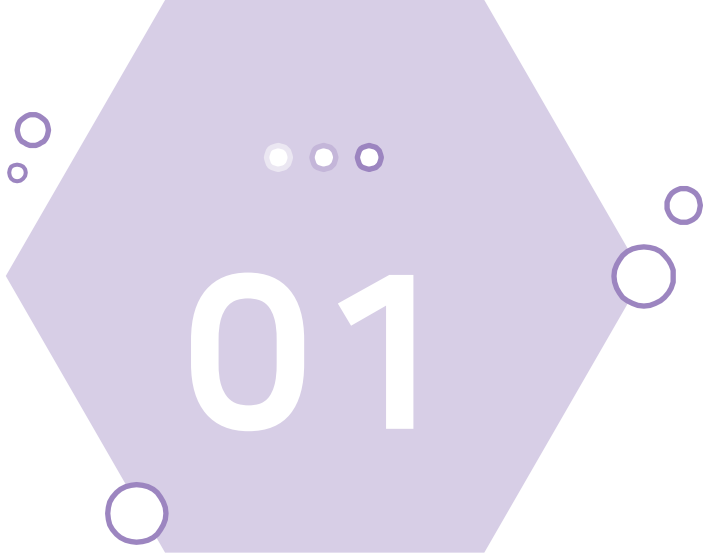
3강. 스케줄링 알고리즘

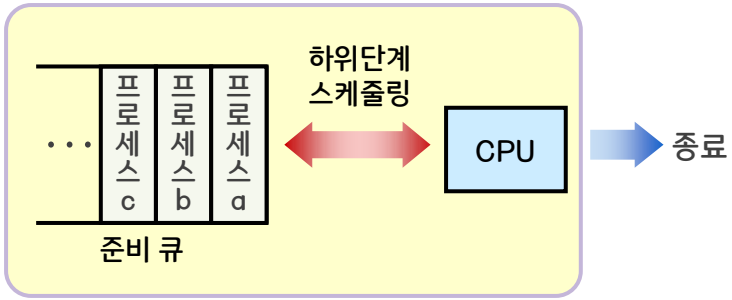
방송대 컴퓨터과학과
김진욱 교수

목차



스케줄링 성능 평가 기준

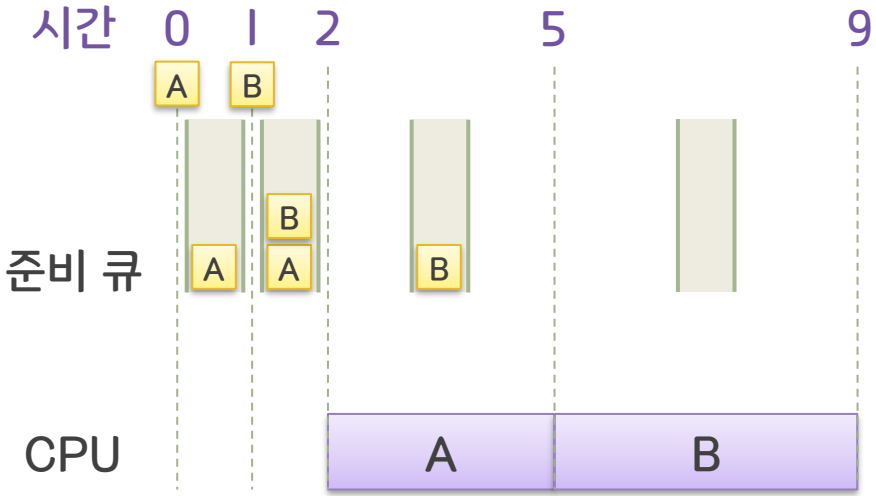
스케줄링 성능 평가 기준



- **평균 대기시간**
  - 각 프로세스가 수행이 완료될 때까지 준비 큐에서 기다리는 시간의 합의 평균값

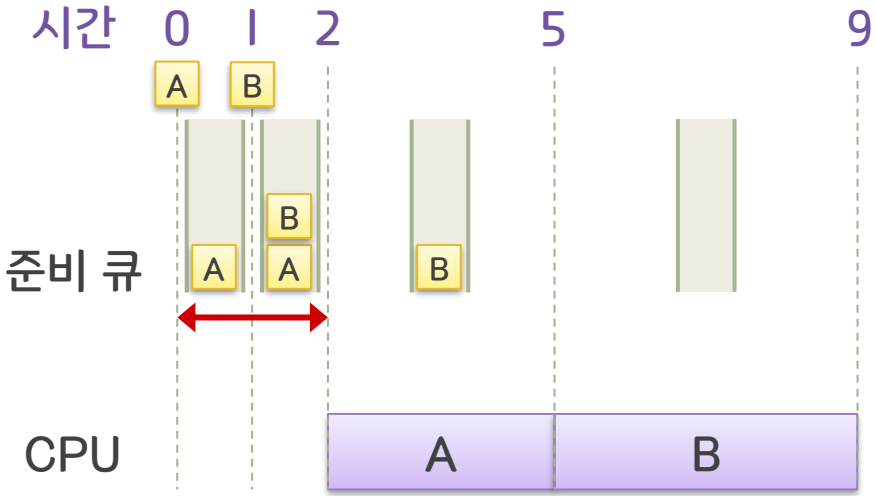
- **평균 반환시간**
  - 각 프로세스가 생성된 시점부터 수행이 완료된 시점까지의 소요시간의 평균값

스케줄링 성능 평가 기준



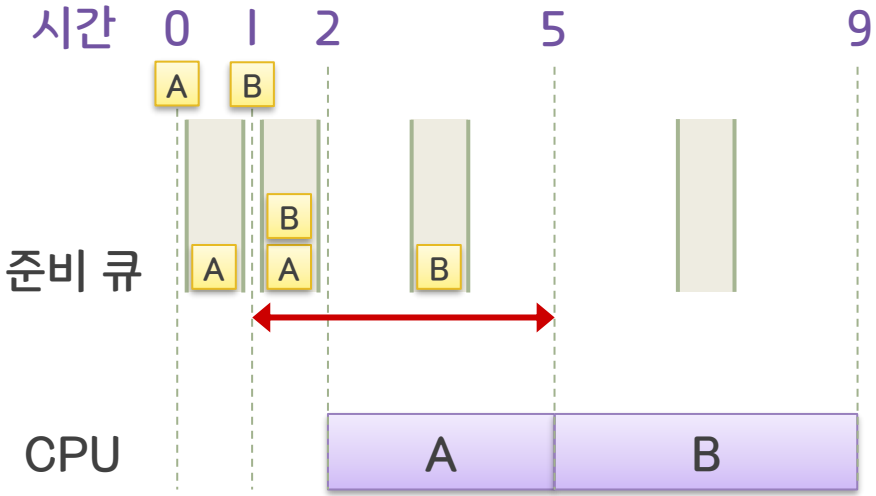
프로세스	A	B
대기시간		
반환시간		

스케줄링 성능 평가 기준



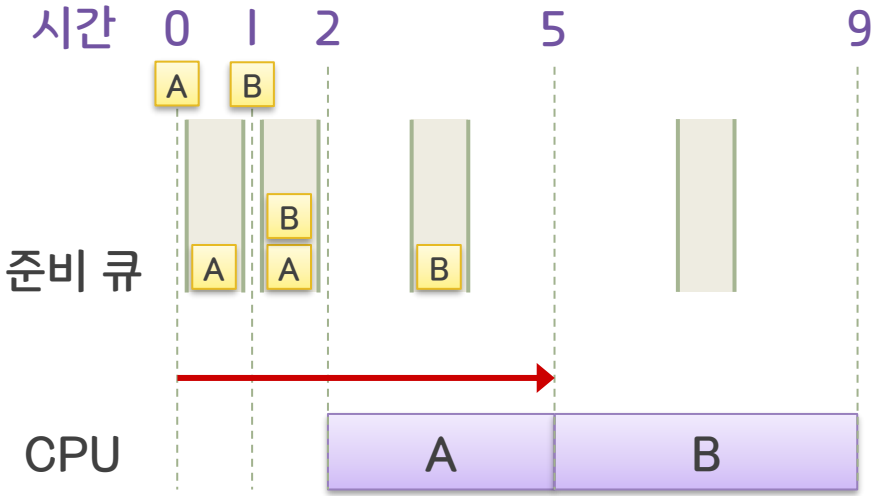
프로세스	A	B
대기시간		
반환시간		

스케줄링 성능 평가 기준



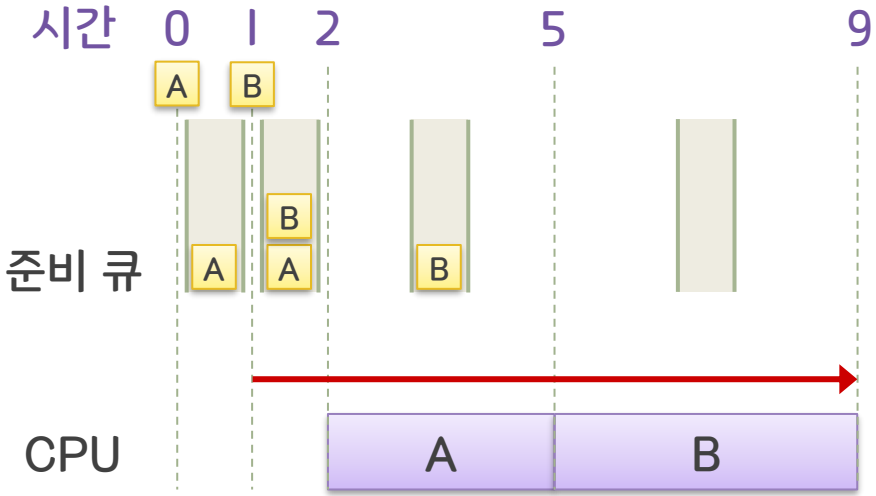
프로세스	A	B
대기시간		
반환시간		

스케줄링 성능 평가 기준



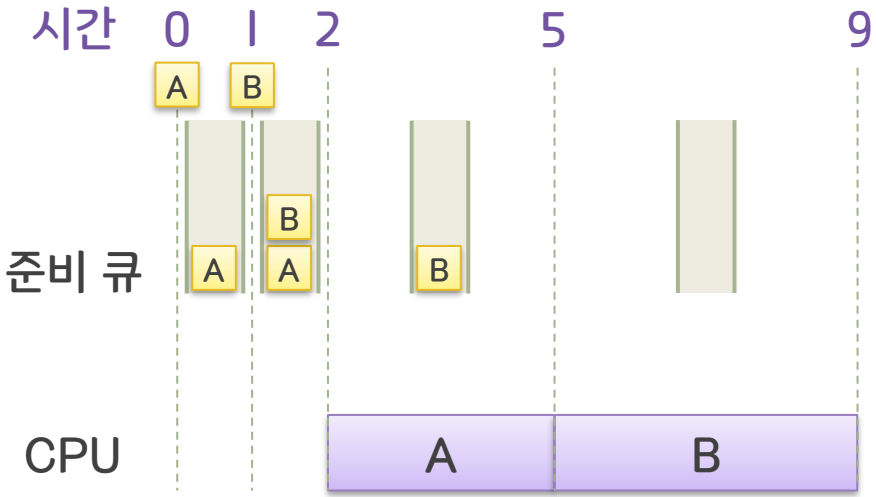
프로세스	A	B
대기시간		
반환시간		

스케줄링 성능 평가 기준



프로세스	A	B
대기시간		
반환시간		

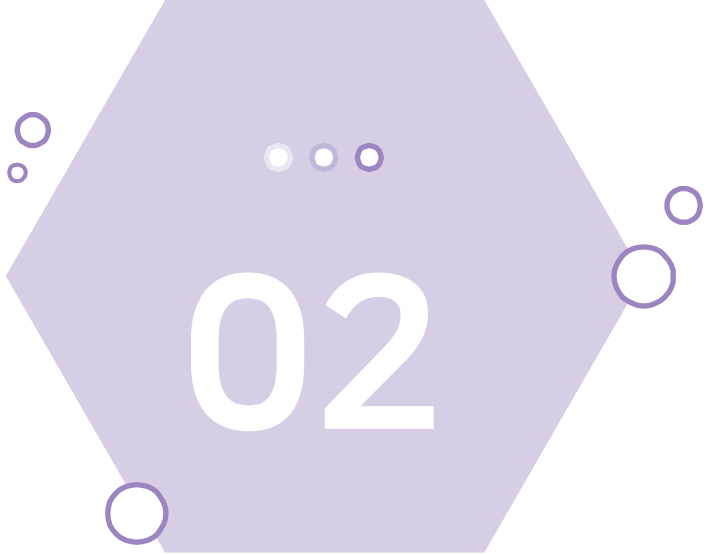
스케줄링 성능 평가 기준



- 평균 대기시간 = $\frac{2+4}{2}$ = 3

프로세스	A	B
대기시간		
반환시간		

- 평균 반환시간 = $\frac{5+8}{2}$ = 6.5



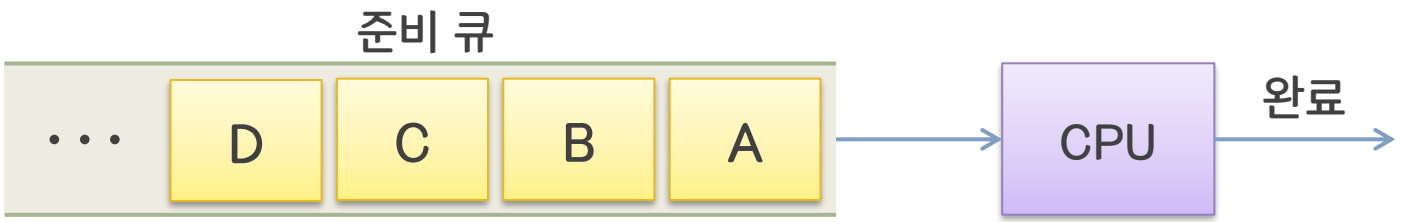
다양한 스케줄링 알고리즘

프로세스와 쓰레드

- FCFS 스케줄링
- SJF 스케줄링
- SRT 스케줄링
- RR 스케줄링
- HRN 스케줄링
- 다단계 피드백 큐 스케줄링

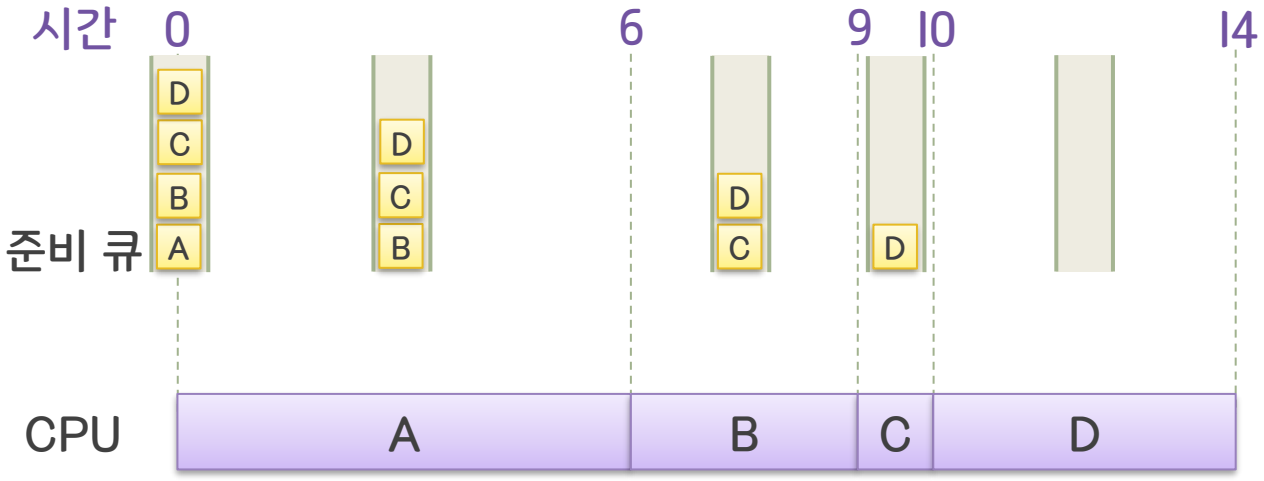
FCFS 스케줄링

- FCFS (First-Come First-Served) 스케줄링
  - 비선점 스케줄링 알고리즘
  - 준비 큐에 도착한 순서에 따라 디스패치



FCFS 스케줄링의 예

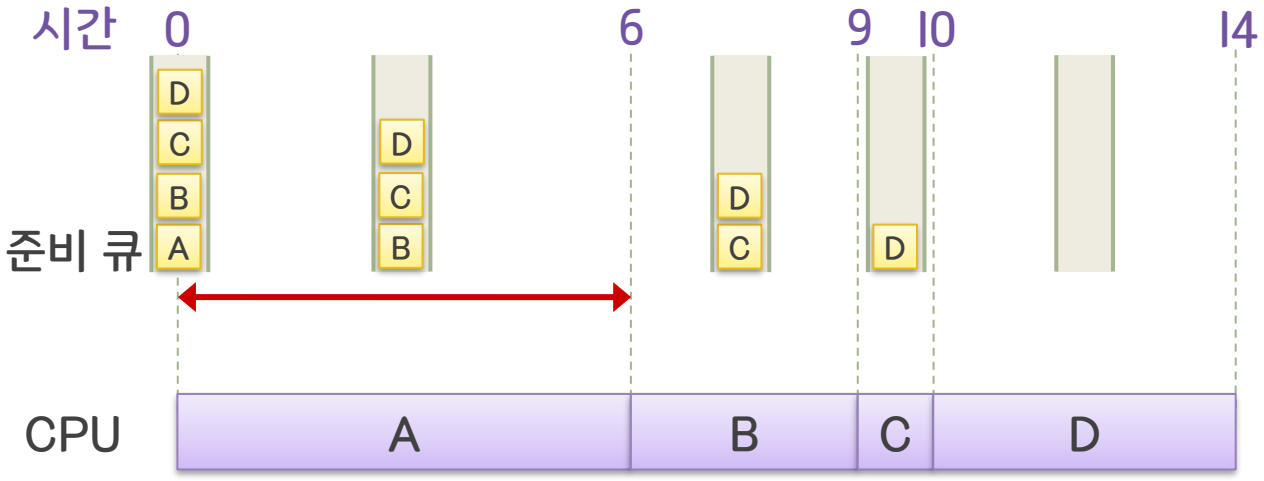
도착시간	0	0	0	0
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	0	6	9	10
반환시간	6	9	10	14

FCFS 스케줄링의 예

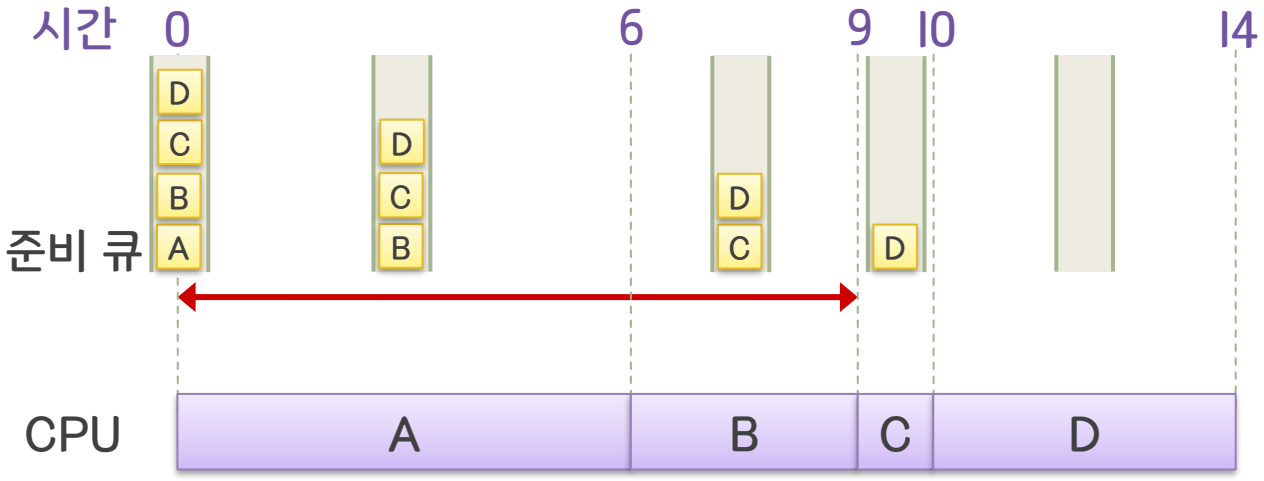
도착시간	0	0	0	0
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	0	6	9	10
반환시간	6	9	10	14

FCFS 스케줄링의 예

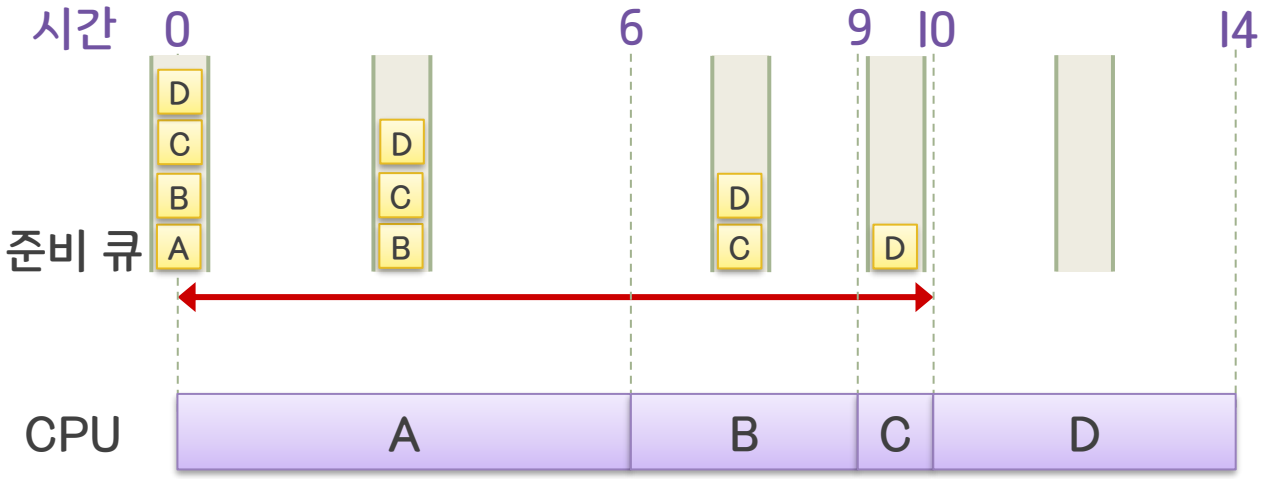
도착시간	0	0	0	0
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	0	6	9	10
반환시간	6	9	10	14

FCFS 스케줄링의 예

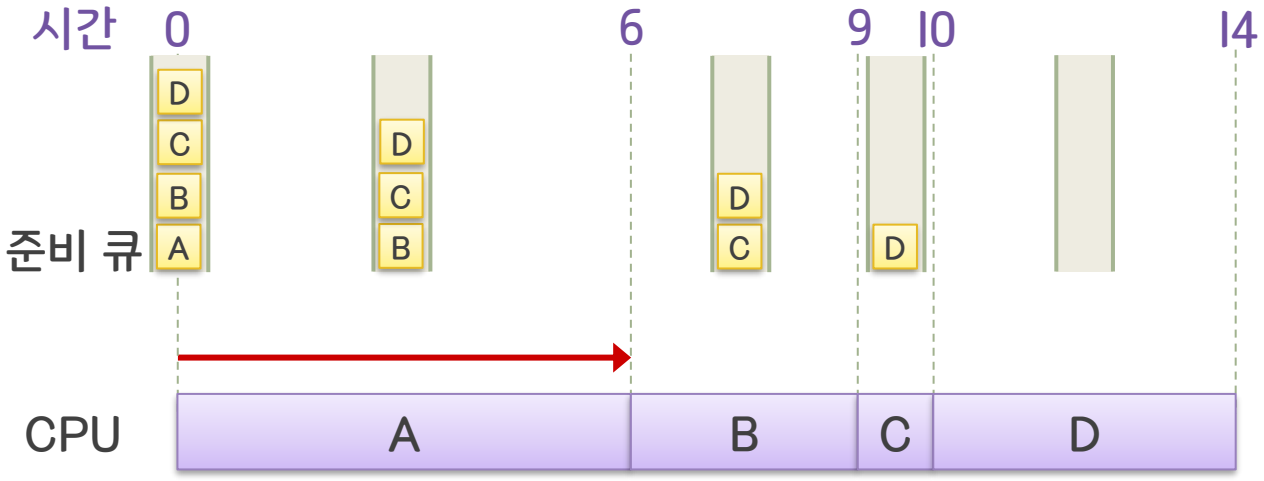
도착시간	0	0	0	0
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	0	6	9	10
반환시간	6	9	10	14

FCFS 스케줄링의 예

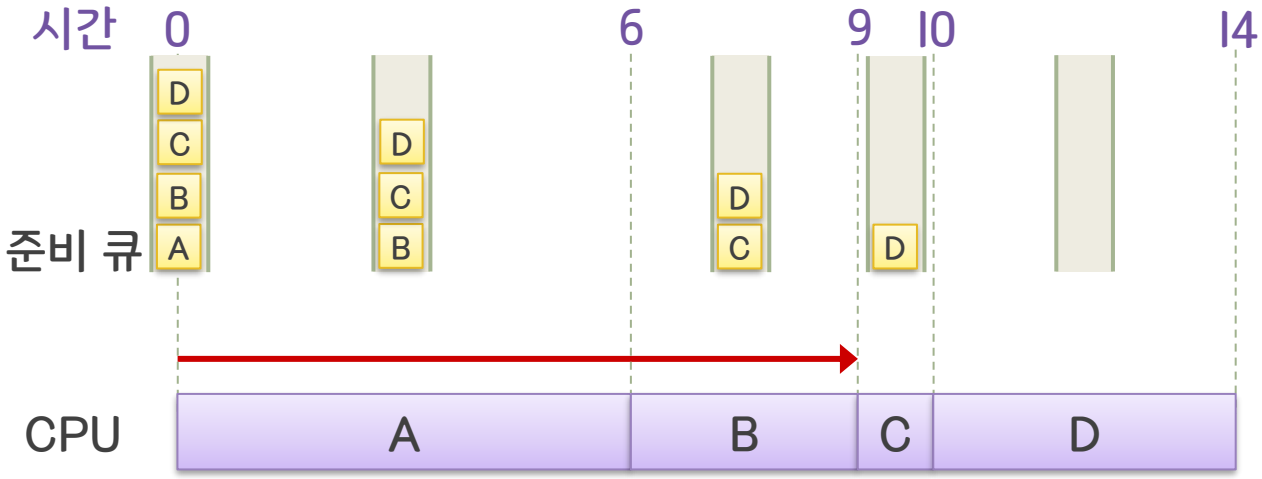
도착시간	0	0	0	0
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	0	6	9	10
반환시간	6	9	10	14

FCFS 스케줄링의 예

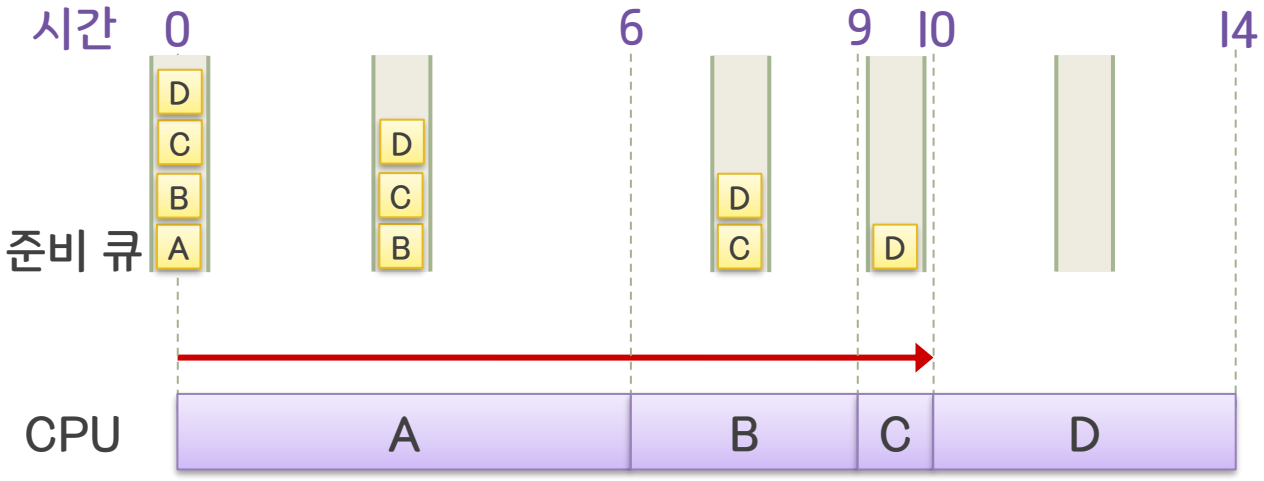
도착시간	0	0	0	0
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	0	6	9	10
반환시간	6	9	10	14

FCFS 스케줄링의 예

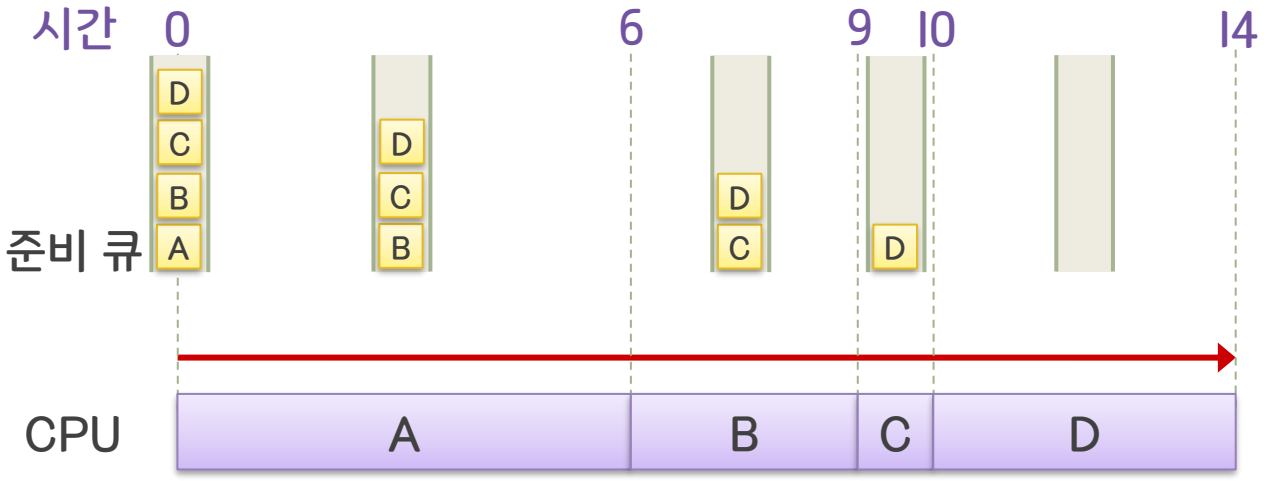
도착시간	0	0	0	0
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	0	6	9	10
반환시간	6	9	10	14

FCFS 스케줄링의 예

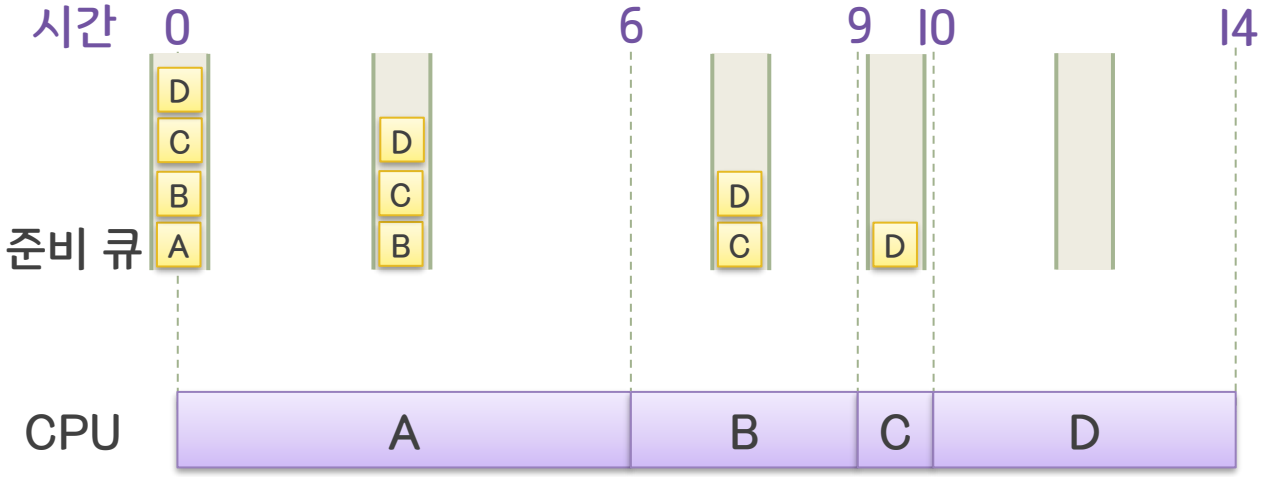
도착시간	0	0	0	0
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	0	6	9	10
반환시간	6	9	10	14

FCFS 스케줄링의 예

도착시간	0	0	0	0
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	0	6	9	10
반환시간	6	9	10	14

- 평균 대기시간 = $\frac{0+6+9+10}{4} = 6.25$
- 평균 반환시간 = $\frac{6+9+10+14}{4} = 9.75$

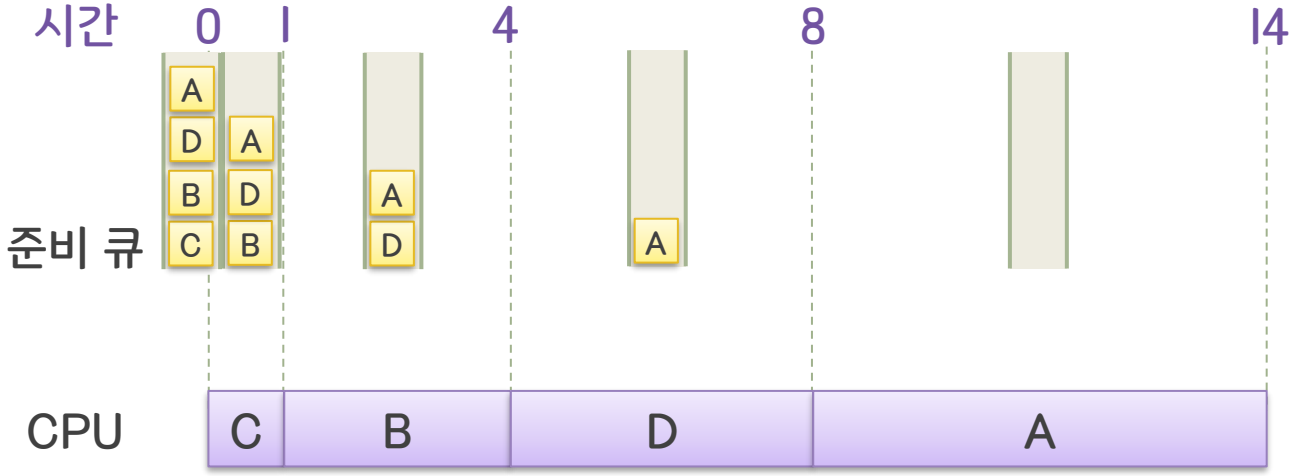
FCFS 스케줄링

- FCFS (First-Come First-Served) 스케줄링
  - 비선점 스케줄링 알고리즘
  - 준비 큐에 도착한 순서에 따라 디스패치
- 장점
  - 가장 간단한 스케줄링 기법
- 단점
  - 짧은 프로세스가 긴 프로세스를 기다리거나,
    중요한 프로세스가 나중에 수행될 수 있음
  - 프로세스들의 도착 순서에 따라 평균 반환시간이 크게 변함

FCFS 스케줄링의 예

도착시간	0	0	0	0
프로세스	A	B	C	D
CPU 사이클	6	3	1	4

- 도착 순서가 다른 경우

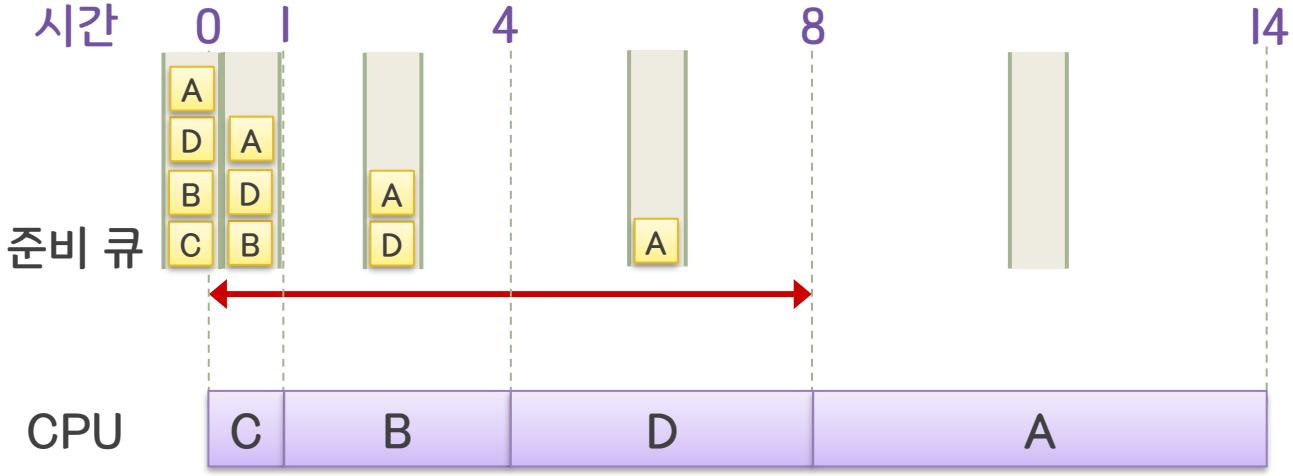


프로세스	A	B	C	D
대기시간	8	1	0	4
반환시간	14	4	1	8

FCFS 스케줄링의 예

도착시간	0	0	0	0
프로세스	A	B	C	D
CPU 사이클	6	3	1	4

- 도착 순서가 다른 경우

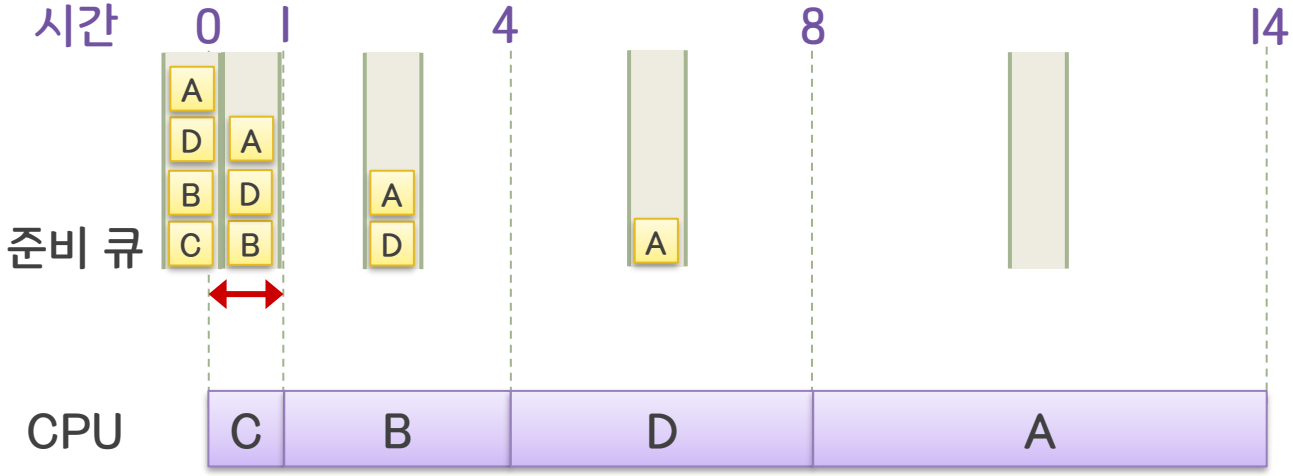


프로세스	A	B	C	D
대기시간	8	1	0	4
반환시간	14	4	1	8

FCFS 스케줄링의 예

도착시간	0	0	0	0
프로세스	A	B	C	D
CPU 사이클	6	3	1	4

- 도착 순서가 다른 경우

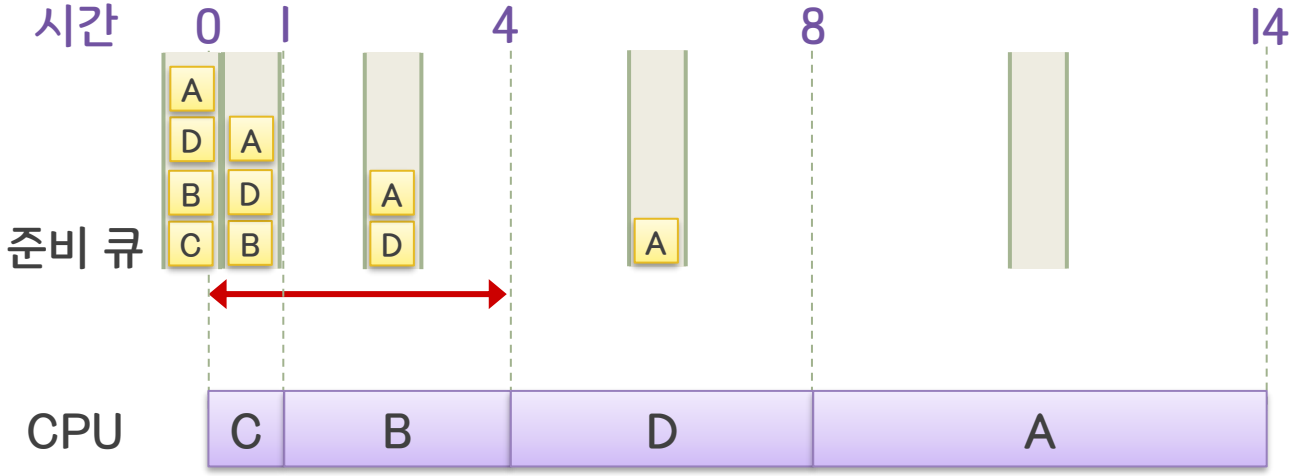


프로세스	A	B	C	D
대기시간	8	1	0	4
반환시간	14	4	1	8

FCFS 스케줄링의 예

도착시간	0	0	0	0
프로세스	A	B	C	D
CPU 사이클	6	3	1	4

- 도착 순서가 다른 경우

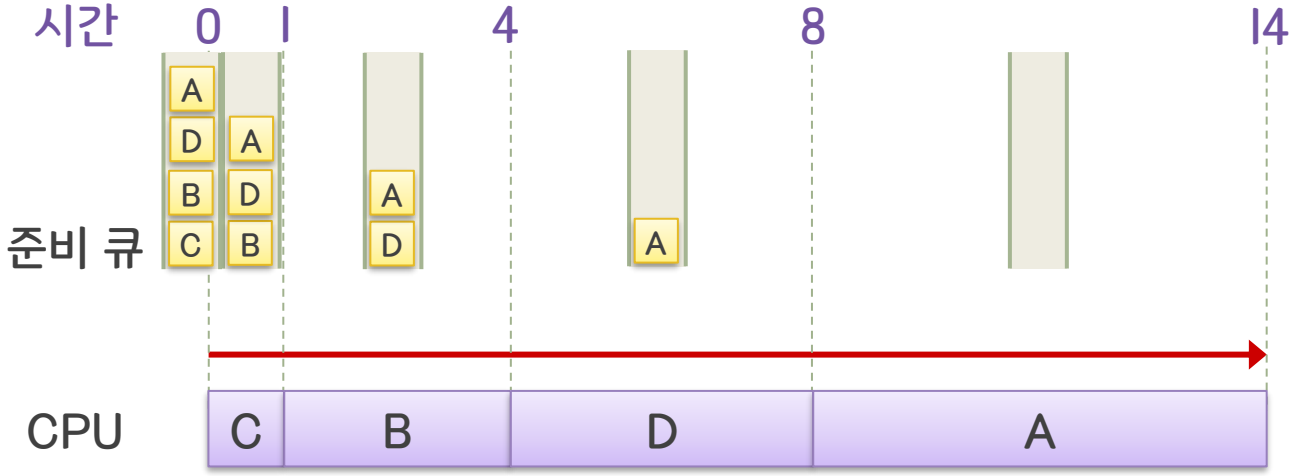


프로세스	A	B	C	D
대기시간	8	1	0	4
반환시간	14	4	1	8

FCFS 스케줄링의 예

도착시간	0	0	0	0
프로세스	A	B	C	D
CPU 사이클	6	3	1	4

- 도착 순서가 다른 경우

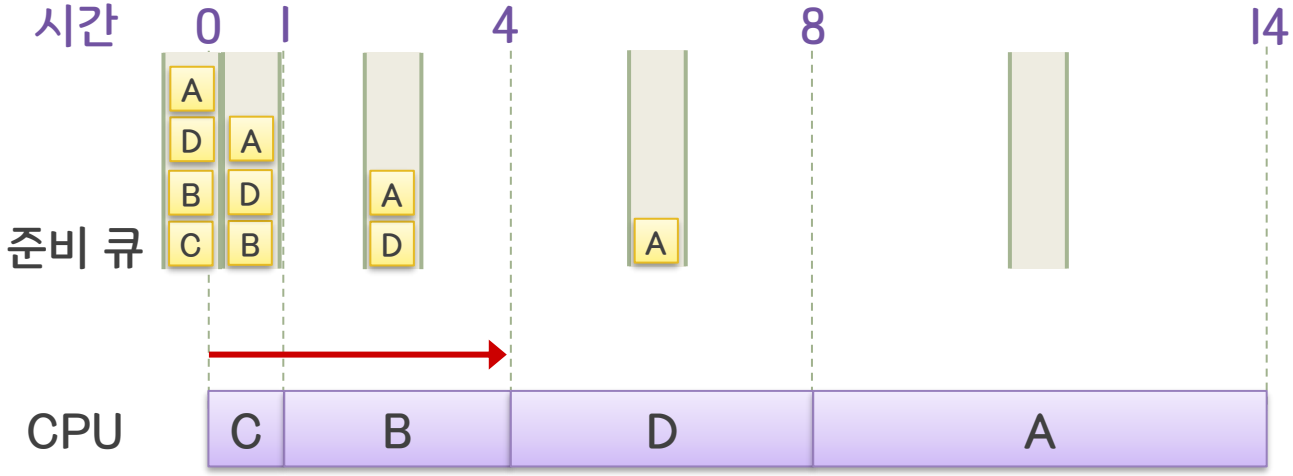


프로세스	A	B	C	D
대기시간	8	1	0	4
반환시간	14	4	1	8

FCFS 스케줄링의 예

도착시간	0	0	0	0
프로세스	A	B	C	D
CPU 사이클	6	3	1	4

- 도착 순서가 다른 경우

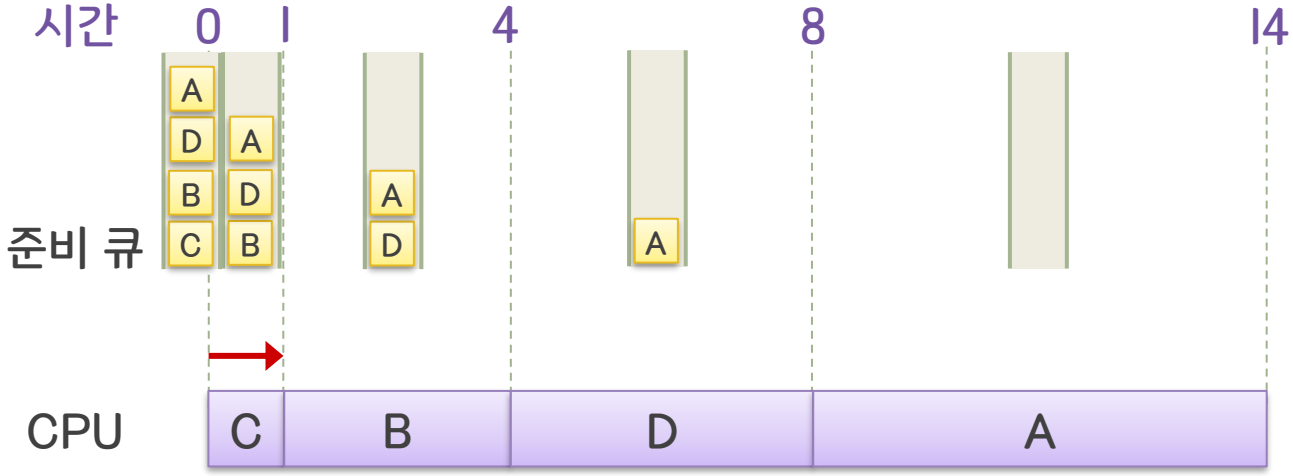


프로세스	A	B	C	D
대기시간	8	1	0	4
반환시간	14	4	1	8

FCFS 스케줄링의 예

도착시간	0	0	0	0
프로세스	A	B	C	D
CPU 사이클	6	3	1	4

- 도착 순서가 다른 경우

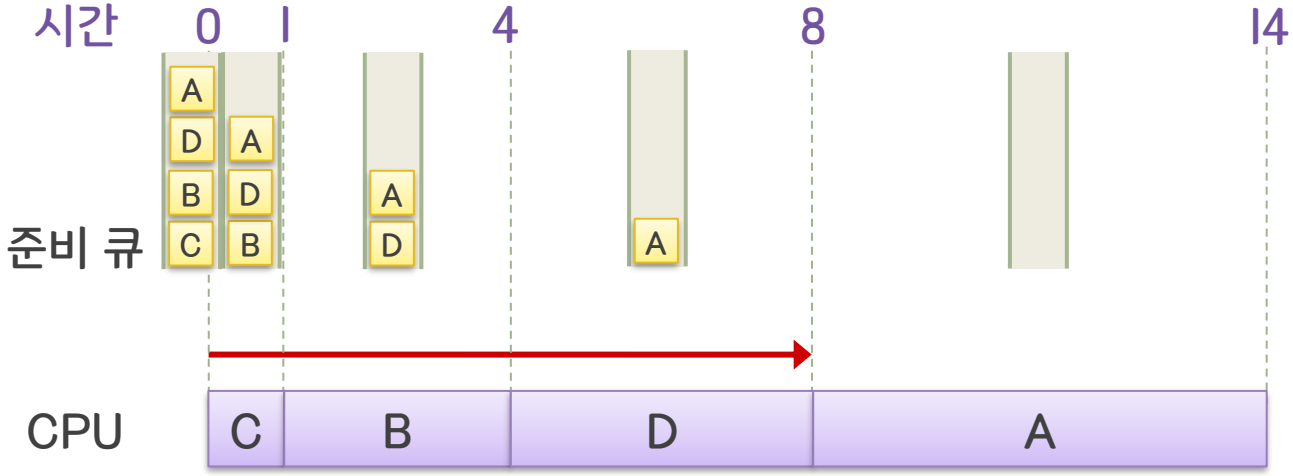


프로세스	A	B	C	D
대기시간	8	1	0	4
반환시간	14	4	1	8

FCFS 스케줄링의 예

도착시간	0	0	0	0
프로세스	A	B	C	D
CPU 사이클	6	3	1	4

- 도착 순서가 다른 경우

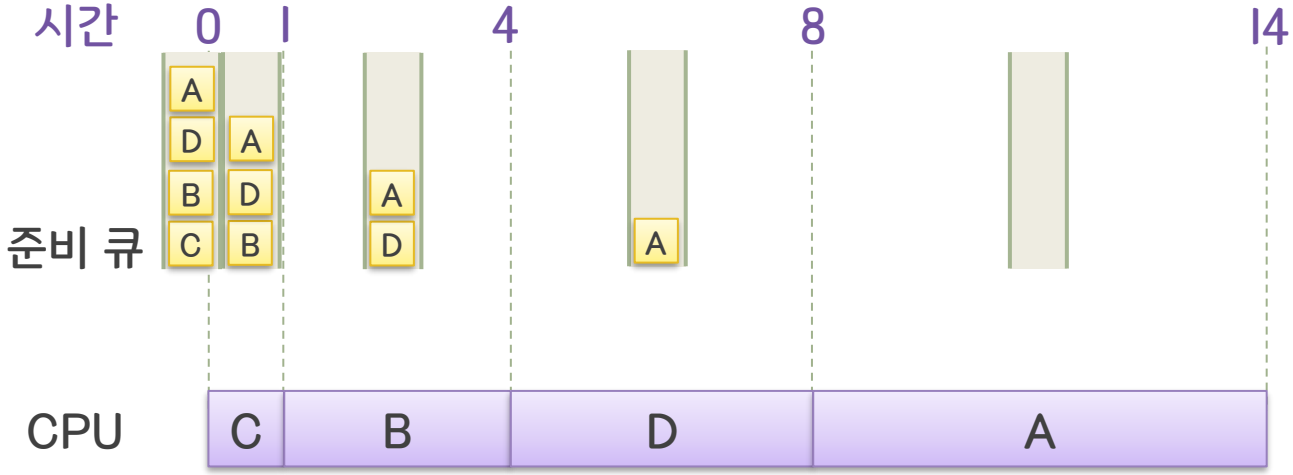


프로세스	A	B	C	D
대기시간	8	1	0	4
반환시간	14	4	1	8

FCFS 스케줄링의 예

도착시간	0	0	0	0
프로세스	A	B	C	D
CPU 사이클	6	3	1	4

- 도착 순서가 다른 경우



✪ 앞의 예(A,B,C,D순)

- 평균 대기시간=6.25
- 평균 반환시간=9.75

프로세스	A	B	C	D
대기시간	8	1	0	4
반환시간	14	4	1	8

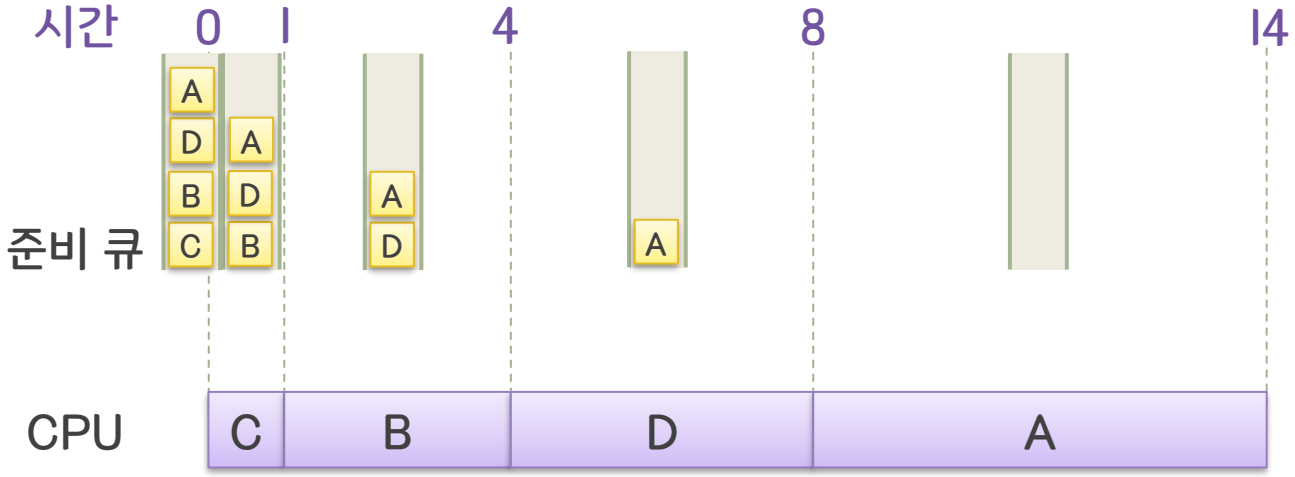
- 평균 대기시간 $= \frac{8+1+0+4}{4} = 3.25$
- 평균 반환시간 $= \frac{14+4+1+8}{4} = 6.75$

SJF 스케줄링

- SJF (Shortest Job First) 스케줄링
  - 비선점 스케줄링 알고리즘
  - 준비 큐에서 기다리는 프로세스 중 실행시간이 가장 짧다고 예상된 것을 먼저 디스패치

SJF 스케줄링의 예

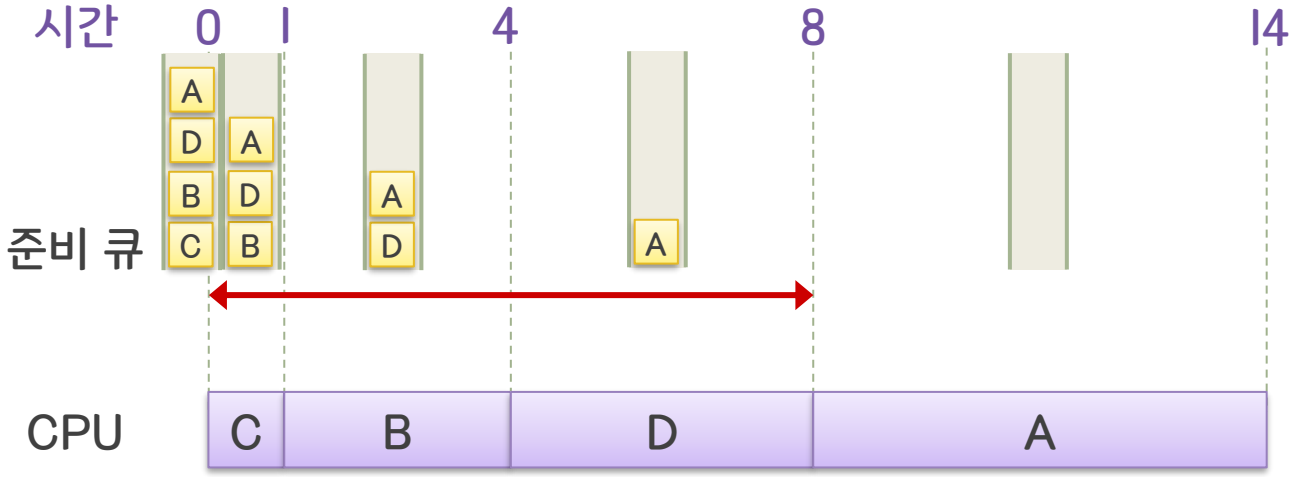
도착시간	0	0	0	0
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	8	1	0	4
반환시간	14	4	1	8

SJF 스케줄링의 예

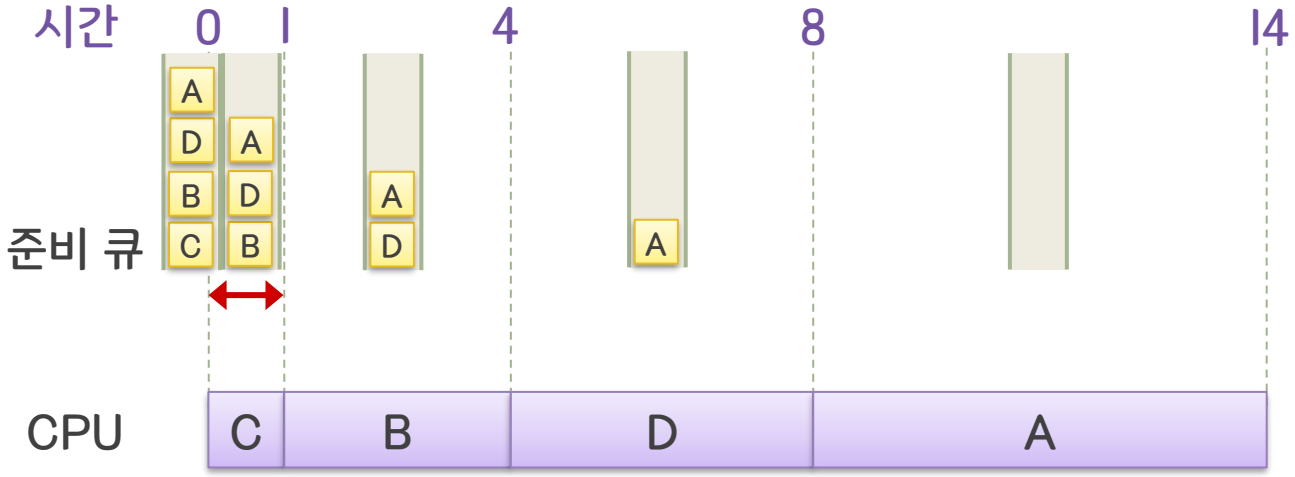
도착시간	0	0	0	0
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	8	1	0	4
반환시간	14	4	1	8

SJF 스케줄링의 예

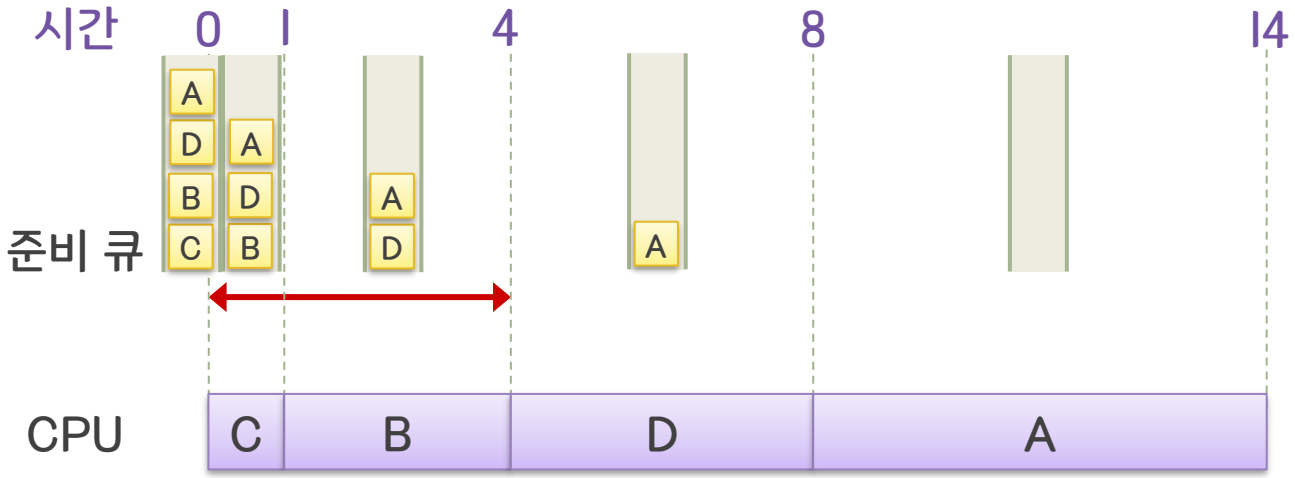
도착시간	0	0	0	0
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	8	1	0	4
반환시간	14	4	1	8

SJF 스케줄링의 예

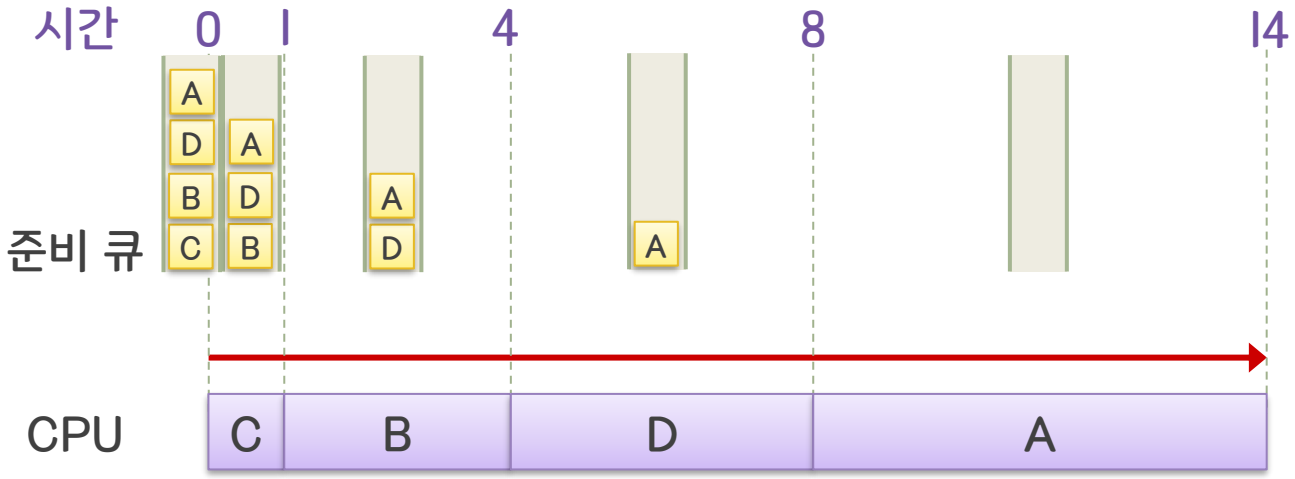
도착시간	0	0	0	0
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	8	1	0	4
반환시간	14	4	1	8

SJF 스케줄링의 예

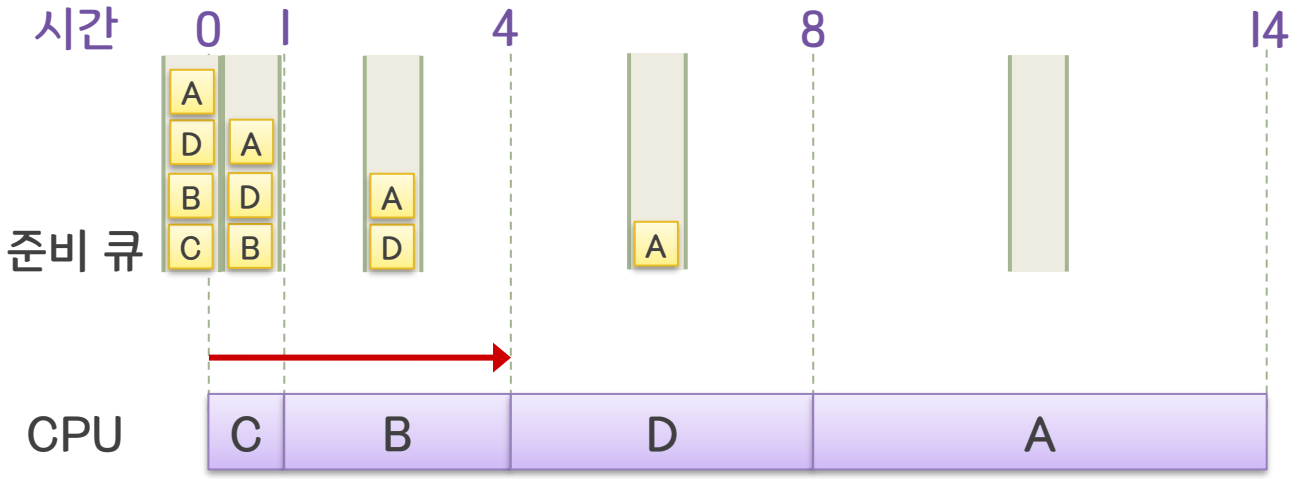
도착시간	0	0	0	0
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	8	1	0	4
반환시간	14	4	1	8

SJF 스케줄링의 예

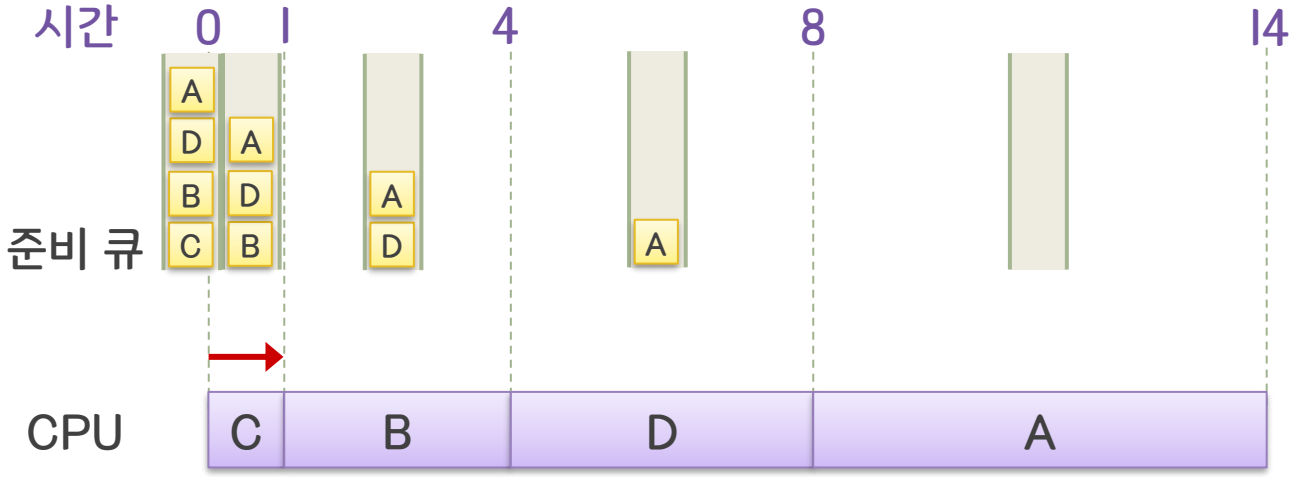
도착시간	0	0	0	0
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	8	1	0	4
반환시간	14	4	1	8

SJF 스케줄링의 예

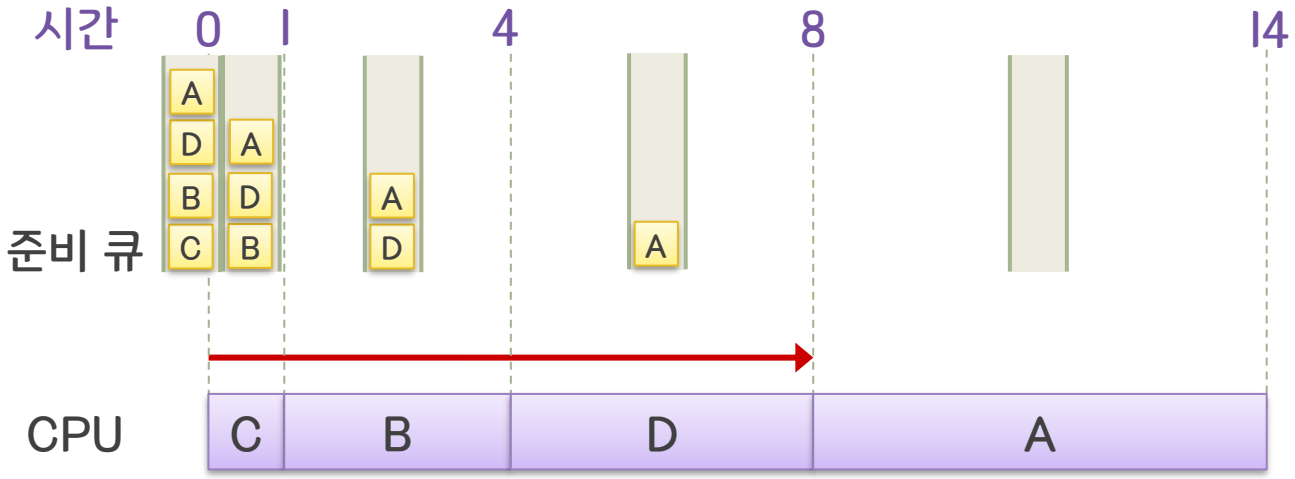
도착시간	0	0	0	0
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	8	1	0	4
반환시간	14	4	1	8

SJF 스케줄링의 예

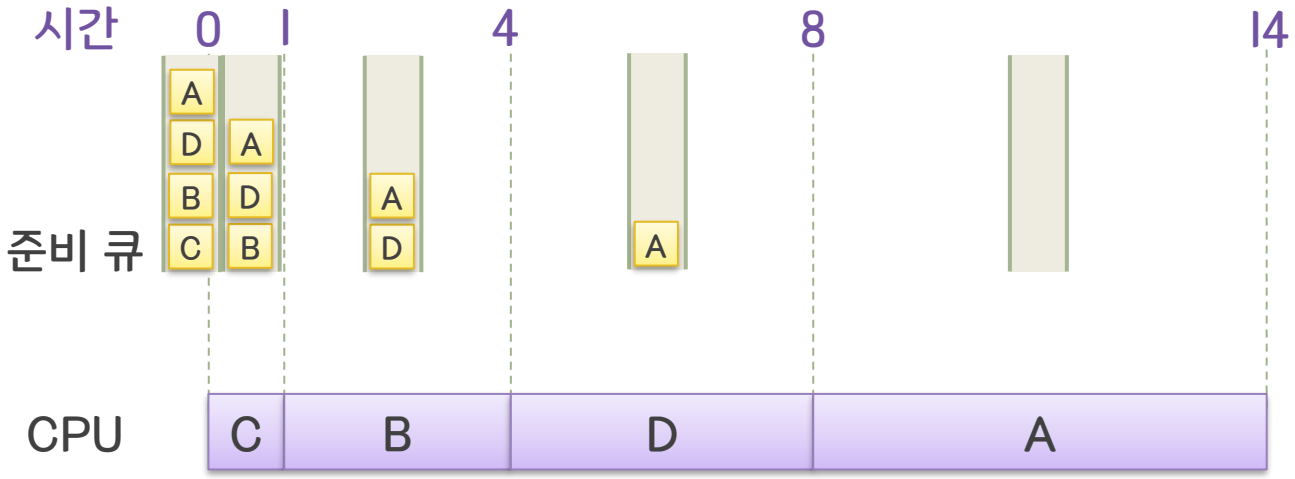
도착시간	0	0	0	0
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	8	1	0	4
반환시간	14	4	1	8

SJF 스케줄링의 예

도착시간	0	0	0	0
프로세스	A	B	C	D
CPU 사이클	6	3	1	4

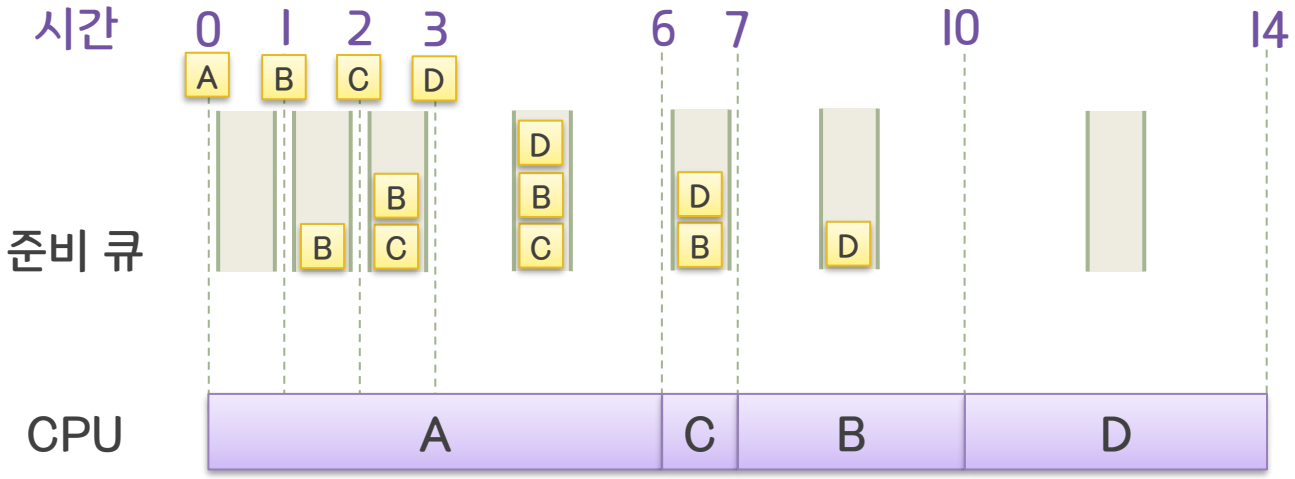


프로세스	A	B	C	D
대기시간	8	1	0	4
반환시간	14	4	1	8

- 평균 대기시간 $= \frac{8+1+0+4}{4} = 3.25$
- 평균 반환시간 $= \frac{14+4+1+8}{4} = 6.75$

SJF 스케줄링의 예

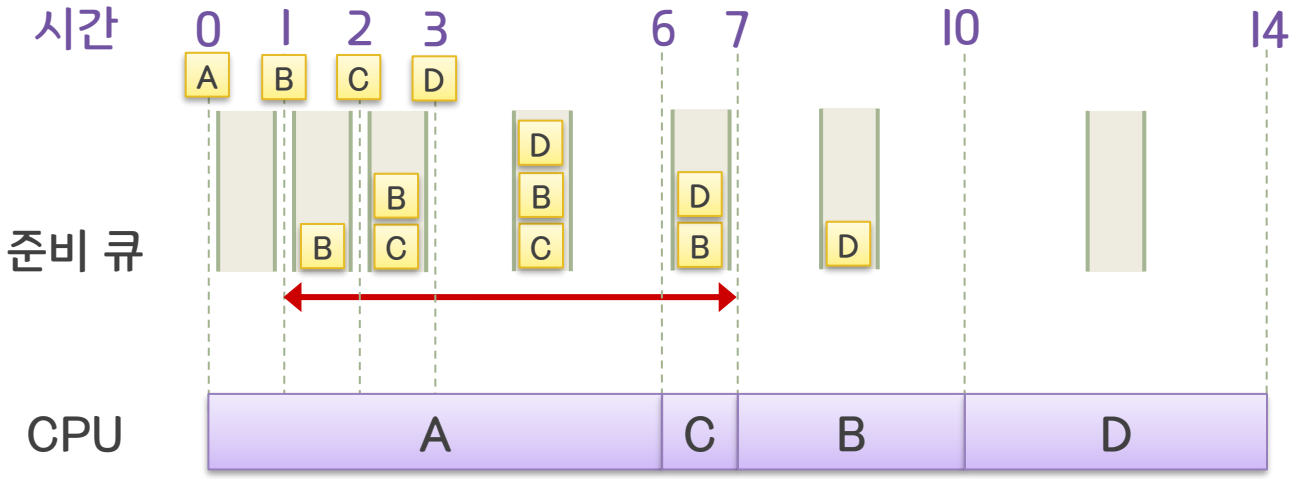
도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	0	6	4	7
반환시간	6	9	5	11

SJF 스케줄링의 예

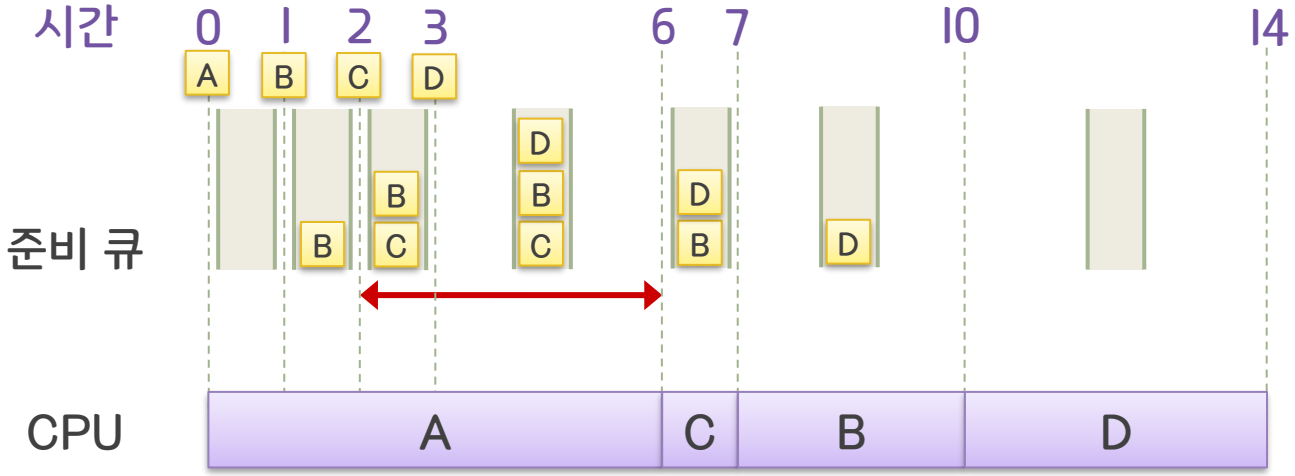
도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	0	6	4	7
반환시간	6	9	5	11

SJF 스케줄링의 예

도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4

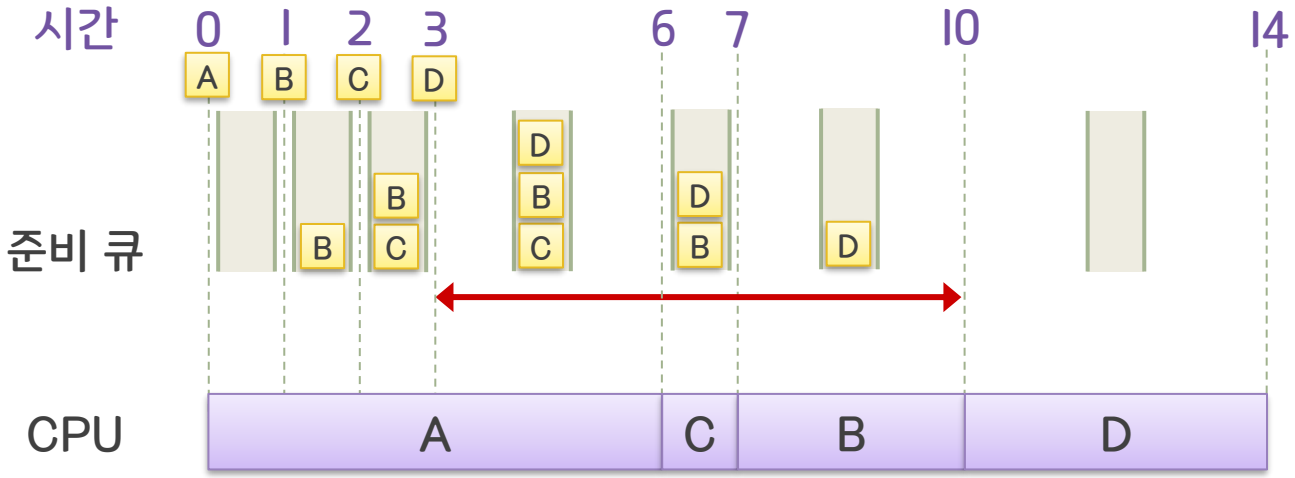


프로세스	A	B	C	D
대기시간	0	6	4	7
반환시간	6	9	5	11

- 평균 대기시간 $= \frac{0+6+4+7}{4} = 4.25$
- 평균 반환시간 $= \frac{6+9+5+11}{4} = 7.75$

SJF 스케줄링의 예

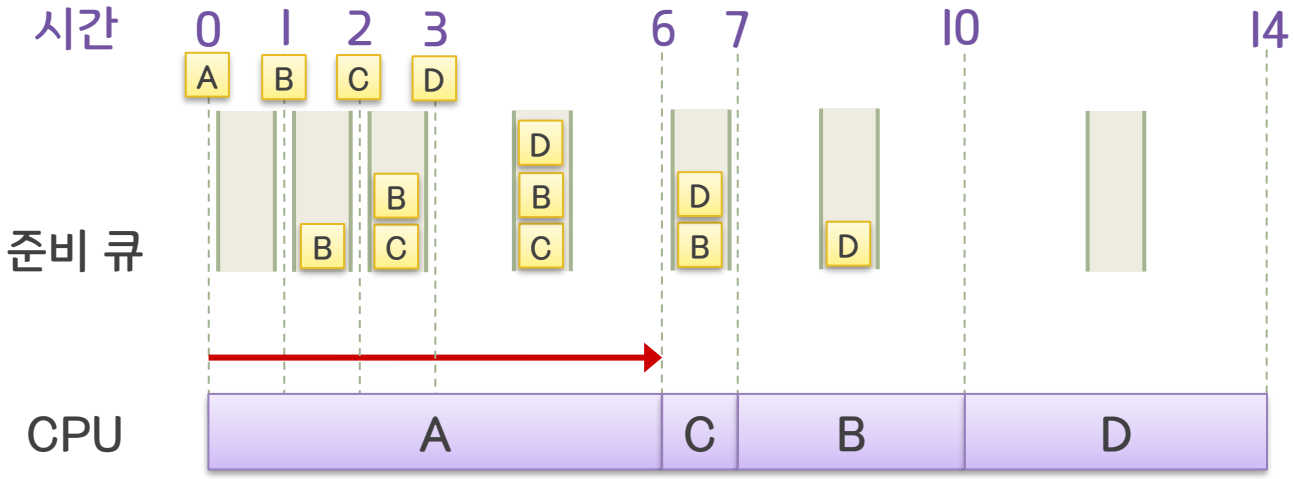
도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	0	6	4	7
반환시간	6	9	5	11

SJF 스케줄링의 예

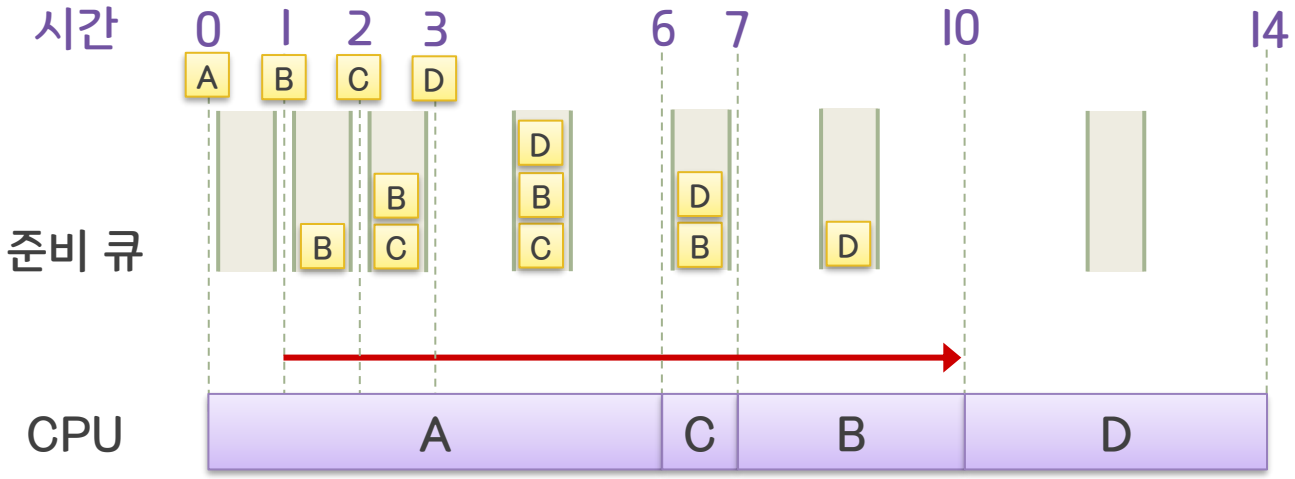
도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	0	6	4	7
반환시간	6	9	5	11

SJF 스케줄링의 예

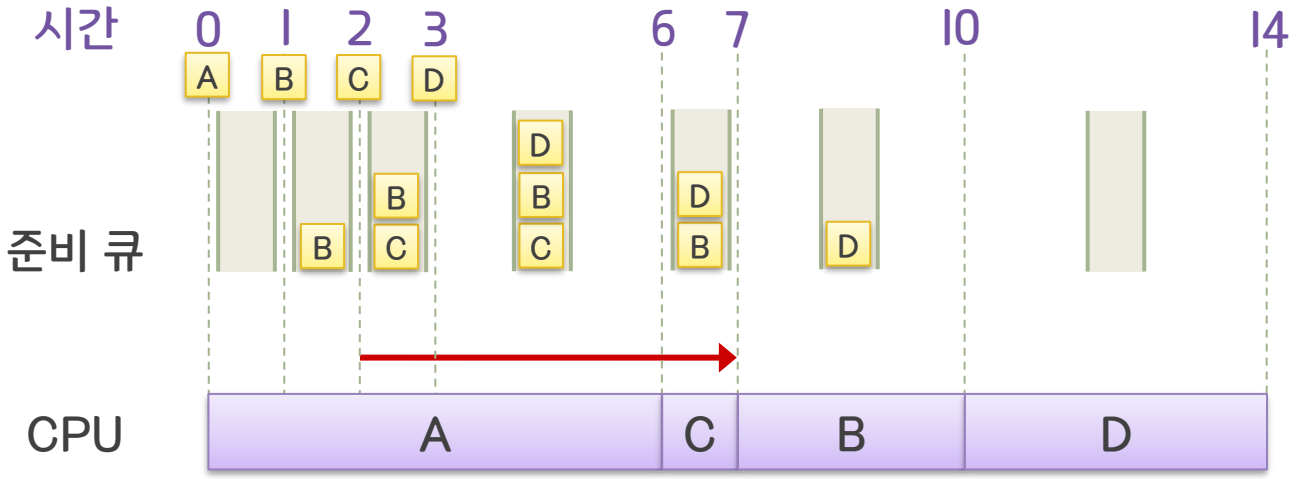
도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	0	6	4	7
반환시간	6	9	5	11

SJF 스케줄링의 예

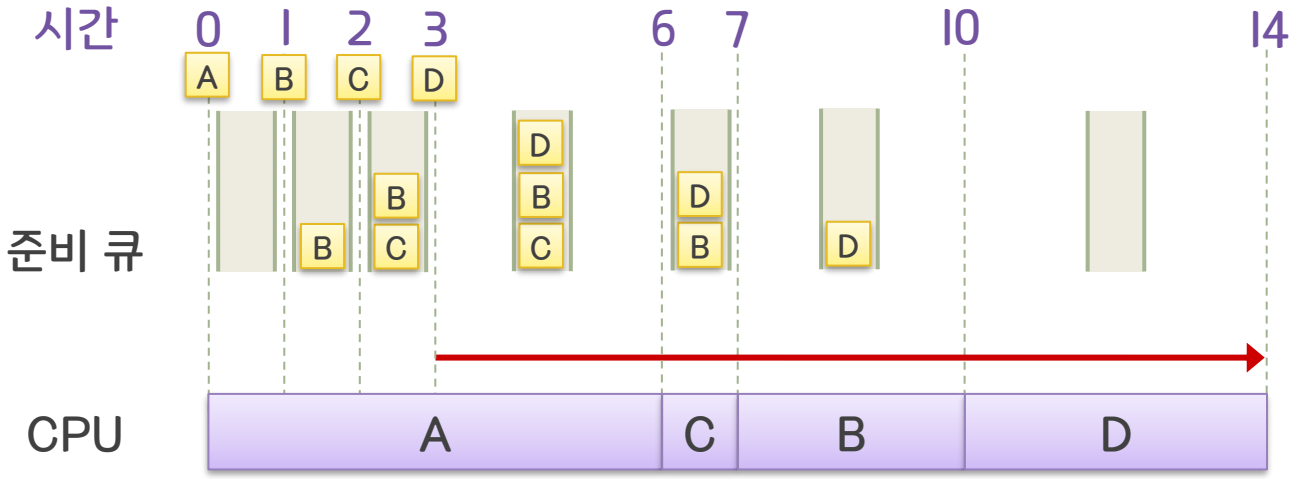
도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	0	6	4	7
반환시간	6	9	5	11

SJF 스케줄링의 예

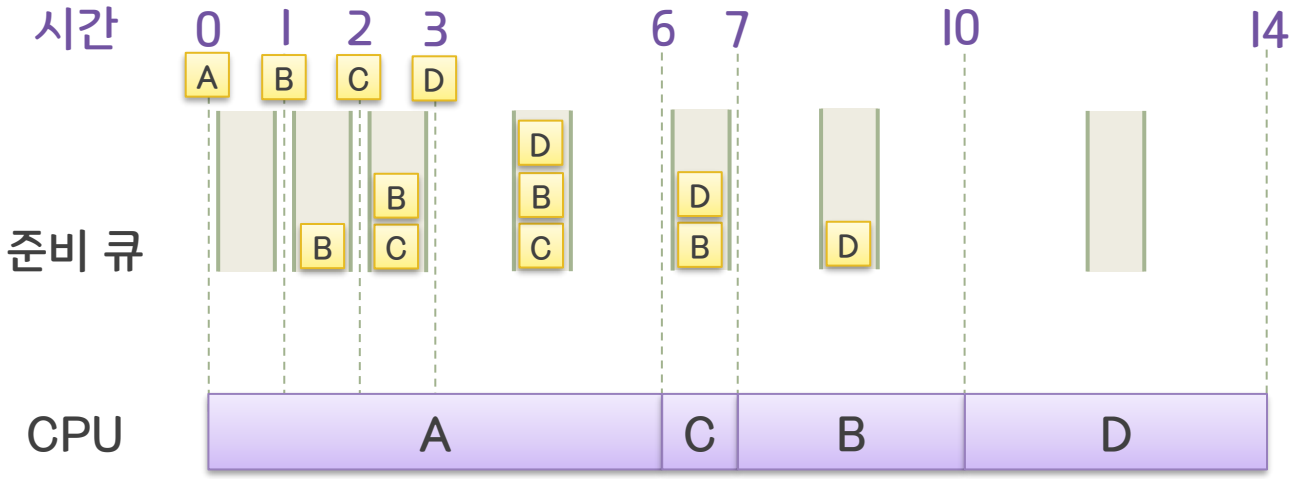
도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	0	6	4	7
반환시간	6	9	5	11

SJF 스케줄링의 예

도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	0	6	4	7
반환시간	6	9	5	11

- 평균 대기시간 $= \frac{0+6+4+7}{4} = 4.25$
- 평균 반환시간 $= \frac{6+9+5+11}{4} = 7.75$

SJF 스케줄링

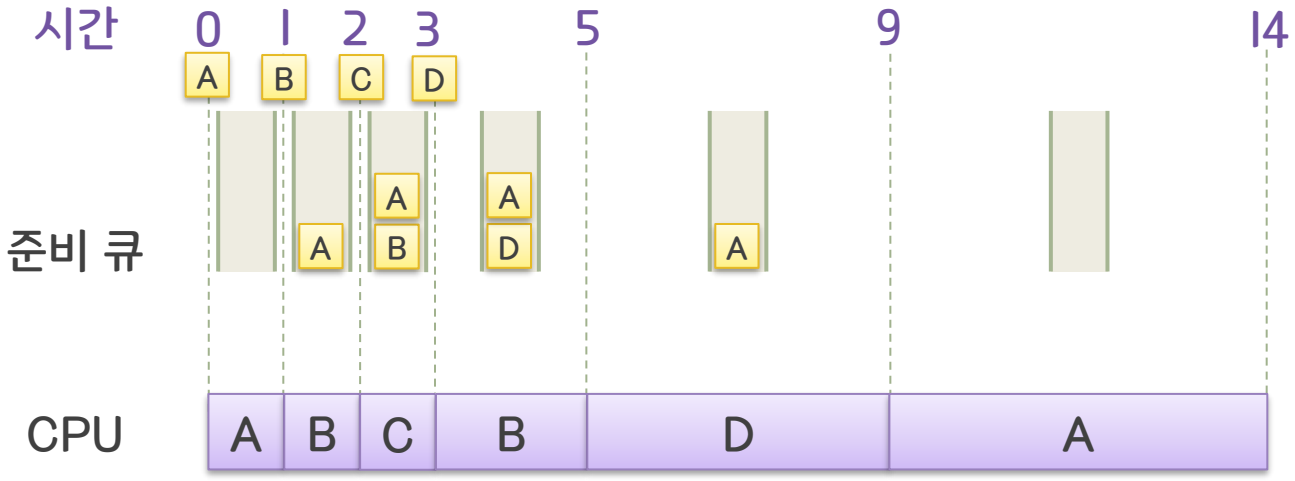
- SJF (Shortest Job First) 스케줄링
  - 비선점 스케줄링 알고리즘
  - 준비 큐에서 기다리는 프로세스 중 실행시간이 가장 짧다고 예상된 것을 먼저 디스패치
- 장점
  - 일괄처리 환경에서 구현하기 쉬움
- 단점
  - 실행 예정 시간 길이를 사용자의 추정치에 의존하기 때문에 실제로는 먼저 처리할 작업의 CPU 시간을 예상할 수 없음

SRT 스케줄링

- SRT (Shortest Remaining Time) 스케줄링
  - 선점 스케줄링 알고리즘
  - 실행이 끝날 때까지 남은 시간 추정치가 가장 짧은 프로세스를 먼저 디스패치

SRT 스케줄링의 예

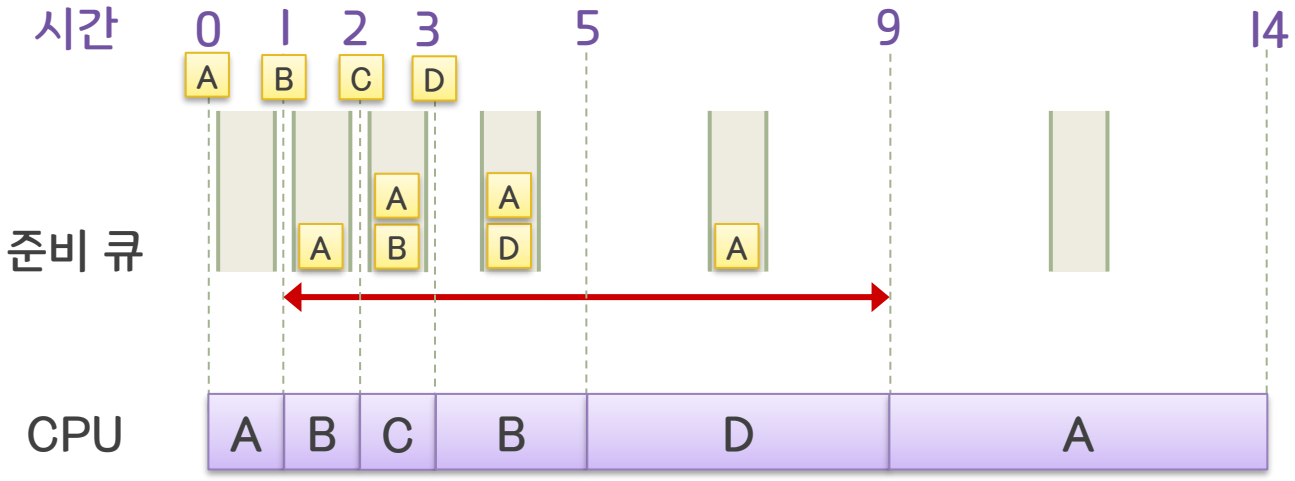
도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	8	1	0	2
반환시간	14	4	1	6

SRT 스케줄링의 예

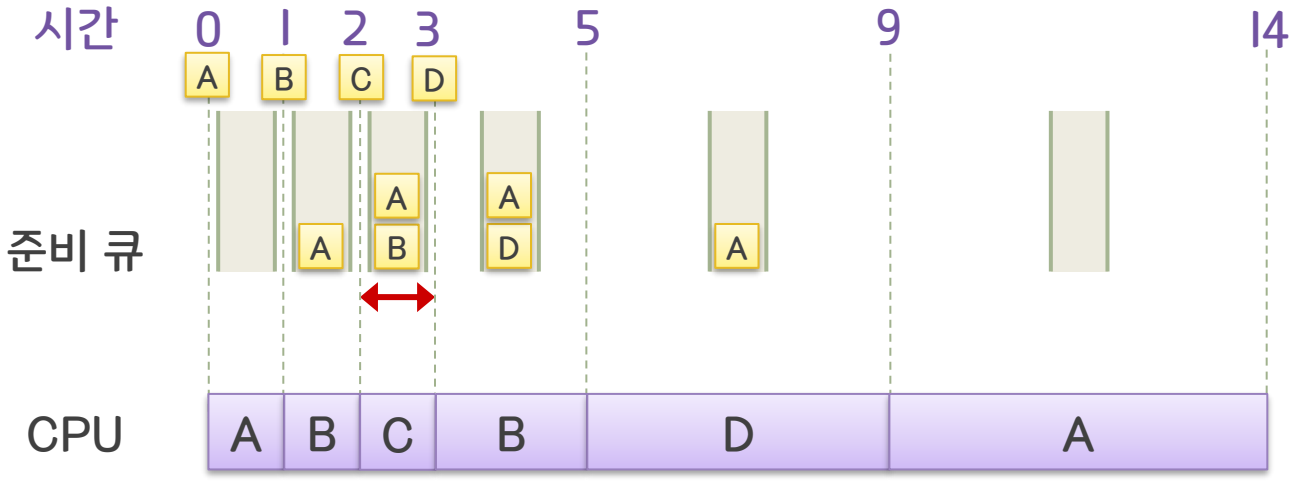
도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	8	1	0	2
반환시간	14	4	1	6

SRT 스케줄링의 예

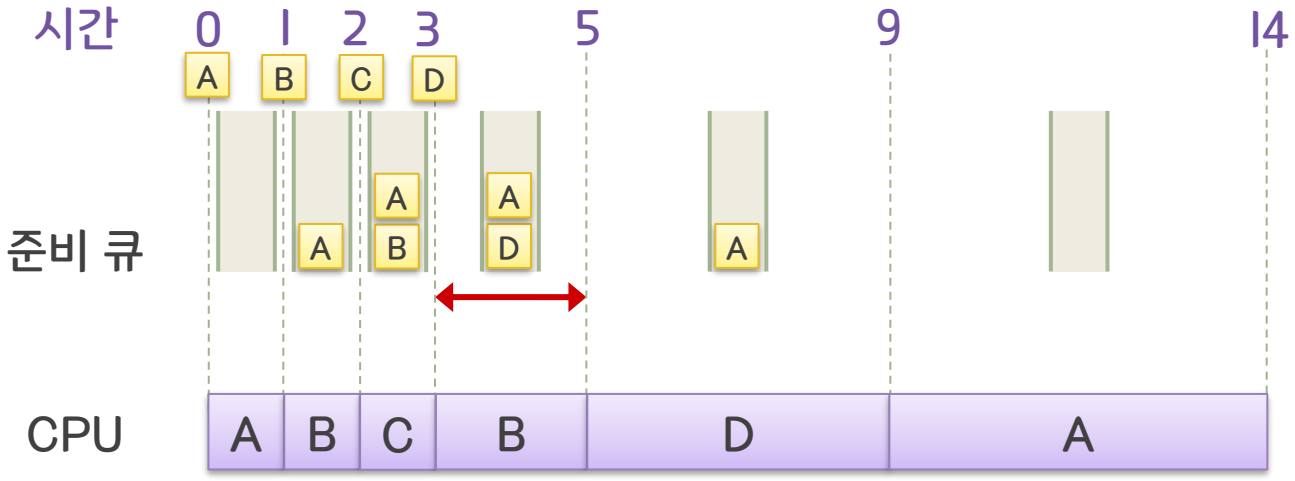
도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	8	1	0	2
반환시간	14	4	1	6

SRT 스케줄링의 예

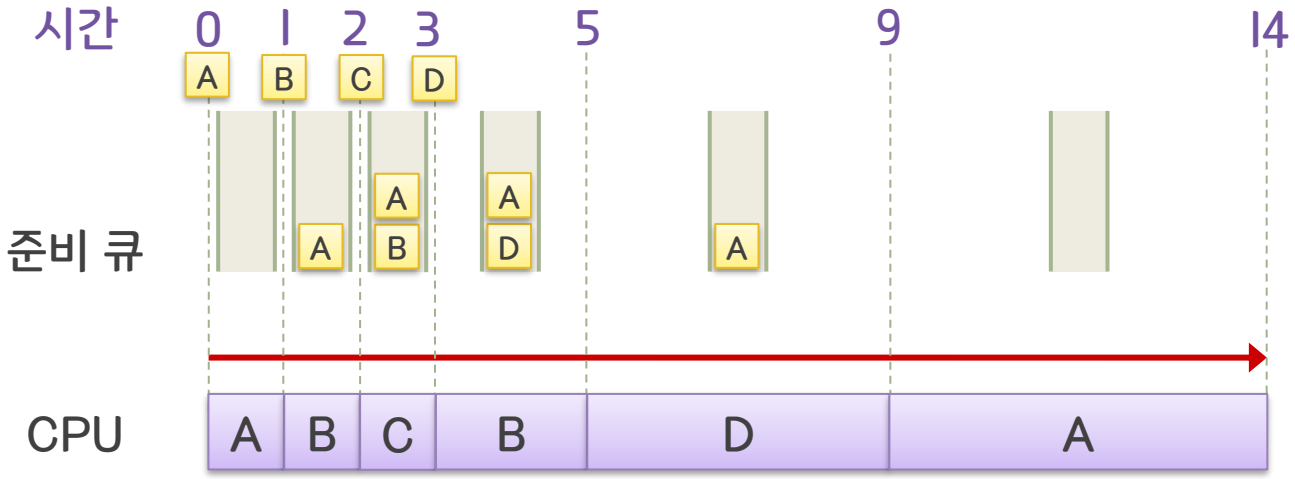
도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	8	1	0	2
반환시간	14	4	1	6

SRT 스케줄링의 예

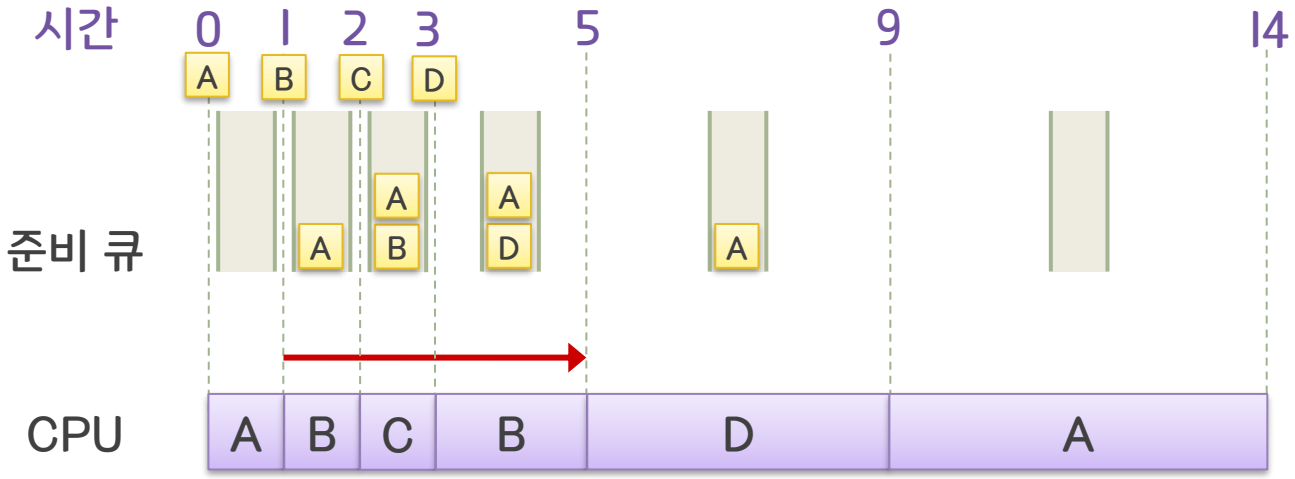
도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	8	1	0	2
반환시간	14	4	1	6

SRT 스케줄링의 예

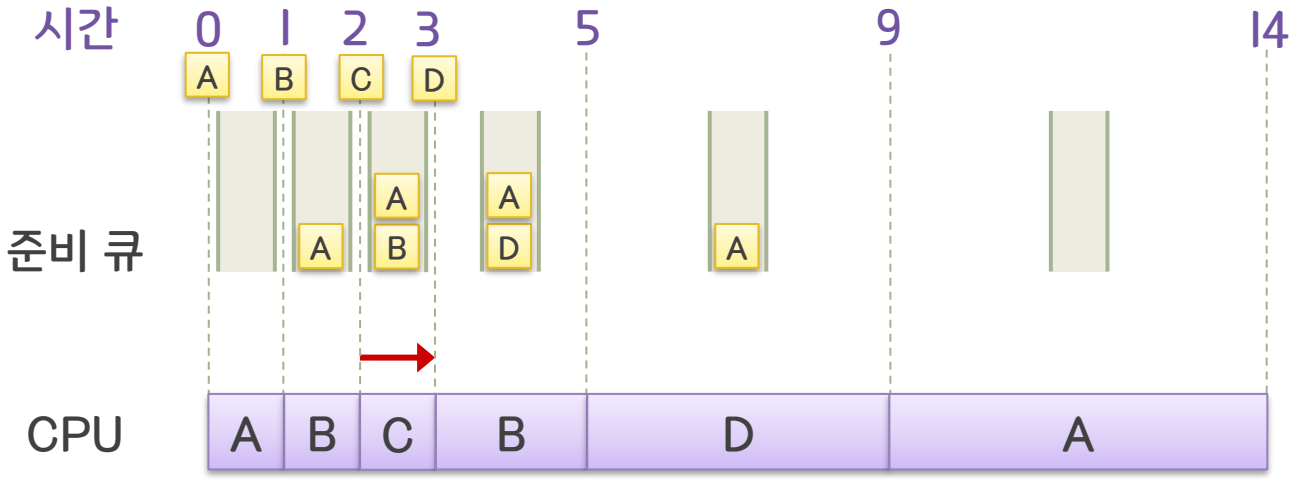
도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	8	1	0	2
반환시간	14	4	1	6

SRT 스케줄링의 예

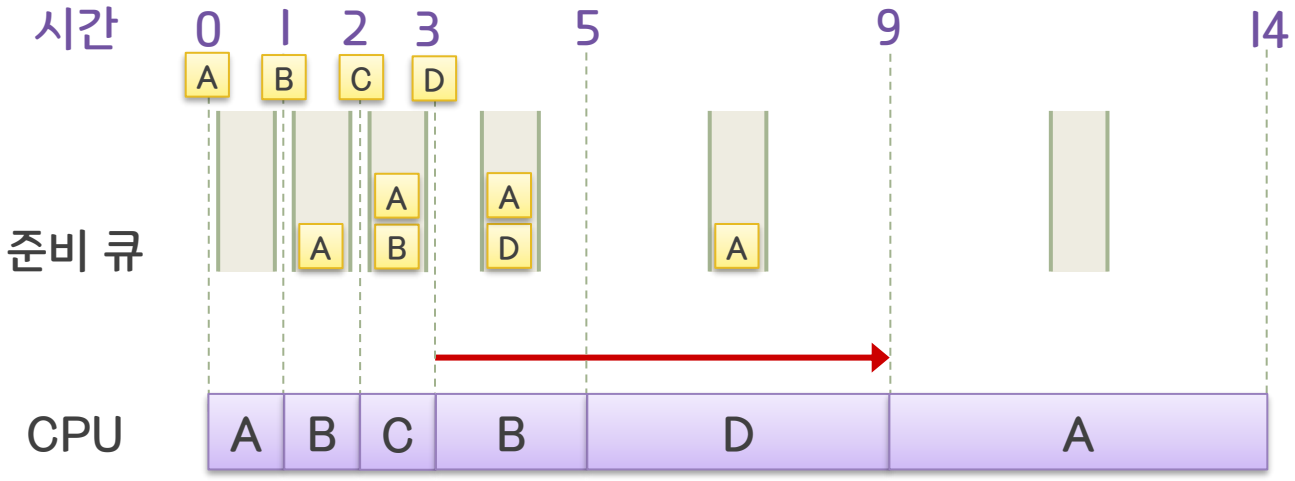
도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	8	1	0	2
반환시간	14	4	1	6

SRT 스케줄링의 예

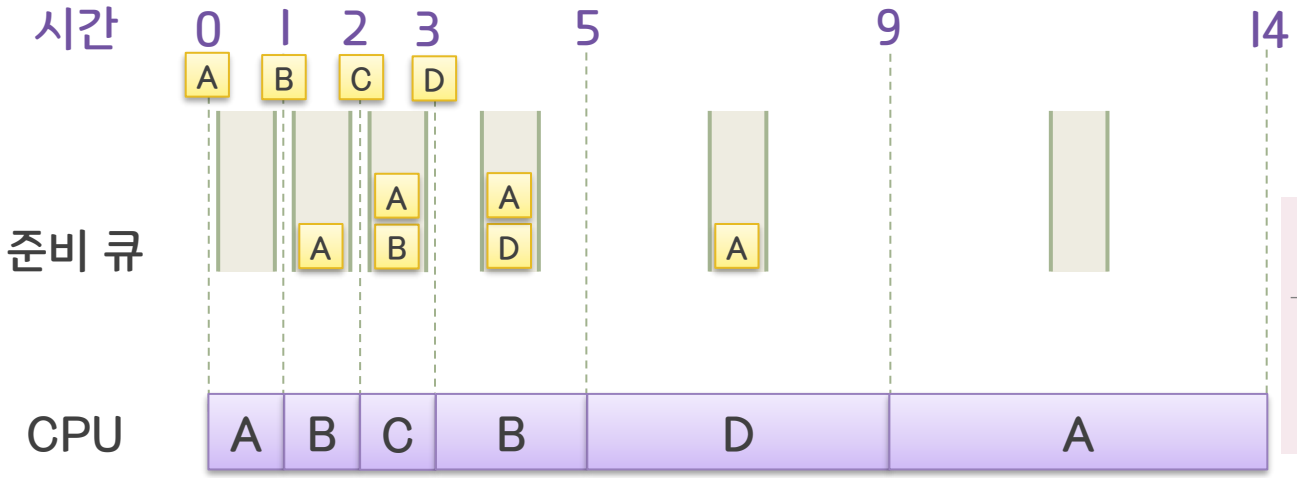
도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



프로세스	A	B	C	D
대기시간	8	1	0	2
반환시간	14	4	1	6

SRT 스케줄링의 예

도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



SJF 스케줄링

- 평균 대기시간=4.25
- 평균 반환시간=7.75

프로세스	A	B	C	D
대기시간	8	1	0	2
반환시간	14	4	1	6

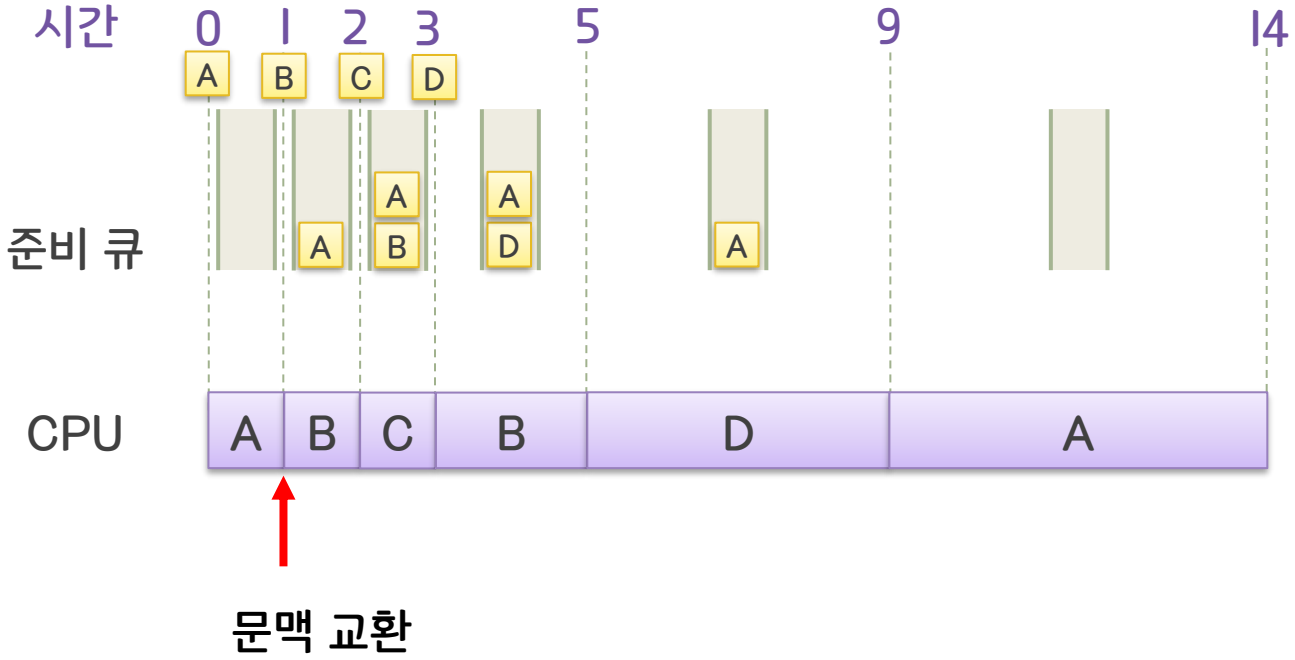
- 평균 대기시간 $= \frac{8+1+0+2}{4} = 2.75$
- 평균 반환시간 $= \frac{14+4+1+6}{4} = 6.25$

SRT 스케줄링

- SRT (Shortest Remaining Time) 스케줄링
  - 선점 스케줄링 알고리즘
  - 실행이 끝날 때까지 남은 시간 추정치가 가장 짧은 프로세스를 먼저 디스패치
- 장점
  - SJF보다 평균 대기시간이나 평균 반환시간에서 효율적
  - 대화형 운영체제에 유용
- 단점
  - 각 프로세스의 실행시간 추적, 선점을 위한 문맥 교환 등 SJF보다 오버헤드가 큼

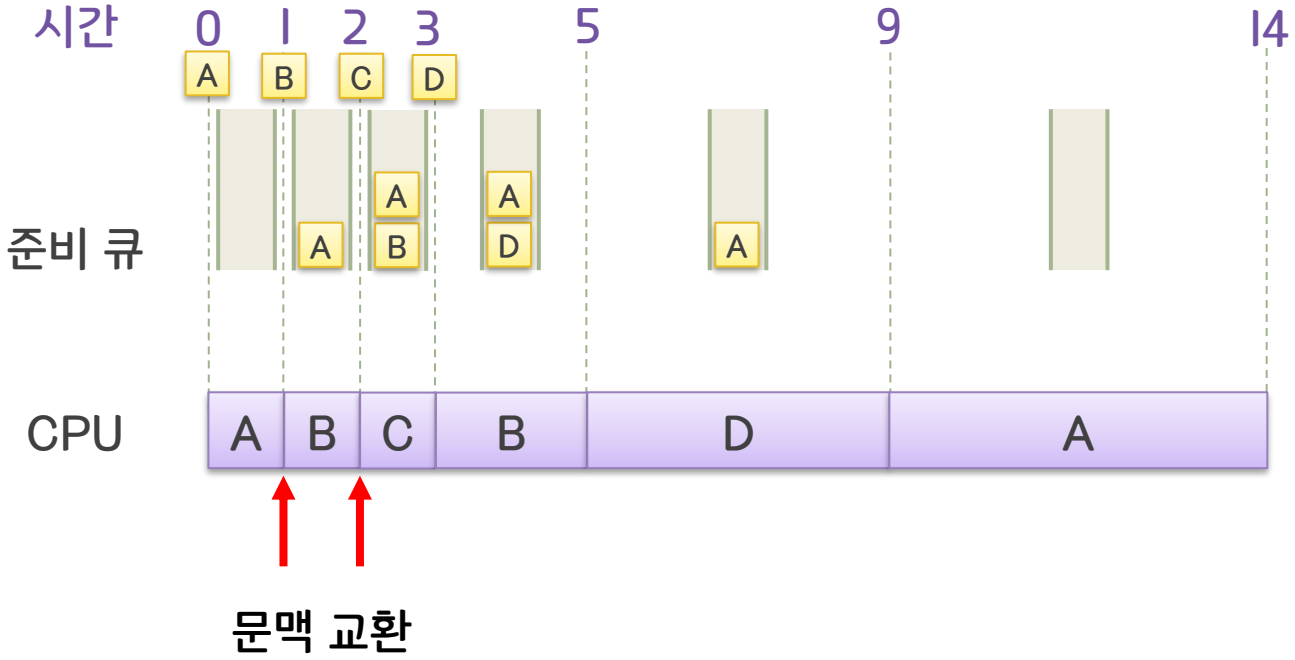
SRT 스케줄링의 예

도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



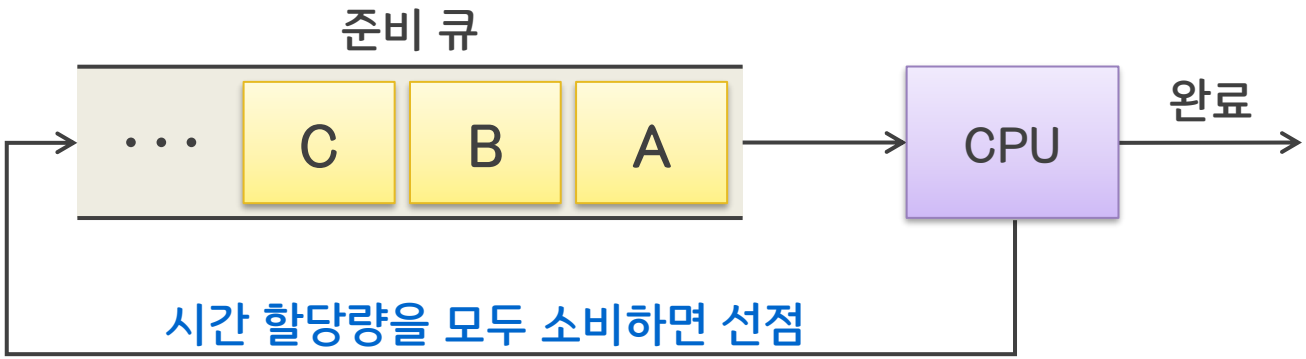
SRT 스케줄링의 예

도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4



RR 스케줄링

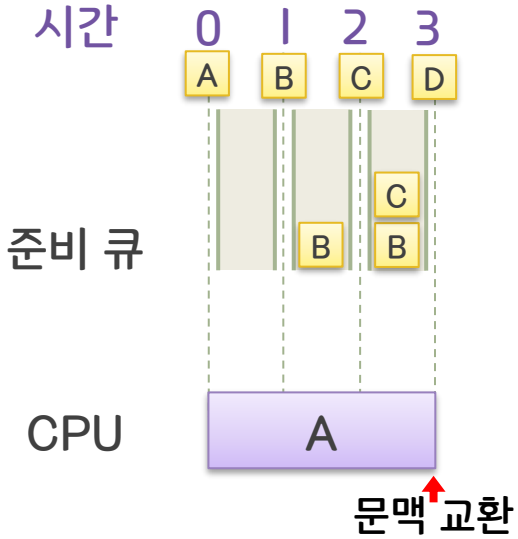
- RR (Round Robin) 스케줄링
  - 선점 스케줄링 알고리즘
  - 준비 큐에 도착한 순서에 따라 디스패치하지만 정해진 시간 할당량에 의해 실행을 제한
  - 시간 할당량 안에 완료되지 못한 프로세스는 준비 큐의 맨 뒤에 배치



RR 스케줄링의 예

도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4

- 시간 할당량 = 3

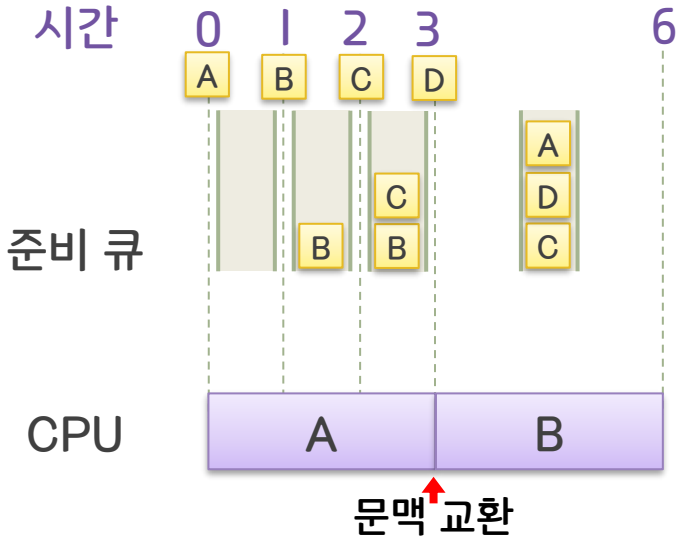


프로세스	A	B	C	D
대기시간	7	2	4	7
반환시간	13	5	5	11

RR 스케줄링의 예

도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4

- 시간 할당량 = 3

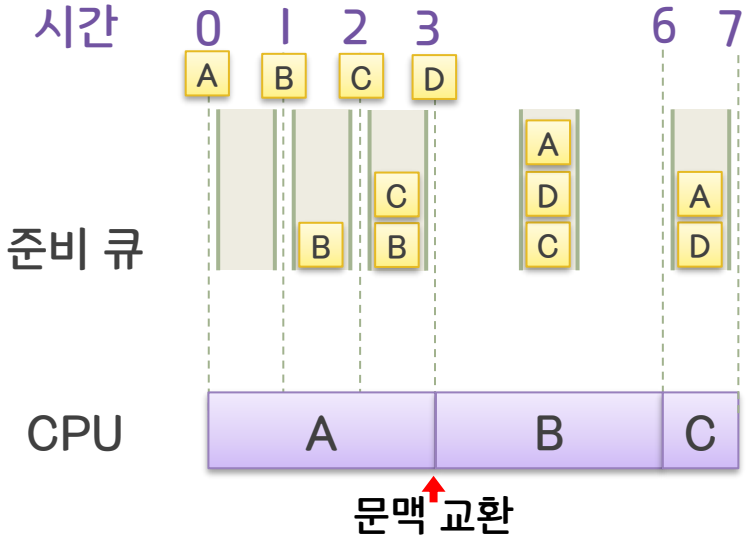


프로세스	A	B	C	D
대기시간	7	2	4	7
반환시간	13	5	5	11

RR 스케줄링의 예

도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4

- 시간 할당량 = 3

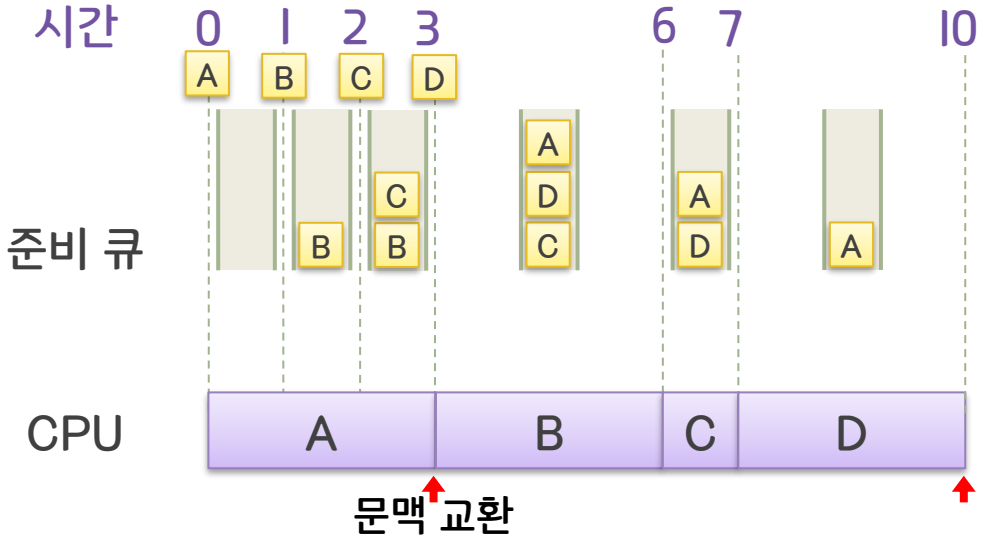


프로세스	A	B	C	D
대기시간	7	2	4	7
반환시간	13	5	5	11

RR 스케줄링의 예

도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4

- 시간 할당량 = 3

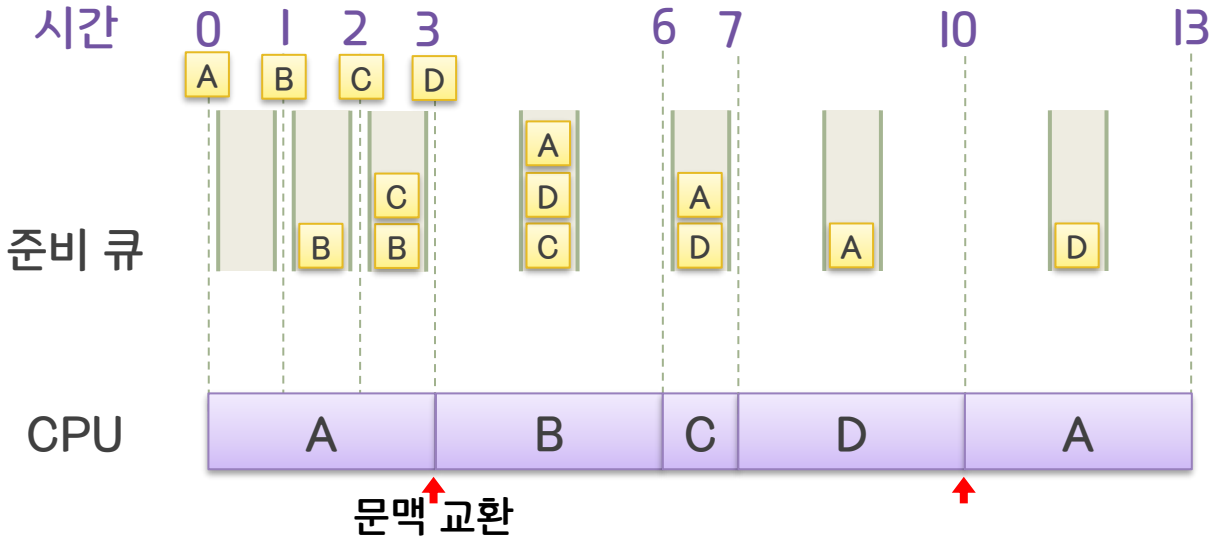


프로세스	A	B	C	D
대기시간	7	2	4	7
반환시간	13	5	5	11

RR 스케줄링의 예

도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4

- 시간 할당량 = 3

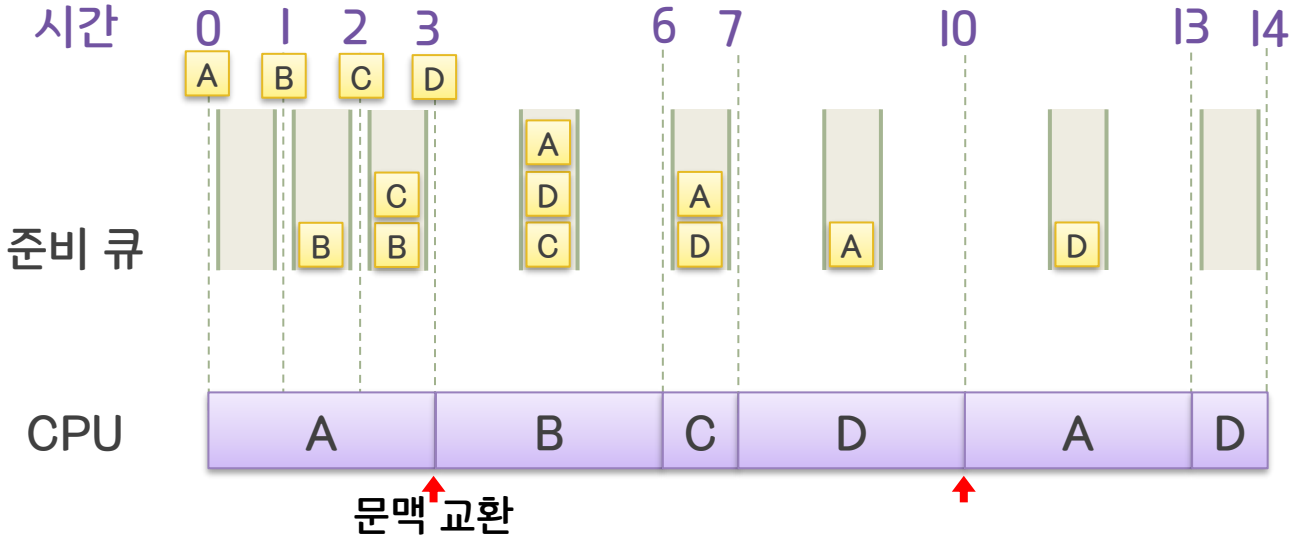


프로세스	A	B	C	D
대기시간	7	2	4	7
반환시간	13	5	5	11

RR 스케줄링의 예

도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4

- 시간 할당량 = 3

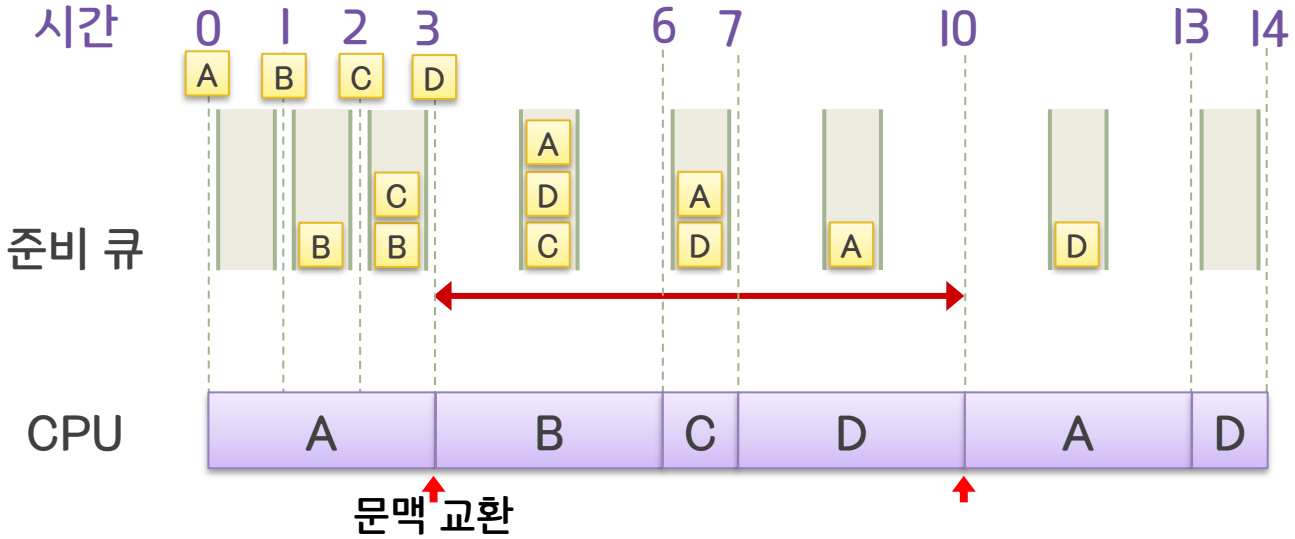


프로세스	A	B	C	D
대기시간	7	2	4	7
반환시간	13	5	5	11

RR 스케줄링의 예

도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4

- 시간 할당량 = 3

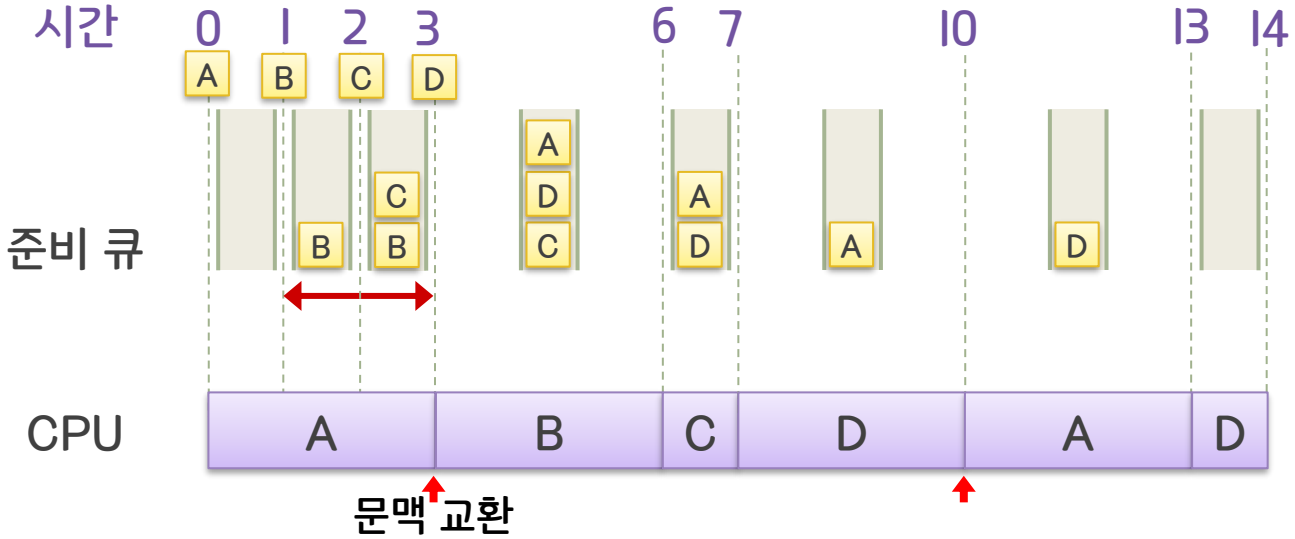


프로세스	A	B	C	D
대기시간	7	2	4	7
반환시간	13	5	5	11

RR 스케줄링의 예

도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4

- 시간 할당량 = 3

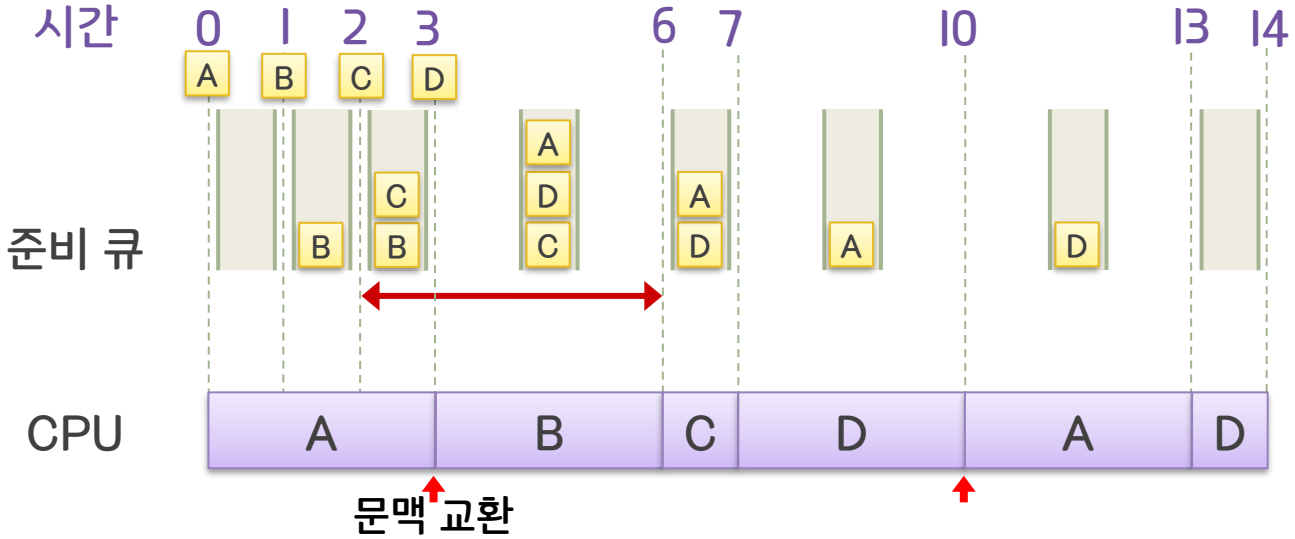


프로세스	A	B	C	D
대기시간	7	2	4	7
반환시간	13	5	5	11

RR 스케줄링의 예

도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4

- 시간 할당량 = 3

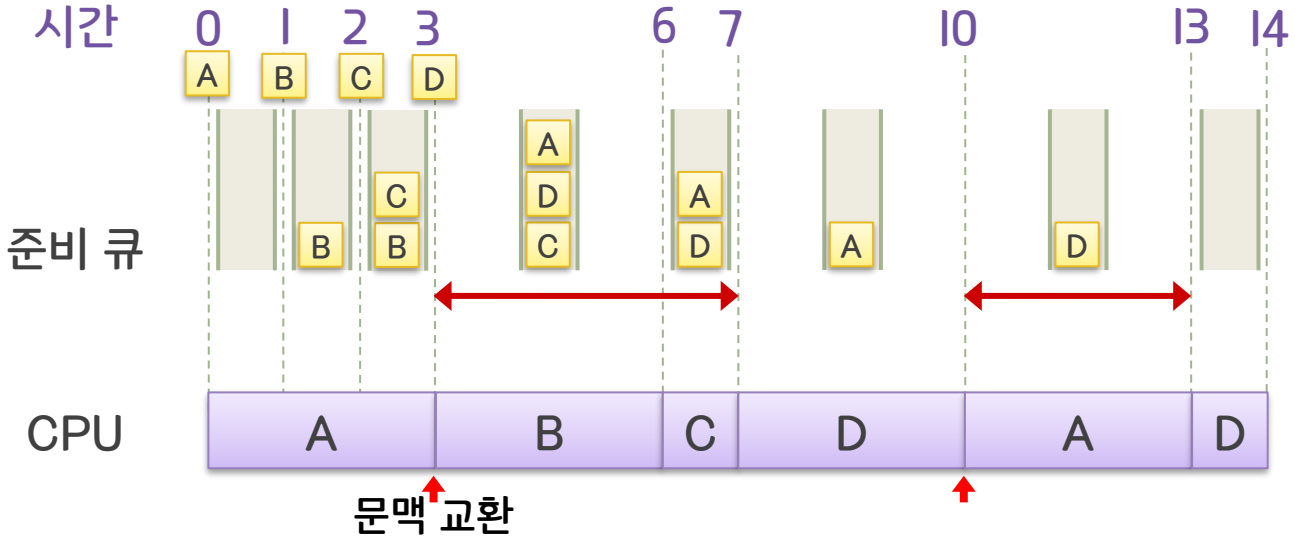


프로세스	A	B	C	D
대기시간	7	2	4	7
반환시간	13	5	5	11

RR 스케줄링의 예

도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4

- 시간 할당량 = 3

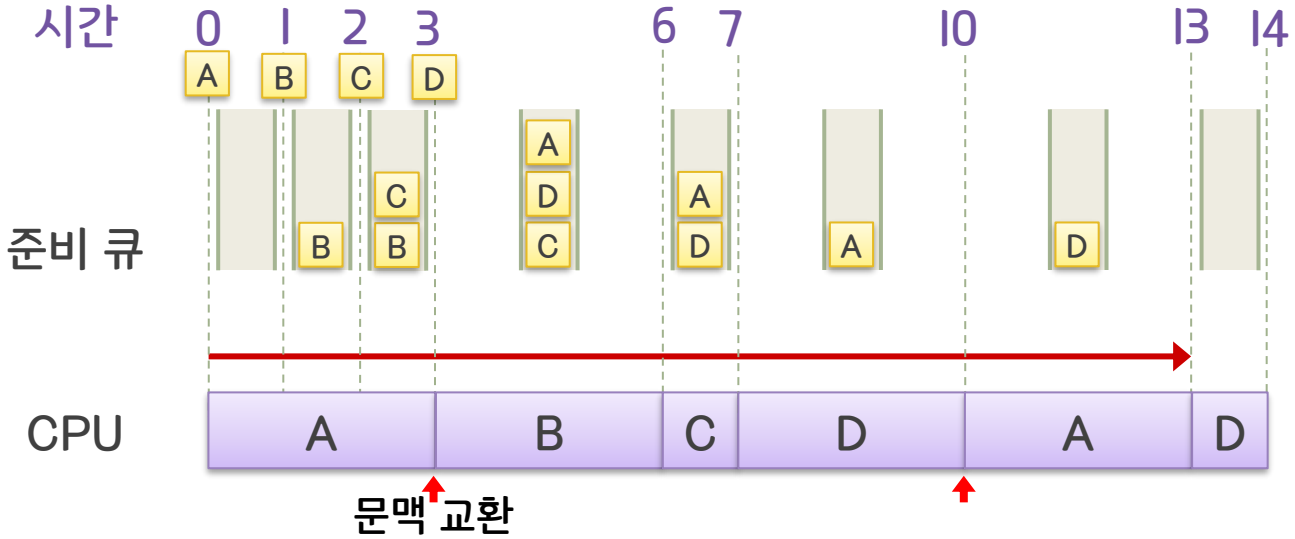


프로세스	A	B	C	D
대기시간	7	2	4	7
반환시간	13	5	5	11

RR 스케줄링의 예

도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4

- 시간 할당량 = 3

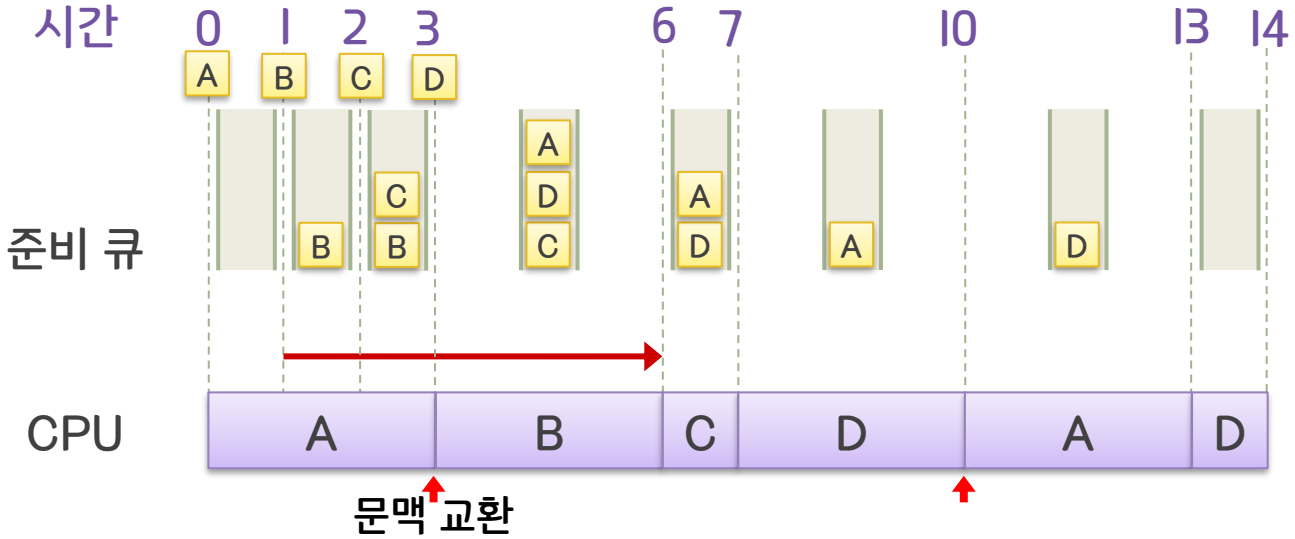


프로세스	A	B	C	D
대기시간	7	2	4	7
반환시간	13	5	5	11

RR 스케줄링의 예

도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4

- 시간 할당량 = 3

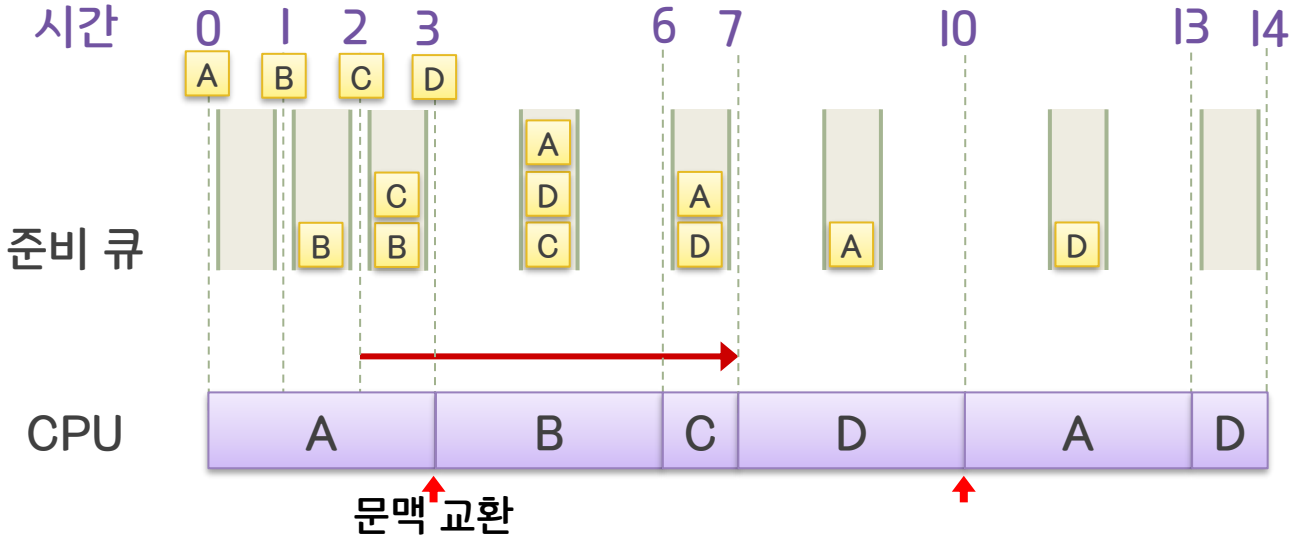


프로세스	A	B	C	D
대기시간	7	2	4	7
반환시간	13	5	5	11

RR 스케줄링의 예

도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4

- 시간 할당량 = 3

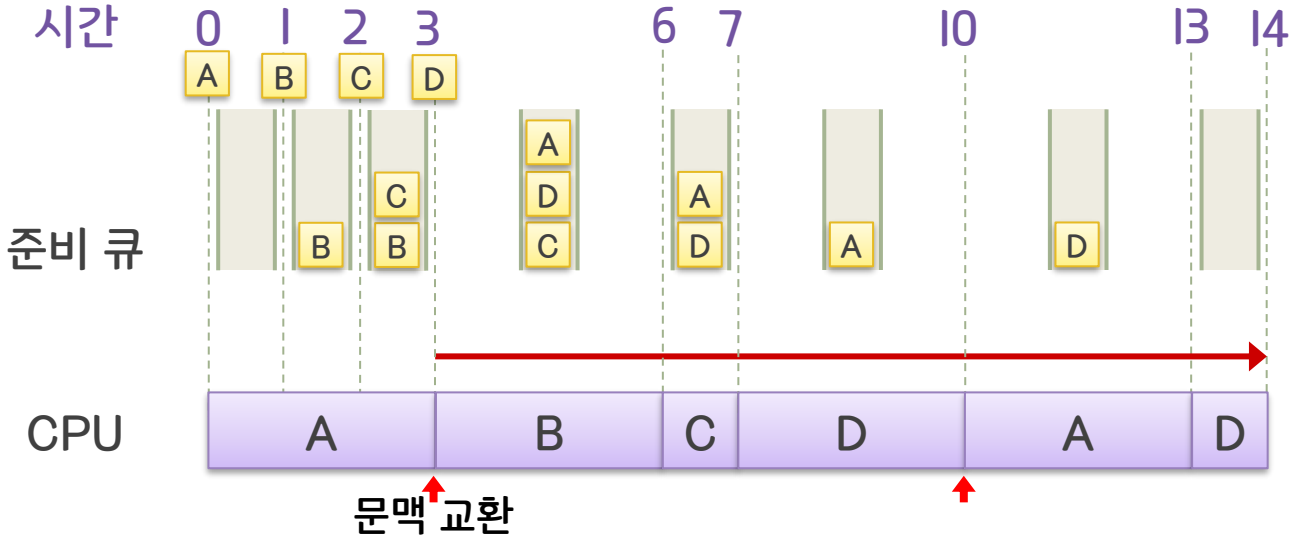


프로세스	A	B	C	D
대기시간	7	2	4	7
반환시간	13	5	5	11

RR 스케줄링의 예

도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4

- 시간 할당량 = 3

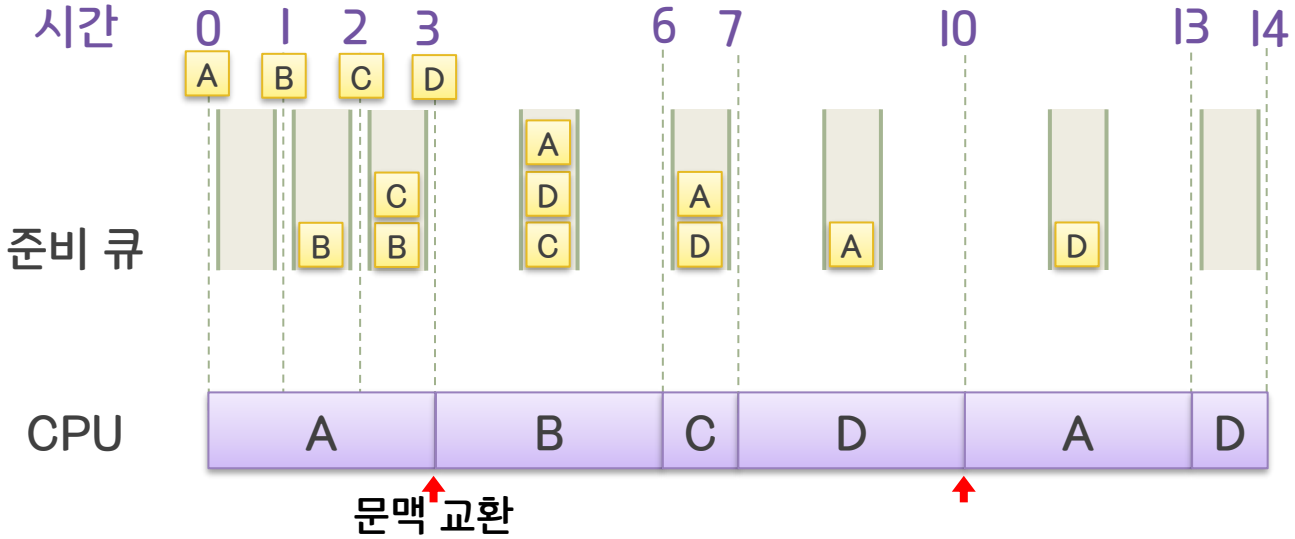


프로세스	A	B	C	D
대기시간	7	2	4	7
반환시간	13	5	5	11

RR 스케줄링의 예

도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4

- 시간 할당량 = 3



프로세스	A	B	C	D
대기시간	7	2	4	7
반환시간	13	5	5	11

- 평균 대기시간 = $\frac{7+2+4+7}{4} = 5$
- 평균 반환시간 = $\frac{13+5+5+11}{4} = 8.5$

RR 스케줄링

- RR (Round Robin) 스케줄링
  - 선점 스케줄링 알고리즘
  - 준비 큐에 도착한 순서에 따라 디스패치하지만
    정해진 시간 할당량에 의해 실행을 제한
  - 시간 할당량 안에 완료되지 못한 프로세스는 준비 큐의 맨 뒤에 배치
- 장점
  - CPU를 독점하지 않고 공평하게 이용
  - 대화형 운영체제에 유용

RR 스케줄링

단점

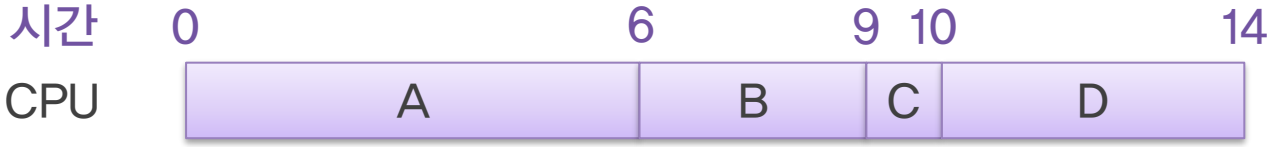
- 시간 할당량이 너무 크면 FCFS 스케줄링과 같아짐
- 시간 할당량이 너무 작으면 문맥 교환에 따른 오버헤드가 크게 증가함

시간 할당량	1	2	3	4	5	6
평균 대기시간	4.5	5.25	5	5.25	6	4.75
평균 반환시간	8	8.75	8.5	8.75	9.5	8.25
문맥 교환 횟수	10	4	2	1	1	0

RR 스케줄링의 예

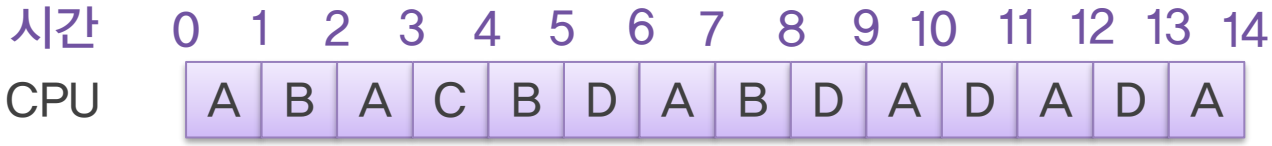
도착시간	0	1	2	3
프로세스	A	B	C	D
CPU 사이클	6	3	1	4

- 시간 할당량 = 6



- 평균 대기시간 = $\frac{0+5+7+7}{4} = 4.75$
- 평균 반환시간 = $\frac{6+8+8+11}{4} = 8.25$

- 시간 할당량 = 1



- 평균 대기시간 = $\frac{8+4+0+6}{4} = 4.5$
- 평균 반환시간 = $\frac{14+7+1+10}{4} = 8$

HRN 스케줄링

HRN (Highest Response Ratio Next) 스케줄링

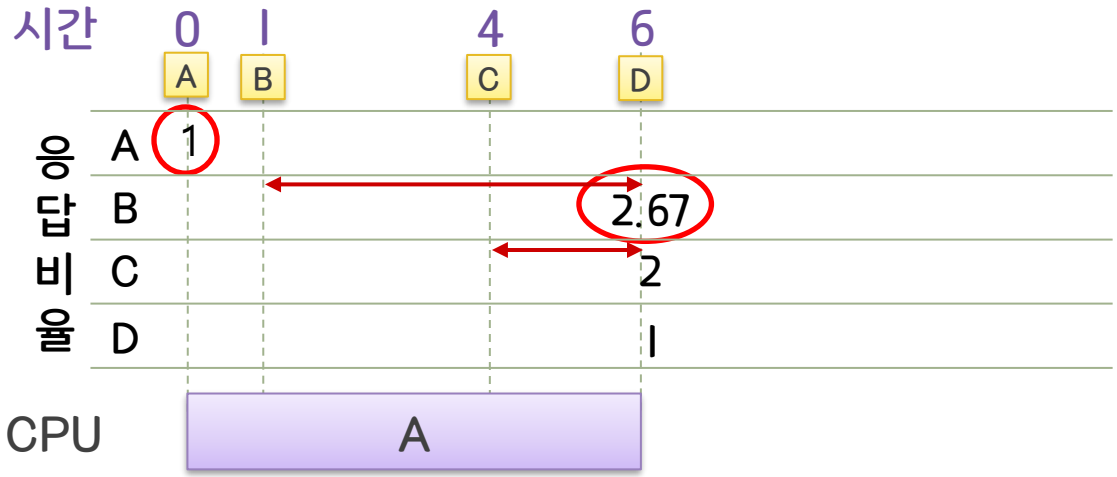
- 비선점 스케줄링 알고리즘
- 준비 큐에서 기다리는 프로세스 중 응답비율이 가장 큰 것을 먼저 디스패치

» $$\text{응답비율} = \frac{\text{대기시간} + \text{예상실행시간}}{\text{예상실행시간}} = \frac{\text{대기시간}}{\text{예상실행시간}} + 1$$

- 예상 실행시간이 짧을수록, 대기시간이 길수록 응답비율이 커짐

HRN 스케줄링의 예

도착시간	0	1	4	6
프로세스	A	B	C	D
CPU 사이클	6	3	2	1

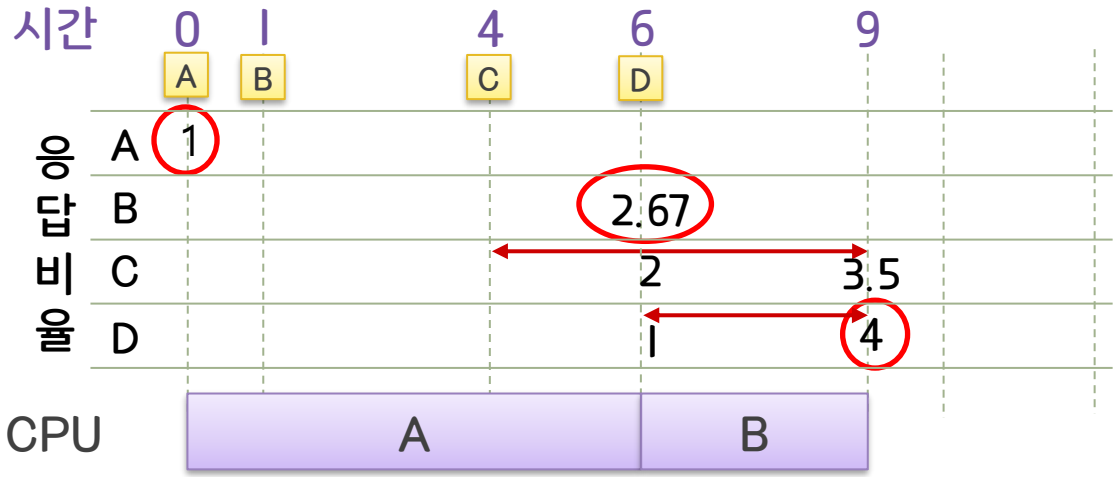


응답비율

$$\frac{\text{대기시간}}{\text{예상 실행시간}} + 1$$

HRN 스케줄링의 예

도착시간	0	1	4	6
프로세스	A	B	C	D
CPU 사이클	6	3	2	1

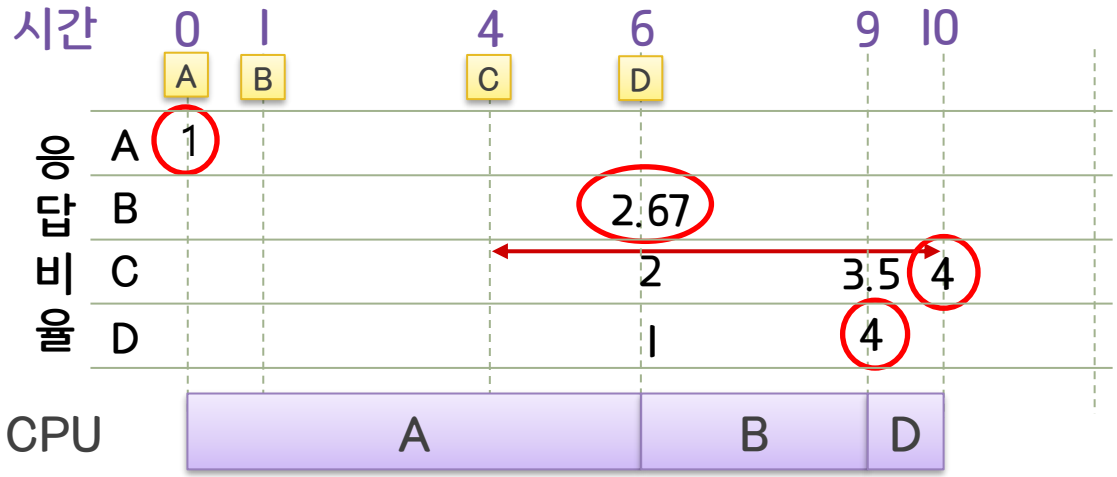


응답비율

$$\frac{\text{대기시간}}{\text{예상 실행시간}} + 1$$

HRN 스케줄링의 예

도착시간	0	1	4	6
프로세스	A	B	C	D
CPU 사이클	6	3	2	1

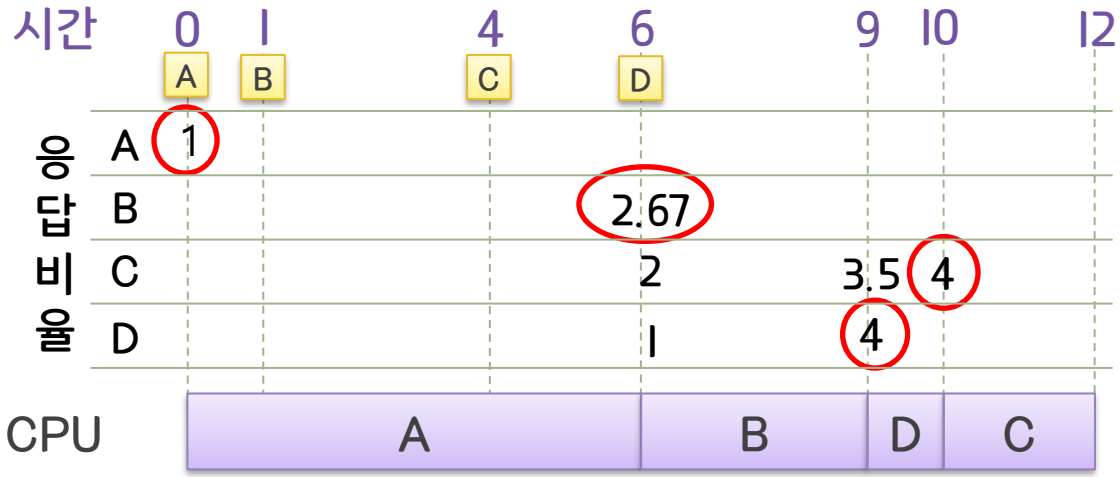


응답비율

$$\frac{\text{대기시간}}{\text{예상 실행시간}} + 1$$

HRN 스케줄링의 예

도착시간	0	1	4	6
프로세스	A	B	C	D
CPU 사이클	6	3	2	1

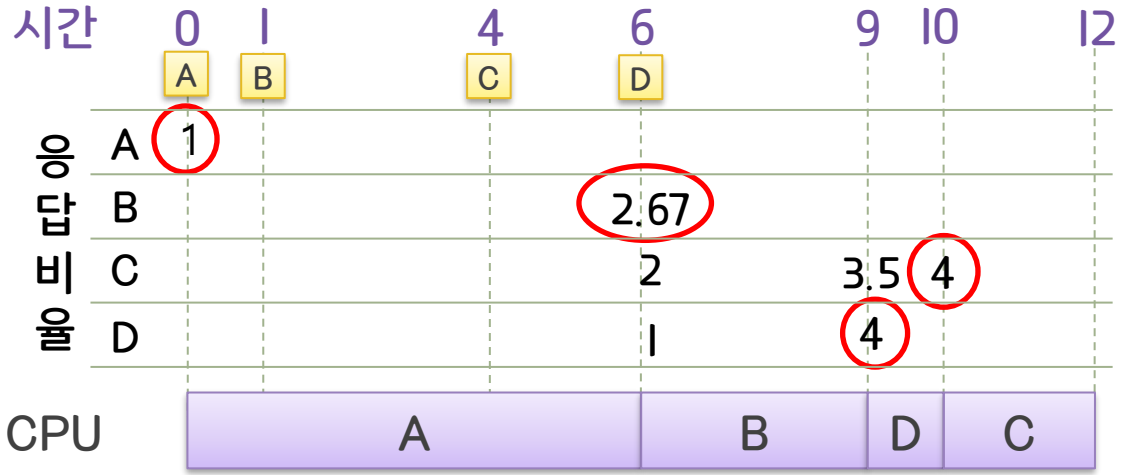


응답비율

$$\frac{\text{대기시간}}{\text{예상 실행시간}} + 1$$

HRN 스케줄링의 예

도착시간	0	1	4	6
프로세스	A	B	C	D
CPU 사이클	6	3	2	1



응답비율

$$\frac{\text{대기시간}}{\text{예상 실행시간}} + 1$$

프로세스	A	B	C	D
대기시간	0	5	6	3
반환시간	6	8	8	4

- 평균 대기시간 $= \frac{0+5+6+3}{4} = 3.5$
- 평균 반환시간 $= \frac{6+8+8+4}{4} = 6.5$

HRN 스케줄링

- HRN (Highest Response Ratio Next) 스케줄링
  - 비선점 스케줄링 알고리즘
  - 준비 큐에서 기다리는 프로세스 중 응답비율이 가장 큰 것을 먼저 디스패치
  - 예상 실행시간이 짧을수록, 대기시간이 길수록 응답비율이 커짐

- 장점
  - SJF의 단점을 보완

SJF 스케줄링

도착시간	0	1	4	6
프로세스	A	B	C	D
CPU 사이클	6	3	2	1

HRN 스케줄링

- HRN (Highest Response Ratio Next) 스케줄링
  - 비선점 스케줄링 알고리즘
  - 준비 큐에서 기다리는 프로세스 중 응답비율이 가장 큰 것을 먼저 디스패치
  - 예상 실행시간이 짧을수록, 대기시간이 길수록 응답비율이 커짐

- 장점
  - SJF의 단점을 보완

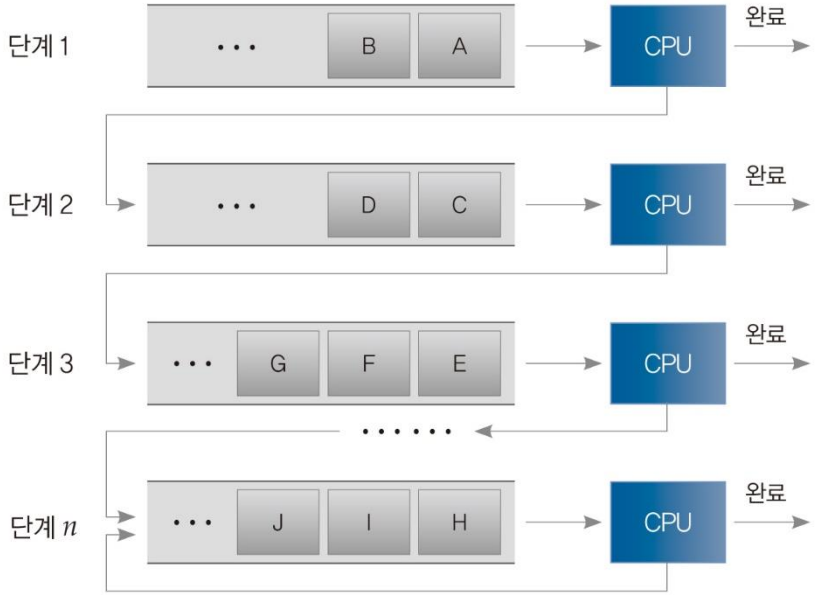
SJF 스케줄링

도착시간	0	1	4	6	8	10
프로세스	A	B	C	D	E	F
CPU 사이클	6	3	2	1	1	2

다단계 피드백 큐 스케줄링

다단계 피드백 큐 스케줄링

- 선점 스케줄링 알고리즘
- I/O 중심 프로세스와 CPU 중심 프로세스의 특성에 따라 서로 다른 시간 할당량 부여
- n개의 단계(단계 1 ~ 단계 n)
- 각 단계마다 하나씩의 큐 존재
- 단계가 커질수록 시간 할당량도 커짐



다단계 피드백 큐 스케줄링

스케줄링 방법

- 신규 프로세스는 단계 1의 큐에서 FIFO 순서에 따라 CPU 점유
- 입출력 같은 이벤트가 발생하면 CPU를 양보하고 대기상태로 갔다가 다시 준비상태가 될 때에는 현재와 동일한 단계의 큐에 배치
- 시간 할당량을 다 썼지만 프로세스가 종료되지 못했다면 다음 단계의 큐로 이동 배치
- 마지막 단계 n에서는 RR 스케줄링 방식으로 동작
- 단계 k 의 큐에 있는 프로세스가 CPU를 할당 받으려면 단계 1 부터 단계 $k-1$ 까지 모든 큐가 비어있어야만 함

다단계 피드백 큐 스케줄링

- 장점
  - I/O 위주의 프로세스(대화형)는 높은 우선권 유지
  - 연산 위주의 CPU 중심 프로세스는 낮은 우선권이지만 긴 시간 할당량 가짐

- 적응적 다단계 피드백 큐 스케줄링
  - 시간 할당량을 다 쓰기 전에 CPU를 반납하는 경우 하나 작은 단계의 큐로 이동 배치
  - 연산 위주의 프로세스가 I/O 위주로 바뀐다면 점점 작은 단계로 배치 가능

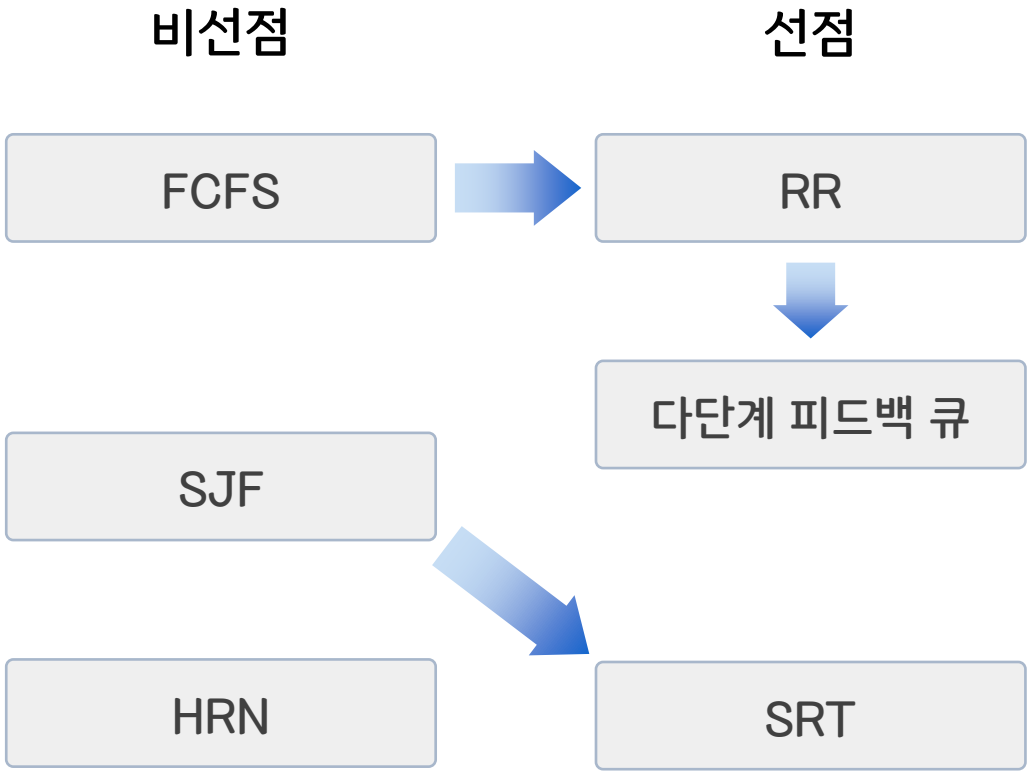
스케줄링 알고리즘

비선점	선점
FCFS	RR
SJF	다단계 피드백 큐
HRN	SRT

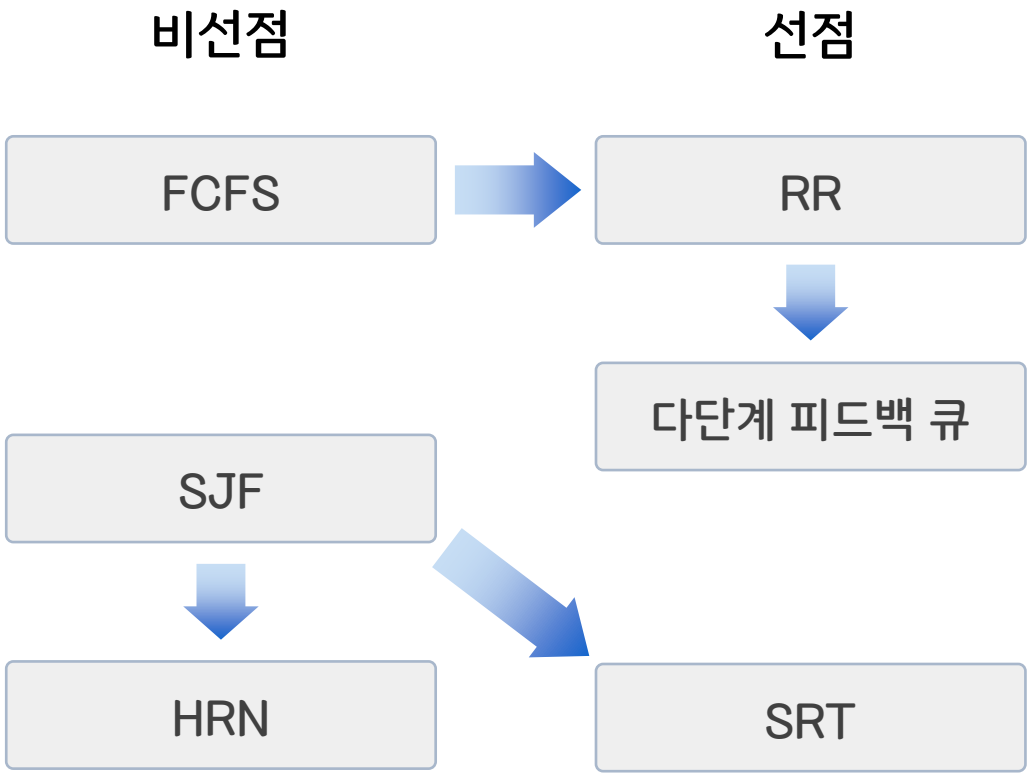
스케줄링 알고리즘

스케줄링 알고리즘

스케줄링 알고리즘



스케줄링 알고리즘



강의를 마쳤습니다.

다음시간에는

4강. 병행 프로세스 I