운영체제 과제 #1

2020년 4월 15일

1 과제 개요

- 이진 파일 a.dat에 있는 행렬 A[ ][ ]와 이진 파일 b.dat에 있는 행렬 B[ ][ ]를 읽어 그 행렬곱 C[ ][ ] (C = A * B)을 구하여 이진 파일 c.dat에 저장
- 개인 과제로 수행하며 보고서와 소스코드를 압축하여 파일명을 hw1-20141234홍길동.zip의 형식으로 elearning 시스템에 제출

2 과제 설명

2.1 준비사항

- POSIX 표준이 지원되는 시스템에서 컴파일 되고 수행되는 프로그램 작성
  - 과제 수행여부는 Ubuntu 16.04가 설치되어 있는 Linux 시스템에서 gcc로 컴파일하여 확인할 예정
  - 대부분의 Linux 배포판, Mac OS X와 Windows의 cygwin에서 POSIX 표준을 지원함
- 이진 파일 a.dat, b.dat, 그리고 테스트를 위한 c.dat는 과제 첨부 파일에서 다운로드 할 것. 또한 두 이진 파일의 소수점 차이를 감안하기 위한 간단한 diff 프로그램인 hwdiff.c 파일이 제공됨.

2.2 과제 명세

- 프로그램 동작
  - 동작 예: hw1 a.dat b.dat c.dat
  - 입력으로 주는 a.dat와 b.dat 파일은 이진 float data가 2048 * 2048개 들어있으며, row-oriented로 구성되어 있다. 즉, A[0][0], A[0][1], A[0][2], ..., A[0][1023], A[1][0], A[1][1], A[1][2], ..., A[1023][1023]의 순서로 4 byte 크기 (`sizeof(float)`)로 들어 있다.

– 이러한 파일 구성에 대해서 다음과 같은 코드로 배열 A[i][j]를 읽을 수 있다. 여기서 행렬은 $M \times N$ 행렬이라고 하자.

```
#include <unistd.h>

lseek(fp, sizeof(float) * (i * N + j), SEEK_SET);
read(fp, &(A[i][j]), sizeof(float));
```

반드시 lseek()나 read()로 읽을 필요는 없으며 fseek()와 fread()를 사용해도 된다. 행렬도 2차원 배열뿐만 아니라 1차원 배열로 구성해도 되며 심지어 read() 계열 대신 mmap()을 사용해도 된다.

- 제대로 수행되는지 확인

  – 생성된 c.dat 파일과 과제 명세서에 첨부된 c.dat 파일을 비교하여 제대로 수행되는지 확인할 수 있다. 첨부된 c.dat파일을 c1.dat로 이름을 변경한 뒤 다음과 같이 수행한다. diff에서 아무런 메시지가 나오지 않으면 두 파일이 동일한 것이며 제대로 결과가 나왔음을 의미한다.

```
$ ./hw1 a.dat b.dat c.dat

$ diff c.dat c1.dat
```

  – diff 명령문은 두 파라미터가 가리키는 파일의 내용이 동일하면 아무런 메시지를 출력하지 않는다.

  – 컴퓨터의 실수 연산으로 인한 round off 에러가 발생하면 linux의 diff 함수는 잘못된 결과를 출력할 수 있다. 이 경우에는 첨부된 hwdiff 파일을 다음과 같이 수행한다. -lm 옵션을 잊지 않도록 하자. 기존 diff와 마찬가지로 두 파일이 동일한 경우 아무런 메시지가 출력되지 않는다.

```
$ gcc -g -Wall -o hwdiff hwdiff.c -lm
$ ./hwdiff c.dat c1.dat
```

  – hw1 프로그램에 수행시간을 출력하도록 했으면 위와 같이 빈 칸으로 나오지 않고 수행시간이 출력될 것이다.

3 보고서 내용

보고서에는 다음과 같은 내용을 포함해야 한다. 출력할 보고서가 아니므로 표지는 만들필요가 없다.

- 학과, 학번, 이름
- 자료구조

  - 핵심 자료구조를 기술한다. 첫 단계의 과제는 매우 간단하지만 2차원 배열로 행렬을 표현했는지, 1차원 배열로 표현했는지, 또는 포인터로 표현한 뒤 malloc()으로 할당받았는지, 전역 변수로 할당했는지와 같은 다양한 옵션이 존재한다. 또한, read()나 fread() 대신 mmap()을 사용했을 때는 파일이 어떤 구조체에 매핑되는지를 구체적으로 기술해야 한다.

- 알고리즘
  - 핵심 문제와 이를 해결하기 위한 알고리즘을 기술해야 한다. 이 과제는 두 행렬의 곱을 구하는 것이므로 우선 수식으로 곱을 구하는 방식을 표현한다. 다음, 그 수식을 유사 코드(pseudo code)나 자연어로 표현한다. 프로그래밍 언어를 사용해도 된다.

- 스냅샷
  - 프로그램을 수행하는 것과 diff를 이용하여 결과를 검증하는 과정이 담겨있는 스냅샷. 1개의 그림으로 충분.

- 구현시 어려웠던 점과 해결 방안
  - 구현이 매우 쉬웠고 아무런 문제가 없었으면 '쉬웠음'이라고 적으면 된다. 그렇지 않은 경우 어떤 부분이 어떻게 어려웠고 어떻게 해결했는지를 기술한다.

- 성능 평가
  - 두 데이터 파일을 읽는 데 걸리는 총 시간
  - 행렬 곱을 구하는데 걸리는 총 시간
  - 결과 행렬을 파일에 저장하는 시간

- 결론
  - 이 과제를 통해 자신이 얻은 것을 기술한다.