**OS Project Report 2: Advanced Scheduler**

20205031 김소정, 20205135 이수린, Team F(team 5)

1. **Problem definition**

Priority donation 방식은 스레드의 실행을 주어진 priority로만 판단하여 수행한다. 따라서 priority가 낮은 스레드는 CPU할당을 받지 못할 가능성이 크다. 스레드는 I/O bounded(I/O 처리 ↑, CPU 사용 ↓)와 CPU bounded(I/O 요청↓ CPU 사용 ↑)로 분류되며 스케줄러가 이 스레드들의 처리 순서를 결정해서 스레드들의 평균 응답 시간을 줄이고 CPU를 효율적으로 사용해야 한다. 이에 스레드의 priority를 실시간으로 수정하는 multi-level feedback scheduler를 구현하여 스레드의 수행 시간을 보정해주고자 한다.

1. **Policy and algorithm design**

* Policy
  + BSD 스케줄러 방식을 차용하여 multi-level feedback scheduler를 구현한다. BSD 스케줄러 방식은 Ready queue가 여러 개 있고 각 큐는 서로 다른 우선순위를 가진다. 항상 가장 높은 우선순위의 큐에 있는 프로세스부터 실행되며 큐에 여러 스레드가 있으면 round robin방식으로 실행한다. 스레드의 우선순위는 처음 스레드의 초기화 때 정해지고, 매 4번째 tick마다 새로 계산된다.
    - round robin: 정해진 시간만큼 스레드를 실행하고 큐의 맨 뒤로 보내 모든 스레드를 일정 시간만큼 실행
  + priority = PRI\_MAX - (recent\_cpu / 4) - (nice \* 2)
    - Recent\_cpu: 얼마나 최근에 CPU를 사용했는지를 나타내며 최근에 CPU를 사용했을수록 우선순위가 낮아지게 한다. 효율적으로 CPU점유율을 나타내기 위해 Exponentially Weighted Moving Average 방식을 사용한다. 처음에 0으로 초기화되거나 부모 값을 물려받는다. 1초마다 모든 스레드의 값을 업데이트하며 1 tick마다 idle스레드가 아닌 running 스레드의 recent\_cpu 값에 1을 더해준다.
      * recent\_cpu = (2\*load\_avg)/(2\*load\_avg + 1) \* recent\_cpu + nice
    - Load\_avg: 지난 1분동안 수행 가능한 평균 스레드 개수를 나타내며 system-wide값으로 부팅될 때 0으로 초기화되고 1초마다 업데이트된다.
      * load\_avg = (59/60)\*load\_avg + (1/60)\*ready\_threads
      * ready\_threads: running or ready-to-run인 스레드 개수
    - Nice: 얼마나 다른 스레드에게 잘 양보하는지를 나타낸다. -20 ≤ nice < 0일 때는 다른 스레드에게서 CPU 사용 시간을 뺏어오므로 우선순위가 높아진다. Nice = 0일 때는 우선순위에 영향을 주지 않는다. 0 < nice ≤ 20일 때는 다른 스레드에게 CPU 사용 시간을 양보하므로 우선순위가 낮아진다. Nice는 0으로 초기화되거나 부모의 값을 물려받는다.
  + Fixed-point: 실수를 표현하는 방식 중 Floating point 방식은 표현할 수 있는 범위가 크지만 계산 시간이 길고 복잡해서 kernel이 느려지는 단점이 있다. 따라서 핀토스에서는 소수점이 고정된 fixed point 방식을 사용한다.
    - 1 sign bit / 17 정수 bit / 14 소수 bit
* Algorithm design
  + Step (1): argument로 -mlfqs를 받으면 multi-level feedback 스케줄러를 활성화한다. BSD 스케줄러가 활성화되면, 스레드는 직접 우선순위를 제어할 수 없다.
  + Step (2): 고정 소수점 연산을 구현한다.
  + Step (3): nice, load\_avg, recent\_cpu 값을 계산하고 이를 이용하여 priority를 계산하는 함수를 만든다.
  + Step (4): 기준 시간별로 load\_avg, recent\_cpu를 업데이트해준다. Load\_avg와 recent\_cpu는 1초에 한 번, priority는 4tick에 한 번 업데이트하면 된다.

1. **Mechanism**

* ADD
  + /home/name/pintos/src/threads/fixed-point.h
* Modified file names
  + /home/name/pintos/src/threads/synch.c
  + /home/name/pintos/src/devices/timer.c
  + /home/name/pintos/src/threads/thread.h
  + /home/name/pintos/src/threads/thread.c
* CODES
  + Fixed-point.h: fixed point 연산을 구현했다.
  + Synch.c: lock\_acquire(), lock\_release()

A screen shot of a computer code

Description automatically generatedA computer screen shot of code

Description automatically generated

Mlqfs일 때는 동작하지 않도록 수정한다.

* + Timer.c: timer\_interrupt()

A computer screen shot of a program code

Description automatically generated

Mlfqs일 때 1초(ticks % TIMER\_FREQ == 0)마다 load\_avg와 모든 스레드의 recent\_cpu를 업데이트하고, 4tick마다 모든 스레드의 mlfqs를 업데이트한다.

* + Thread.h: thread 구조체에 nice와 recent\_cpu값을 추가해준다. mlqfs 함수 선언들도 추가하였다.
  + Thread.c: load\_avg을 선언해준다.



* + Thread.c: thread\_set\_priority()

A screen shot of a computer

Description automatically generated

Mlfqs일 때는 스레드는 직접 우선순위를 제어할 수 없으므로 바로 return한다.

* + Thread.c: thread\_set\_nice(), thread\_get\_nice()

A screen shot of a computer program

Description automatically generated

thread\_get\_nice(): 현재 nice값을 return한다.

thread\_set\_nice(): 주어진 nice값으로 업데이트한다. 해당 스레드의 우선순위 값을 새로 계산해서 업데이트한다. Idle 스레드일 때는 nice를 바꾸지 않는다.

* + Thread.c: thread\_get\_load\_avg(), thread\_set\_load\_avg()

A computer screen with text and images

Description automatically generated

thread\_get\_load\_avg(): 현재 load\_avg값에 100을 곱해 return한다.

thread\_set\_load\_avg(): load\_avg 값을 새로 계산해서 업데이트한다.

* + Thread.c: thread\_get\_recent\_cpu(), thread\_set\_recent\_cpu(), thread\_renew\_recent\_cpus(), thread\_increment\_recent\_cpu()

A screen shot of a computer program

Description automatically generated

thread\_get\_recent\_cpu(): 현재 recent\_cpu에 100을 곱해 return한다.

thread\_set\_recent\_cpu(): recent\_cpu를 새로 계산해서 업데이트한다. Idle 스레드는 업데이트하지 않는다.

thread\_increment\_recent\_cpu(): 실행 중인 스레드의 recent\_cpu 값을 1 증가시킨다. Idle 스레드는 업데이트하지 않는다.

thread\_renew\_recent\_cpus(): 모든 스레드의 recent\_cpu를 계산해서 업데이트한다.

* + Thread.c: thread\_set\_priority\_mlfqs(), thread\_renew\_priorities\_mlfqs()

A screen shot of a computer program

Description automatically generated

Thread\_set\_priority\_mlfqs: idle스레드가 아닐 때 스레드의 우선순위를 계산해서 업데이트한다. 최소, 최대 값을 넘어가면 조정한다.

Thread\_renew\_priorities\_mlqfs: 모든 스레드의 우선순위를 계산해서 업데이트한다.