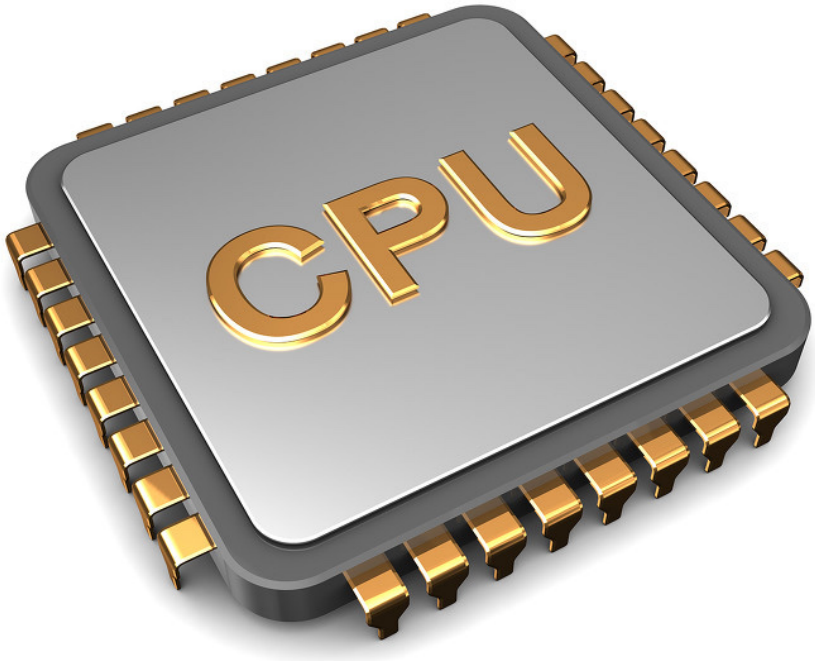
CPU 스케줄링



CPU 스케줄링은 여러 개의 프로세스가 동시에 실행될 때 CPU가 어떤 프로세스에게 우선순위를 부여하고 어떤 방식으로 처리할지를 결정하는 작업입니다.

CPU 스케줄링의 종류

선점형 스케줄링

선점형 스케줄링은 실행 중인 프로세스가 다른 프로세스가 실행되어야 할 시점에 강제로 CPU를 빼앗겨 다른 프로세스가 실행될 수 있는 방식입니다. 이러한 방식으로 다른 프로세스에게 CPU를 양보하는 것을 컨텍스트 스위칭(Context Switching)이라고 합니다. 선점형 스케줄링은 우선순위가 높은 프로세스가 존재할 때 중요하며, 대표적으로 **라운드 로빈, 다단계 큐 스케줄링** 등이 있습니다.

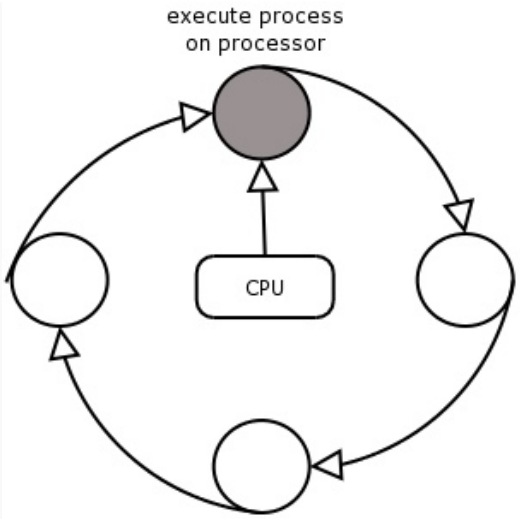
🚀 장점

- 응답 시간이 더 짧아지므로, 시스템의 대화형 성능이 향상됩니다.
- 우선순위가 높은 작업에 더 많은 CPU 시간이 할당되므로, 시스템 전체적인 처리율이 향상됩니다.
- 높은 우선순위를 가진 작업이 바로 처리되므로, 실시간 시스템에서 중요한 이벤트 처리가 가능해집니다.

💔 단점

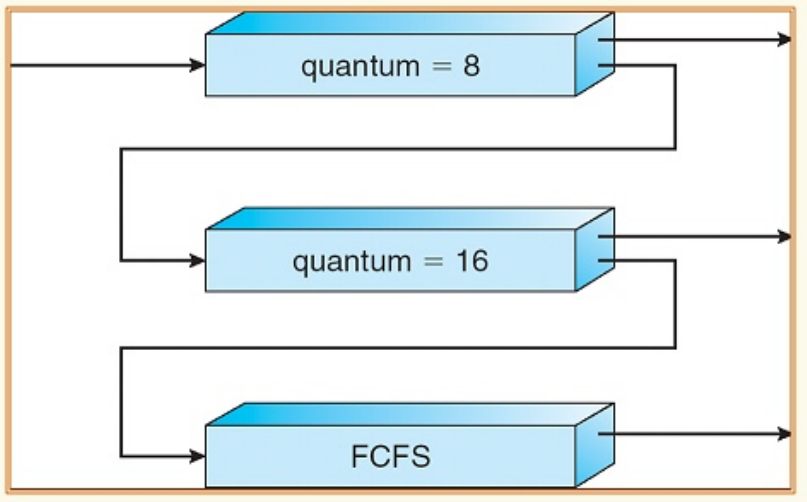
- CPU 자원을 더 많이 사용하므로, 오버헤드가 증가합니다.
- 높은 우선순위 작업이 지나치게 많아지면, 낮은 우선순위 작업이 무시될 수 있습니다.
- CPU 시간을 할당받아 실행 중인 작업이 갑작스럽게 중단될 수 있습니다.

라운드 로빈 (Round Robin)



프로세스들 사이에 우선순위를 두지 않고, 순서대로 시간단위로 CPU를 할당하는 방식입니다. 보통 시간 단위는 10ms ~ 100ms 정도이고 시간 단위동안 수행한 프로세스는 준비 큐의 끝으로 밀려나게 됩니다. 문맥 전환의 오버헤드가 큰 반면, 응답 시간이 짧아지는 장점이 있어 실시간 시스템에 유리하고, 할당되는 시간이 클 경우 비선점 FIFO기법과 같아지게 됩니다.

다단계 큐 (Multi-level Queue)



다단계 큐 스케줄링은 프로세스를 여러 개의 큐로 구분하고 각 큐마다 다른 스케줄링 알고리즘을 적용하여 우선순위를 부여해 실행하는 스케줄링 알고리즘입니다.

보통 우선순위가 높은 프로세스가 먼저 실행되어야 할 때 사용됩니다. 큐는 보통 세 가지 종류로 나뉘며, 각각의 큐마다 우선순위에 따라 다른 스케줄링 알고리즘을 적용합니다.

1. 최상위 큐 (Top-level Queue) : 우선순위가 가장 높은 프로세스들을 위한 큐입니다. 일반적으로 Round Robin 스케줄링을 사용하며, 일정 시간이 지나면 하위 큐로 이동합니다.

2. 중간 큐 (Mid-level Queue) : 중간 우선순위의 프로세스들을 위한 큐입니다. 우선순위가 높은 최상위 큐에서 이동한 프로세스나 우선순위가 낮아져 내려온 프로세스들이 위치합니다. 보통 Shortest Job First (SJF) 또는 Round Robin 스케줄링을 사용합니다.
3. 최하위 큐 (Lowest-level Queue) : 우선순위가 가장 낮은 프로세스들을 위한 큐입니다. 일반적으로 First-In-First-Out (FIFO) 스케줄링을 사용합니다.

비선점형 스케줄링

반면, 비선점형 스케줄링은 CPU가 한 번 할당되면 해당 프로세스가 종료되거나 대기 상태가 될 때까지 CPU를 계속 사용하게 됩니다. 따라서 해당 프로세스가 종료되거나 I/O 등 다른 이벤트가 발생할 때까지 다른 프로세스가 CPU를 사용할 수 없습니다. 대표적인 예로는 **First-Come**, **First-Served**, **Shortest-Job-First** 등이 있습니다.

🚀 장점

- 오버헤드가 적으므로, 시스템 전체적인 처리율이 향상됩니다.
- 짧은 작업이 긴 작업에 의해 블록되지 않으므로, 기아 상태(starvation)가 발생할 가능성이 적습니다.

💔 단점

- 응답 시간이 더 길어지므로, 시스템의 대화형 성능이 저하됩니다.
- 우선순위가 높은 작업이 먼저 처리되지 않을 수 있으므로, 실시간 시스템에서 중요한 이벤트 처리가 어려워집니다.
- 긴 작업이 지속적으로 실행되면, CPU 사용률이 낮아져 CPU 자원의 낭비가 발생할 수 있습니다.

First-Come, First-Served

프로세스가 도착한 순서대로 CPU를 할당하는 비선점형 스케줄링 방식입니다. CPU가 할당되면 해당 프로세스는 CPU를 반환할 때까지 실행됩니다. FCFS 스케줄링은 구현이 간단하며, CPU 버스트 시간이 동일한 경우 평균 대기 시간을 최소화할 수 있습니다. 그러나 CPU 버스트 시간이 큰 프로세스가 먼저 도착하면 평균 대기 시간이 길어질 수 있습니다.

Shortest-Job-First

프로세스의 예상 CPU 버스트 시간에 따라 우선순위를 결정하고, 가장 짧은 CPU 버스트 시간을 가진 프로세스에게 먼저 CPU를 할당하는 비선점형 스케줄링 방식입니다. 이 방식은 평균 대기 시간을 최소화하는 최적의 스케줄링 알고리즘 중 하나로 알려져 있습니다. 하지만, 프로세스의 CPU 버스트 시간을 예측하는 것이 어렵기 때문에, 실제로는 사용이 어려울 수 있습니다.

CPU 스케줄링 평가 기준 (CPU Scheduling Criteria)

1. CPU 이용률(CPU utilization)
    - 시간당 CPU를 사용한 시간의 비율
    - 프로세서를 실행상태로 항상 유지하려고 해야 한다.
2. 처리율(Throughput)
    - 시간당 처리한 작업의 비율

- 단위 시간당 완료되는 작업 수가 많도록 해야 한다.

3. 반환시간(Turnaround Time)
   - 프로세스가 생성된 후 종료되어 사용하던 자원을 모두 반환하는 데까지 걸리는 시간
   - 작업이 준비 큐(ready queue)에서 기다린 시간부터 CPU에서 실행된 시간, I/O 작업 시간의 합이다.
4. 대기시간(Waiting Time)
   - 대기열에 들어와 CPU를 할당받기 까지 기다린 시간
   - 준비 큐에서 기다린 시간의 총합
5. 반응시간(Response Time)
   - 대기열에서 처음으로 CPU를 얻을 때까지 걸린 시간
   - 대기시간과 비슷하지만 다른 점은, 대기시간은 준비 큐에서 기다린 모든 시간을 합친 것이지만 반응 시간은 CPU를 할당받은 최초의 순간까지 기다린 시간 한번 만을 측정한다.