CSE 520 Computer Architecture II – Spring 2020

Programming Assignment 2 (100 Points)

Cache Replacement Policies

In this assignment, you will implement a cache replacement policies primarily based on reference counter for L2 cache in gem5, i.e., RefCount (Counter-based cache replacement policy with AIP) [1]. For performance comparison, the built-in LRU policy from gem5 will also be considered.

The cache replacement policy that you will implement in gem5 require program counter information who cause the cache miss, which is not provided in the existing replacement policies infrastructure. Download the gem5.patch from canvas and apply it under the gem5 root folder. This patch allows you to access PC in replacement policy by calling getPC() from a ReplacementData. Please refer to the appendix for how to apply the patch. The patch also include changes to enable a command line option for specifying L2 cache replacement policy.

When creating the policies in gem5, the formal name for each policy needs to follow the list above. For example, "--l2_rpp=RefCount()" will be given to gem5 when RefCount cache replacement policy is used. In addition to L2 cache replacement policies, the following CPU configuration in gem5 is used during this assignment.

- AtomicSimpleCPU
  - 5 CPU cores for 4-thread fluidanimate, 6 CPU cores for 4-thread bodytrack
- L1 cache
  - 32 KB for instruction, 32 KB for data
- L2 cache
  - 1MB, and 4MB
  - 16-way set associative
  - Replacement Policies LRU and RefCount
- All other settings remain in default value.

You will run your gem5 with two workloads (fluidanimate and bodytrack) for studying different L2 cache replacement policies and different L2 cache size in 1MB, and 4MB. The inputs and the benchmark programs are in benchmark_assign2.zip. The estimated simulation time for each run should be less than 20 minutes. Hence, the total simulation time is approximately 160 minutes (2 policies, 2 cache sizes, and 2 benchmarks)

In the report, please give a brief introduction to RefCount cache replacement policy, followed by the table and graph showing overall miss rate of L2 cache collected from your simulation results.

[1] M. Kharbutli and Y. Solihin, Counter-Based Cache Replacement Algorithms, IEEE Transactions on Computers (Volume: 57, Issue: 4, April 2008), Page(s): 433 – 447.

Due Date

This assignment will be due at 11:59pm, March 17.

What to turn in for grading

1. Create a working directory for each combination of the benchmark program and cache configuration to include the output files from gem5. Prepare a **single patch file** containing your implementation of RefCount replacement policy and a **readme** file describing how you generate the patch and run the simulation. Also, please comment on your code properly before generating the patch.
2. Compress the directories, the patch file, the readme file, and your report (in pdf) into a zip archive file named **cse520-lastname_firstname-assign02.zip.** The directory structure before zipped will be look like this

```
cse520-XXX_OOO-assgn02/
├── bodytrack_LRU_1MB/
|   ├── config.ini
|   ├── config.json
|   └── state.txt
├── bodytrack_LRU_4MB/
|   └── …
├── fluidanimate_LRU_1MB/
|   └── …
├── fluidanimate_LRU_4MB/
|   └── …
├── bodytrack_RefCount_1MB/
|   └── …
├── bodytrack_RefCount_4MB/
|   └── …
├── fluidanimate_RefCount_1MB/
|   └── …
├── fluidanimate_RefCount_4MB/
|   └── … ├── patch
├── readme
└── report.pdf
```

3. Submit the zip archive to Canvas by the due date and time. No email submission will be accepted.
4. There will be 20 points penalty per day if the submission is late. Note that submissions are time stamped by Canvas. If you have multiple submissions, only the newest one will be graded. If needed, you can send an email to the instructor and TA to drop an early submission.
5. The assignment must be done individually. No collaboration is allowed, except the open discussion in the forum on Canvas. The instructor reserves the right to ask any student to explain the work and adjust the grade accordingly.
6. ASU Academic Integrity Policy (http://provost.asu.edu/academicintegrity), and FSE Honor Code (http://engineering.asu.edu/integrity) are strictly enforced and followed.

Appendix

Benchmarks to be used:

In this assignment, we will only use fluidanimate and bodytrack from previous assignment with a much smaller input dataset since the gem5 simulator is roughly 10000x slower than a real CPU.

The following 2 commands execute gem5 to simulate each benchmark with the proper input arguments to the benchmark program, assuming the benchmarks are put inside benchmarks folder in the gem5 root folder. In this 2 command, the appropriate number of CPU cores are given after '-n' flag for each benchmark while each benchmark generates 4 threads. (Do not miss the double quotation marks after -o flag)

- For flluidanimate in gem5

```
build/X86/gem5.opt configs/example/se.py -n 5 -c
benchmarks/fluidanimate/fluidanimate -o "4 5
benchmarks/fluidanimate/in_5K.fluid"
```

- For bodytrack in gem5

```
build/X86/gem5.opt configs/example/se.py -n 6 -c
benchmarks/bodytrack/bodytrack -o
"benchmarks/bodytrack/sequenceB_1 4 1 100 5 2 4"
```

In addition to the benchmark programs, a simple program finding the maximum number using 4 threads is included under benchmarks/max_thread. This is only for you to quickly debug your code as it only requires less than 10 seconds to complete in gem5. This program does not require any argument to run, but do need you to specify **5** cpu cores for gem5 to simulate this program.

Generate and Apply a Patch

Linux Patch file is a file format containing the difference between 2 given files or file groups. The main usage of a patch file is to deploy the update in small size as the patch file only lists the differences instead of the whole context of the files.

To make a patch file, it is required to prepare 2 working folders, as for the comparison between prior and post modification. The following command gives an example, which essentially lists all the differences from all files between the 2 folders into a file named `PATCH_FILE`.

```
diff -Naur folder_old folder_new > PATCH_FILE
```

In this assignment, you only modify the code under gem5/src/. Please only diff this folder with the unmodified src folder (**AFTER** applying gem5.patch). Also make sure you generate the patch in gem5 root folder. **DO NOT** diff the entire gem5 folder otherwise the compiled binary would also be added into the patch file. Failing to do so will likely get you 0 point as TAs will be nearly impossible to reproduce your modification for gem5.

Alternatively, since the gem5 is downloaded from Github and is tracked by git, you can use git to generate a patch, provided you are familiar with git operations. How to use git to generate a patch is out of the scope of this assignment though. (But do state how you generate your patch in your readme)

You should verify your patch file by applying it to a clean gem5 folder using the following command, the same command to apply the gem5 patch downloaded from Canvas.

```
patch -p0 -i PATCH_FILE
```