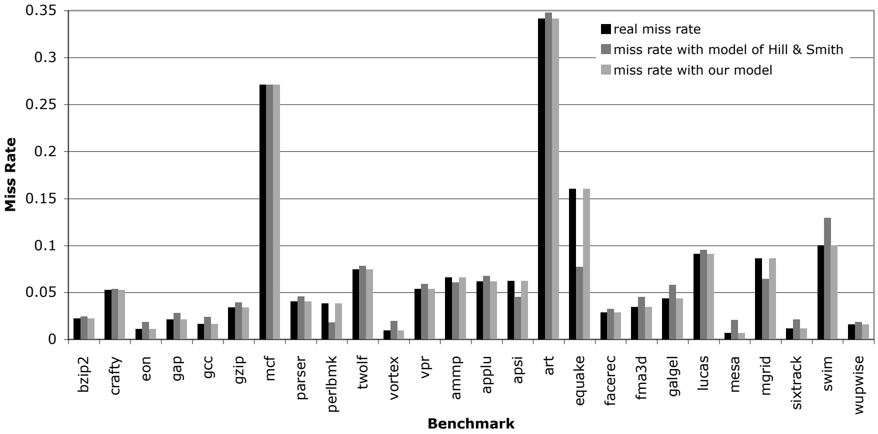

Figure 1: Evaluation of the microarchitecture-independent cache models for a direct mapped cache of 32KB with a 64B cache line size

computing the number of conflicting accesses. Note that two memory references touch the same set if their distance modulo the number of sets equals 0. We can express P_i as follows.

$$\begin{aligned} P_i &= \sum_{j=i}^{\infty} \text{Pr(LRU stack distance } i \text{ with S sets} \mid \text{LRU stack distance } j \text{ with 1 set)} \\ &= \sum_{j=i}^{\infty} Q_j \cdot R_{ij} \end{aligned}$$

R_{ij} is the probability that $i-1$ intermediate references touch the same set as the reference at depth j.

4 Experimental Setup

In our experiments we assume an L1 data cache with bit selection as set mapping algorithm and no prefetching. Table 1 gives the cache configurations. The benchmarks along with their reference inputs used in this study are the SPEC CPU 2000 benchmarks, taken from the SimpleScalar website. We note that in this paper the profiles are still collected dependent on the cache line size. Also, when an LRU-stack distance is larger than 4096 we will assume this is a miss.

5 Preliminary Results

We now evaluate our model and compare its accuracy to the model of Hill and Smith [Hill89]. In both models we estimate the miss rates on a per instruction basis which we then use to compute the overall miss rate.

Figure 1 shows the miss rates obtained with detailed cache simulation, the model of Hill and Smith and our model for a direct mapped 32KB data cache; we obtained similar results for the other cache configurations. We can see that the model of Hill and Smith in some cases can lead to huge absolute errors – up to 0.083 for equake. In most cases the absolute errors are rather small, although they can lead to relative errors up to 400% (*mesa*). We believe this is due to the assumption that each set is equally likely to be touched on a memory access. The results we achieve with our model confirm this believe. With the spatial en temporal locality modeling the results are very accurate. The absolute miss rate errors are not larger than 0.0018 and the relative miss rate errors are smaller than 1% in most cases. The largest relative error we have measured is smaller than 2.5%.

One drawback for our model is the size of the statistical profile, which for some benchmarks such as *ammp* is very large – up to 6GB for a 100M instruction trace. Typically the sizes of the statistical profiles are in the range of 100MB to 2GB.

6 Summary & Future Work

In this paper we introduced a cache model that models both spatial and temporal locality and we compared it against the model by Hill and Smith. Our experiments show that by explicitly modeling spatial locality we can obtain substantial reductions in miss rate prediction errors. With our model we obtain relative miss rate errors below 2.5%.

In future work we will reduce the size of the statistical profiles. Pruning the information tracked during the statistical profiling should be no problem since we focus on statistical simulation that uses very small synthetic traces. We will also make our model independent on the cache line size and provide means to estimate the L2 cache miss rate.

References

[Berg04] E. Berg and E. Hagersten. StatCache: A Probabilistic Approach to Efficient and Accurate Data Locality Analysis. In *Proceedings of the 2004 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS-2004)*, Austin, Texas, USA, mar 2004.

[Genb06] D. Genbrugge, L. Eeckhout, and K. De Bosschere. Accurate Memory Data Flow Modeling in Statistical Simulation. In *ICS '06: Proceedings of the 20th ACM International Conference on Supercomputing*, June 2006.

[Hill89] M. Hill and A. Smith. Evaluating Associativity in CPU Caches. *IEEE Trans. Comput.*, 38(12):1612–1630, 1989.

[Smit78] A. Smith. A Comparative Study of Set Associative Memory Mapping Algorithms and Their Use for Cache and Main Memory. *IEEE Trans. Software Eng.*, 4(2):121–130, 1978.