6강

교착상태 I

컴퓨터과학과 김진욱 교수

한국방송통신대학교

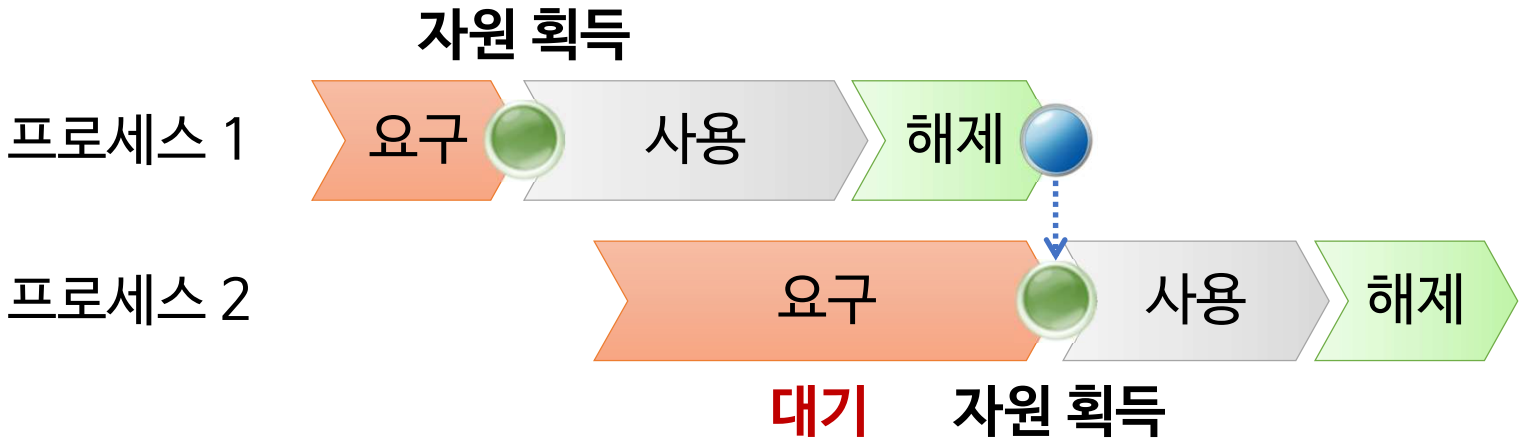
목차

01

교착상태의 개요

프로세스의 자원 사용 절차

- 요구 → 사용 → 해제
- 요구과정에서 가용한 자원이 없으면
  - 자원을 획득할 때까지 대기

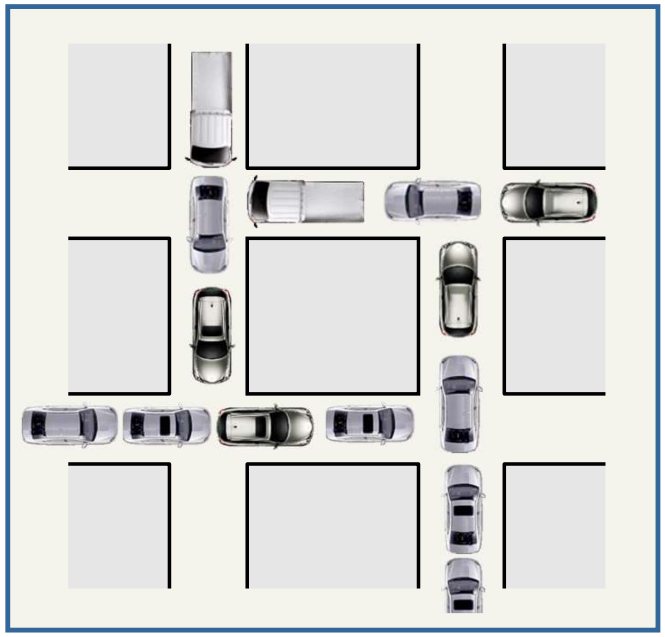


교착상태(deadlock)

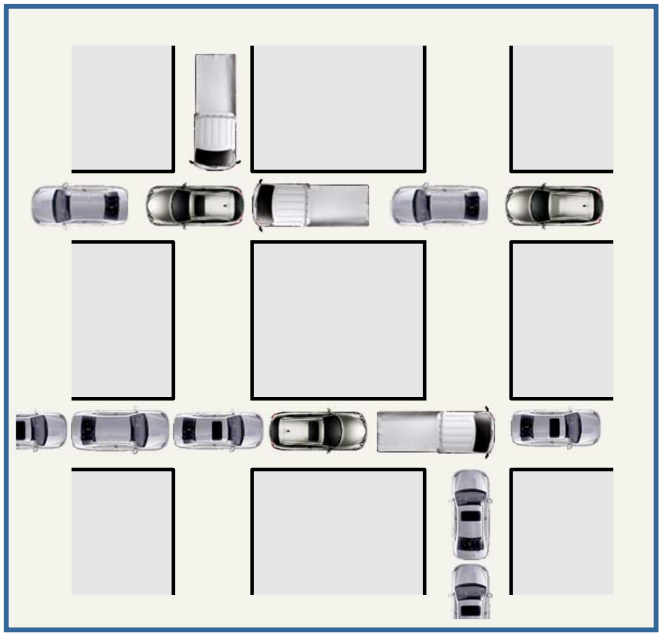
- 여러 개의 프로세스가
  서로 상대방의 작업이 끝나기만 기다리고 있어
  어느 쪽도 영원히 진행하지 못하는 상태

교착상태와 기아상태의 차이

교착상태



기아상태



02

교착상태의 특성

교착상태의 필요조건

- 상호배제
- 점유대기
- 비선점
- 환형대기

- 네 가지 조건이 동시에 만족될 때 교착상태 발생 가능

상호배제(mutual exclusion) 조건

- 프로세스가 자원에 대한 배타적인 통제권을 요구
- 적어도 하나 이상의 자원은 여러 프로세스에 의해 동시에 사용될 수 없음
- 다른 프로세스가 점유한 자원이 필요하면 반드시 대기

점유대기(hold and wait) 조건

- 프로세스가 이미 한 자원을 할당받아 점유하고 있는 상황에서 다른 프로세스가 점유하고 있는 또 다른 자원을 요구하여 해제되기를 기다리는 상황

비선점(no preemption) 조건

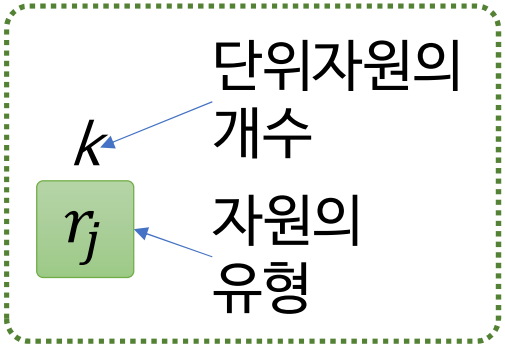
- 프로세스에 할당된 자원은 그 프로세스가 사용을 마치고 스스로 반환하기 전에는 해제되지 않음
- 할당된 자원은 타의에 의해서는 해제되지 않음

환형대기(circular wait) 조건

- 프로세스의 자원 점유 및 점유된 자원의 요구 관계가 환형을 이루며 대기하는 상황

자원할당 그래프 $G = (V, E)$

- V: 정점의 집합 $V = P \cup R$
  - $P = \{p_1, p_2, \cdots, p_n\}$: n개의 프로세스 p_i
  - $R = \{r_1, r_2, \cdots, r_m\}$: m개의 자원 r_j



- E: 방향 있는 간선의 집합 $E = Q \cup S$
  - $Q = \{(p_i, r_j): p_i \in P,\ r_j \in R\}$: 프로세스 p_i가 자원 r_j를 요구

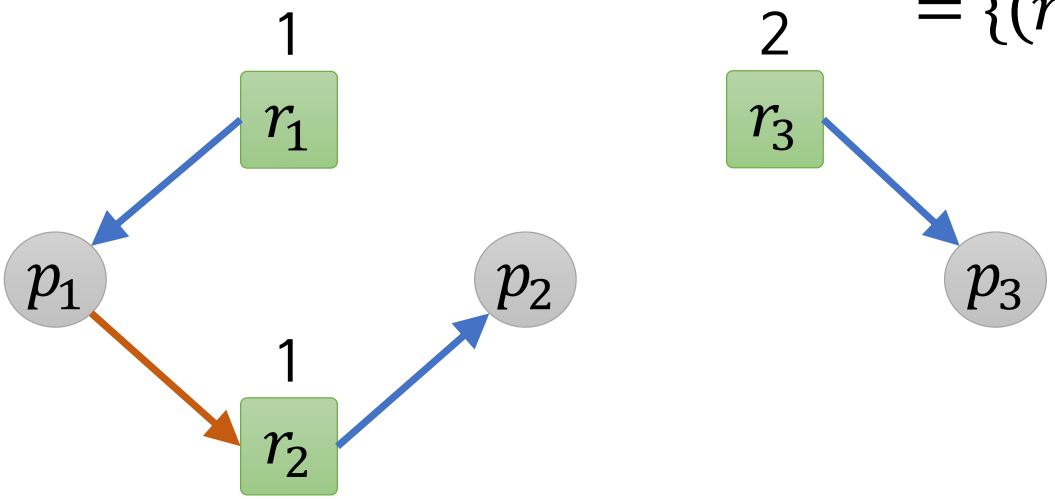
    요구간선 

  - $S = \{(r_j, p_i): r_j \in R,\ p_i \in P\}$: 자원 r_j가 프로세스 p_i에 할당

    할당간선 

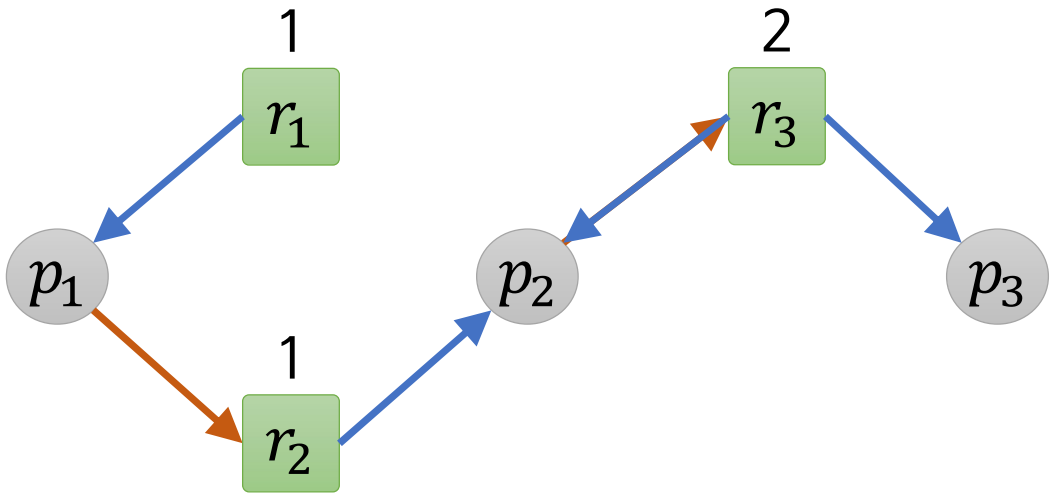
자원할당 그래프 예

- 정점의 집합 $V = P \cup R$
  - 프로세스 집합 $P = \{p_1, p_2, p_3\}$
  - 자원 집합 $R = \{r_1, r_2, r_3\}$
- 방향 있는 간선의 집합 $E = Q \cup S$
  - 요구간선 집합 $Q = \{(p_1, r_2)\}$
  - 할당간선 집합 S
    $= \{(r_1, p_1), (r_2, p_2), (r_3, p_3)\}$



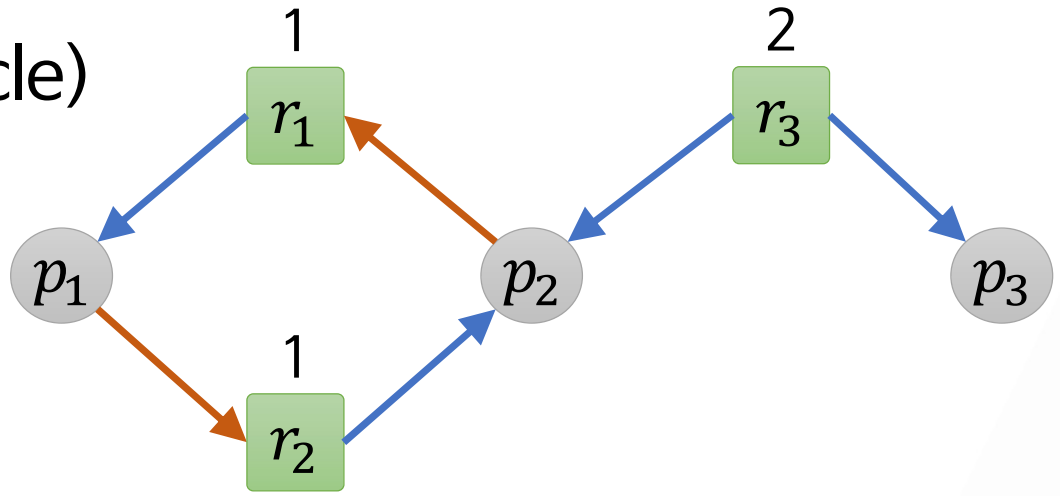
자원할당 그래프의 변화

- p_2가 r_3을 요구하는 경우
  - 요구간선 (p_2, r_3) 추가
  - 가용한 단위자원 존재하면 할당간선 (r_3, p_2)로 바꿈



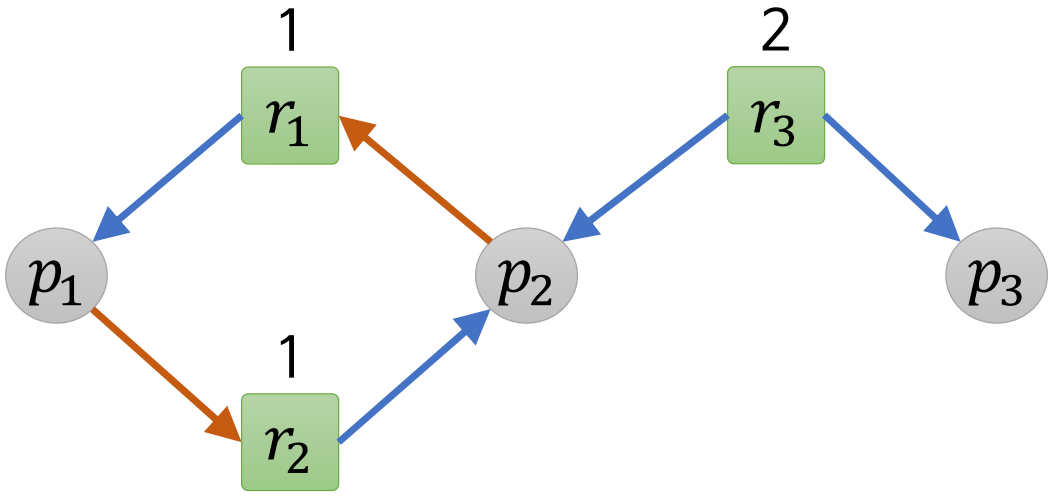
자원할당 그래프

- 교착상태의 필요조건 표현
  - 상호배제: 하나의 할당간선
  - 점유대기: 한 프로세스에 할당간선과 요구간선 연결
  - 비선점: 요구간선
  - 환형대기: 사이클(cycle)



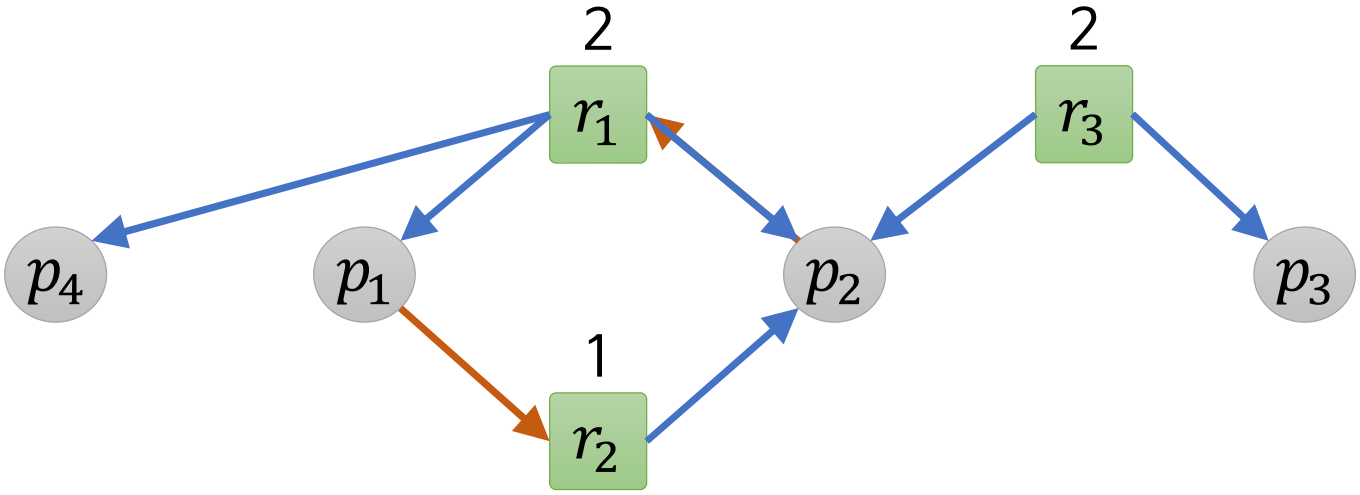
자원할당 그래프

- 사이클 없음 → 교착상태 없음
- 사이클 있음 → 교착상태 발생 가능
  - 교착상태 예



자원할당 그래프

- 사이클 없음 → 교착상태 없음
- 사이클 있음 → 교착상태 발생 가능
  - 교착상태 아닌 예



교착상태의 처리기법

- 교착상태 예방
  - 교착상태의 네 가지 필요조건이 동시에 만족되는 것을 피하여 교착상태가 발생하지 않도록 하는 방법
- 교착상태 회피
  - 프로세스에 필요한 자원의 최대량에 대한 정보를 이용하여 교착상태가 발생하지 않도록 하는 방법
- 교착상태 탐지 및 복구
  - 교착상태의 발생 여부를 조사하여 발생한 경우에는 적절한 조치를 취해 정상상태로 복구하는 방법

03

교착상태 예방

상호배제 조건 제거

- 공유할 수 있는 자원: 상호배제 필요 없음
  - 예: 읽기 전용 파일

- 공유할 수 없는 자원: 반드시 상호배제 필요
  - 예: 프린터
  - ➔ 상호배제 조건 제거로 교착상태 예방은 불가능

점유대기 조건 제거

- 자원을 점유했을 때 대기하지 않아야 함
  - 프로세스가 앞으로 필요한 모든 자원을 처음에 한꺼번에 요구하여 할당받음
    - 자원이용률 낮아짐, 기아상태 가능
- 대기할 때 자원을 점유하고 있지 않아야 함
  - 새로운 자원을 요구할 때 할당받았던 자원을 모두 해제
    - 점유 도중 해제할 수 없는 자원에는 적용 불가

비선점 조건 제거

- 선점이 가능하도록 해야 함
  - 자원의 특성에 따라 불가능한 경우 존재
    - 예: 프린터
- 다른 프로세스가 대기할 가능성 줄이기
  - 점유대기 상황이 발생하면 할당받았던 자원을 모두 해제
    - 프린터 같은 자원에는 적용 불가

환형대기 조건 제거

- 모든 자원에 일련번호를 지정
  - 함수 $f: R \rightarrow N$ (R: 자원 집합, N: 자연수 집합)
    - $r_i \neq r_j$이면 $f(r_i) \neq f(r_j)$
- 방법1
  - 프로세스는 자원을 요구할 때 일련번호 기준으로 항상 오름차순이 되도록 요구
  - 가장 최근에 할당받은 자원이 r_i이면 다음에 요구할 자원 r_j는 $f(r_i) < f(r_j)$ 만족

환형대기 조건 제거

- 모든 자원에 일련번호를 지정
  - 함수 $f: R \rightarrow N$ (R: 자원 집합, N: 자연수 집합)
    - $r_i \neq r_j$이면 $f(r_i) \neq f(r_j)$
- 방법2
  - 프로세스가 자원을 요구할 때
    그보다 일련번호가 작은 자원만 점유하도록 함
  - 자원 r_j 요구하려면 점유 중인 자원 중
    $f(r_j) \leq f(r_i)$인 자원 r_i는 모두 해제

환형대기 조건 제거

- 점유대기 중인 프로세스는 점유 중인 자원의 일련번호보다 대기 중인 자원의 일련번호가 큼
  - 환형대기 발생 불가능

- 프로세스마다 요구순서가 다를 수 있어 자원의 일련번호 설정 어려움
- 요구순서가 일련번호 오름차순을 못 지키면 점유자원 해제 필요하나 적용 불가한 자원 존재

다음시간 안내

7강

교착상태 II