Evaluate different cache replacement policies for their effectiveness.

The work requires modifying an open source cache simulator such as Dinero (http://pages.cs.wisc.edu/~markhill/DineroIV/) or Moola (https://github.com/fernandomosquera/Moola---Multicore-Cache-Simulator)

Learn how to use the cache simulator -- what input trace format is used, and how to configure the caches.

Next, identify different cache replacement policies, implement them and evaluate with a set of benchmarks. Identify which replacement policy achieves the lowest cache miss rate and its complexity in implementing it.

1. Install Dinero
2. Learn how to use dinero
   1. What input trace format
   2. How to configure caches
3. Pick which cache policies we want to use
   1. **FIFO (oldest from load)\***
   2. **LRU (last recently used)\***
   3. **Random replacement\***
   4. **MRU\***
   5. LIFO\*
4. Implement different caches in dinero
5. Create a list of benchmarks
6. Identify which cache replacement system is best and which is easiest to implement

Sorting Algorithms

* Need a trace
* Run Dinero through the trace
* Dinero will report cache hit misses

Intel Pin Tool - Linux Env

Pin -t ./obj-intel64/pintrace.so -- /bin/ls

Executables

* Lower level
* **Take array and sort**
* **Matrix Mult**
* Graph driver (BFS or DFS of tree)
* **File read and writes**
* Pull from existing or open source

Intel standard

32Kb size

Associativity 4 or 8

Block size 64 Bytes