Computational Thinking

2017 2학기

11장-동시적 행동

박종혁 교수 (컴퓨터공학과)

jhpark1@seoultech.ac.kr

http://www.parkjonghyuk.net

서울과학기술대학교

SEOULTECH SEOUL NATIONAL UNIVERSITY OF SCIENCE & TECHNOLOGY

학습개요

- 11장 동시적 행동
  1. 병렬성 또는 동시성?
  2. 스케쥴링
  3. 정렬 연결망
  4. 동시성 효과 측정하기
  5. 동시성에서의 해결 과제
  6. 참고문헌
- 조별발표
  - 조별 10분 발표

학습목표

- 현대의 문제 해결을 위한 수퍼컴퓨터와 분산컴퓨팅의 역할과 병렬성과 동시성 사이의 차이를 설명할 수 있다.
- 동시적인 실행을 금지하는 기본적인 제약들을 인지한다.
- 동시적인 행동을 설명하는 시간 맞춤 도표들을 해석할 수 있다.
- 특정한 데이터 집합을 위해 주어진 정렬 연결망의 동작을 추적할 수 있다.
- 성능 개선을 위한 동시성의 잠재력을 설명할 수 있다.
- 공유자원이 어떻게 동시성을 제약하는지 설명할 수 있다.
- 톡토우(TOCTOU-경쟁조건)를 인지하고 이 상황이 어떻게 오류를 야기하는지 설명할 수 있다.
- 데드락(deadlock)과 라이브락(livelock) 상황을 인지한다.

❖ 동시적 실행 (concurrent execution), 동시성 (concurrency)

- 동시에 여러 개의 작업들을 수행

❖ 멀티코어 프로세서 (multi-core processors)

- 하나의 집적회로에 여러 개의 프로세서들을 포함

❖ 병렬 실행(parallel execution)

- 여러 개의 장치들에 의한 동시적인 실행은 각 장치들이 병렬로 실행

❖ 수퍼컴퓨터 (supercomputer)

- 속도 : 페타플롭 (petaflops) (초당 100만의 4제곱개의 명령어들이 실행)
- 수백만 개의 프로세서들을 동시에 실행

❖ **분산컴퓨팅 (distributed computing)**
- 인터넷에서의 병렬 실행
- 공통된 문제를 해결하기 위해 여러 개의 대규모의 독립적인 컴퓨터들이 집단적으로 작업

❖ **그리드 컴퓨팅 (grid computing)**
- 분산컴퓨팅. 일기예보와 금융 모델링에서 사용

❖ **클라우드 컴퓨팅 (cloud computing)**
- 대용량의 공유 데이터를 포함하는 분산컴퓨팅

❖ **동시성(concurrency)**
- 소프트웨어 시스템과 관련

❖ **병렬성(parallelism)**
- 하드웨어 형태의 동시적인 실행

- 축구 토너먼트와 컴퓨터
- 각 토너먼트 경기 → 하나의 소프트웨어 작업
- 축구 경기장 → 작업이 실행되는 프로세서
- 하나의 경기장에서 전체 토너먼트를 진행

  → 모든 작업을 하나의 프로세서에서 실행
- 각 경기 3시간 소모

그림 11.1 4개의 팀의 단일 예선을 위한 토너먼트 묶음

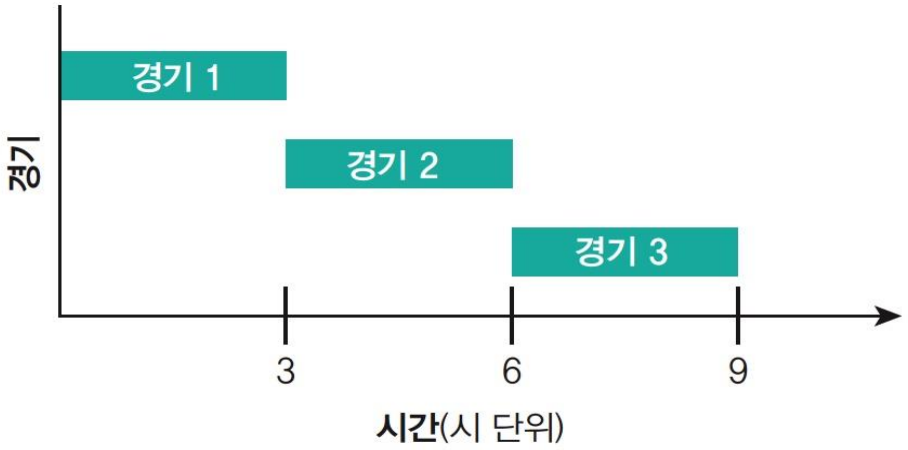


그림 11.2 한 개의 경기장에서 진행된 4개의 팀에 대한 토너먼트의 시간 도식

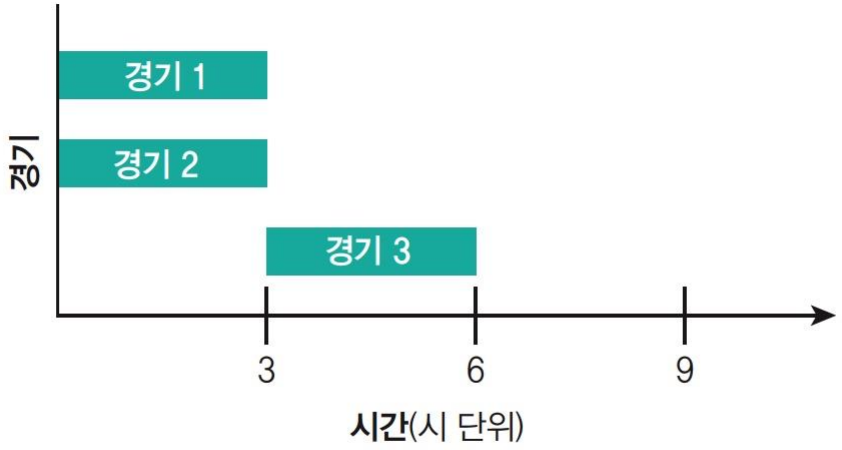


그림 11.3 두 개의 경기장에서 진행된 4개의 팀에 대한 토너먼트의 시간 도식

❖ 스케쥴링(scheduling)

- 어떤 시점에 어느 프로세서에서 작업들을 실행할 것인지를 결정

❖ 다중작업

- 장점 : 시간을 절약할 수 있다는 것이다.
- 더 많은 프로세서가 항상 성능을 향상시키는 것은 아님

Ex) 경기1(프로세스1)과 경기2 (프로세스2) 의 승자가 결정된 후에
경기3 (프로세스3) 은 진행되야 하는 경우

- 작업이 계획될 때 프로세스들 사이의 이러한 종속관계를 고려해야 한다.

11.2 스케쥴링

그림 11.4 8개의 팀의 단일 예선에 대한 토너먼트 묶음

그림 11.5 4개의 경기장에서 진행된 8개의 팀에 대한 토너먼트의 시간 도식

❖ **성능 개선**

❖ **여러 개의 경기장(다중 프로세서)에서 7개 경기(7개의 작업)가 진행되는 경우**

→ 9시간 만에 종료

❖ **동일한 경기장(단일 프로세서)에서 모든 게임(21개의 작업)이 진행**

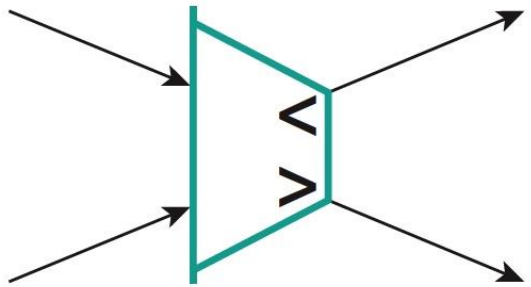
→ 21시간(3시간 x 7개의 경기)만에 종료

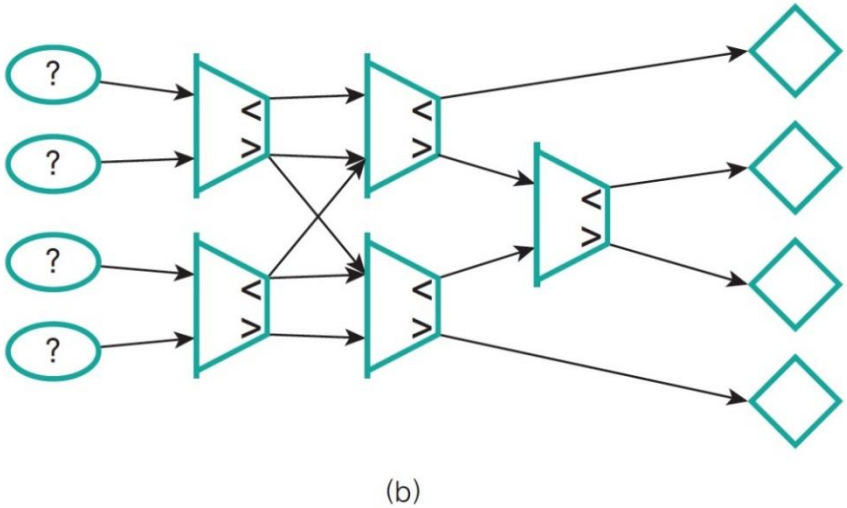
→ 따라서 다중작업을 사용하면 $2\frac{1}{3}$ 배 빠름

❖ 정렬 알고리즘(sorting algorithm)

- 오름차순(가장 작은 것부터 가장 큰 것까지)이나
  내림차순(가장 큰 것부터 가장 작은 것까지)으로 배열되도록 정렬하기

그림 11.6 돌덩이를 정렬하는 동굴 주거인





❖ 정렬 연결망(sorting network)

- 단순 프로세서들의 집합
- (그림 (a) 단일 정렬 연결망 프로세서) : (1) 연결망에서 두 개의 값을 받아들이고, (2) 두 개의 값을 비교하고, (3) 두 개의 비교된 값을 연결망에서 전송함
- (그림 (b) 5개의 프로세서로 이루어진 정렬 연결망) : 4개의 데이터를 받아들여 5개의 프로세서를 이용해서 정렬하기

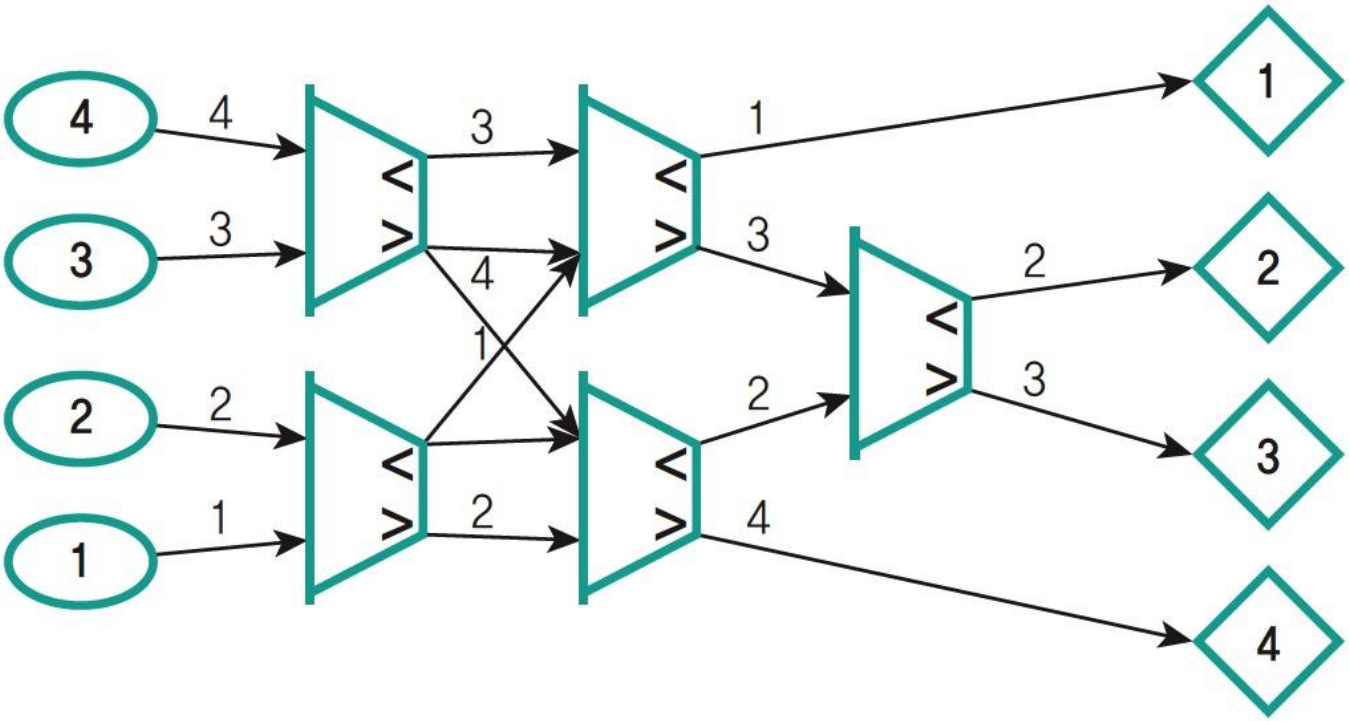
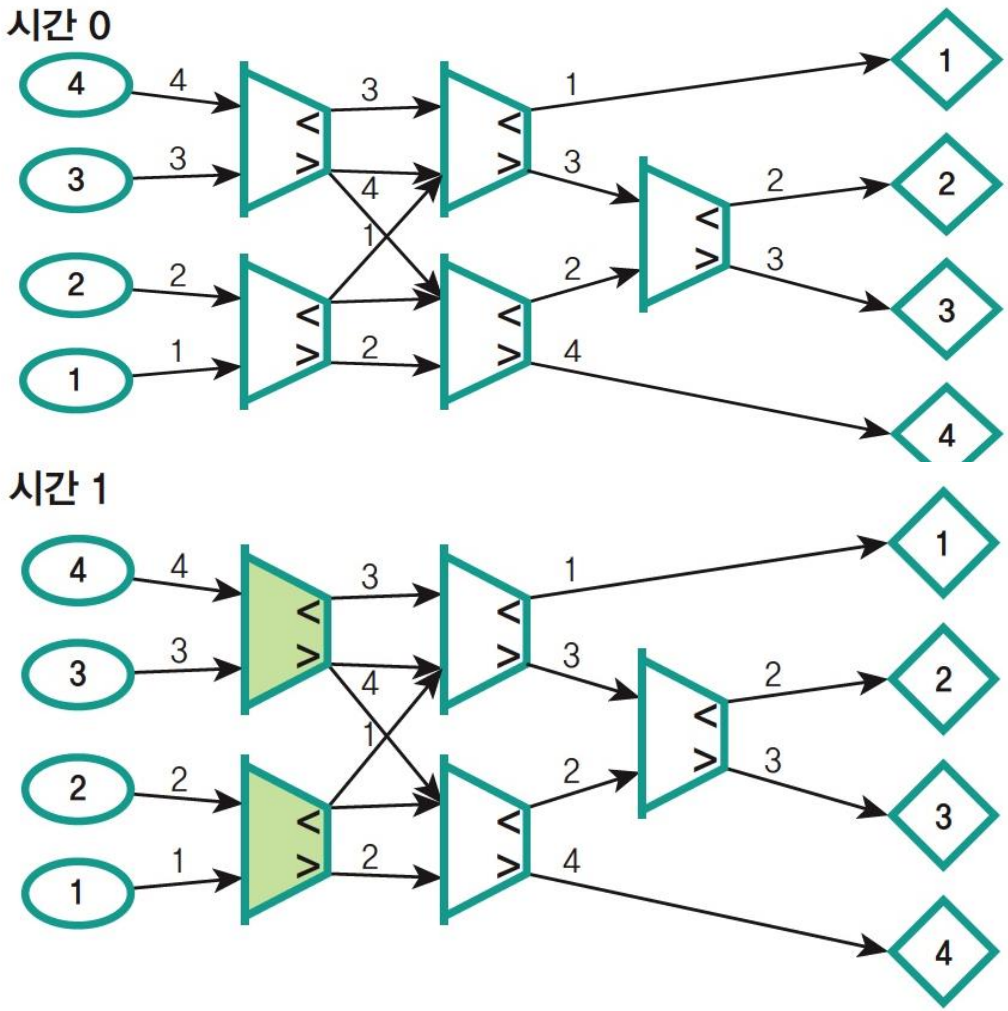
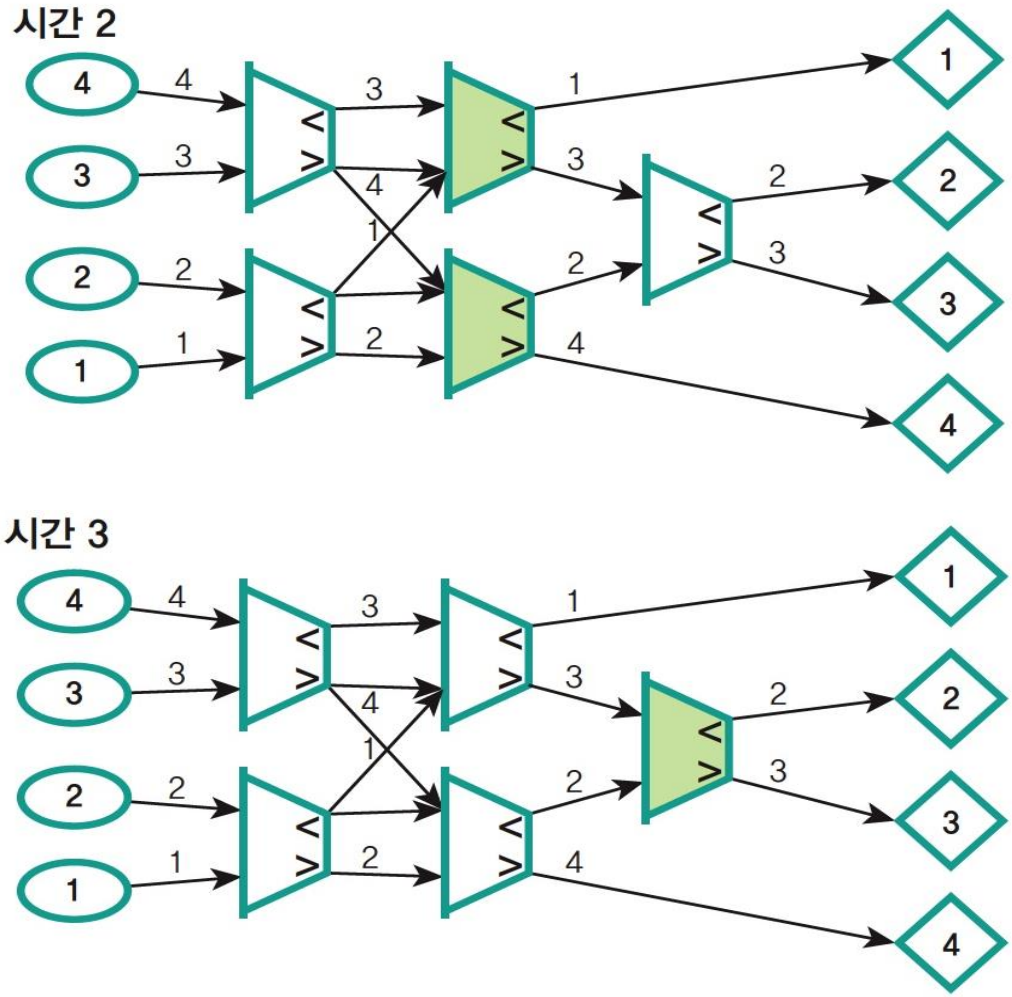


그림 11.8 4개의 값을 정렬하는 연결망의 예

11.4 동시성 효과 측정하기



11.4 동시성 효과 측정하기



11.4 동시성 효과 측정하기

- 연결망은 프로세서들의 행렬로 시작하는데, 이때 행과 열의 수는 데이터 크기의 절반
- 정방행렬의 오른쪽에는 왼쪽에서 오른쪽으로 가면서 열의 높이가 하나씩 감소하는 프로세서들의 삼각형이 존재
- 정렬 연결망에서 정렬을 위해 필요한 시간은 열의 수와 같음
- 필요한 시간은 정렬될 값의 수보다 하나가 작게 됨

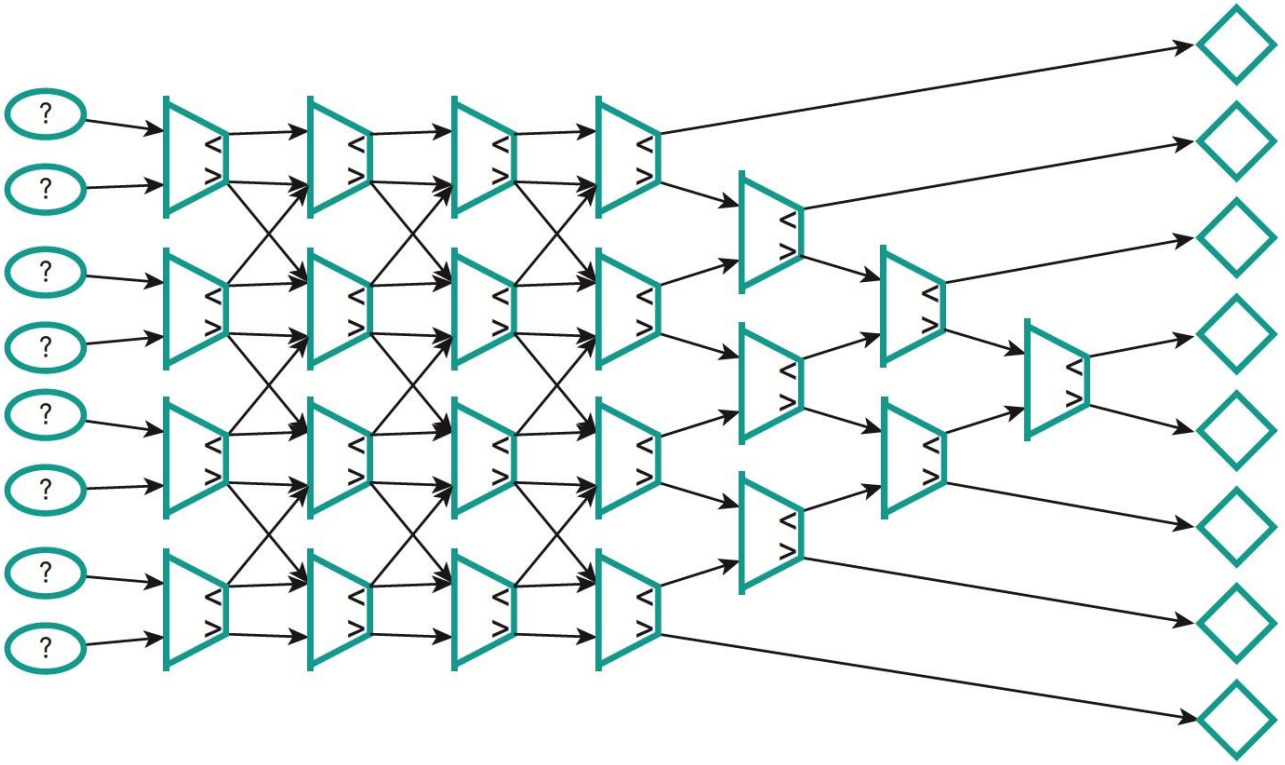


그림 11.10 8개의 값을 정렬하는 연결망

❖ **100개의 프로세서들은 단일 프로세서보다 100배 빨리 작업을 완료할 수 있을까?**

❖ **공유 자원**

- 동시적으로 프로세서들이 실행을 할 때 제한
- 프로세서들은 각각의 입력 데이터 값이 사용가능하게 될 때까지 실행을 시작할 수 없음
- 하지만 이러한 값들은 다른 프로세서들(적어도 이전에 이 값들을 비교했던 프로세서들)과 공유되는 일종의 공유 자원

- 동시성 제약을 지키지 못하면 다중작업이 실패함
- 동시적인 읽기 접근 OK
- 한 명의 사용자가 문서를 읽고 있을 때 누군가 문서를 변경하고 있다면?
- 공유 문서에 대한 여러 개의 동시적인 편집(갱신)이 있다면?

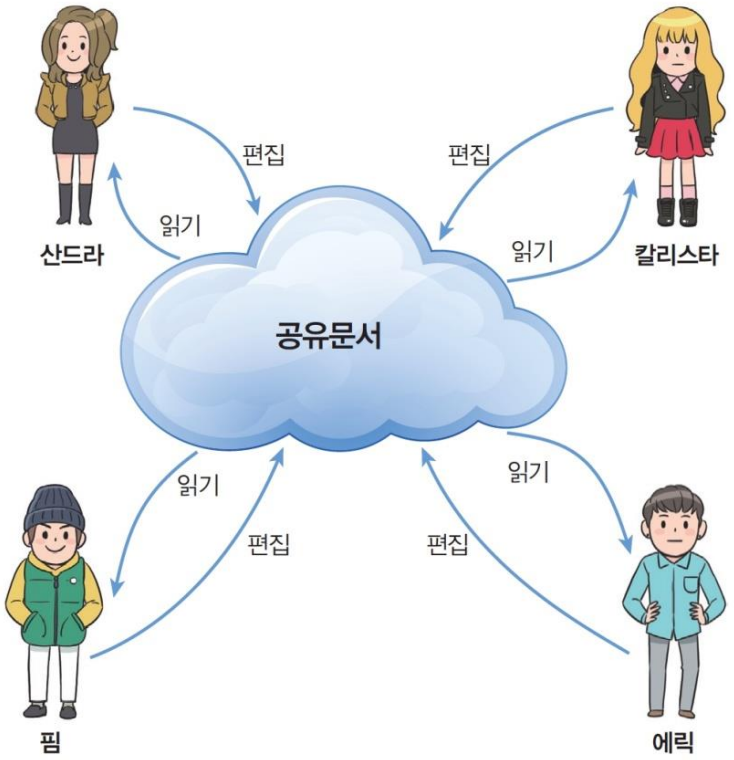


그림 11.11 4명의 사람들이 공유하는 문서 시스템

❖ **TOCTOU(time-of-check time-of-use: 확인시간 사용시간) 조건, 경쟁조건(race condition)**

- 공유자원에 대한 동시적인 갱신 접근이 이루어질 때 발생
- TOC(time-of-check): 프로세서가 자원의 상태를 확인하는 시간.
  문서의 제목을 읽는 것
- TOU(time-of-use): 프로세서가 자원의 상태를 갱신하는 시간.
  제목을 편집하는 것
- 회피해야할 TOCTOU 상황 : 첫 번째 프로세스가 공유자원에 대한 확인을 수행하고 자신의 갱신을 완료하기 전에, 두 번째 프로세스가 동일한 자원에 대하여 확인과 갱신을 둘 다 수행할 때

❖ 자원을 잠금

- 일단 하나의 프로세서가 확인을 개시하면, 그 자원은 갱신이 완료될 때까지 잠금이 풀리지 않음
- 다른 프로세서들은 잠긴 자원에 어떤 식으로든 접근할 수 없음
- 잠금을 수행한 프로세스만 잠금을 풀 수 있음
- 동시성에 대한 제약

그림 11.12 경쟁조건을 경험하는 두 대의 자동차

❖ 데드락(deadlock)

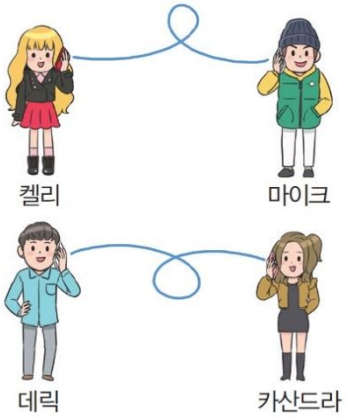
- 프로세서들이 자원에 대해 계속해서 진행할 수 없는 방법으로 잠금을 할 때 데드락이 발생
- 하나 이상의 자원을 해제하지 않고는 프로세서들이 실행하는 것이 불가능

단계 1

켈리는 마이크와 카산드라에게 전화회의를 정하려고 한다.
데릭도 또한 마이크와 카산드라에게 전화회의를 하려고 한다.

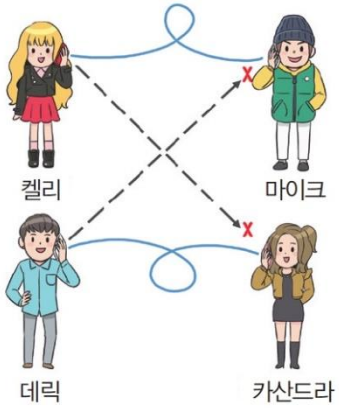
단계 2

켈리는 마이크에게 전화를 걸고 응답을 받는다.
동시에 데릭은 카산드라에게 전화를 걸고 응답을 받는다



단계 3

켈리는 카산드라에게 전화걸기를 시도하고
데릭은 마이크에게 전화걸기를 시도한다.
데드락 상태에 빠진다 – 전화회의를 완성하지 못하게 된다.



❖ 그리드락(grid lock)

- 교통량에 데드락이 발생할 때

❖ 라이브락(live lock)

- 일부 프로세서들에게만 데드락이 발생
- 컴퓨터 시스템에서 프로세서들이 자원을 공평하게 부여받지 못할 때 발생
- 일부 특정한 프로세서나 특정한 작업은 때때로 낮은 우선순위가 배정되어서 더 이상 진행되지 못하게 됨

그림 11.14 그리드락 상태에 있는 6대의 자동차

❖ 크라우드 소싱(crowd sourcing)

- 크라우드 소싱 해법에서 사람들은 동시성 프로세서의 역할을 함
- 그룹(크라우드)에 있는 모든 사람들은 동일한 문제가 주어지고 문제를 해결하는 과정에서 집단적으로 공유

11.7 참고문헌

- 생각하는 프로그래밍, 윤성준,조상민 역, 인사이트, 2014
- 벤츠 타는 프로그래머, 정금호 저, 제이펍, 2013
- 컴퓨터과학이 여는 세계, 이광근 저, 인사이트, 2015
- 소프트웨어 전쟁, 백일승 저, 더하기북스,2015
- 김대수, “소프트웨어와 컴퓨팅 사고”, 생능출판, 2017
- 천인국, “어서와 파이썬은 처음이지!”, 인피니티북스, 2016

10장 – 계산의 한계

- 발표방식: ppt 자료 활용 2개 조 각각 10분 발표 (질의응답 포함)

- 문제 1
  - 다음 제시된 알고리즘에 대하여 각각 가능, 불가능, 비실용적으로 구별하고 이유를 설명하여라. 또한 아래의 예시 외에 가능, 불가능, 비실용적 각각의 상황에 대한 예를 한 가지 이상 들고 이유를 설명하시오.

  1) 대한민국 인구조사 프로그램
  2) 거대한 자연수의 약수를 찾는 문제

- 문제 2
  - 튜링 테스트란 기계가 인간과 얼마나 비슷하게 대화할 수 있는지를 기준으로 기계에 지능이 있는지를 판별하고자 하는 테스트이다. 튜링 테스트를 통과한 인공지능 컴퓨터가 개발이 되고 상용화 된다면 미래의 우리의 사회는 어떻게 될 것인가?’에 대해 토의하고 발표하시오.