**OS 중간고사 정리**

시험 출제

Chapter5. Process scheduling

CPU burst: CPU 명령어 연속 수행 구간

I/O burst: I/O의 완료를 기다리며 블록되는 구간.

CPU-bound process: CPU burst가 긴 프로세스. 계산이 중요 ex) 과학계산, 일기예보

I/O-bound process: I/O burst가 CPU burst보다 긴 프로세스. 사용자에게 응답하는 것이 중요.

대부분 프로그램이 이것!

Preemptive scheduling: 선점적, 우선순위가 높은 게 생긴다면 수행하던 프로세스를 중단하고 우선순위가 높은 것을 수행한다. 컴퓨터에선 이게 일반적이다.

Non-preemptive scheduling: 비선점적, 우선순위가 높은 게 생기더라도 수행하던 프로세스를 종료할 때까지 계속 수행한다. 일상에서는 이게 대부분이다.

Ex) 중요한 손님이 오더라도 먼저 먹던 손님이 다 먹는다!

**스케쥴링 시 고려할 척도**

**↑** CPU utilization: CPU가 일하는 시간.

**↑** Throughput: 단위시간 당 처리한 양. ex) 정해진 시간에 몇 개의 프로세스를 실행했냐!

**↓** Turnaround time: 프로세스의 시작부터 끝까지의 시간

**↓** Waiting time: 다른 프로세스 때문에 ready queue에서 기다리는 시간

**↓** Response time: 응답 시간

스케쥴링의 목표: Fairness (공평하게) & Balance (하나만 바쁘지 않게)

Starvation: ready queue에서 스케쥴링 되지 못하고 계속 순서가 밀리는 상태.

Aging을 이용하여 해결!

\*Aging: 기다리는 시간만큼 중요도를 증가시킨다!

**FCFS / FIFO : First-Come, First-Served**

프로세스가 도착한 순서대로 실행하는 것.

장점: Fair

단점: waiting time이 길다. (수행시간이 긴 프로세스 뒤에 있으면 엄청 기다려야 한다.)

**SJF : Shortest Job First**

수행시간이 짧은 프로세스부터 실행하는 것.

장점: Waiting time을 줄이는 데 최적의 방법.

단점: starvation을 유발한다. CPU burst(수행시간)을 알기 힘들다.

Non-preemptive: 수행시간이 짧은 프로세스가 도착하더라도 계속 이어서 수행.

Preemptive: SRTF(Shortest-Remaining-Time-First), 남은 수행시간을 기준으로 수행시간이 더 짧은 프로세스가 도착하면 수행하던 것을 중단하고 수행한다.

**Priority Scheduling**

정해진 중요도에 따라 우선 순위 부여. (ex OS 프로세스 > 사용자 프로세스)

단점: starvation을 유발한다.

우선 순위가 같은 레벨에서는 RR(Round Robin) 수행.

**RR (Round Robin)**

Time quantum을 주고 순차적으로 돌아가면서 조금씩 수행한다.

FCFS의 preemptive 버전으로 볼 수 있다.

Time quantum이 작을수록 context switch가 많아진다. CPU burst의 80%를 커버할 수 있을 정도가 적당하다. (약 60ms). 대부분은 I/O 요청을 기다리면서 자발적으로 종료하는 경우가 많다.

\* 현재 윈도우 유닉스 등은 priority scheduling + RR

**Multilevel Feedback Queue**

처음의 time quantum을 다 채우지 않고 수행 완료일 경우 반복한다.

그러나 다 채우는 경우 time quantum을 증가시킨 레벨로 이동시킨다.

다음 레벨에서도 다 채우는 경우, FCFS를 사용한다

⇒ 다 채우는 경우는 CPU-bound process일 가능성이 높기 때문에 FCFS가 유리하다.