운 영 체 제

2장 프로세스 관리(3)

학습 내용

- 2장 프로세스 관리
  - 시스템 소프트웨어의 종류와 기능
    - 링커와 로더
  - 프로세스 개요
    - 프로세스 제어 블록(PCB)
    - 프로세스의 상태
    - 프로세스 상태의 전이
  - 프로세스 스케줄링 개요
  - 프로세스 스케줄링
    - 비선점 스케줄링
      - FCFS(First Come First Service)
      - SJF(Shortest Job First)
      - HRN (Highest Response-ratio Next)
      - 기한부(Deadline)
      - 우선순위(Priority)
    - 선점 스케줄링
      - RR, SRT, 선점우선순위, 다단계큐, 다단계피드백 큐

2장 프로세스 관리(3)

시스템 소프트웨어의 종류와 기능
프로세스 개요
프로세스 스케줄링 개요
프로세스 스케줄링
  -비선점 스케줄링 기법
  -선점 스케줄링 기법

학습 내용

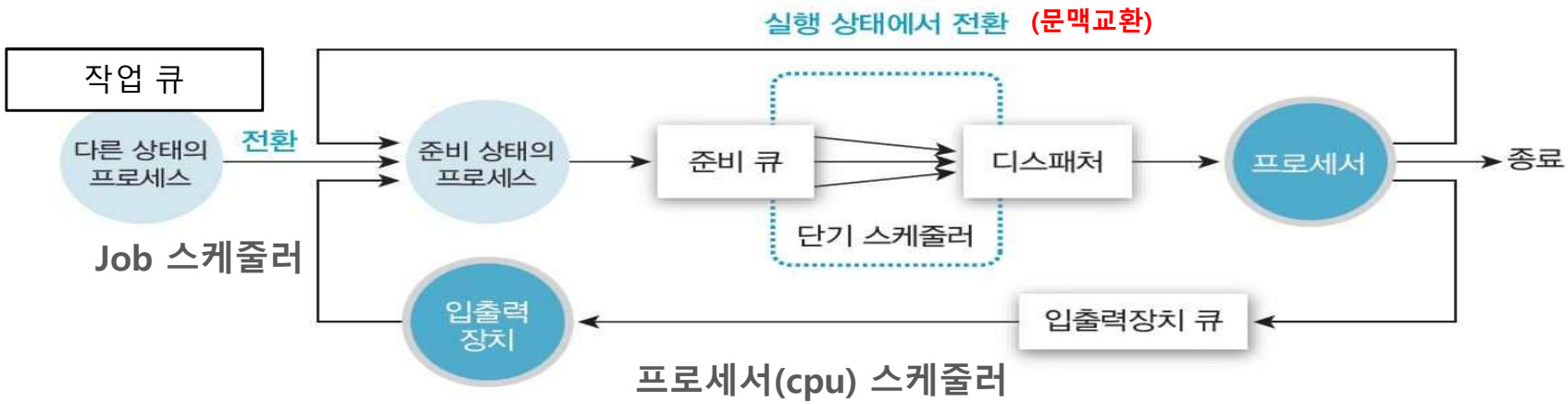
- 프로세스 스케줄링
  - 비선점 스케줄링
    - FCFS, SJF, HRN, 비선점 우선순위, 기한부
  - 선점 스케줄링
    - 선점 스케줄링 개념
    - RR(Round Robin) ⇔ FCFS(비선점) + 선점 형태
    - SRT(Shortest Remaining Time) ⇔ SJF(비선점) + 선점 형태
    - 선점우선순위 ⇔ 비선점 우선순위(비선점) + 선점 형태
    - 다단계큐(MQ : Multi-level Queue)
    - 다단계피드백 큐(MFQ : Multi-level Feedback Queue)

스케줄링의 개요

- **스케줄링 개념**
  - 스케줄링은 프로세스가 생성되어 실행될 때 필요한 시스템의 여러 자원을 해당 프로세스에게 할당하는 작업을 의미한다.
  - 프로세스가 생성되어 완료될 때까지 프로세스는 여러 종류의 스케줄링 과정을 거치게 된다.

❖ **선점/비 선점(preemptive/nonpreemptive) 스케줄링**

- 비 선점스케줄링
  - 이미 할당된 CPU를 다른 프로세스가 강제로 빼앗아 사용할 수 없는 스케줄링 기법(일괄 처리 방식에 적합)
- 선점스케줄링
  - 이미 할당된 CPU를 다른 프로세스가 강제로 빼앗아 사용할 수 있는 스케줄링 기법. (시분할 시스템에 적합)



비선점(nonPreemptive) 스케줄링 기법

비선점(nonPreemptive) 스케줄링 기법

FCFS(First Come First Service)	• 준비상태 큐에 도착한 순서에 따라 차례로 CPU를 할당하는 기법 • 먼저 도착한 것이 먼저 처리되어 공평성은 유지되지만 짧은 작업이 긴 작업을, 중요한 작업이 중요하지 않은 작업을 기다리게 됨
SJF(Shortest Job First)	• 실행 시간이 가장 짧은 프로세스에 먼저 CPU를 할당하는 기법 • 가장 적은 평균 대기 시간을 제공하는 최적 알고리즘
HRN(Hightest Response-ratio Next)	• 실행 시간이 긴 프로세스에 불리한 SJF 기법을 보완하기 위한 것으로, 대기 시간과 서비스(실행) 시간을 이용하는 기법 • 우선순위 계산 공식 = $\frac{\text{대기 시간} + \text{서비스 시간}}{\text{서비스 시간}}$
기한부(Deadline)	• 프로세스에게 일정한 시간을 주어 그 시간 안에 프로세스를 완료하도록 하는 기법 • 시스템은 프로세스에게 할당할 정확한 시간을 추정해야 하며, 이를 위해서 사용자는 시스템이 요구한 프로세스에 대한 정확한 정보를 제공해야 함
우선순위 (Priority)	• 기다리는 각 프로세스마다 우선순위를 부여하여 그 중 가장 높은 프로세스에게 먼저 CPU를 할당하는 기법 • 우선순위의 등급은 내부적 요인과 외부적 요인에 따라 부여할 수 있음 • 각 작업마다 우선순위가 주어지며, 우선순위가 제일 높은 작업에게 먼저 프로세서가 할당됨 • 가장 낮은 순위를 부여받은 프로세스는 무한 연기 또는 기아 상태(Starvation)가 발생할 수 있음

선점 (Preemptive) 스케줄링 개념

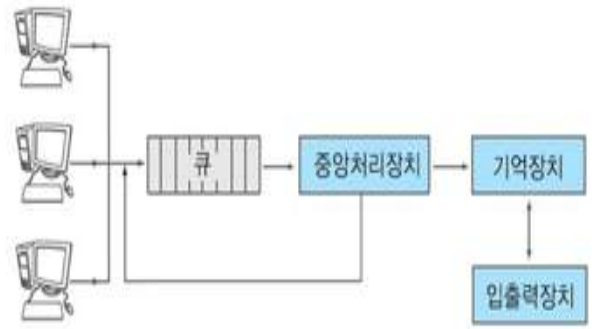
- 하나의 프로세스가 중앙처리장치를 차지하고 있을 때 다른 프로세스(우선순위가 높은)가 현재 수행 중인 프로세스를 중지시키고 자신이 중앙처리장치를 차지할 수 있는 기법
- 우선 순위가 높은 프로세스를 빠르게 처리 할 수 있음
- 시분할 시스템, 온라인 응용들에 적합
- 선점이 가능하도록 일정시간 배당에 대한 인터럽트을 타이머클럭이 필요
- 문맥 교환을 위한 많은 오버헤드를 초래
- RR, SRT, 선점우선순위, 다단계큐, 다단계 피드백 큐 등의 알고리즘

Cf)비선점스케줄링 : FCFS, SJF, 우선순위, HRN, 기한부

(1) RR(Round Robin)

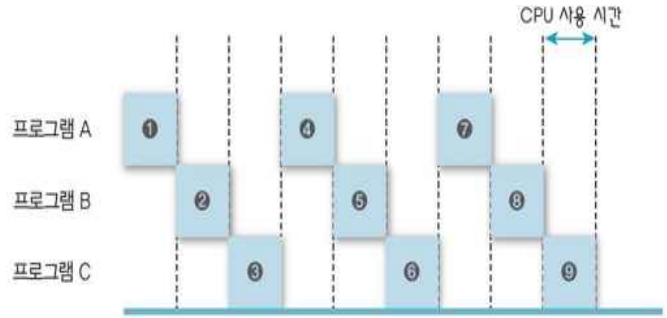
- 시분할 시스템(Time Sharing System)을 위해 고안된 방식으로, FCFS 알고리즘을 선점 형태로 변형한 기법
- FCFS 기법과 같이 준비상태 큐에 먼저 들어온 프로세스가 먼저 CPU를 할당받지만 각 프로세스는 할당된 시간(Time Slice, Quantum) 동안만 실행한 후 실행이 완료되지 않으면 다음 프로세스에게 CPU를 넘겨주고 준비상태 큐의 가장 뒤로 배치됨
- 할당되는 시간이 클 경우 FCFS 기법과 같아지고, 할당되는 시간이 작을 경우 문맥 교환 및 오버헤드가 자주 발생됨
- 보통 할당 시간량은 10×10밀리초에서 100×10밀리초 범위
- 대화식 시스템에 유용함

Cf) FCFS(First Come First Service)=FIFO(First In First Out)는 준비 상태 큐에 도착한 순서에 따라 차례로 CPU를 할당하는 기법으로 , 선입선출로 가장 간단한 알고리즘이고 문제점은 먼저 도착한 것이 먼저 처리되어 공평성은 유지되지만 짧은 작업이 긴 작업을 , 중요한 작업이 중요하지 않은 작업을 기다리게 한다.



시분할 시스템

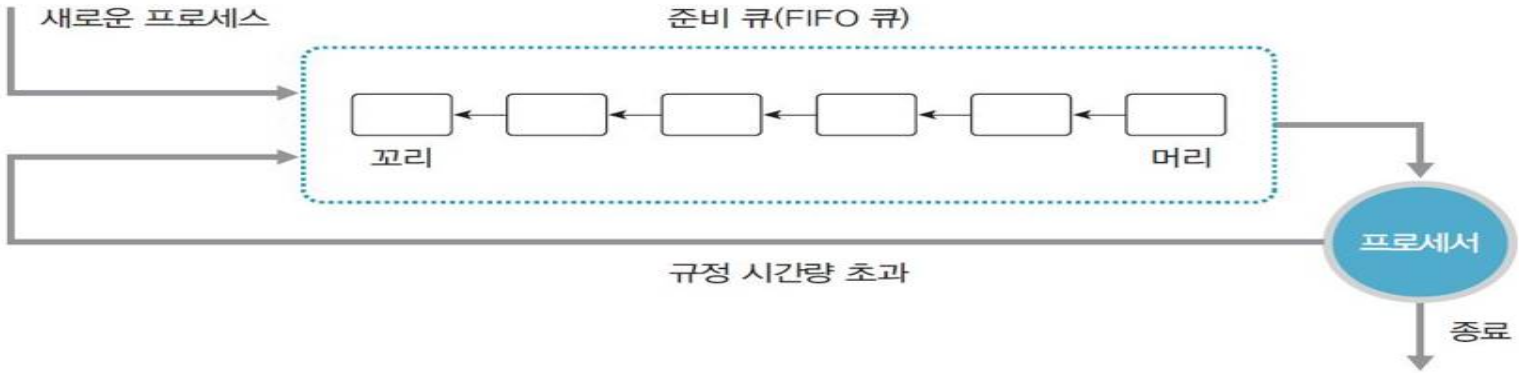
사용자 대기 시간을 줄여 다수의 사용자가 동시에 컴퓨터 자원을 공유하도록 하는 방식 (일괄처리의 단점 보완)



시분할 시스템의 처리 방법 예

(1) RR(Round Robin)

- 시분할 시스템을 위하여 고안된 선점 스케줄링 방식
- 각 프로세스는 같은 크기의 중앙처리장치 시간을 할당 받음
- 할당시간(time quantum)의 크기는 보통 10×10밀리초에서 100×10밀리초 범위
  - 할당시간이 너무 크면
    - FCFS 방식과 같은 형식이 된다.
  - 할당시간이 너무 적으면
    - 문맥 교환을 위한 오버헤드가 무시못할 요소가 되어 결과적으로 대부분의 시간이 중앙처리장치를 분배하는 데 소모



라운드 로빈 스케줄링

(1) RR(Round Robin)

❖ 문맥 교환 시간이 라운드 로빈 스케줄링에 미치는 영향

할당시간	문맥 교환 횟수
11	0
8	1
2	4

할당시간에 따른 문맥 교환의 횟수

RR(Round Robin) 예제

대기시간 : 프로세스의 대기한 시간으로 , 바로 앞 프로세스까지의 진행시간을 구함
반환시간 : 프로세스의 대기시간과 실행 시간의 합

- 다음과 같은 프로세스들이 차례로 준비상태 큐에 들어왔다고 가정할 때 평균 대기시간, 평균반환시간을 구하시오.(단, Time Slice는 4초이다.)

프로세스번호	p1	P2	P3
실행시간	20	4	6

- 주어진 시간할당량(time slice)동안 실행되지 못할 경우 준비상태 큐의 가장 마지막으로 재배치하여 차례를 기다리므로 다음과 같이 표시할 수 있다.

0 4 8 12 16 18 22 26 30

프로세스번호	p1	p2	p3	p1	p3	p1	p1	P1
실행시간	4	4	4	4	2	4	4	4

프로세스번호	**p1**	**p2**	**p3**	**평균**
반환시간	**30**	**8**	**18**	**56/3=18.6**
대기시간	**26-16=10**	**4**	**16-4=12**	**26/3=8.6**

대기 시간은 구하고자 하는 프로세스의 가장 마지막 실행이 시작되기 전까지의 진행시간에 해당 프로세스가 앞에서 여러 번 실행되었을 경우 실행된 시간은 제외한다.

RR(Round Robin) 문제

- round-robin방식으로 스케줄링 할 경우, 입력된 작업이 다음과 같고 각 작업의 cpu할당 시간이 4시간일 때, 모든 작업을 완료하기 위한 CPU의 사용 순서가 옳게 나열된 것은?

가.ABCABCBCC　　나.AAABBBCCC　　다.ABCABCACA　　라.ACCCCCBBA

작업	입력시간	작업수행시간
A	10:00	5시간
B	10:30	10시간
C	12:00	15시간

(2) SRT(Shortest Remaining Time)

- 비선점 스케줄링인 SJF기법을 선점형태로 변경한 기법
- 선점SJF기법이라고도 한다.
- 현재 실행중인 프로세스의 남은 시간과 준비상태 큐에 새로 도착한 프로세스의 실행시간을 비교하여 가장 짧은 실행시간을 요구하는 프로세스에게 CPU를 할당하는 기법
- 새로 도착한 프로세스를 포함하여 처리가 완료되는 데 가장 짧은 시간이 소요된다고 판단되는 프로세스를 먼저 수행
- 시분할 시스템에 유용하다.
- 준비상태큐에 있는 각 프로세스의 실행시간을 추적하여 보유하고 있어야 하므로 오버헤드가 증가한다.

Cf) SJF(Shortest Job First)은 준비상태큐에서 기다리고 있는 프로세스들 중에서 실행시간이 가장 짧은 프로세스에게 먼저 CPU를 할당하는 기법으로 가장 적은 평균 대기시간을 제공하는 최적 알고리즘이다. 문제점은 실행시간이 긴 프로세스는 실행시간이 짧은 프로세스에게 할당 순위가 밀려 무한 연기 상태가 발생할 수 있다.

SRT(Shortest Remaining Time) 예제1

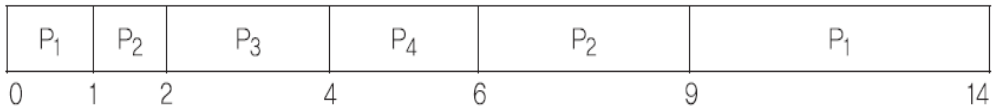
- 다음과 같은 프로세스P1,P2,P3,P4의 중앙처리 장치 버스트 시간(실행 시간)과 도착 시간이 다음과 같을 경우 평균 대기시간과 평균 반환시간을 구하시오. (도착 시간이 있는 경우)

대기시간 : 프로세스의 대기한 시간으로 , 바로 앞 프로세스까지의 진행시간을 구함
반환시간 : 프로세스의 대기시간과 실행 시간의 합

프로세스	버스트 시간	도착시간
P_1	6	0
P_2	4	1
P_3	2	2
P_4	2	3

1) SRT(Shortest Remaining Time)

- Gantt 차트



- 평균 반환 시간
- P1=(14-0)=14, P2=(9-1)=8, P3=(4-2)=2, P4=(6-3)=3 이므로 (14+8+2+3)/4=6.75
- 평균 대기 시간
- P1=0+(9-1)=8, P2=6-2=4, P3=0, P4=4-3=1 이므로 (8+4+0+1) /4=3.25

- 간트 차트(Gantt chart)는 프로젝트 일정관리를 위한 바(bar)형태의 도구로서 각 업무별로 일정의 시작과 끝을 그래픽으로 표시하여 전체 일정을 한눈에 볼 수 있는 차트
- CPU Burst Time은 cpu가 할당되어서 실제로 cpu를 점유하고 있는 시간(실행 시간).

SRT(Shortest Remaining Time) 예제2

- 다음과 같은 프로세스P1,P2,P3,P4의 중앙처리 장치 버스트 시간(실행 시간)과 도착 시간이 다음과 같을 경우 평균 대기시간과 평균 반환시간을 구하시오. (도착 시간이 있는 경우)

프로세스	버스트 시간	도착시간
P_1	6	0
P_2	4	1
P_3	2	2
P_4	2	3

대기시간 : 프로세스의 대기한 시간으로 , 바로 앞 프로세스까지의 진행시간을 구함
반환시간 : 프로세스의 대기시간과 실행 시간의 합

2) SJF(Shortest Job First)

- Gantt 챠트
-
- 평균 반환 시간
-
- 평균 대기 시간

- SRT는 대기 시간을 최소로 하지만 각 프로세스가 서비스를 받을 시간이 기록 보유되어야 하기 때문에 오버 헤드가 늘어나기 때문에 SJF로 처리하는 것이 나을 때도 있다.

(3) 선점우선순위(Preemptive Priority)

- 준비상태큐의 프로세스들 중에서 우선순위가 가장 높은 프로세스에게 먼저 CPU를 할당하는 기법이다.
- 비선점 우선순위기법을 선점 형태로 변경한 것으로 , 준비상태큐에 새로 들어온 프로세스의 순위가 높을 경우 현재의 프로세스를 보류하고 새로운 프로세스를 실행한다.
  - 우선순위를 내부적 또는 외부적으로 정의
    - 내부적 우선순위 : 제한 시간, 기억장소 요청량, 사용 파일 수, 평균 프로세서 버스트에 대한 평균 입출력 버스트의 비율 등
    - 외부적 우선순위 : 프로세스 중요성, 사용료 많이 낸 사용자, 작업을 지원하는 부서, 정책적인 요인 등

Cf) 비선점 우선순위(Priority)은 준비상태 큐에서 기다리는 각 프로세스마다 우선순위를 부여하여 그 중 가장 높은 프로세스에게 먼저 CPU를 할당하는 기법으로 비선점 기법이다 우선순위가 동일할 경우 FCFS기법으로 CPU를 할당한다.문제점은 가장 낮은 순위를 부여받는 프로세스는 무한연기 또는 기아상태(Starvation) 가 발생할 수 있다. 이 문제를 해결 하기 위한 기법으로 Aging기법이 있다.

(3) 선점우선순위(Preemptive Priority)

- 준비상태큐의 프로세스들 중에서 우선순위가 가장 높은 프로세스에게 먼저 CPU를 할당하는 기법
- 선점 우선 순위와 비선점 우선 순위 예(숫자가 큰것이 우선 순위가 높다)

프로세스	도착 시간	실행 시간	우선순위
P_1	0	10	3
P_2	1	28	2
P_3	2	6	4
P_4	3	4	1
P_5	4	14	2

(a) 준비 큐

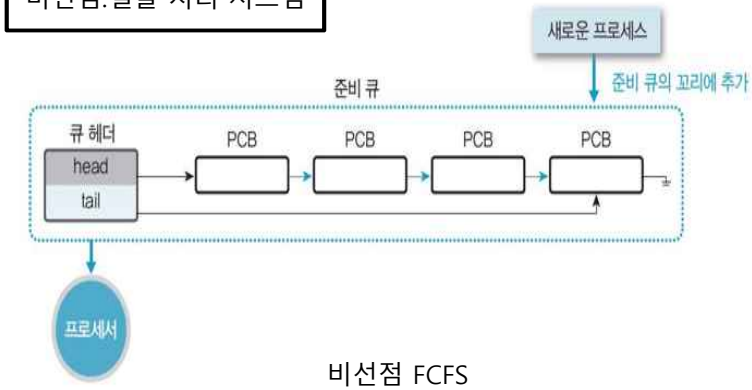
(b) 선점 우선순위 간트 차트

(c) 비선점 우선순위 간트 차트

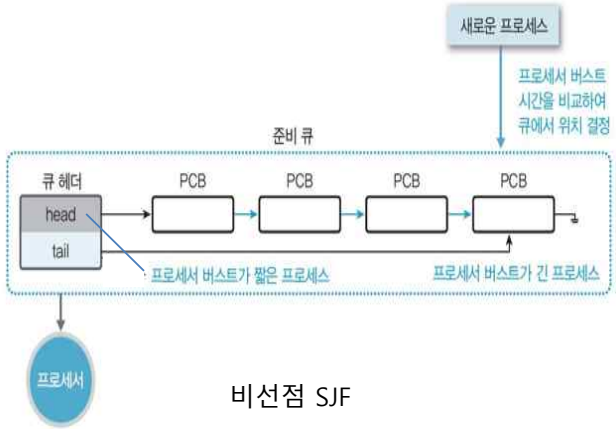
우선순위 스케줄링 예

선점과 비선점 스케줄링 비교

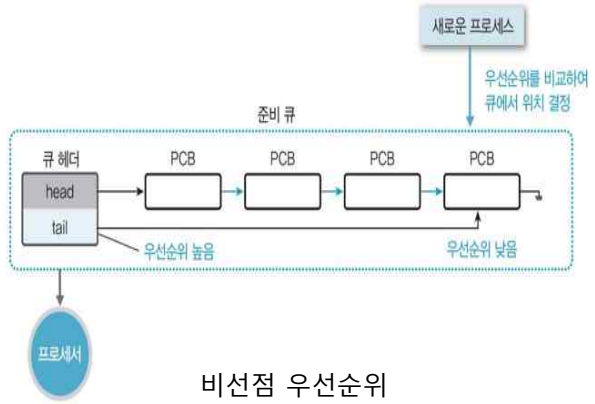
비선점:일괄 처리 시스템



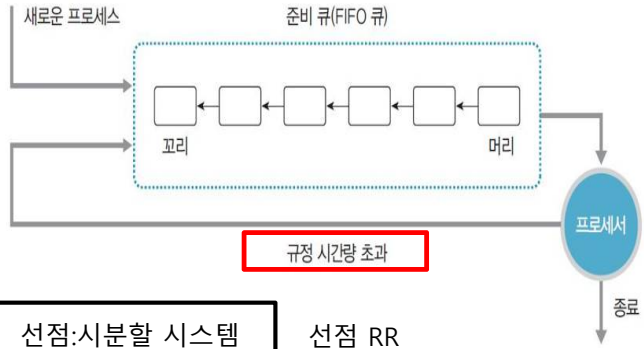
비선점 FCFS



비선점 SJF

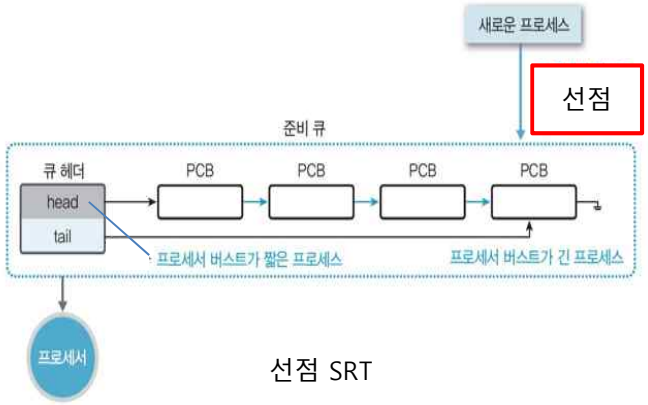


비선점 우선순위

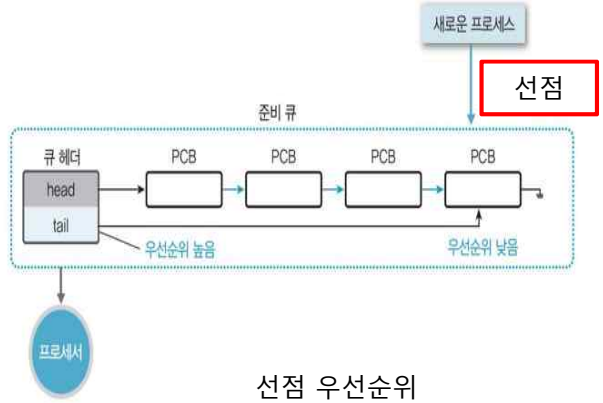


선점:시분할 시스템

선점 RR



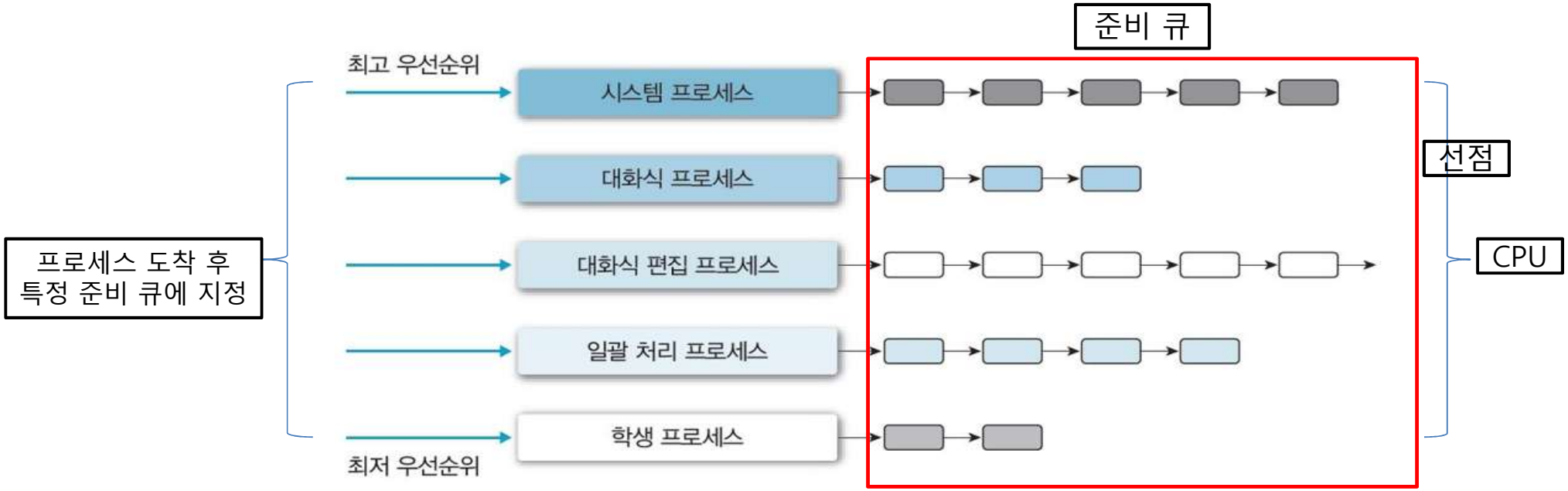
선점 SRT



선점 우선순위

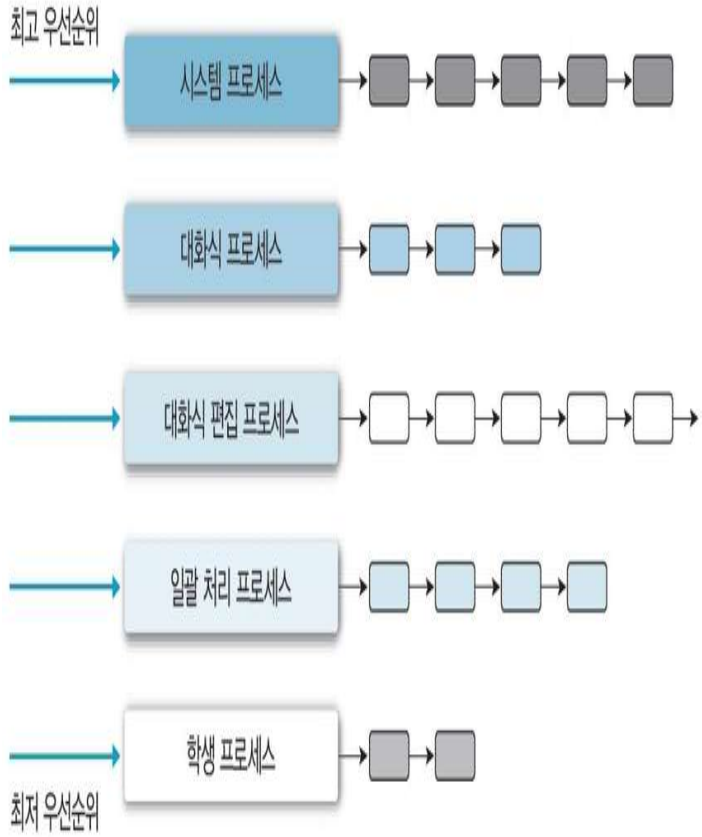
(4) 다단계 큐(MQ : Multi-level Queue)

- 작업들을 여러 그룹으로 나누어 여러 개의 준비 상태 큐를 이용하는 기법
  (각 작업을 서로 다른 묶음으로 분류할 수 있을 때 사용)
- 작업을 메모리의 크기나 프로세스 형태(특성)에 따라 특정 큐에 지정
  - 예를 들어 전면작업 프로세스과 후면 작업 프로세스로 분류를 하면 두 유형의 요청 반응 시간은 다르며 서로 다르게 스케줄링해야 한다. 또한 전면작업 프로세스는 후면작업 프로세스들보다도 높은 우선순위가 높다.



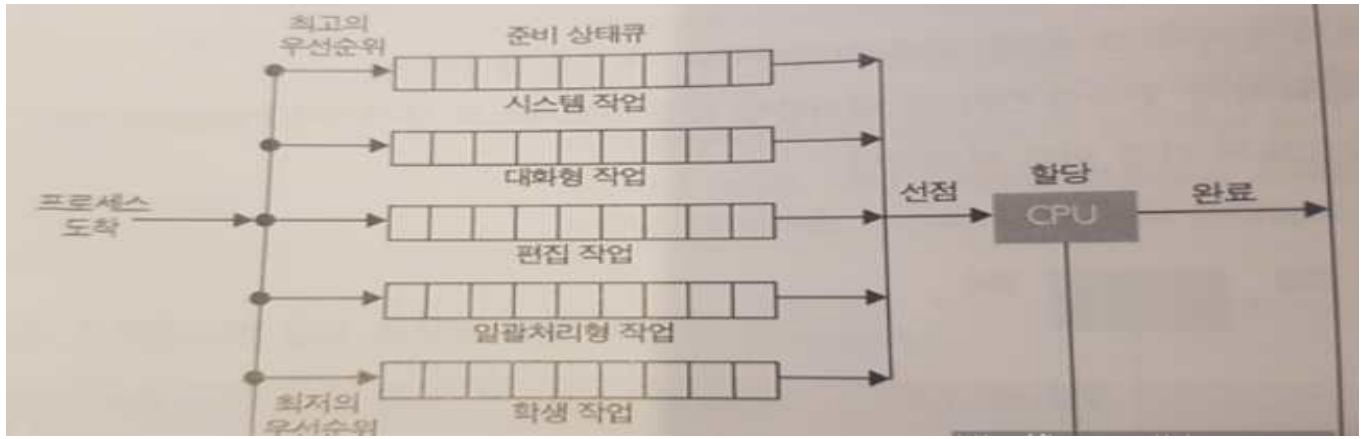
(4) 다단계 큐(MQ : Multi-level Queue)

- 작업들을 여러 그룹으로 나누어 여러 개의 준비 상태 큐를 이용하는 기법(각 작업을 서로 다른 묶음으로 분류할 수 있을 때 사용)
- 프로세스들을 우선순위에 따라 시스템프로세스, 대화형프로세스, 편집프로세스, 일괄처리프로세스등으로 상위,중위,하위 단계의 단계별 준비 큐를 배치하는 CPU 스케줄링 기법
- 전면작업 프로세스들은 후면작업 프로세스들보다도 높은 우선순위
- 큐별로 스케줄링
- 각 준비상태큐는 독자적인 스케줄링을 가지고 있으므로 각 그룹의 특성에 따라 서로 다른 스케줄링 기법을 사용할 수 있다.



(4) 다단계 큐(MQ : Multi-level Queue)

- 준비 상태 큐를 종류별로 여러 단계로 분할, 그리고 작업을 메모리의 크기나 프로세스의 형태에 따라 특정 큐에 지정. 각 큐는 자신만의 독자적인 스케줄링 갖음
- 하위 단계 준비상태 큐에 있는 프로세스를 실행하는 도중이라도 상위 단계 준비 상태 큐에 프로세스가 들어오면 상위 단계 프로세스에게 CPU를 할당해야 한다.
- 프로세스가 특정 그룹의 준비 상태 큐에 들어갈 경우 다른 준비 상태 큐로 이동 할 수 없다.
- 응답이 빠르지만 여러 준비 큐와 스케줄링 알고리즘 때문에 오버 헤드가 발생하며 우선 순위가 낮은 큐의 프로세스는 무한정 대기하는 기아가 발생할 수 있다.



다단계 큐

(5) 다단계 피드백 큐
(MFQ : Multi-level Feedback Queue)

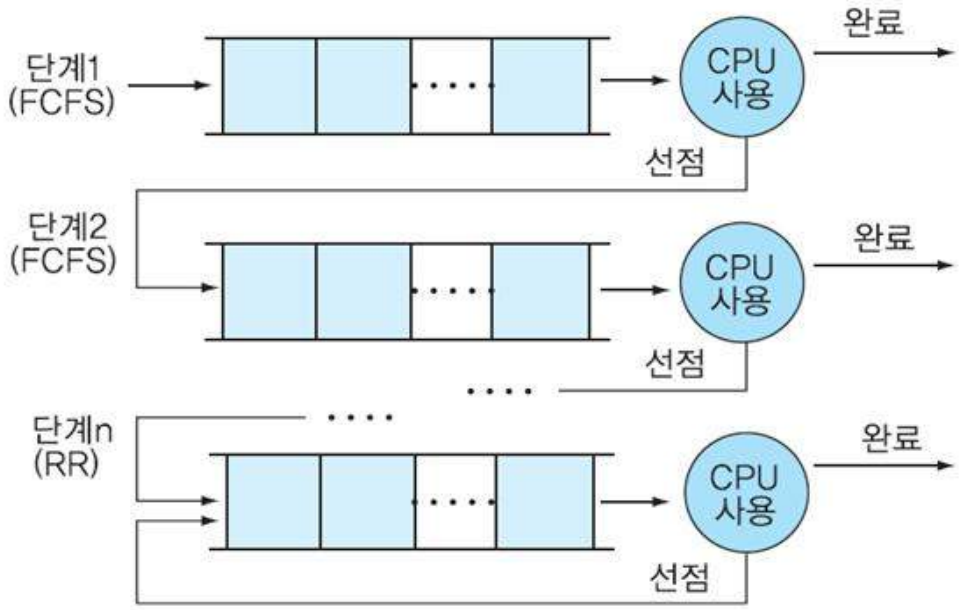
- 특정 그룹의 준비상태 큐에 들어간 프로세스가 다른 준비상태 큐로 이동 할 수 없는 다단계 큐 기법을 준비상태 큐 사이를 이동 할 수 있도록 개선한 기법
  (다단계의 융통성이 떨어지는 단점 보완)

- 각 준비 상태 큐마다 시간 할당량을 부여하여 그 시간 동안 완료 되지 못한 프로세스는 다음 단계의 준비상태 큐로 이동한다.(프로세서 실행의 특성에 따라)
  - 작업 요청의 프로세서 실행 시간이 너무 길면 낮은 단계큐로 이동하고 특정 큐에서 오래 기다린 프로세스를 우선 순위가 높은 큐로 이동시켜 기아 상태를 예방한다.
  - 작업 분류는 입출력 중심 작업과 전면 작업과 프로세서 실행 시간이 짧은 프로세스는 높은 우선 순위 큐를 할당하여 일찍 종료 시킨다.

- 하위 단계 준비상태 큐에 있는 프로세스를 실행하는 도중이라도 상위 단계 준비 상태 큐에 프로세스가 들어오면 상위 단계 프로세스에게 CPU를 할당하며 마지막 단계 큐에서는 작업이 완료 될 때까지 RR스케줄링 기법을 사용한다

(5) 다단계 피드백 큐 (MFQ : Multi-level Feedback Queue)

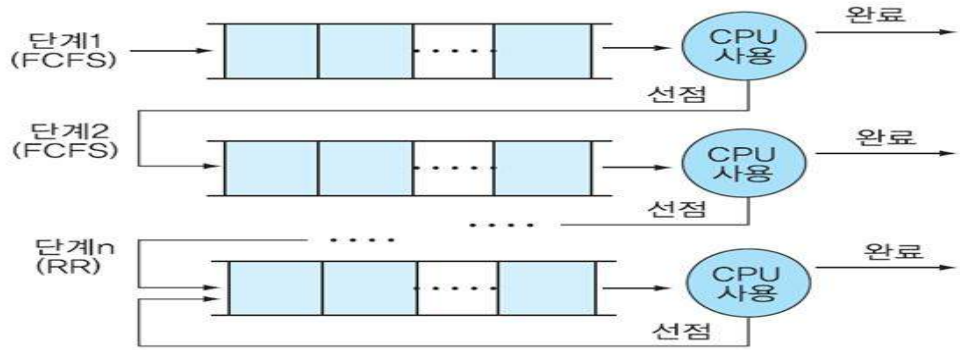
- 대부분의 다단계 피드백 체계에서는 프로세스가 하위 단계의 큐로 옮겨갈수록 주어진 할당시간은 점차 크게 설정하고 마지막 단계 큐에서는 작업이 완료 될 때까지 RR 스케줄링 기법을 사용한다.



다단계 피드백 큐

(5) 다단계 피드백 큐 (MFQ : Multi-level Feedback Queue)

- 일반적의 다단계 피드백 큐 스케줄러는 다음 매개변수에 의해 정의 된다.
  - 큐의 개수
  - 각 큐를 위한 스케줄링 알고리즘
  - 한 프로세스를 높은 우선순위 큐로 올려주는 시기를 결정하는 방법
  - 한 프로세스를 낮은 우선순위 큐로 강등시키는 시기를 결정하는 방법
  - 프로세스가 서비스를 필요로 할 때 프로세스가 들어갈 큐를 결정하는 방법



다단계 피드백 큐

- 다단계 피드백 큐는 가장 일반적인 CPU 스케줄링 알고리즘이지만 모든 매개 변수들의 값을 선정하는 특정 방법이 필요하기 때문에 가장 복잡한 알고리즘이기도 하다.

학습 내용 정리

선점(Preemptive) 스케줄링 기법

선점 우선순위	준비상태 큐의 프로세스들 중에서 우선순위가 가장 높은 프로세스에게 먼저 CPU를 할당하는 기법
SRT(Shortest Remaining Time)	비선점 기법인 SJF 알고리즘을 선점 형태로 변경한 기법으로, 현재 실행중인 프로세스의 남은 시간과 준비상태 큐에 새로 도착한 프로세스의 실행 시간을 비교하여 가장 짧은 실행 시간을 요구하는 프로세스에게 CPU를 할당하는 기법
RR(Round Robin)	• 시분할 시스템(Time Sharing System)을 위해 고안된 방식으로, FCFS 알고리즘을 선점 형태로 변형한 기법 • FCFS 기법과 같이 준비상태 큐에 먼저 들어온 프로세스가 먼저 CPU를 할당받지만 각 프로세스는 할당된 시간(Time Slice, Quantum) 동안만 실행한 후 실행이 완료되지 않으면 다음 프로세스에게 CPU를 넘겨주고 준비상태 큐의 가장 뒤로 배치됨 • 할당되는 시간이 클 경우 FCFS 기법과 같아지고, 할당되는 시간이 작을 경우 문맥 교환 및 오버헤드가 자주 발생됨 • 시간 할당량이 너무 작으면 스래싱에 소요되는 시간의 비중이 커짐 • 대화식 시스템에 유용함
다단계 큐 (Multi level Queue)	프로세스들을 우선순위에 따라 시스템 프로세스, 대화형 프로세스, 일괄 처리 프로세스 등으로 상위, 중위, 하위 단계의 단계별 준비 큐를 배치하는 CPU 스케줄링 기법
다단계 피드백 큐(Multi level Feedback Queue)	• 특정 그룹의 준비상태 큐에 들어간 프로세스가 다른 준비상태 큐로 이동할 수 없는 다단계 큐 기법을 준비상태 큐 사이를 이동할 수 있도록 개선한 기법 • 특정 큐에서 오래 기다린 프로세스나 I/O 작업 주기가 큰 프로세스 또는 Foreground 큐에 있는 프로세스는 우선순위가 높은 단계의 준비 큐로 이동시키고, CPU의 점유 시간이 긴 작업은 우선순위가 낮은 하위 단계의 준비 큐로 이동시킴 • 마지막 단계 큐에서는 작업이 완료될 때까지 RR 스케줄링 기법을 사용함