CSE 490/590 PROJECT 2

ANALYSIS OF CACHE PARAMETERS AND TRADE-OFFS ON X86 PROCESSOR USING GEM5

DUE: 6th MAY 2019, 11:59 pm

You can work in groups of two or individually on this project.

All students should send an **email** to both TAs choosing one of the options below:

1. Working in the same group as in Project 1
2. Working individually as in Project 1
3. Was working in a group for Project 1 and going to work individually for Project 2
4. Was working individually for Project 1 and going to work in a group for Project 2

If you have chosen 4, make sure you let us know who you'll be working with. Again, CSE490 students can work with another CSE490 students only and CSE590 students can work with another CSE590 students only.

Cache parameters influence the performance of a processor to a great extent. In this project you are asked to tune various cache parameters and analyze performance in each case.

The architecture you will have to analyze is the X86 architecture.

Part 1: Gem5 Introduction

The Gem5 simulator is a modular platform for computer-architecture research. Read through the below documentation for more details on how to use the tool and male necessary modifications.

1. Gem5 Wiki: http://gem5.org/Main_Page
2. Gem5 tutorial from ASPLOS 2018: http://learning.gem5.org/book/index.html

Follow Part 2 or Part 3 depending on how you want to use Gem5

Part 2: Setting up Gem5 on your computer

Gem5 needs to be installed on an Ubuntu system. If you do not have a Linux operating system, you need to install a Virtual Machine with the information given below:

1. Download and install the free software VirtualBox, a virtualization product from: http://www.virtualbox.org/
2. Download and install the latest stable version of Ubuntu. Get an ISO disk image from: http://www.ubuntu.com/desktop/
3. Fire up the virtual box
   a. Create a new virtual machine with the default settings
   b. Point the CD image to the Ubuntu ISO image
   c. Start the Virtual machine: the installation of Ubuntu should proceed with minimum or no delays/problems.

Open the terminal, and download Gem5 from https://gem5.googlesource.com/public/gem5

Part 3: Accessing Gem5 on CSE server metallica

The CSE server on which Gem5 has been installed for your use is 'metallica.cse.buffalo.edu', which is accessible from on-campus machines or via VPN from off-campus machines.
To login, you need to use your UBIT names and passwords.

The Gem5 simulator is found in "/util/gem5"
You can find and use all related folders for X86 architecture which is built and ready for analysis. However, you will not be able to modify the scripts here. If you want to output files from this directory, you need to define an output directory in your local network drive and use this as the destination directory.

Also, you may need to update your PATH and LD_LIBRARY_PATH environment variables to use the /util versions of gcc. That is, add /util/gcc/bin to your PATH and /util/gcc/lib64 to your LD_LIBRARY_PATH

Part 4: Setting up CPU Benchmarks

CPU benchmarks are used to simulate the architectural designs and can be downloaded from https://github.com/timberjack/Project1_SPEC.

The benchmarks we will be using are:

1. 401.bzip2
2. 429.mcf
3. 456.hmmer
4. 458.sjeng
5. 470.lbm

For purposes of this project, these benchmarks have already been downloaded and are ready for use under the directory: "/util/gem5/benchmark".

There are two modes of simulating Gem5: SE (System Call Emulation) mode and FS (Full System) mode. For this project, we will use the SE mode.
To execute a benchmark program using SE mode, command line instruction need to follow the given format (example given here is for X86):

```
time $GEM5_DIR build/X86/gem5.opt
        -d <output directory>                   #output directory
        -I 100000000                            # max instructions
        $GEM5_DIR/configs/example/se.py         #define system script
        -c <program>                            #define benchmark
        -o <argument>                           #arguments of benchmark
        <other options>                         #cache parameter settings
```

Within each benchmark directory, you can find a "src" directory that contains the source code and an executable binary, and a "data" directory that contains any necessary input files/arguments. Within the "src" directory, the executable file named "benchmark" is the program that you need to simulate with. As some of the benchmark programs can take a long time, you can define a max number of instructions to be executed, e.g. a maximum of 100000000 instructions.
-d <output directory> is not needed if the project is done on your personal computer's VM.

Part 5: Cache memory parameter analysis

The parameters that can be varied are:

1. L1 D Cache size
2. L1 I Cache size
3. L2 Cache size
4. L1 D Associativity
5. L1 I Associativity
6. L2 Associativity
7. Block size
8. Benchmark

During analysis, modify only one of the parameters at a given time.

Every parameter should be tested with at least 5 different cases, for all 5 benchmarks.

For each case, calculate the following and plot graphs to show the trend in the relationship between the parameters. After each modification is made and the corresponding script is executed, view the stats.txt file to read the hit rates and miss rates.

1. Hit rate of L1 D cache
2. Miss rate of L1 D cache
3. Hit rate of L1 I cache
4. Miss rate of L1 I cache
5. Hit rate of L2 cache
6. Miss rate of L2 cache
7. CPI

Also, for each case, calculate the CPI using the formula given below:

$$CPI = 1 + \frac{(IL1.miss\_num + DL1.miss\_num) \times 6 + L2.miss\_num \times 50}{Total\_Inst\_num}$$

Part 6: Useful things to know

In Gem5, there is one unified mode, which contains separated L1 cache and unified L2 cache. For the cache replacement policy, LRU and Pseudo LRU policy are available. The cache replacement policy can be modified through the configuration script. For this project however, you do not need to consider changes to the replacement policy.

Do not exceed 512KB on L1 (in total) and 4MB on L2 cache.

Part 7: Submission

On completion, each project team should turn in a compressed folder with the following items:

1. A detailed project report with a step by step explanation of how you went about this project, including all the commands you used, graphs/charts/plots and explanation on why the graph trends are obtained in that particular way. Report should also contain tabulated benchmarks, hit rates, miss rates and CPI obtained in each case.
2. Another compressed folder that contains all the stats.txt files, with proper naming for identification.