운영체제 2차과제 보고서

학 과: 컴퓨터학과

학 번: 2017320215

이 름: 임준상

제출일: 2019.06.06

Free Day 사용 일수: 2

I**.** 과제 개요 및 프로세스와 스케줄러의 개념

**과제 개요:**

본 과제는 프로세스의 CPU 점유 시간을 측정하여 기록한 정보를 통해 리눅스의 프로세스 및 프로세스 스케줄러를 이해한다.

**프로세스와 스케줄러의 개념:**

프로세스:

프로세스는 자원을 분배하는 단위(unit)며 실행 중인 프로그램의 추상화, 프로그램의 실행 과정이다. 운영체제는 CPU를 프로세스에 할당해준다.

스케줄러:

Multiprogramming 시스템에서 프로세스의 개수는 항상 CPU 자원보다 많다. 프로세스는 CPU의 자원을 사용하기 위해 자원 경쟁은 피해갈 수 없는 상황이다. 스케줄러는 CPU를 분배하는 것이다. Queue에서 알고리즘에 근거로 프로세스를 CPU에 할당해준다. 프로세스를 CPU에 할당해주어야 되는데 “어떤 프호세스를 CPU에 할당해주는 것인가?”에 대한 것은 프로세스 스케줄링이다.

II.CPU burst에 대한 그래프 및 결과분석

**그래프:**

**一張含有 文字 的圖片

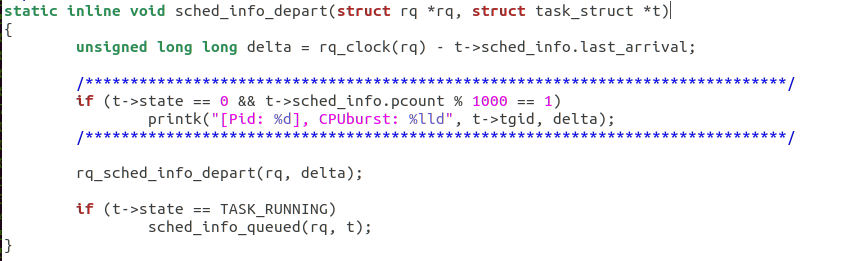
描述是以高可信度產生**

**분석:**

이번 과제는 Firefox에서 YouTube로 음악을 30분 재생하는 프로세스를 했다. top를 통해 Firefox의 PID는 1853인 것을 알 수 있다. 이 그림에서 대부분의 CPU burst duration은 1ms 안에 끝나고 4ms를 넘지 않다. 가장 큰 burst duration은 44.558347ms이다. CPU-bound 프로세스는 긴 CPU burst이고 I/O-bound 프로세스는 짧은 CPU burst이며 CPU-bound프로세스는 CPU를 많이 쓰고 I/O-bound 프로세스는 CPU를 짧게 쓴다. 이 그림을 보면 짧은 burst duration이 많고 긴 burst duration이 작다. 그러므로 I/O-bound 프로세스가 CPU-bound 프로세스보다 훨씬 많은 것이 나타난다.

III. **작성한 모든 소스코드에 대한 설명**

소스코드:



소스코드는 “/\*\*\*/” 주석으로 표시하였다.

설명:

task\_struct 구조체를 찾아서 t->state의 값은 0일 때 runnable 상태인 것을 알 수 있다.



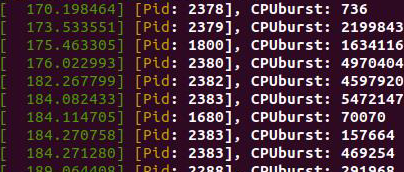
t->sched\_info.pcount의 주석은 /\* # of times we have run on this CPU: \*/이며 이 변수를 통해 task(process)가 이 CPU에서 실행된 횟수를 표현할 수 있다.



CPU burst 값이 1,000회 바꿀 때 1회를 기록해야 하기 때문에 (pcount % 1000)를 하면 천번이 될 때 다음 함수를 실행한다. printk함수 안에 있는 tgid는 process의 식별자이고 delta는 CPU burst이다.

이 코드의 전체 흐름은 프로세스가 CPU 점유를 마쳤을 때 함수가 호출되어 프로세스의 상태는 runnable이고 CPU burst값이 1,000회 바꾸면 프로세스의 식별자와 이 프로세스의 CPU burst 정보를 출력할 것이다.

이렇게 수정한 후 커널 컴파일해서 dmesg 명령어를 입력하면 나온 부분 결과는 아래에 있다. 전체 output는 log.txt 파일에 출력했다.



IV.문제점과 해결방법

1. CPU burst를 측정하는 방법을 몰라서 첫번째 과제처럼 새로운 syscall 함수를 작성했는데 필요한 함수(rq\_clock 등)를 쓰지 못했다.

2. 과제 요구는 CPU burst가 1,000회 바꿀 때 1 번 기록한다. 어떻게 1,000회를 알 수 있는지 몰랐다.

3. excel로 그래프를 작성했는데 지난 번 excel를 쓴지 7년이 되었다. 어떻게 그래프를 만드는지 몰랐다.

4. 과제를 쓸 때 나오는 결과가 틀리는지 맞는지 잘 확실하지 못했다.

해결방법:

1. stats.h에서 코딩하면 필요한 함수가 다 있다.

2. 소스코드를 분석해서 sched\_info.pcount는 CPU에서 실행된 횟수를 기록한다.

3. excel를 배우면서 그래프를 작성했다.

4. 정답을 확실하지 못하지만 아무래도 포기하는 것보다 좋다.