
ALOHA
ALOHA
심화 - 알고리즘심화반
4째날
이준엽
Algorithm research team of Hanyang Univ.

# INDEX

# Topological Sort
# (위상정렬)

# Topological Sort

## 의존성 그래프? (Dependency graph)

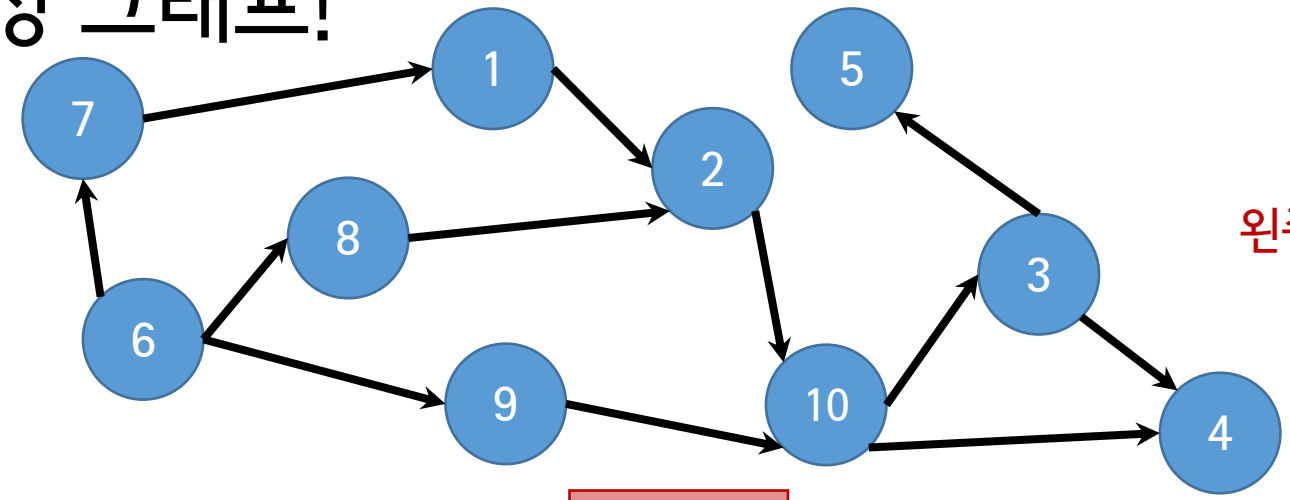
1. Task사이의 의존성을 DAG(사이클 없는 유향 그래프)로 나타낸 것

2. 의존성이 있는 Task? 라면 끓인다고 했을 때,
   (1) 라면을 먹는다
   (2) 파를 썬다
   (3) 스프를 넣은 물에 후레이크와 면을 넣고 3분간 끓인다
   (4) 라면을 산다
   (5) 라면을 뜯는다
   (6) 계란을 넣는다
   (7) 끓는 물에 스프를 넣는다
   (8) 물을 끓인다

3. (4)→(5) , (8)→(7)→(3) 등의 관계에서 의존성 발견!

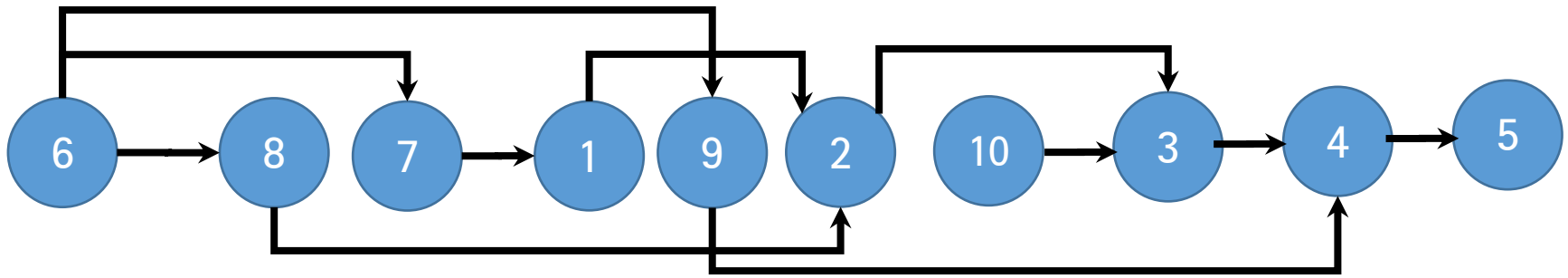
# Topological Sort

의존성 그래프!




왼쪽부터 차례대로 일 처리 해야겠다!

Topological Sort

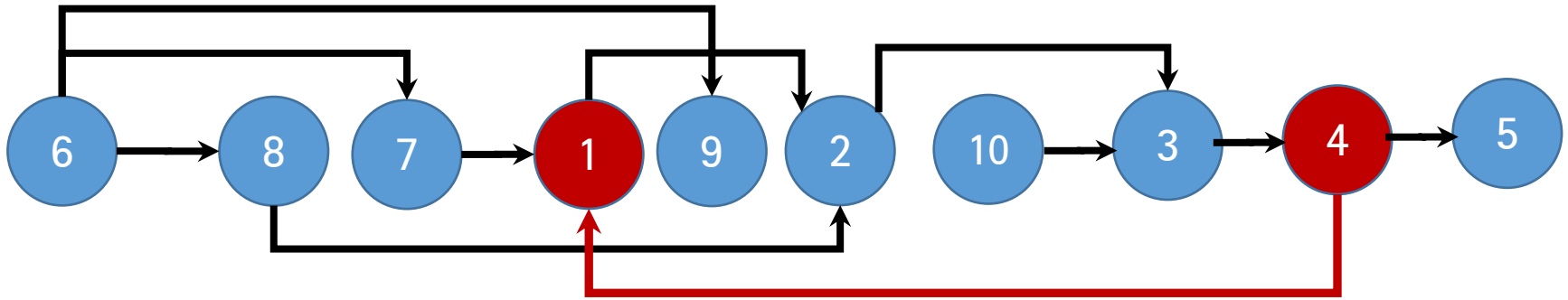


# Topological Sort

## Topological Sort 어떻게 하나요?

방법은 여러가지 이지만 DFS를 이용한 방법이 가장 간단.

1. 임의의 노드에서 DFS 출발! 방문노드를 체크하면서 간다.
2. 방문이 완료된 노드를 vector에 저장
3. 1~2 과정을 방문하지 않은 노드가 없을 때 까지 반복
4. vector를 거꾸로 뒤집어 준다!! (방문 완료의 역순이 TS후 순서)




DFS 종료 역순으로 TS했는데 이런 경우가 발생할 수는 없나요?ㅠㅠ

귀류법으로 상위 경우가 발생할 수 없음이 증명가능.

만약 TS후에 저런 엣지가 있었다고 가정.

(1) visited[1] == false 였다면? dfs(4)가 dfs(1)을 재귀 호출 하게됨 → dfs(1)이 종료한 후에 dfs(4)가 종료하므로 위 경우 처럼 될 수 없음

(2) visited[1] == true 였다면? dfs(1)이 한 번 호출 되었다는 뜻. dfs(4)가 dfs(1)보다 늦게 끝났다는건 dfs(1)이 실행중이라는 의미. dfs(1)→ … → dfs(4)로 호출되는 경로가 있다는 뜻인데, 그림에서 dfs(4)→dfs(1) 엣지가 존재하므로 그래프가 DAG 라는 가정에 모순

* 위 그림은 (2)에 해당

# 예시문제

음악프로그램

https://www.acmicpc.net/problem/2623

```
void dfs(int p)
{
    check[p] = true;
    for( int i = 1 ; i <= n ; i++ )
        if( !check[i] && map[p][i] == 1 )
            dfs(i);
    order.push_back(p);
    return;
}
```

# 위상정렬과제

**Sorting It All Out**
http://poj.org/problem?id=1094

**Window Pains**
http://poj.org/problem?id=2585

**무어기계 (2013 ACM-ICPC 인터넷 예선 문제)**
https://www.acmicpc.net/problem/3300

**Genealogical tree**
http://poj.org/problem?id=2367

**Following Orders**
http://poj.org/problem?id=1270

# 오일러 서킷

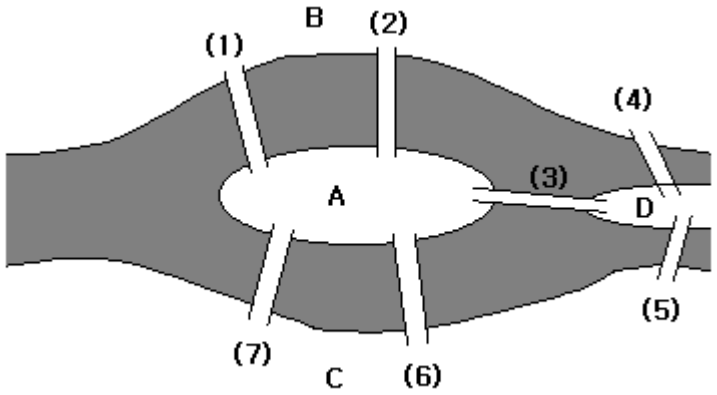
# 오일러 서킷

## 오일러 서킷?(회로)

- 그래프에서 임의의 정점에서 출발해서 그래프상의 모든 간선을 지난 뒤 시작점으로 돌아오는 경로

- 주어진 그래프에 오일러 서킷이 존재하는가? 만 확실하면 찾는건 어렵지 않아요 :D

- 있는지 확인! 있다면 탐색!



# 오일러 서킷

있는지 확인! 사실 확인도 간단합니다.

- 조건1: 주어진 그래프가 하나의 컴포넌트로 연결
- 조건2: 모든 정점의 차수(Degree)가 짝수

두 개의 조건을 만족하는 그래프라면 언제나 오일러 서킷이 존재합니다.

# 오일러 서킷

있다면 탐색! 이해만 하면 쉬워요

- 있다면? 그래프는 하나의 컴포넌트로 연결 && 모든 정점의 차수가 짝수
- 아무 정점에서나 하나의 서킷(회로) A를 찾음 (DFS)
- A가 오일러 서킷이라면? 찾았으니까 종료

# 오일러 서킷

있다면 탐색! 이해만 하면 쉬워요

- 그게 오일러 서킷이아니라면? 서킷상에 아직 사용하지 않은 간선이 있는 정점(오일러서킷 존재조건에 의해 항상존재)으로 부터 새로운 서킷 B를 찾음

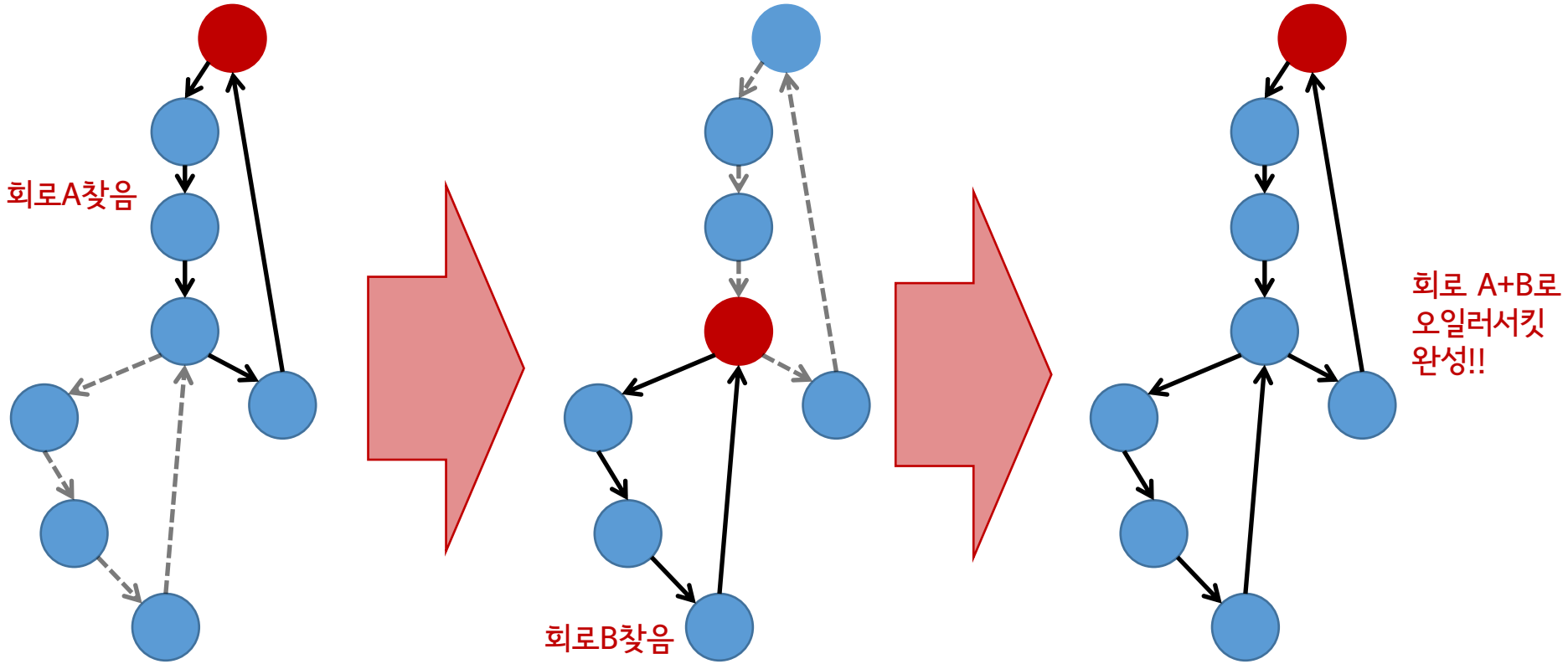
- A사이에 B를 끼워 넣음!

- 간선 다 쓸 때 까지 반복하면 오일러 회로 완성~

- 무슨 소린지 모르겠지

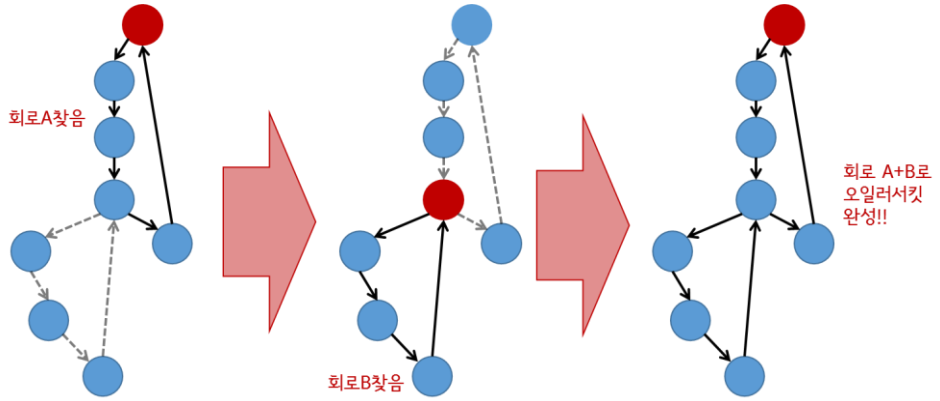
# 오일러 서킷

## 있다면 탐색! 이해만 하면 쉬워요



# 오일러 서킷

## 있다면 탐색! 이해만 하면 쉬워요




## 근데 이걸 어떻게 구현하죠… DFS!

- 회로 탐색을 DFS로 돈다
- 더 이상 방문할 간선이 없는 정점에서 return
- return하는 도중에 정점에 방문하지 않은 간선이 남아있다면 계속 재귀 호출!
- 방문 완료된 정점을 vector에 넣어준다!

# 예시문제

오일러 회로

https://www.acmicpc.net/problem/1199

# 오일러서킷과제

**Sightseeing tour**
http://poj.org/problem?id=1637

**That Nice Euler Circuit**
http://poj.org/problem?id=2284

**Catenyms**
http://poj.org/problem?id=2337

# 오일러 서킷