Final Project

1 Objective

The objective is to extend your PA3 cache bypassing assignment, model different L1 cache bypassing logic, and do a performance study. The reference paper is the ICS'15 paper, Locality-Driven Dynamic GPU Cache Bypassing by Li et al.
(https://hzhou.wordpress.ncsu.edu/files/2022/12/ICS_15.pdf)

2. Ground rules

(1) All students must work alone. Sharing codes between students will be considered as cheating and receive appropriate action based on the University Policy.
(2) A forum has been created on Moodle for posting questions and discussing and debating issues, but please do not share your code directly on Moodle.
(3) Every student should follow the submission requirement exactly. Otherwise, you will receive a point deduction.

3 Description

3.1 Task 1: cache efficiency analysis.

The ICS paper aims at cache-unfriendly benchmarks. So your first task would be identifying the categories of benchmarks. There would be three categories for the benchmarks: Cache Unfriendly (L1 cache bypassing can improve the IPC), Cache Insensitive (The IPCs have no significant difference when enabling or disabling the bypass logic), Cache Friendly (L1 cache bypassing will cause performance degradation). You can run the benchmarks by enabling and disabling the ”−gmem_skip_L1D'' option in gpgpusim.config. This option can enable or disable data cache bypass logic for global memory access.

The configuration you should use for this task 1 is **SM2_GTX480**. You can identify the categories of benchmarks by comparing the IPCs of simulation with and without cache bypass. If you suffer from long running time with the benchmarks in your machine, you can set the maximum instructions that can finish within a reasonable time in your machine for each benchmark, and report the number of instructions you used in your report such that I can verify your results. But do not use a number less than 100M, in which case you may not be able to observe the correct behavior of benchmarks.

The benchmarks you should use for this task 1 are three benchmarks from Rodinia and three benchmarks from ISPASS. So you need to run **a total number of 6 benchmarks**. For the Rodinia

benchmarks, you can use the provided commands from run.txt to run it, and you can find run.txt for each of the benchmarks from the rodinia.zip. For the ISPASS benchmarks, you can use the commands which are provided in PA2 to run the benchmarks, and you can choose three benchmarks from the ISPASS by yourself.

3.2 Task 2: profiling-based cache bypassing

The ICS paper has done an intensive performance study for different cache bypassing logic. For this task 2, you are required to re-implement the profile-based cache bypass. The profilebased cache bypass can be divided into 2 steps:

(1) In the first run of the simulation, profile the reference frequency of each data block in the data cache(l1 cache in the simulator). And you should use a per SM local profiler to record the intra-SM data reference frequency. For per SM profiling, you need to have a data structure for each SM and save the reference counter with the following format:

< SM 0, kernel 1 : addr1 ref, addr2 ref, ...addrn ref kernel 2 :
addr1 ref, addr2 ref, ...addrn ref
...
kernel n : addr1 ref, addr2 ref, ...addrn ref >
...
< SM n, kernel 1 : addr1 ref, addr2 ref, ...addrn ref kernel 2 :
addr1 ref, addr2 ref, ...addrn ref
...
kernel n : addr1 ref, addr2 ref, ...addrn ref >

After profiling, you need to dump out your reference statistics into some files for later use.

(2) In the second run of the simulation, you need to first load your per SM local reference statistics into the simulator before running the simulation, then use these profiled reference statistics to implement your cache bypass logic. And the cache bypass logic is: for the request with a reference counter less than 3, bypass the L1 data cache.

Please use **cache unfriendly benchmarks** for this task 2. You should already know which benchmark is cache unfriendly from doing task 1.

The configuration you should use for this task 2 is **SM7_QV100**. (You may notice that the configuration we are using for task 2 is different from task 1, the reason is the SM2 configuration may lead to deadlock for task 2.)

3.3 Task 3: Experiment with different cache replacement policies

For this task, read about gpu-cache.cc and go through the replacement policies it supports. For now gpgpusim supports FIFO and LRU cache replacement policies. This can configured in gpgpusim.config file. Pick any three benchmarks (out of the 6 mentioned above) and compare performance of FIFO vs LRU. Also report any secondary stats which justify the IPC drop/gain with a brief explanation in your report. The config to be used for this task is **SM2_GTX480.**

NOTE: You can check out the configs in this directory : /root/gpgpu-sim_distribution/configs/tested-cfgs

4 Report

In your report:

(1) For task 1, you need to report the result of different benchmarks, use a table to show the results of cache unfriendly, cache insensitive and cache friendly benchmarks. Your table should look like the following format:

benchmark name	kernel _name	kernel_launch _uid	IPC with no cache bypassing	IPC with cache bypassing	percentage change of comparing the IPC with/without cache bypassing	benchmark category

(2) For task 2, explain your implementation of the profile-based cache bypass. And also, in your report, please give detailed commands about how to use your profiled data and show the benchmark results when you use the profile-based cache bypass.

(3) For task 3, report the result in the table format below. Explain the IPC drop/gain observed with secondary stats.

benchmark name	kernel _name	kernel_launch _uid	IPC with LRU	IPC with FIFO	percentage change of comparing the IPC of LRU vs FIFO	benchmark category

5 Submission

(1)Create a directory named ”task1”, copy the output files you generated for task1 to this directory.

(2)Create a directory named ”task2”, copy the files you modified for task2, your profiled data, and the output files you generated for task2 to this directory.

(3)Create a directory named “task3” and copy the output files generated for task3

(4)Zip ”task1”, ”task2” , “task3” and your report as unity id.zip.