CS422 Project Report

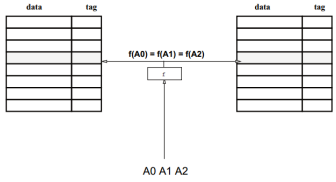
Vineet Purswani

April 2015

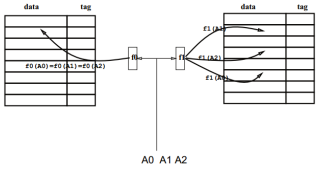
1 Introduction

This project tries to compare between two very popular cache replacement policies, k-way Set Associative replacement and k-way Skew Associative replacement. It is highly inspired from the work done by Professor Andre Seznec, and I have used the simulator provided on his website along with the INTEL PIN tool to generate data for SPEC benchmarks applications.

k-way Set Associative Caches are Caches divided into k banks or sets. All the k banks have a same indexing function. An address can be mapped to any of the k-banks, depending on the replacement policy used. Random mapping and LRU policies are used in this project.



k-way Skew Associative Caches are similar to Set associative caches, they have k banks or sets and are indexed using various functions. The only difference is, Set associative caches use only one indexing function at a time, while Skew associative caches use different indexing functions for different sets. This helps to reduce data conflicts, and results in decreasing a substantial number of data misses. The same can be seen from the observations given below.



2 Replacement Policies

2.1 Set Associative Caches

Direct Mapping - Although, this replacement policy is not set associative, but we can name it as 1-way set associative cache. This is the most primitive type of replacement algorithms, the simplest ones. It maps the address to some cache line using a simple mapping function, usually Modulo. For the multi-banked caches, we can use direct mapping inside the banks, after using the indexing function to identify which bank to use.

1. Random Replacement - Random indexing, as the name suggests, selects any one bank, randomly and uses it for the further operations.

2. Least Recently Used Replacement - LRU indexing, selects the least recently used bank, and uses it in further replacement operations.

2.2 Skew Associative Caches

Skewing function - Skewing function used in this project is based on the research work done by Professor Andre, in his paper *Skewed associativity improves program performance and enhances predictability.*

Inter-bank Dispersion - Main motive of the skewing technique is to reduce data conflicts, or cache line conflicts, i.e. if two data lines map to the same cache line in one bank (through cache indexing function), they should not conflict in any of the other banks. Thus, we need different mapping functions for each bank, that ensure that such conflicts are minimum. The above paper explains one such technique. I will brief out the concepts of the Skewing functions used in the project.

An address can be written as, $A2 * 2^{n+c} + A3 * 2^c + A1$. So a mapping function of the type,

$$fn = \sigma^n\{A1\} \oplus A2$$

would have a better inter-bank dispersion. σ should be not be a identity function, else it will fail to serve our purpose. It should be a perfect bit shuffling function, so that we don't get the same cache line value for different banks. fn represents the Skewing function for n^{th} bank. n is the number of cache lines in a bank and 2^c is the cache line size.

3 Observation

Due to shortage of time I was not able to run the PIN tool on SPEC benchmarks, rather I have provided the data for *ls* & *uname -a* shell commands.

Replacement Policy	# of Banks	Miss Rate
DirectMap	1	6.443022
AssocLRU	4	4.267563
AssocRAND	4	4.776958
SkewPseudoLRU	2	4.457341

Table 1: Miss Rate for various Replacement Policies for *ls* command with 572247 data access.

Replacement Policy	# of Banks	Miss Rate
DirectMap	1	8.117865
AssocLRU	4	7.081889
AssocRAND	4	7.587395
SkewPseudoLRU	2	7.169741

Table 2: Miss Rate for various Replacement Policies for *uname -a* command with 208306 data access.

The above data clearly shows that 2-way Skew Associative Caches give the similar output as 4-way Set Associative Caches provide. The extra hardware required in Skew Associative caches for computing Skewing functions could be a problem, but many good Skewing functions with optimizations perform well with almost equal hardware requirements as Set Associative Caches. In all, Skew associative caches are an upgrade to the cache community over Set associative caches.