

Scalable Cross-Architecture Predictions of Memory Latency for Scientific Applications

Gabriel Marin
mgabi@cs.rice.edu

John Mellor-Crummey
johnmc@cs.rice.edu

Introduction

The gap between processor and memory speeds grows with each new generation of microprocessors, making memory hierarchy response a critical factor limiting performance.

Traditional methods for understanding how an application uses the memory hierarchy:

- cache simulation and profile based methods - need full simulation for each combination of input parameters and cache configuration that must be analyzed
- compile time analysis techniques - applicable only to certain array references, cannot characterize reuse that spans different loop nests
- dynamic code monitoring and optimization - limited applicability

This poster presents a method for characterizing the data access pattern of an application in a machine independent fashion. We collect memory reuse distance histograms for each memory reference in a program during repeated executions with small data inputs. Then, we model the structure and scaling of each references reuse distance histogram as a function of problem size. This approach enables us to predict the number of cache misses at the instruction level for architectures and problem sizes that we did not measure.

Static Analysis & Binary Instrumentation

Use static analysis and binary rewriting tools to instrument application executables for collecting memory reuse distance (MRD) for each reference.

- build control flow graph
- compute loop nesting structure
- compute symbolic formulas for each reference's access pattern

Example for static analysis of memory access pattern. Compute symbolic formulas for:

```

    { int i, j, k;
      for ( i=0 ; i<N ; i+=1 )
        for ( j=0 ; j<N ; j+=1 ) {
          c[i*N+j] = 0;
          for ( k=0 ; k<N ; k+=2 )
            c[i*N+j] += A[i*N+k]*B[k*N+j];
        }
    }
  
```

• **base formula (BF)** = first accessed location

S₁, ..) for each loop level from the innermost to the outermost loop (S₁ corresponds to the innermost loop).

SPARC assembly for innermost loop

Next we model the structure and scaling of memory reuse distance histograms as a function of problem size.

- aggregate histograms corresponding to sets with similar access pattern from adjacent leaf loops which are the result of compiler optimizations (e.g. loop unrolling, etc.)

Translate each histogram into 3D Cartesian coordinates:

- x axis: problem size
- y axis: bins normalized execution frequency
- z axis: reuse distance

Model bins with constant distance first.

Recursively split rest of data:

• normalize frequency ratio the same across all problem sizes at each split

• select ratio s.t. the two subsets cover equal sized ranges of reuse distance

Reuse distance data translates directly into cache miss predictions for fully-associative caches:

- cache hit, if reuse distance is less than the cache size
- cache miss otherwise

Data Modeling

Assuming an uniform distribution of the accessed memory blocks, we can combine the MRD predictions with a probabilistic model [1,2] to approximate the number of cache misses for a set-associative cache with s sets and associativity k. Probability that a memory access with reuse distance d misses in a set-associative cache is:

$$P_{\text{miss}}(d, s, k) = 1 - \sum_{i=0}^{k-1} \left(\frac{1}{s}\right)^i \left(\frac{s-1}{s}\right)^{d-i} C(d, i)$$

where d' = memory reuse distance
 s = number of sets in cache
 k = associativity level

and the number of misses for a single histogram (a set of aggregated references) is:

$$Num_{\text{misses}}(Hist, s, k) = \sum_{bin_i \in Hist} (P_{\text{miss}}(D_{bin_i}, s, k) F_{bin_i})$$

D_{bin_i} = average MRD for bin_i
 F_{bin_i} = execution frequency for bin_i

Validation

Construct MRD models for ASCI Sweep3D and several NAS benchmarks. Compare predictions versus measurements with hardware performance counters.

Itanium2:

- 256 KB 8-way set-associative L2 cache
- 1.5 MB 6-way set-associative L3 cache
- 128 entries fully-associative L2 TLB
- 16 KB memory page

Legend

- L2 measured MRD
- L2 predicted probabilistic
- L3 measured MRD
- L3 predicted MRD
- TLB measured MRD (x5)
- TLB predicted MRD (x5)

Model Evaluation

Evaluate the MRD models for the desired problem size.

- 8 MB, 2-way set-associative L2 cache
- 64 double entries fully-associative TLB
- 16 KB memory page

For a fully-associative cache, the number of cache misses with reuse distance larger than the size of the cache.

size of the cache.