**CS570 “Advanced Computer Architecture”**

**Homework 4**

***Submission:***

***Due by end of the day of 03/03/2020***

***This is an individual assignment!***

***Total points 100 - Late penalty: 10% penalty for each day late***

***Please upload your assignment on Blackboard with the following name:***

***CS570\_SectionNumber\_LastName\_FirstName\_HW4.*pdf*.***

***Please do NOT email your assignment to the instructor and/or TA!***

In this assignment, you need to run gem5 with some configurations such as CPU configurations and cache configurations. We recommend a **64-bit Linux machine** or **virtual machine** to do this assignment. If you use the virtual machine, I suggest increasing the amount of memory allocated to it.

1. **Background**

Gem5 can be configured as different computer systems with different CPUs, caches and memories.

**2. Gem5 Explanation/Tutorial**

* We can provide command line arguments for programs running under gem5, for example:

|  |
| --- |
| build/X86/gem5.opt ./configs/example/se.py --caches --l1d\_size=32kB --l1d\_assoc=2 --l1i\_size=32kB --l1i\_assoc=2 --l2cache --l2\_size=2MB --l2\_assoc=8 --cacheline\_size=64 --cpu-type=DerivO3CPU --mem-type=SimpleMemory --mem-size=8192MB -c './queens' -o '-c 10' --cpu-clock=2GHz |

runs the command ./queens -c 10 in the simulator.

* The simulator will output some spurious warning messages like:

|  |
| --- |
| info: Entering event queue @ 0. Starting simulation...  info: Increasing stack size by one page.  info: Increasing stack size by one page.  warn: ignoring syscall access(...)  warn: ignoring syscall mprotect(...)  warn: ignoring syscall mprotect(...) |

these are harmless.

* The simulator outputs a message like:

|  |
| --- |
| Exiting @ tick 357959500 because exiting with last active thread context |

Which indicates the simulation tick which the program completed on. By default, each simulation tick represents 1 picosecond of simulation time, and the simulated CPU has a clock rate of 2 GHz, so this simulation represents 357959500 ticks / (500 ticks / clock cycle) = 715919 clock cycles of simulated time.

* The more important outputs from the simulation are in the m5out directory:

config.ini, config.json: contain the full configuration of the components of the simulation.

stats.txt: contain numerous statistics from the simulation. Interesting statistics include:

1. sim\_seconds: simulation time

2. system.cpu.ipc: instructions per cycle achieved by the simulated CPU

3. system.cpu.dache.overall\_miss\_rate::cpu.data: L1 data cache miss rate

4. system.l2.overall\_misses::total: L2 cache miss rate

**3. Supplied Benchmark Programs**

Several benchmark programs are selected to explore a range of demands on the simulated processor. Please download from <https://github.com/Xiaoyang-Lu/CS570-benchmark.git>

Two programs are introduce here:

1. BFS: computes a breadth-first search problem. This is taken from the Problem Based Benchmark Suite. Source code for this benchmark, along with utilities for generating graph data is in the breadthFirstSearch directory.

We supply some example graphs in the inputs directory. Our suggested command-line for this program is

|  |
| --- |
| ./BFS path/to/RL3k.graph |

*This program was selected because it should have poor data cache locality.*

2. queens: queens: solves the N-queen problem for an N specified as an argument.

Our suggested command-line for this program is

|  |
| --- |
| ./queens -c 10 |

The -c option indicates to count solutions instead of printing out any solutions.

*This program was selected because it should be very friendly to the cache, but very challenging for branch prediction.*

**4. Building Benchmarks**

You do not need the following instructions if you are using our prebuilt archives on a 64-bit Linux system.

The benchmarks include a Makefile. If you have GNU make and gcc and g++ installed, running make clean, then make should rebuild all the benchmark programs for your system.

**5. Tasks – Optimize Miss Rate for each benchmark**

With a different configuration, we will find that many factors can influence the performance of program and cache hierarchy. By exploring the design space and trying different configurations, in this assignment, we will try to find a good configuration of cache which provides lowest miss rate.

You should run the gem5 with the hardware configurations, with Syscall Emulation (SE) system mode, also with the BFS(RL3k.graph) and queues(-c 10) benchmark.

The default hardware configurations are as follows:

o CPU mode: O3 mode

o The number of CPU cores: 1

o CPU frequency: 2GHz

o Caches: with two level caches which are L1 cache and L2 cache

2-way L1 dcache with the size of 4KB

2-way L1 icache with the size of 4KB

8-way L2 cache with the size of 2MB

the cacheline size of these caches are 64 Byte

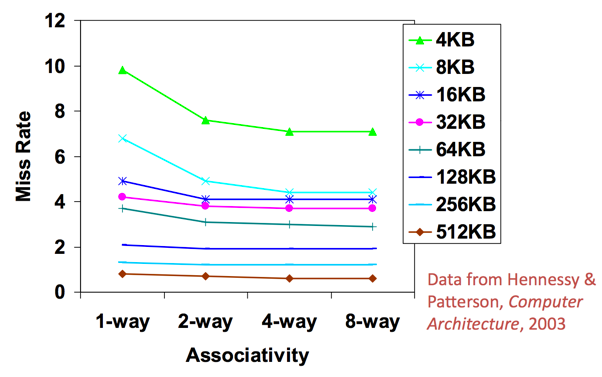
o Memory mode: Simple Memory

o Memory size: 8GB

Please explore the tradeoffs between different configurations and L1 dcache’s miss rate, which include associativity and size allocation for L1 data cache (associativity: 1-way, 2-way, 8-way; size: 4KB, 64KB, 512KB).

For each application, you need to try a total of 9 (3\*3) configurations.

You should explain the reasoning about your result for each configuration and present graphs (as shown below) showing the trade-offs (configuration and L1 dcache miss rate) between the design choices.



If you do not know that how many available configuration options your gem5 has, you can use the command “./build/X86/gem5.opt configs/example/se.py --help” to check these configuration options.

**6. What to turn in**

A written report of the assignment in PDF format. This is your chance to explain your steps of running gem5 with these configurations, the explanation of your results and the graphs about different configurations and L1 dcache miss rate are needed.