Trace-driven Cache Simulator

Ju Su, Tianrui Yang, Yuan Yao

Cache Simulator

In this project, we designed a fully configurable L1 data cache simulator. This cache simulator is implemented by python. We use the trace file extracted from <https://www.cis.upenn.edu/~milom/cis501-Fall12/traces/trace-format.html>. This trace file contains several fields but only memory address is useful in our cache simulator. Our simulator is configurable in a number of parameters and is able to measure basic cache statistics. Due to the large size of trace file, we also implemented a progress bar which indicates the simulator progress.

Configuration

The following parameters is configurable using the configuration file:

* Cache block size
* Cache size
* Associativity
* Replacement policies
* Hit delay (Default is 1 cycle.)
* Workload

Each memory address(64 bit) can be divided into 3 parts: tag, index and offset. The number of bits of offset is determined by cache block size and the number of bits of index is determined by the formulation: cache size/cache block size/associativity. While mapping from memory address to cache, it is required to configure the associativity. Associativity of 1 is direct-mapped cache, while associativity of block number is fully associative cache. Replacement policy is configurable within LRU, FIFO and Random.

Cache Statistics

The following basic cache statistics can be measured in our simulator:

* Total number of cache accesses
* Cache hits
* Cache misses
* Average Memory Access Time

These statistics are saved in the output file *result\_\*\*\*\*\*\*@\*\*-\*\*-\*\*.txt*, where asterisk represent the time when we create the output file.

The following is an example of output file which contains cache statistics and corresponding configuration information:

