2018년 1학기 운영체제 과제 #2

과제 개요

프로세스(Process)는 실행중인 프로그램의 인스턴스로, 운영체제에서 동시에 여러 개가 실행된다. 이를 통해 우리는 여러 개의 프로그램을 거의 동시에 사용할 수 있다. 스레드(Thread)는 한 프로세스 내에서 여러 작업을 동시에 처리하기 위한 기법으로, 빠른 처리를 위해 널리 사용되는 기법이다. 이 과제에서는 K-평균 군집화(K-means Clustering)를 여러 프로세스를 사용하는 멀티프로세싱(Multiprocessing) 기법과 여러 스레드를 사용하는 멀티스레딩(Multithreading) 기법을 통해 구현하여 성능을 비교해 볼 것이다.

1. 과제 내용

✓ 여러 개의 Process나 Thread를 사용하여 K-means Clustering을 구현하고, 그 성능을 비교한다.

✓ Makefile을 작성하여 소스를 빌드한다.

2 과제 목적

✓ Process와 Thread의 차이를 이해한다.

✓ 멀티프로세싱과 멀티스레딩 프로그램 구현의 기초를 습득한다.

✓ 병렬 프로그래밍을 하였을 때의 이점을 이해한다.

✓ 커널 빌드시 사용하였던 make 빌드 시스템에 대한 기초를 습득한다.

3 제출 기한

✓ 2018년 4월 16일 12:00 (정오) 까지

4 제출 방법

✓ 보고서 출력본: 제 4공학관 D814호 앞 과제 제출함

✓ 보고서 파일 (pdf 형식) 및 실습 과제 소스 파일: 과목 홈페이지 과제 제출 게시판

5 제한 사항

✓ 1인 1프로젝트이며 각자 환경에서 작업한다.

✓ 운영체제는 리눅스를 사용한다. 리눅스 종류와 버전은 제한이 없다.

- 채점 서버 OS: Ubuntu 16.04
- 채점 서버 컴파일러 : gcc 5.4.0 / clang 3.8.0

✓ 프로그래밍 언어는 C 또는 C++만을 사용한다.

✓ 프로세스 생성은 fork()만을, 스레드 생성은 pthread만을 직접적으로 사용해야 하며 다른 라이브러리는 사용하면 안된다.

✓ 그 외 구현에서는 C/C++ 표준 라이브러리만을 자유롭게 사용할 수 있다.

✓ 프로그램 구조

- 제출물의 압축을 해제하면 Makefile, noparallel.c, process.c, thread.c 가 존재하여야 한다. C++의 경우 확장자가 cpp가 될 수 있다.
- make시 noparallel, process, thread라는 이름의 실행 가능한 바이너리가 컴파일 되어야 한다. (./noparallel 으로 프로그램 실행 됨)
- 입력과 출력은 표준 입출력으로 한다.
- make clean시 모든 빌드 결과물을 삭제하여야 한다.

6 유의 사항

✓ 적절한 사유 없이 Delay 될 경우 절대 받지 않음

✓ 타 수강생의 보고서 및 과제를 카피 또는 수정하여 제출하였을 경우 모두 0점 처리

✓ 그림, 코드 조각을 포함한 모든 참고 자료는 반드시 출처를 명시

✓ 과제에 대한 문의 사항이 있을 경우 게시판으로 문의한다.

과제 세부 사항

보고서 과제

보고서는 **사전 조사 보고서**와 **실습 과제 수행 보고서**로 구성된다. 이번 과제에서는 보고서 작성이 매우 중요하다.

1 사전 조사 보고서

✓ Process와 Thread의 차이

✓ Process와 Thread의 리눅스에서의 구조 및 구현 등

✓ 멀티프로세싱과 멀티스레딩

✓ 프로세스 간 통신(IPC)

✓ 기타 필요하다고 생각되는 내용들

2 실습 과제 수행 보고서

✓ 작성한 프로그램의 동작 과정과 구현 방법

- 순서도나 블록 다이어그램 등 도식화 자료를 반드시 이용할 것

✓ 작성한 Makefile 에 대한 설명

✓ uname -a 실행 결과 화면과 개발 환경 명시 (CPU, 메모리 등)

✓ 과제 수행 중 발생한 애로사항 및 해결방법

✓ [중요] 결과 화면과 결과에 대한 분석 내용

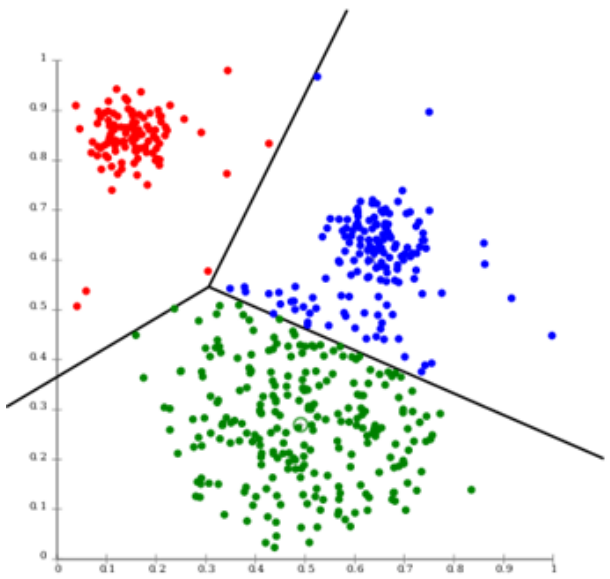
- Thread와 Process의 평균 생성 시간 분석
- 병렬 프로그래밍을 적용하지 않은 것, 멀티스레딩, 멀티프로세싱의 차이를 전체 수행 시간 관점에서 분석
- 인풋 데이터의 크기를 변경해가며 세 프로그램의 성능 비교 분석
- 그 외에 자유롭게 결과에 대해 비교 분석
- 그래프와 도표등을 반드시 활용하여 결과 분석을 성의있게 할 것

실습 과제

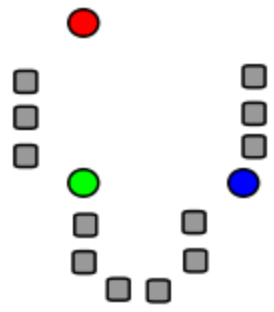
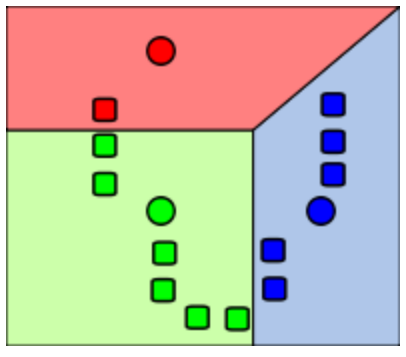
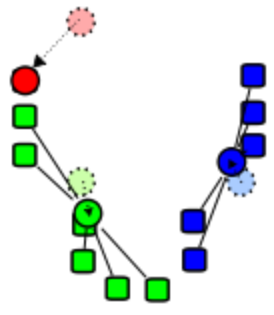
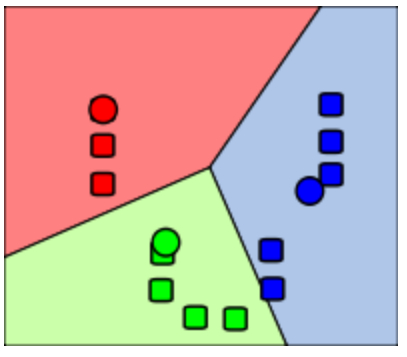
과제 목적

운영체제는 동시에 여러 Process를 실행할 수 있으며, 이를 이용하여 하나의 큰 작업을 여러 개로 나누어 여러 개의 Process에서 동시에 처리할 수 있다. 이러한 기법을 멀티프로세싱이라고 한다. 또한, Thread를 사용하면 하나의 프로세스 내에서도 이러한 병렬 처리가 가능하며, 이를 멀티스레딩이라고 한다. 멀티프로세싱과 멀티스레딩은 흔히 사용되는 병렬 프로그래밍 기법이다. 우리는 리눅스 환경에서 두 가지 병렬 프로그래밍 기법을 적용한 K-means Clustering을 구현하고, 그 성능 차이를 비교해 볼 것이다.

K-means Clustering



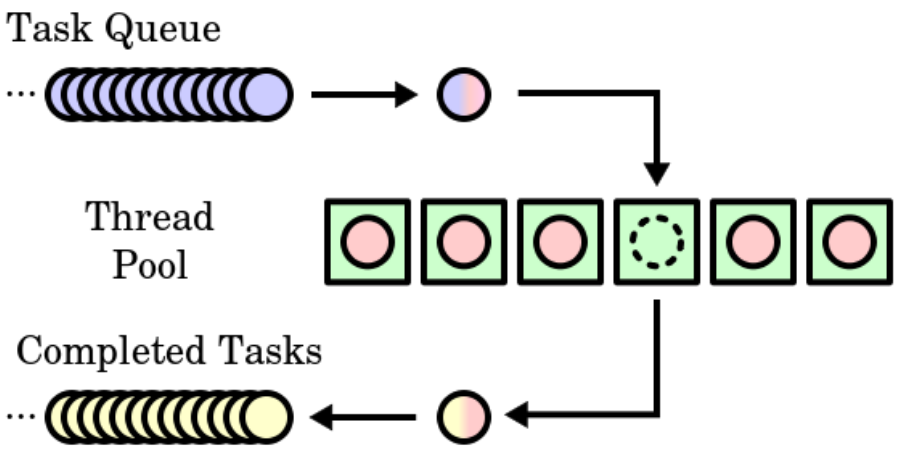
K-means Clustering은 주어진 데이터를 K개의 클러스터(Cluster)로 묶는 알고리즘으로, 각 클러스터와 거리 차이를 분산을 최소화하는 방식으로 동작한다. K-means Clustering의 표준 알고리즘의 실행 과정은 다음과 같다. 아래 설명에서 K의 값은 3이다.

			
1) 초기 K개의 '평균값'을 랜덤으로 선택한다. 단, 이번 과제에서는 채점을 위해 이 초기 '평균값'들이 입력값으로 주어진다.	2) 각 점들은 가장 가까이 있는 '평균값'을 기준으로 묶이게 되어 K개의 클러스터가 생성된다.	3) 각 클러스터들을 이루는 점들의 평균을 새로운 '평균값'으로 설정한다. 이 때, 새로운 '평균값'은 처음에 주어진 점들 중에 있지 않을 수도 있다.	4) 수렴할 때 까지 2), 3) 과정을 반복한다. 단, 이번 과제에서는 채점을 위해 이 과정을 몇 번 반복해야 하는지가 입력값으로 주어진다.

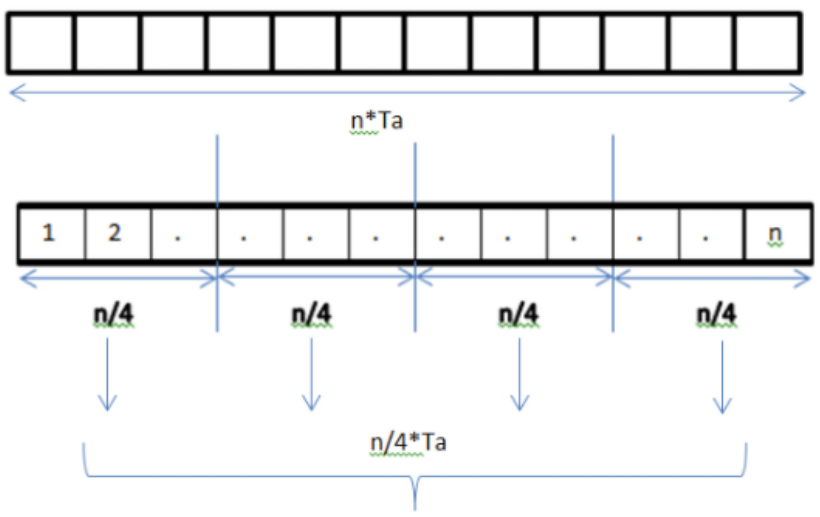
Multiprocessing / Multithreading

위 과정중 특히 2)에서 각 점들이 어느 클러스터에 속하는지를 판단하기 위해서는 꽤 많은 연산이 필요하다. 이 때, 병렬 프로그래밍을 이용하면 이 과정에서 이득을 볼 수 있다. 이 과제에서 병렬 프로그래밍을 어떻게 적용할 지는 과제 구현자의 몫이다. 이에 대해 몇 가지 힌트를 제공하자면 다음과 같은 방법들이 있다.

첫 번째로는 아래 그림과 같이 미리 원하는 수만큼 Thread나 Process를 생성해 놓고 작업들을 빈 Thread나 Process에 할당하는 방법이다. 이 과제에서는 각 점들이 어느 클러스터에 속하게 되는지를 판단하는 것이 작업이 될 수 있을 것이다. 이러한 기법을 Thread Pool(또는 Process Pool) 이라고 한다.



두 번째로는 입력되는 데이터들을 여러 구간으로 나누어 각 구간을 하나의 Thread나 Process가 담당하여 처리하는 방법이다. 이러한 기법을 데이터 병렬 처리(Data Parallelism)라 한다.



이외에도 다양한 방법이 있을 수 있으니 자유롭게 구현하고, 어떤 방식으로 구현했는지를 순서도나 그림과 함께 실습 과제 수행 보고서에 기재하면 된다. 각 테스트 케이스를 하나의 작업으로 생각하고 이를 병렬로 처리할 수도 있으나 출력은 반드시 순서에 맞게 이루어져야 한다.

과제 구현 방법

아래에 주어진 입력 조건과 출력 조건을 만족하는 K-means Clustering을 위의 설명을 바탕으로 세 가지 방법으로 구현한다. 첫 번째로는 병렬 프로그래밍을 사용하지 않고 구현한다. 이 파일의 이름은 noparallel.c로 한다. 두 번째로는 fork()를 사용하는 멀티프로세싱으로 구현한다. 이 파일의 이름은 process.c로 한다. 마지막으로는 pthread를 사용하는 멀티스레딩으로 구현한다. 이 파일의 이름은 thread.c로 한다. 단, C++의 경우에는 확장자가 cpp일 수 있다.

과제 구현은 크게 두가지 목표로 이루어진다. 하나는 K-means Clustering의 기능에 대한 정확한 구현이고, 나머지 하나는 성능 비교 분석을 위한 구현이다. Thread/Process의 평균 생성 시간 분석이나 각 프로그램의 전체 실행 시간 분석 등 실습 과제 수행 보고서를 위해 추가적으로 코드를 작성할 수 있으며, 이 부분은 제출 시에 주석처리를 한 후, 보고서에 이 부분에 대한 설명을 서술하면 된다.

이 프로그램들은 Makefile을 작성하여 make로 빌드를 하도록 한다. make 빌드 시스템으로 빌드를 하게 되면 각각 noparallel, process, thread 라는 이름의 실행 가능한 파일이 생성되어야 한다. 이 프로그램들은 입력과 출력을 표준입출력으로 한다.

입력

첫째 줄에 테스트 케이스의 개수 T가 주어진다. 그 다음 줄부터 각각의 테스트 케이스에 대한 내용이 주어진다. 각 테스트 케이스의 첫 번째줄은 알고리즘 반복 횟수 I, 두 번째 줄은 클러스터의 개수 K, 세 번째 줄은 전체 점의 개수 N이 주어진다. 그 다음 부터 N개의 줄에서 각 점의 좌표(x, y)가 주어진다. 이 때, N개의 점들 중 처음 K개의 점들이 각 클러스터(순서대로 0, 1, …, K-1 번째 클러스터)의 초기 평균값이다. 각 수들의 자료형과 범위는 다음과 같다.

- T, I, K, N은 정수(int)이며 $0 < T \leq 100$, $0 < I \leq 1000$, $0 < K \leq 10000$, $K < N \leq 100000$
- x, y는 실수(float)이며 절댓값의 크기가 10000보다 작음

출력

각 케이스 별로 순서대로 출력한다. 해당 케이스에서 첫 번째 줄에는 몇 번째 케이스인지(0, 1, …, T-1) 출력한다. 두 번째 줄에는 해당 케이스를 처리하는데 걸린 시간을 마이크로초 단위로 출력한다. 그 다음 줄부터는 매 줄마다 각 점이 몇 번째 클러스터(0, 1, …, K-1)에 속하는지를 출력한다.

예제 입력 (주어지는 input에는 주석이 없음)

```
3 // T
100 // Test Case #0의 I
1 // K
5 // N
10.0 0.1 // 0번째 클러스터의 초기 평균값
-7.5 3.333
10.3 5.8
-7.4 3.213
8.8 3.3
300 // Test Case #1
2
5
10.0 0.1  // 0번째 클러스터의 초기 평균값
-7.5 3.333 // 1번째 클러스터의 초기 평균 값
10.3 5.8
-7.4 3.213
8.8 3.3
300 // Test Case #2
5
5
10.0 0.1
-7.5 3.333
10.3 5.8
-7.4 3.213
8.8 3.3
```

예제 출력

Test Case #0

1234 microseconds

0

0

0

0

0

Test Case #1

1235 microseconds

0

1

0

1

0

Test Case #2

1236 microseconds

0

1

2

3

4

과제 제출 주의사항

✓ 실습을 위해 설정한 Linux 환경은 반드시 UTF-8을 사용한다. (한글을 사용할 경우 euc-kr을 사용하지 말고 locale을 ko.utf-8을 사용)

✓ 제출 소스에는 반드시 주석을 달 것, 주석의 내용이 불분명하거나 없을 경우 감점 처리

✓ 과제의 내용을 개인 블로그나 기타 웹 사이트 및 질의 응답 게시판에 포스팅 또는 첨부하는 것을 절대 금지 **(발각 시 F처리)**

✓ 아래 과제 제출 방법을 지키지 않았을 경우 감점 내지는 0점 처리 (특히 커널이 정상 동작하지 않아 시스템이 중지 될 경우 0점)

과제 제출 방법

보고서 pdf 파일(hw2_[학번].pdf)과 소스 압축 파일(hw2_[학번].tar)을 제출한다.

보고서 파일(hw2_[학번].pdf)

PDF 파일로 변환하여 제출한다. **파일의 이름은 반드시 hw2_[학번].pdf**로 한다. (Ex. hw2_2018123123.pdf)

소스 압축 파일(hw2_[학번].tar)

반드시 다음 명령어를 사용해서 압축 파일을 생성한다. **다른 형식의 압축 또는 이중 압축은 절대 허용하지 않으며 보고서 파일을 함께 압축해서는 안된다.**

```
# 과제 제출용 작업 디렉토리를 생성해서 작업한다. 본인이 작업하는 디렉토리가 hw2라고 가정하자.
# 본인의 학번이 2018123123 이라고 가정

$ cd ~/hw2
$ tar cvf hw2_2018123123.tar *
```

보고서 작성 요령

✓ 너무 많은 내용을 담으려 하지 말고 필요한 사항만 정확하게 작성

✓ 절대로 Web에 있는 내용이나 타인의 작성 내용을 그대로 복사해서는 안된다.

✓ Web이나 서적의 그림, 내용 등을 사용할 경우나 과제를 하면서 참고한 자료가 있다면 반드시 출처를 명시한다.

✓ 실습 보고서의 경우 일반적인 보고서 형태를 따른다.
정해진 양식은 없으나 필요한 부분은 섹션 별로 나누어 작성 하고 다음의 내용을 포함하는 것이 좋다.
수행과정/세부 내용/문제점 및 애로사항/해결방법/결과 분석 및 토론/참고 문헌/그 외 기타

과제 감점 요인

✓ 컴파일 불가능할 시 프로그램 점수 0점 부여

✓ 반드시 리눅스에서 작업할 것

✓ 그 외에도 제한 사항, 유의 사항, 보고서 작성 요령을 잘 지키지 않은 경우 감점

참고 사이트

K-means Clustering https://en.wikipedia.org/wiki/K-means_clustering

Thread Pool https://en.wikipedia.org/wiki/Thread_pool

Data parallelism https://en.wikipedia.org/wiki/Data_parallelism

Introduction to Parallel Computing https://computing.llnl.gov/tutorials/parallel_comp/

POSIX Threads Programming https://computing.llnl.gov/tutorials/pthreads/

Multi-Process Programming in C

http://home.deib.polimi.it/fornacia/lib/exe/fetch.php?media=teaching:aos:2016:aos201617_multiprocess_programming_updated20161223.pdf