**CSE 420: Computer Architecture 1  
Summer 2017**

**Project 3**

**Due Date: Friday, July 7, 2017, 11:59 PM**

**Problem 1 [20 Points]:** Understanding LRU Policy

In the design of set associative caches, an important design parameter to analyze - is the block replacement policy. Block replacement policy essentially determines which block (of the blocks in the set) must be evicted from the cache, when it is time to bring in new data.

One of the most popular cache replacement policies is Least Recently Used (LRU) block. In this scheme, we evict the block that has been unused for the longest period. While it seems to be one of the most intuitively promising block replacement schemes, implementation complexity is its main disadvantage. LRU must maintain an access history for each block, which will slow down the cache. Thus, most caches implement only an approximation of LRU.

Study the LRU block replacement policy and explain – **i)** in which scenario, LRU will perform usually better **ii)** scenarios where LRU will be inefficient **iii)** how LRU is implemented in gem5.

Note: You will find the cache replacement policy implementation in the following file in the gem5 source directory – /src/mem/cache/tags/lru.cc

**Problem 2 [30 Points]:** Understanding SRRIP Policy

A. Jaleel et. al. proposed high-performance cache replacement policies using re-reference interval prediction (RRIP) at International Symposium on Computer Architecture (ISCA) 2010. Read [this paper](http://dl.acm.org/citation.cfm?id=1815971) to understand the RRIP schemes as our next goal is to implement 2-bit SRRIP block replacement policy in gem5.

In which scenarios RRIP – **i)** will perform like ideal replacement policy (i.e. performs very well) for given benchmark **ii)** will be inefficient?

Explain the logistics behind implementing 2-bit SRRIP block replacement policy in gem5. Please give an overview about will be the changes required in which source files for such implementation.

Note: You will make changes in the file - lru.cc, and other files (such as blk.cc or blk.hh) depending your implementation.

**Problem 3 [50 Points]**

Run Gem5 with the newly implemented cache replacement policy on the four benchmarks provided to you in the last project. Analyze the performance of the two replacement policies – LRU and SRRIP, by observing the following cache simulation statistics and the characteristics of the benchmark (e.g., memory access patterns) and cache configuration. You will analyze output statistics such as:

* Miss rate for L1 instruction cache
* Miss rate for L1 data cache
* Miss rate for L2 cache
* Execution time (Sim ticks)

Tabulate these statistics and briefly describe your analysis of the results based on given benchmarks.

Note – To simulate the execution with caches, you can use command line options similar to that you used in project 2. However, please fix L1 data cache size as 1 kB, its associativity as 4 and cache-line size to 32.

**Project Deliverables**

1. Source code of your SRRIP implementation – You need to submit only the new versions of the source files that you have modified. Please do not submit the entire source code for gem5. Please comment well on your changes.
2. For Problem 1 and 2: Briefly explain your answers for the respective questions.
3. For Problem 3: Tabulate required simulation statistics. Briefly present your analysis. You may use additional graphs to support the explanation.

**Submission Instructions**

1) Write your answers to the questions, and briefly explain required details. Plot needed graphs and/or tables. Submit single PDF file containing all the answers/plots/tables. Along with reporting the output statistics, analyze the results and try to relate it with the concepts.

2) Submit a single zip file and name it as Project3-Your10DigitAsuId.zip. This zip file should contain *i)* a PDF containing all the answers *ii)* a directory with needed source code inside it *iii)* a directory containing output statistics for all policies.

3) In case where you are doing project in a group, only one group member should make the submission through blackboard.