11.CPU_Scheduling

11.CPU_Scheduling

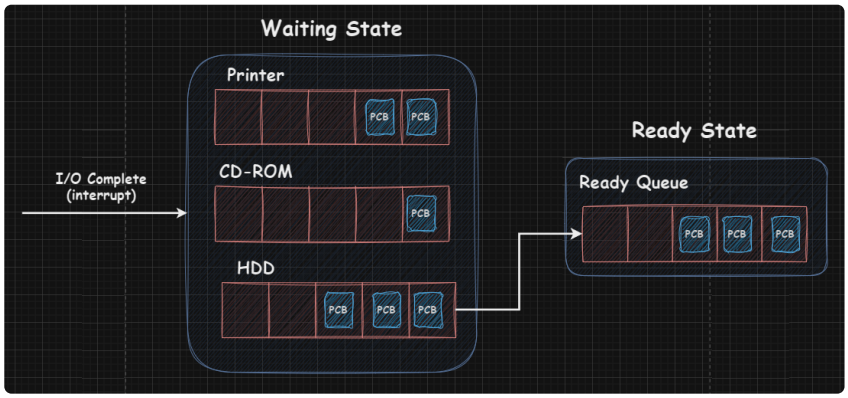
- 운영체제가 프로세스에 CPU 자원을 배분하는 방법

11.1 Introduce

11.1.1. 프로세스 우선순위

- 우선순위(priority)가 높은 프로세스는 대표적으로 입출력 작업이 많은 프로세스
- 프로세스는 대부분 실행/대기 상태를 반복한다.
- I/O bound process
  - 입출력 집중 프로세스
  - 입출력 작업이 많은 프로세스
  - 비디오 재생, 디스크 백업 등
  - 실행보다 주로 대기상태에 머무름
- CPU bound process
  - CPU 집중 프로세스
  - 수학 연산, 컴파일, 그래픽 처리 등
  - 대기보다 주로 실행상태에 머무름
- CPU를 이용하는 작업은 CPU burst, 입출력장치를 기다리는 작업을 I/O burst라 한다.

11.1.2. Scheduling queue



- PCB에 우선순위가 저장되어 있지만, 매번 PCB를 확인하는 것은 비효율적이다.
- 때문에 메모리 적재, 입출력 장치 사용, CPU 사용 등을 위해 프로세스를 줄세우며, 이를 스케쥴링 큐로 구현한다.
- 큐는 대표적으로 ready queue, waiting queue가 있으며, 각각 CPU, I/O를 이용하려는 프로세스들이 들어간다.
- 입출력이 완료되어 완료 인터럽트가 발생하면 운영체제는 대기 큐에서 작업이 완료된 PCB를 찾아 PCB를 준비 상태로 변경한 뒤 대기 큐에서 제거, 준비 큐로 이동시킨다.

11.1.3. (non) Preemptive

- Preemptive scheduling:
  - 선점형 스케줄링
  - 하나의 프로세스가 CPU를 사용중이라도 OS는 CPU를 강제로 빼앗아 다른 프로세스에 CPU를 넘겨주는 방식
  - 하나의 프로세스가 자원을 독점할 수 없다.
  - 대표적으로는 timer interrupt에 의한 context switching이 있으며, 이것에 의한 오버헤드가 존재한다.
- Non-preemptive scheduling:
  - 프로세스가 종료되거나 스스로 대기상태에 들어가기 전까지 다른 프로세스가 끼어들 수 없는 방식
  - 하나의 프로세스가 자원을 독점할 수 있다.
  - 문맥 교환에서 발생하는 오버헤드는 적지만, 자원의 고른 분배가 어렵다.

11.2. CPU scheduling algorithm

11.2.1. FCFS

- First Come First Served Scheduling
- 선입 선처리 스케줄링
- 단순히 준비 큐에 삽입된 순서대로 프로세스들을 처리하는 비선점형 스케줄링 방식
- 먼저 요청한 프로세스부터 CPU를 할당한다.
- 프로세스들이 기다리는 시간이 매우 길어질 수 있다.
- 2ms가 필요한 프로세스 앞에 총 22ms가 필요한 프로세스들(즉, 2ms <- 5ms <- 17ms <- serve 형태)이 있을 수 있으며, 이런 현상을 convoy effect(호위 효과)라 한다.
- 위의 경우 평균 대기 시간은 13ms((22 + 17 + 0) / 3)이 된다.

11.2.2. SJF

- Shortest Job First Scheduling
- 최단 작업 우선 스케줄링
- CPU 사용 시간이 가장 짧은 프로세스를 먼저 실행하는 스케줄링
- 17ms <- 5ms <- 2ms <- serve 순으로 수행되며 총 대기시간은 7ms, 평균 대기시간은 3ms가 안 된다.

11.2.3. RR

- Round robin scheduling
- 선입 선처리 스케줄링에서 타임 슬라이스[1]라는 개념이 더해진 방식
- 즉, 프로세스는 정해진 시간 동안만 CPU를 이용하며, 이후 다른 프로세스에 CPU를 넘겨주는 선점형 스케줄링 방식으로 돌아간다.
- 타임 슬라이스 시간에 따라 호위 효과가 생길 수도 있고, 너무 짧으면 문맥 교환에 발생하는 오버헤드가 커질 수 있다.

11.2.4. SRT

- Shortest Remaining Time scheduling
- 최소 잔여 시간 우선 스케줄링
- SJF와 RR을 합친 스케줄링 방식
- 프로세스들은 정해진 타임 슬라이스만큼 CPU를 사용하고, 다음으로 남은 작업 시간이 가장 적은 프로세스가 할당된다.

11.2.5. Priority

- 우선순위 스케줄링.
- 프로세스에 우선순위를 부여하고, 가장 높은 우선순위 프로세스부터 실행한다.
- 우선순위가 같으면 선입 선처리로 스케줄링 된다.
- SJF, SRT은 넓은 의미에서 우선순위 스케줄링이라 볼 수 있다.
- 다만, 이런 우선순위 스케줄링은 Starvation 현상이 발생할 수 있는데, 우선순위가 높은 프로세스가 계속 먼저 실행된다면 우선순위가 낮은 프로세스는 계속 뒤로 밀려 실행되지 않을 수 있다.
- 이 현상을 방지하기 위해 Aging을 사용한다. 오래 대기할 수록 우선순위를 높이는 방식.

11.2.6. Multilevel queue

- 우선순위 스케줄링의 발전된 형태
- 다단계 큐 스케줄링
- 우선순위 별로 준비 큐를 여러 개 사용하는 방식
- 우선순위가 가장 높은 큐에 있는 프로세스를 먼저 처리하고, 해당 큐가 비었다면 다음 우선순위 큐에 있는 프로세스를 처리한다.
- 우선순위가 높은 순으로, 입출력, 백그라운드, 상호작용이 잦은 프로세스, CPU bound 프로세스가 각각의 큐에 들어갈 수 있으며, 유형별로 우선순위를 구분해 실행하는 것이 편해진다.
- 각각의 큐에 타임 슬라이스나 알고리즘을 다르게 적용할 수 있다.

11.2.7. Multilevel feedback queue

- Multilevel queue는 프로세스들이 큐 사이를 이동할 수 없어, 기아현상이 발생할 수 있다.
- Multilevel feedback queue는 큐 간 프로세스 이동이 가능하며 시나리오는 다음과 같다.
  1. 새로 준비 상태가 된 프로세스를 가장 높은 우선순위 큐에 삽입하고 일정 시간(time slice) 실행된다.
  2. 실행 시간이 남았다면 다음 우선순위 큐에 삽입되어 실행된다.
  3. 또 다시 시간이 남았다면 다음 우선 순위 큐에 삽입된다.
- 이처럼 실행시간이 긴 프로세스는 우선순위가 계속 낮아진다. 즉, CPU bound process는 우선순위가 자연스레 낮아지고 I/O bound process는 높은 우선순위에서 작업을 마칠 수 있다.
- 이 방식 또한 기아현상을 방지하기 위해 Aging 기법을 이용할 수 있다.

---

1. 타임 슬라이스는 각 프로세스가 CPU를 사용할 수 있는 정해진 시간을 말한다.↩