[CPEN-411 2022] Assignment 3:
Deep-Diving into the Architectural Simulator and Code Optimization

This assignment aims to understand the implementation of architectural simulators and how to generate traces for the simulator from applications. The assignment also aims to optimize a naive matrix multiplication code using the architectural knowledge you have learned during the course. **ChampSim** is used to help you have a better understanding of architectural simulator usages and test your optimized matrix multiplication code. The assignment is divided into three parts. The first part discusses the preparatory steps. During the second part, you need to understand the simulator implementation and implement a non-pipelined core by modifying the current out-of-order core implementation of **ChampSim (1.5 points)**. Lastly, you will need to implement two optimized matrix multiplication implementations using your architectural knowledge (e.g., resolving data dependencies and increasing cache hit rate) (**3.5 points).**

Please read this document very carefully!

Preparatory Steps

1. **[0 Marks]** untar a**ssignment3.tar.gz**
   $ tar -zxvf a**ssignment3.tar.gz**

```
➜  ~ scp assignment3.tar.gz jhwoo36@ssh.ece.ubc.ca:~/
jhwoo36@ssh-linux4:~$ ls
assignment1  assignment3.tar.gz  solution  ta
jhwoo36@ssh-linux4:~$ tar -zxvf assignment3.tar.gz
```

2. **[0 Marks]** Change directory into **ChampSim** directory.

```
jhwoo36@ssh-linux4:~$ cd assignment3/ChampSim/
jhwoo36@ssh-linux4:~/assignment3/ChampSim$ ls
algorithms  champsim_config.json  inc       output      results        tracer
bin         compile_champsim.sh   LICENSE   prefetcher  run_all.sh
branch      config.sh             Makefile  README.md   run_single.sh
btb         create_submission.sh  obj       replacement src
jhwoo36@ssh-linux4:~/assignment3/ChampSim$
```

3. **[0 Marks]** Change directory into **src** directory.

```
jhwoo36@ssh-linux4:~/assignment3/ChampSim$ cd src/
jhwoo36@ssh-linux4:~/assignment3/ChampSim/src$ ls
cache.cc            main.cc     ptw.cc          vmem.cc
dram_controller.cc  ooo_cpu.cc  tracereader.cc
```

   a. The cache.cc file implements the cache.
   b. The dram_controller.cc file implements the memory controller.
   c. The ooo_cpu.cc file implements the out-of-order processor.
   d. The ptw.cc file implements the page-table walk.
   e. The vmem.cc file implements the virtual memory.
   f. The main.cc file is the main function of the simulator
   g. The tracereader.cc file is used to read the trace files.

Please go through each source code carefully and try to understand the implementation. You will need to modify a few files to implement a non-pipelined processor.

4. **[0 Marks]** Change directory into **Champsim** directory.

```
jhwoo36@ssh-linux4:~/assignment3/ChampSim$ ls
algorithms  champsim_config.json  inc       prefetcher   run_all.sh
bin         compile_champsim.sh   LICENSE   README.md    run_single.sh
branch      config.sh             Makefile  replacement  src
btb         create_submission.sh  output    results      tracer
```

5. **[0 Marks]** Build **Champsim**.

```
jhwoo36@ssh-linux4:~/assignment3/ChampSim$ bash compile_champsim.sh
g++ -Wall -O3 -std=c++17 -Iinc -MMD -MP  -c -o src/ptw.o src/ptw.cc
g++ -Wall -O3 -std=c++17 -Iinc -MMD -MP  -c -o src/tracereader.o src/tracereader.c
c
g++ -Wall -O3 -std=c++17 -Iinc -MMD -MP  -c -o src/ooo_cpu.o src/ooo_cpu.cc
src/ooo_cpu.cc: In member function ‘void O3_CPU::init_instruction(ooo_model_instr)
```

6. **[0 Marks]** Change directory into **algorithms** folder.

```
jhwoo36@ssh-linux4:~/assignment3/ChampSim$ cd algorithms/
jhwoo36@ssh-linux4:~/assignment3/ChampSim/algorithms$ ls
assignment3.o  matmul_basic.c  matmul_opt1.c  matmul_opt2.c  README.md
Makefile       matmul_basic.h  matmul_opt1.h  matmul_opt2.h  runall.sh
```

7. **[0 Marks]** Understand the basic matrix multiplication in **matmul_basic.c** and **matmul_basic.h**

```
jhwoo36@ssh-linux4:~/assignment3/ChampSim/algorithms$ vim matmul_basic.c
```

```
#include "matmul_basic.h"

void matmul_basic(unsigned long long int* mat_a, unsigned long long int* mat_b, unsigned long long int* mat_c, unsigned long long int matrix_size){
    for(unsigned long long int i=0; i<matrix_size; i++){
        for(unsigned long long int j=0; j<matrix_size; j++){
            for(unsigned long long int k=0; k<matrix_size; k++){
                mat_c[j*matrix_size + i] += mat_a[k*matrix_size + i] * mat_b[j*matrix_size + k];
            }
        }
    }
}
```

8. **[0 Marks]** Compile the source files in the **algorithms folder**.

```
jhwoo36@ssh-linux4:~/assignment3/ChampSim/algorithms$ make
gcc -c matmul_opt1.c
gcc -c matmul_opt2.c
gcc -c matmul_basic.c
gcc -O3 assignment3.o matmul_opt1.o matmul_opt2.o matmul_basic.o -o assignment3.bin
jhwoo36@ssh-linux4:~/assignment3/ChampSim/algorithms$ ls
assignment3.bin  matmul_basic.c  matmul_opt1.c  matmul_opt2.c  README.md
assignment3.o    matmul_basic.h  matmul_opt1.h  matmul_opt2.h  runall.sh
Makefile         matmul_basic.o  matmul_opt1.o  matmul_opt2.o
```

9. **[0 Marks]** Execute your compiled code.

```
jhwoo36@ssh-linux4:~/assignment3/ChampSim/algorithms$ bash runall.sh
```

10. **[0 Marks]** Change directory into **output** folder.

```
jhwoo36@ssh-linux4:~/assignment3/ChampSim/algorithms$ cd ../output/
jhwoo36@ssh-linux4:~/assignment3/ChampSim/output$ ls
matmul_basic.txt  matmul_opt1.txt  matmul_opt2.txt
```

11. **[0 Marks]** Make sure that **matmul_basic.txt** has **PASSED**.

```
jhwoo36@ssh-linux4:~/assignment3/ChampSim/output$ vim matmul_basic.txt
jhwoo36@ssh-linux4: ~/assignment3/ChampSim/output
PASSED
```

12. **[0 Marks]** Change directory into **tracer** folder.

```
jhwoo36@ssh-linux4:~/assignment3/ChampSim/output$ cd ../tracer/
jhwoo36@ssh-linux4:~/assignment3/ChampSim/tracer$ ls
champsim_tracer.cpp  obj-intel64                          README.md
generate_trace.sh    pin-3.22-98547-g7a303a835-gcc-linux  traces
```

13. **[0 Marks]** Create a trace for the **basic matrix multiplication code (matmul_basic).**

```
jhwoo36@ssh-linux4:~/assignment3/ChampSim/tracer$ bash generate_trace.sh 0
```

You need to **change the number within the green box following the trace you want to generate.**

- 0: Trace for **the matmul_basic.**
- 1: Trace for **the matmul_opt1.**
- 2: Trace for **the matmul_opt2.**

14. **[0 Marks]** Change directory into **traces** folder, and check if the trace of **matmul_basic** is generated correctly.

```
jhwoo36@ssh-linux4:~/assignment3/ChampSim/tracer$ cd traces/
jhwoo36@ssh-linux4:~/assignment3/ChampSim/tracer/traces$ ls -l
total 753032
-rw-r--r-- 1 jhwoo36 compeng 640000000 Oct 26 21:18 matmul_basic-720M.champsim
```

15. **[0 Marks]** Compress your trace file as an xz file.

```
jhwoo36@ssh-linux4:~/assignment3/ChampSim/tracer/traces$ xz -v3 matmul_basic-720M.
champsim
matmul_basic-720M.champsim (1/1)
 49.4 %         0.2 MiB / 302.1 MiB = 0.001    38 MiB/s       0:08         8 s

jhwoo36@ssh-linux4:~/assignment3/ChampSim/tracer/traces$ ls
matmul_basic-720M.champsim.xz
```

NOTE: An uncompressed trace file is almost **640MB**, so **always compress your trace first** before generating other traces.

16. **[0 Marks]** Change into **Champsim** folder.

```
jhwoo36@ssh-linux4:~/assignment3/ChampSim$ ls
algorithms  champsim_config.json  inc       output       results       tracer
bin         compile_champsim.sh   LICENSE   prefetcher   run_all.sh
branch      config.sh             Makefile  README.md    run_single.sh
btb         create_submission.sh  obj       replacement  src
```

17. **[0 Marks]** Run your trace with Champsim. **(NOTE: It would take ~5 minutes)**

```
jhwoo36@ssh-linux4:~/assignment3/ChampSim$ bash run_single.sh 0
```

You need to **change the number within the green box following to run the specific trace you have generated.**

- 0: Trace for **the matmul_basic.**
- 1: Trace for **the matmul_opt1.**
- 2: Trace for **the matmul_opt2.**

18. **Or, if you have generated all the traces, you could use the following command. (NOTE: It would take ~15 minutes)**

```
jhwoo36@ssh-linux4:~/assignment3/ChampSim$ bash run_all.sh
```

19. **[0 Marks]** Once your simulations finish executing, change directory into **results** folder.

```
jhwoo36@ssh-linux4:~/assignment3/ChampSim$ cd results/
jhwoo36@ssh-linux4:~/assignment3/ChampSim/results$ ls
matmul_basic.result
```

20. Check your simulation results.

```
jhwoo36@ssh-linux4:~/assignment3/ChampSim/results$ vim matmul_basic.result

ChampSim completed all CPUs

Region of Interest Statistics

CPU 0 cumulative IPC: 0.771648 instructions: 10000000 cycles: 12959285
cpu0_DTLB TOTAL     ACCESS:    3072659  HIT:    2952812  MISS:     119847
cpu0_DTLB LOAD      ACCESS:    2717092  HIT:    2597245  MISS:     119847
cpu0_DTLB RFO       ACCESS:     355567  HIT:     355567  MISS:          0
cpu0_DTLB PREFETCH  ACCESS:          0  HIT:          0  MISS:          0
cpu0_DTLB WRITEBACK ACCESS:          0  HIT:          0  MISS:          0
cpu0_DTLB TRANSLATION ACCESS:          0  HIT:          0  MISS:          0
cpu0_DTLB PREFETCH  REQUESTED:          0  ISSUED:          0  USEFUL:          0  USELESS:          0
cpu0_DTLB AVERAGE MISS LATENCY: 9.27769 cycles
cpu0_ITLB TOTAL     ACCESS:      35348  HIT:      35348  MISS:          0
cpu0_ITLB LOAD      ACCESS:      35348  HIT:      35348  MISS:          0
cpu0_ITLB RFO       ACCESS:          0  HIT:          0  MISS:          0
cpu0_ITLB PREFETCH  ACCESS:          0  HIT:          0  MISS:          0
cpu0_ITLB WRITEBACK ACCESS:          0  HIT:          0  MISS:          0
cpu0_ITLB TRANSLATION ACCESS:          0  HIT:          0  MISS:          0
cpu0_ITLB PREFETCH  REQUESTED:          0  ISSUED:          0  USEFUL:          0  USELESS:          0
cpu0_ITLB AVERAGE MISS LATENCY: -nan cycles
cpu0_STLB TOTAL     ACCESS:     119847  HIT:     119687  MISS:        160
cpu0_STLB LOAD      ACCESS:     119847  HIT:     119687  MISS:        160
cpu0_STLB RFO       ACCESS:          0  HIT:          0  MISS:          0
cpu0_STLB PREFETCH  ACCESS:          0  HIT:          0  MISS:          0
cpu0_STLB WRITEBACK ACCESS:          0  HIT:          0  MISS:          0
cpu0_STLB TRANSLATION ACCESS:          0  HIT:          0  MISS:          0
cpu0_STLB PREFETCH  REQUESTED:          0  ISSUED:          0  USEFUL:          0  USELESS:          0
cpu0_STLB AVERAGE MISS LATENCY: 208 cycles
cpu0_L1D TOTAL     ACCESS:    2835927  HIT:    2567359  MISS:     268568
cpu0_L1D LOAD      ACCESS:    2358109  HIT:    2089541  MISS:     268568
cpu0_L1D RFO       ACCESS:     477658  HIT:     477658  MISS:          0
cpu0_L1D PREFETCH  ACCESS:          0  HIT:          0  MISS:          0
cpu0_L1D WRITEBACK ACCESS:          0  HIT:          0  MISS:          0
cpu0_L1D TRANSLATION ACCESS:        160  HIT:        160  MISS:          0
cpu0_L1D PREFETCH  REQUESTED:          0  ISSUED:          0  USEFUL:          0  USELESS:          0
cpu0_L1D AVERAGE MISS LATENCY: 23.0189 cycles
cpu0_L2C TOTAL     ACCESS:     269499  HIT:     137014  MISS:     132485
cpu0_L2C LOAD      ACCESS:     268568  HIT:     136640  MISS:     131928
cpu0_L2C RFO       ACCESS:          0  HIT:          0  MISS:          0
cpu0_L2C PREFETCH  ACCESS:          0  HIT:          0  MISS:          0
cpu0_L2C WRITEBACK ACCESS:        931  HIT:        374  MISS:        557
cpu0_L2C TRANSLATION ACCESS:          0  HIT:          0  MISS:          0
cpu0_L2C PREFETCH  REQUESTED:          0  ISSUED:          0  USEFUL:          0  USELESS:          0
cpu0_L2C AVERAGE MISS LATENCY: 24.3643 cycles
LLC TOTAL     ACCESS:     132859  HIT:     127256  MISS:       5603
LLC LOAD      ACCESS:     131928  HIT:     126325  MISS:       5603
LLC RFO       ACCESS:          0  HIT:          0  MISS:          0
LLC PREFETCH  ACCESS:          0  HIT:          0  MISS:          0
LLC WRITEBACK ACCESS:        931  HIT:        931  MISS:          0
LLC TRANSLATION ACCESS:          0  HIT:          0  MISS:          0
LLC PREFETCH  REQUESTED:          0  ISSUED:          0  USEFUL:          0  USELESS:          0
LLC AVERAGE MISS LATENCY: 81.6373 cycles

DRAM Statistics
 CHANNEL 0
 RQ ROW_BUFFER_HIT:       4524  ROW_BUFFER_MISS:       1079
 DBUS AVG_CONGESTED_CYCLE: -
 WQ ROW_BUFFER_HIT:          0  ROW_BUFFER_MISS:        118  FULL:          0

CPU 0 Branch Prediction Accuracy: 99.2239% MPKI: 0.1868 Average ROB Occupancy at Mispredict: 175.246
Branch type MPKI
BRANCH_DIRECT_JUMP: 0
BRANCH_INDIRECT: 0
BRANCH_CONDITIONAL: 0.1868
BRANCH_DIRECT_CALL: 0
```

21. **[0 Marks]** Throughout the assignment you are allowed to change only the **matmul_opt1, matmul_opt2 files** in the algorithms **folder,** and the files in the **src folder**. Also, README files in each directory have the necessary information for the assignment.

Evaluated Steps [5 marks]: Implementing a non-pipelined processor and 2 optimized matrix multiplication codes

These will take you FULL 2 weeks. Please start this TODAY ITSELF!

1. **PART-A [1.5 Marks]** Implement a **non-pipelined processor**. In this part, your processor fetches one instruction per core and waits for it to be completed/committed by the Reorder Buffer before fetching a new instruction again.

   For this part, first, you need to understand the simulator's overall implementation by carefully going through each source code in the **src** folder. Then, modifying the current out-of-order pipelined processor implementation into a non-pipelined one. You can edit any source code in the **src** folder to do this. Once you implement this you need to **REBUILD YOUR CHAMPSIM** following **step 5** in the preparatory steps. Once you successfully rebuild your Champsim, run the basic matrix multiplication algorithm with your non-pipelined processor and compare the results. You need to report the IPC of this implementation in the **Assignment-3-PART-A** Entry on Canvas.

2. **PART-B [3 Marks]** For this part, you will need to use the original pipelined implementation of Champsim. So please be careful to not use the code from PART-A.

   This part implements **two** optimized matrix multiplication based on the "architectural knowledge" that you have learned during the course. For instance, you could optimize your code to have cache-friendly access or to avoid data dependencies.

   a. **[2 Marks]** Edit the *matmul_opt1 files* to perform the same matrix multiplication task and achieve more than 30% speed-up with respect to the basic matrix multiplication code. You will need to run both of these traces on ChampSim to measure speedup.
   b. **[1 Mark]** Edit the *matmul_opt2 files* to perform the same matrix multiplication task and reduce the L1D cache miss-rate by 8X as compared to the basic matrix multiplication code. You will need to run both of these traces on ChampSim to measure the reduction in the L1D cache miss rate.

3. **PART-B [0.5 Mark]** Write a report in pdf format explaining how you optimized matrix multiplication implementations and why you thought that would be effective. Also, analyze your results compared to others. This pdf should also have graphs that compare:
   a. **Cumulative IPC**
   b. **L1D Miss Rate**
   c. **L2C Miss Rate**
   d. **LLC Miss Rate**
   e. **Branch Prediction Accuracy**

   These graphs should include all your optimized matrix multiplication implementations.
   Please place your pdf report in the main ChampSim folder. The report should have a .pdf extension. If you do not do this, the submission script will not include your report.

Submission [PART-B]

To submit, please execute the following command within the **ChampSim folder.**

```
jhwoo36@ssh-linux4:~/assignment3/ChampSim$ bash create_submission.sh
```

You need to submit a compressed file in the **Assignment-3-PART-B** Entry on Canvas.

Thus, be extremely careful and double check if your results, algorithm folder, etc. are the right ones We have ~80 students in this course and we will not re-evaluate your Assignment if you submit an incorrect or stale work.