Project 5

Due: 12/12/2025 10:30 AM

서론

(1) 각 프로젝트의 파일은 Project<번호> -> Problem<번호> 폴더 구조로 구성한다. 예를 들어, Project5->Problem1 의 폴더에 Problem1 에서 요구하는 모든 파일을 저장한다. Project 폴더를 압축(zip)하고 압축 파일 이름은 학번으로 하여 제출한다. 명시된 폴더 구조를 따르지 않아 발생하는 불이익에 대한 책임은 본인에게 있음.

(2) 보고서가 필요한 경우는 pdf 형식으로 제출하며, 언어는 영어/한글 모두 무방하다.

(3) 프로젝트의 deadine 까지 LMS 의 <과제 및 평가>란에 제출한다.

(4) 프로젝트의 마감기한을 지키지 못한 경우, 감점 없이 0 점 처리한다.

(5) 프로젝트는 개별로 수행하며, 다른 사람 또는 자료에서 도움을 받거나 이를 참고로 하였을 경우 관련 소스코드와 보고서에 출처를 명확히 제시하여야 한다. 또한, 다른 학생에게 도움을 주었을 경우에도 소스코드와 보고서에 이에 관련하여 서술한다. 그렇지 않을 경우 모두 0 점 처리한다.

문제 1(50 점): gem5 실행을 위한 환경설정

Project5_overview 자료의 [Environmental Setup]에 제시된 gem5 실행환경 구축

(1) (20 점) Project5_overview 자료의 [Environmental Setup]에 제시된 대로 gem5 를 빌드
- 빌드 후 출력 터미널(problem1_1.jpg)
- 빌드 후 "ls -al" command 로 ~/path/to/gem5/build/X86 디렉토리를 출력하고, "./gem5.opt –version" command 로 gem5 버전을 출력한 터미널(problem1_2.jpg)

(2) (30 점) 제공된 간단한 코드(test.cpp)를 Project5_overview 자료의 reference virtual CPU spec 을 활용하여 gem5 로 실행
- Core 수는 1 로 설정하여 실행 후, profiling 결과(stats.txt)를 data.xlsx 의 test tab 에 기록한다.

문제 2(200점): GeMM code 최적화 및 gem5 기반 성능측정 및 분석

이 프로젝트에서는 생성형 인공지능 모델의 실행시간에서 많은 부분을 차지하는 MatMul(Matrix Multiplication) 연산의 최적화를 목표로 한다. 이를 위해, 우선 최적화를 적용하기 이전의 MatMul 연산에 대한 baseline(matmul_base.cpp)의 실행 성능을 측정한다. 그리고, baseline 코드를 수업시간에 배운 내용을 참고하여 최적화한다. 또한, 최적화 전후에 대한 성능지표를 분석한다. 문제 2의 진행은 Project5_overview 자료의 [Benchmark & Modification]와 [Profiling Guide] 파트를 참고한다.

(1)(30 점) MatMul 연산의 baseline 코드를 gcc 로 컴파일하여 gem5 실행 후, Project5_overview 에 제시된 성능 지표 측정

- Project5_overview 자료의 reference virtual CPU spec 을 활용한다.
- Matrix 크기 (M, K, N)={(16, 512, 512), (16, 1024, 1024), (16, 2048, 2048)}에 대해 MatMul 연산을 3 번 반복 실행하였을 때 성능 측정
- OpenMP thread 수를 1 로 설정하여 실행시간을 측정한다.
  - OpenMP 의 thread 수를 설정하는 command: "export OMP_NUM_THREADS=1"
- 성능 지표 측정결과를 data.xlsx 의 perf tab 에 기록하고, 성능 분석 내용은 보고서(project5.pdf)에 정리한다.
  - 각 성능 지표에 대한 분석

(2) (50 점) MatMul 연산의 baseline 코드에 적용할 최적화에 대한 자세한 설명과 성능 이를 적용한 optimized 코드(matmul_opt.cpp)를 보고서(project5.pdf)에 제시

(3) (20 점) Optimized MatMul 코드를 gcc 로 컴파일한 후, baseline MatMul code 실행결과간 correctness 확인

- Matrix 크기 (M, K, N)={(16, 512, 512), (16, 1024, 1024), (16, 2048, 2048)}에 대해 MatMul 연산을 3 번 반복 실행하였을 때 동일한 결과가 출력된 터미널 (problem2.jpg)

(4) (50 점) Optimized MatMul 코드를 1)과 같이 gem5 실행 후 성능지표 측정

- (1)과 동일하게 진행한다.

(5) (50 점) Baseline 과 optimize 간 성능 비교 내용을 분석한 보고서(project5.pdf)

- Speed up 의 근거를 성능지표들간 관계를 통해 제시한다.
- Matrix size 변화에 대한 성능 분석