CPU 스케줄링

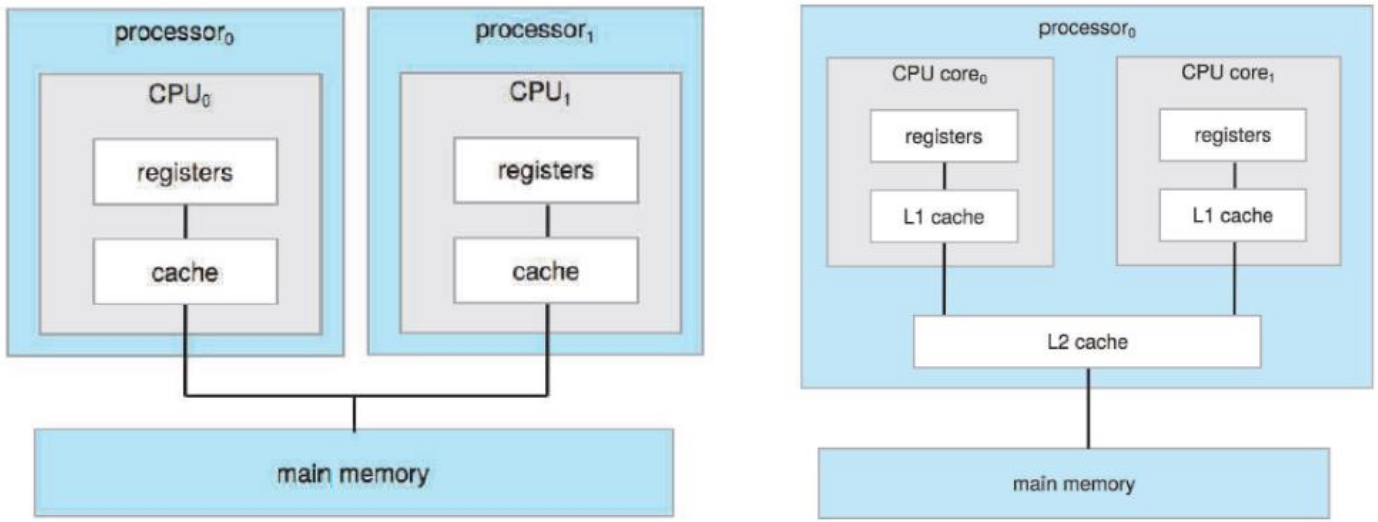
김현수

왜 필요할까요?

- 하나의 프로세서에서 프로그램의 동작 효율을 높이기 위해!
- 멀티 프로세서?
- 멀티 프로그래밍?
- 멀티 프로세서 vs 멀티 프로그래밍

멀티 프로세서

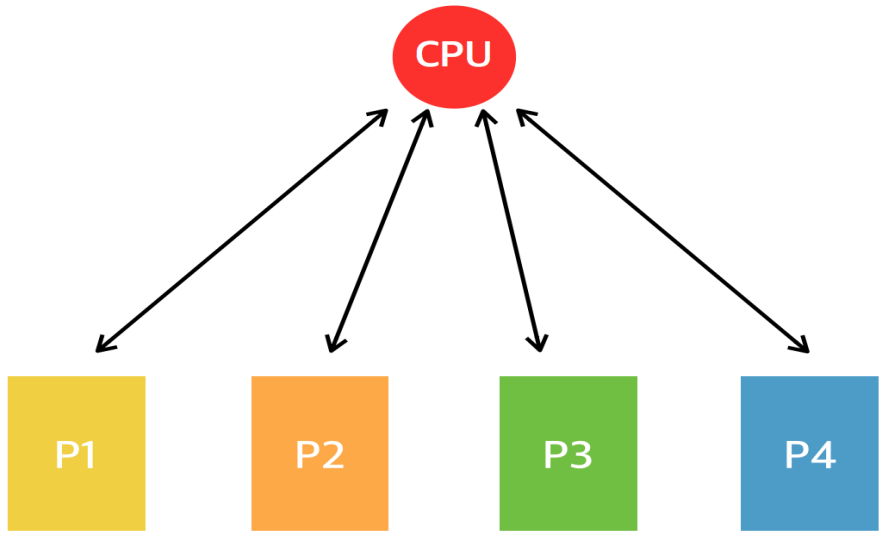
- 하나의 컴퓨터, 여러 개의 프로세서(CPU)



- 다수의 프로세서가 서로 협력하여 일을 처리

멀티 프로그래밍

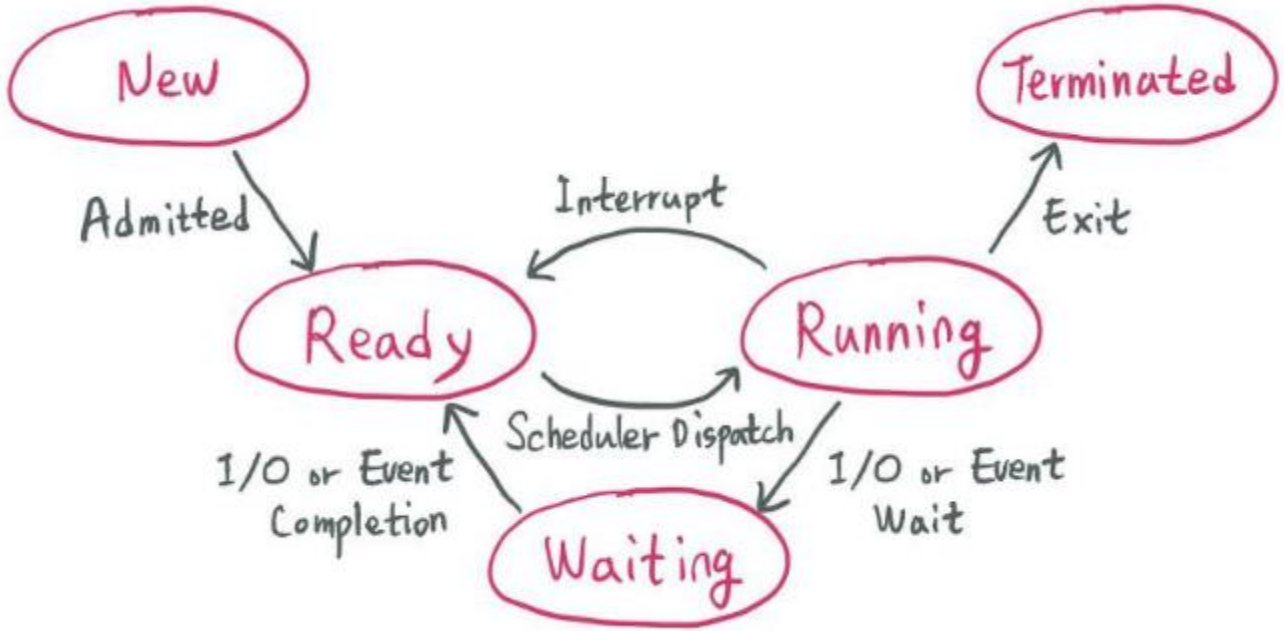
- 하나의 CPU, 여러 개의 프로그램



- CPU가 프로그램을 처리할 때 보다 효율적으로 수행할 수 있음

CPU와 프로그램

- 프로그램이 실행되려면 반드시 CPU가 할당되어야 한다.
- 프로세스 상태



CPU 스케줄러의 역할

- 멀티 프로그래밍
  - 프로그램이 동시에 실행되는 것 X.
  - 번갈아가면 실행되어서 동시에 실행되는 것처럼 보이는 것.

- CPU 스케줄러 : CPU를 효율적으로 사용하기 위해 프로그램의 순서를 정하는 것
  - Batch System : 처리량
  - Interactive System : 빠른 응답시간
  - Real-time System : 데드라인

스케줄링 척도

- Throughput 처리율 : 단위시간당 처리하는 작업의 수
- Turnaround time 반환시간 : 프로세스의 모든 작업이 끝나고 종료하는데 걸린 시간
- Waiting time 대기시간 : 프로그램이 ready 상태에서 기다린 시간
- Response time 응답시간 : 프로그램이 할당되기까지 걸린 시간

여러가지 스케줄러

- First In First Out
- Shortest Job First

- Priority Scheduling
- Round Robin
- Multilevel Feedback Queue

FIFO

- 먼저 들어온 프로그램 먼저 실행

Process	Burst Time(msec)
P1	24
P2	3
P3	3

- Turnaround time = 24+27+30 = 81
- Response time = 0+24+27 = 51
- 너무 많은 시간

SJF

- 실행시간이 작은 프로그램 먼저 실행

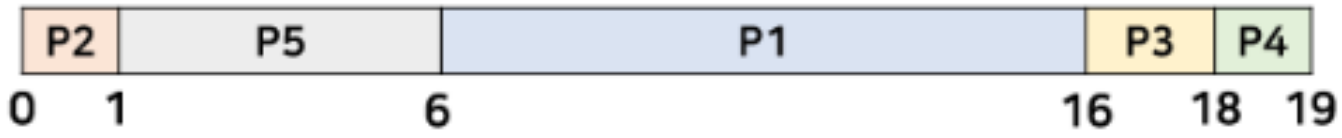
Process	Burst Time(msec)
P1	24
P2	3
P3	3

- Turnaround time = 3+6+30 = 39
- Response time = 0+3+6 = 9
- 비현실적

Priority

- 우선순위가 높은 프로그램을 먼저 실행

Process	Burst Time(msec)	Priority
P1	10	3
P2	1	1
P3	2	4
P4	1	5
P5	5	2

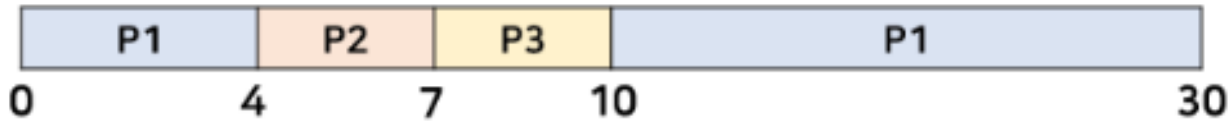


- 현재 실행 중인 프로그램보다 우선순위가 높은 프로그램이 등록되면 현재 실행 중인 프로그램을 중지하고 우선순위가 높은 프로그램 실행
- Starvation

Round Robin

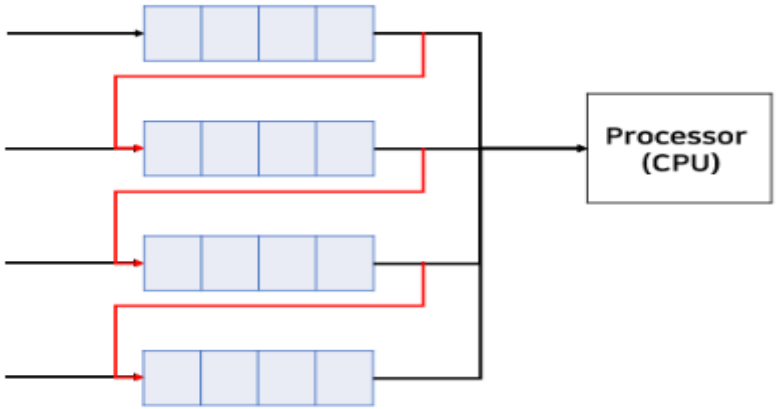
- 일정시간만큼 프로그램 실행 후 다음 프로그램 실행

Process	Burst Time(msec)
P1	24
P2	3
P3	3



- Time quantum에 매우 의존적

MLFQ



- Queue -> FIFO
- 각 Queue의 레벨 -> Priority
- 상위 Q에서 너무 오랜 시간 점유하면 하위 Q로 조정 -> RR
- 모든 Q가 연결되어 있음 -> Starvation 발생 시 상위 Q로 조정
- 각 Q는 서로 다른 스케줄링 적용 가능

질문

땡큐베리머치