CO262 - Course Project

Objective

Design and program (preferably in C) a simple one level cache simulator and implement the following cache replacement policies:

- LRU – Least Recently Used
- LFU – Least Frequently Used
- NRU – Not Recently Used
- PLRU – Pseudo Least Recently Used
- SRRIP – Static Re-Reference Interval Prediction

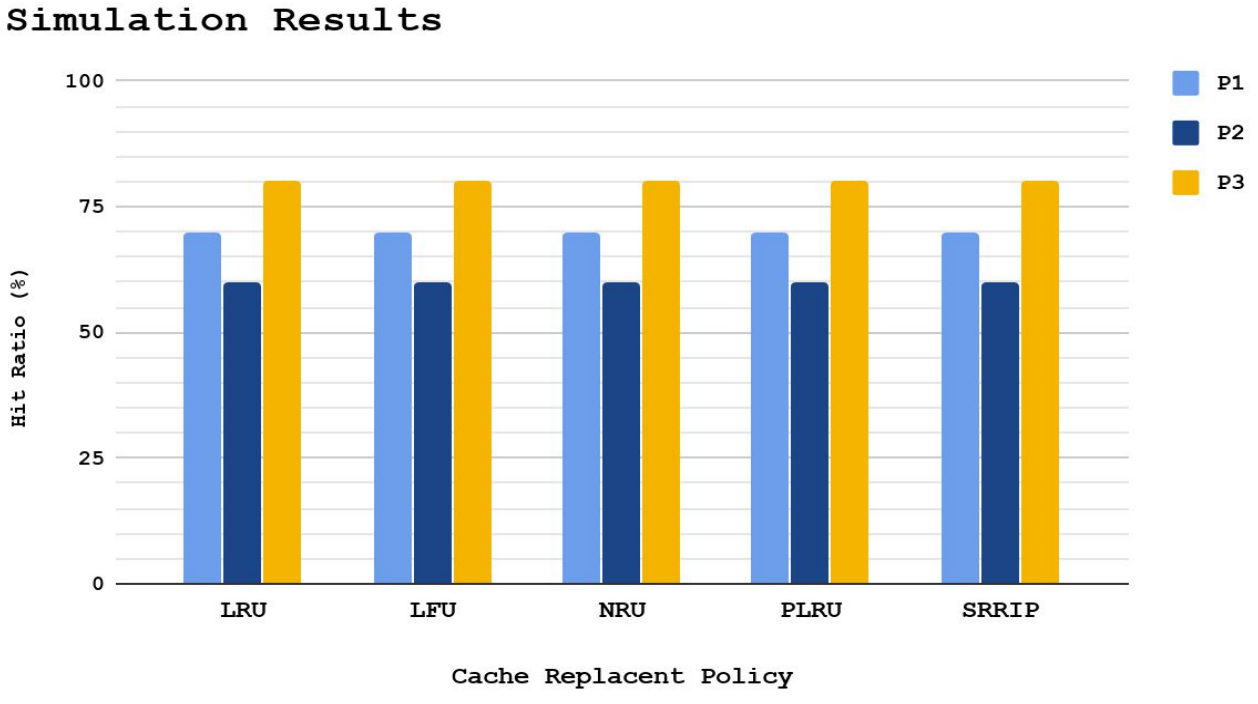
The simulator must simulate a set associative cache with configurable values for the number of sets, ways and block size.
The simulator must take in as inputs: a memory reference trace file (generated using Intel Pin 3.7, specifically “pinatrace.so” from the ManualExamples in the source) and the name of a replacement policy.
The simulation output must include the number of memory accesses (reads and writes separately), the hit ratio and the simulation time.

Assessment

You will have to generate 3 traces from the Pin tool for the following programs written (by you) in C:

- P1 – DGEMM (Double precision General Matrix Multiply)
- P2 – DGEMM with row-major accesses only
- P3 – Tiled DGEMM

Run your DGEMM programs with large input matrices, preferably larger than 1000 x 1000.
For every trace you will run 5 simulations, each with a different replacement policy. After running all the 15 simulations, plot a clustered bar graph showing the hit ratios obtained for all simulations, categorized by the replacement policies. Example:



Analysis

A group may pick one of the topics to do analysis on:

- Impact of cache size on hit ratio: Varying block size
- Impact of cache size on hit ratio: Varying number of sets
- Impact of cache size on hit ratio: Varying number of ways per set
- Impact of tile size (in P3) on hit ratio
- Comparison of the three traces in terms of scans/thrashing.
- Plot out which matrix elements receive hits for the different algorithms, e.g. address A[10][20] always gets cache hit with SRRIP, but never with LFU.