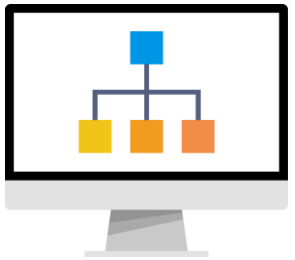


Data Structure & Algorithm
자료구조 및 알고리즘

27. P = NP? 및 마무리

25강 보고서 돌아보기

1) 크루스칼 알고리즘의 최악의 경우 시간복잡도는 얼마일까? 우리가 구현한 알고리즘은 $O(E^2)$
   - 모든 간선을 내림차순 또는 오름차순으로 정렬할 때 힙을 쓴다면 O(E log E)
   - 실제로는 O(E log E)

```
void ConKruskalMST(ALGraph * pg)     // 크루스칼 알고리즘 기반 MST의 구성
{
    Edge recvEdge[20];    // 복원할 간선의 정보 저장
    Edge edge;
    int eidx = 0;
    int i;

    // MST를 형성할 때까지 아래의 while문을 반복
    while(pg->numE+1 > pg->numV)     // MST 간선의 수 + 1 == 정점의 수
    {
        edge = PDequeue(&(pg->pqueue));
        RemoveEdge(pg, edge.v1, edge.v2);

        if(!IsConnVertex(pg, edge.v1, edge.v2))
        {
            RecoverEdge(pg, edge.v1, edge.v2, edge.weight);
            recvEdge[eidx++] = edge;
        }
    }

    // 우선순위 큐에서 삭제된 간선의 정보를 회복
    for(i=0; i<eidx; i++)
        PEnqueue(&(pg->pqueue), recvEdge[i]);
}
```

25강 보고서 돌아보기

1) 그래프 내의 모든 정점을 포함하는 트리 중 **최대**의 가중치 합을 가지는 트리를 **최대 비용** 신장 트리라고 하자. 어떻게 계산할 수 있을까?
   - 간선들의 가중치에 모두 -1을 곱하여 최소 비용 신장 트리를 계산한다.

26강 보고서 돌아보기

- 어떤 그래프 G의 정점의 개수를 V 간선의 개수를 E라고 하고 인접행렬을 사용하여 표현했을 때,

- 다익스트라 알고리즘의 시간복잡도는?
- 플로이드-워셜 알고리즘의 시간복잡도는?
- (선택 질문) 다익스트라 알고리즘을 우리가 배운 자료구조를 활용하여 더 빨리 만들 수 있을까?

26강 보고서 돌아보기

- 다익스트라 알고리즘의 시간복잡도: $O(V^2)$

```
int main() {
    int i, j, v;

    for (i = 0; i < N; i++) for (j = 0; j < N; j++) distance[i][j] = INF;
    for (i = 0; i < N; i++) shortest[i] = INF;

    set_edge(0, 1, 2); set_edge(0, 2, 10); set_edge(1, 2, 1); set_edge(1, 3, 3);
    set_edge(2, 3, 4); set_edge(2, 4, 8); set_edge(3, 4, 2); set_edge(3, 5, 7);
    set_edge(4, 5, 6);

    int start = 0;
    shortest[start] = 0;

    for (i = 0; i < N; i++) {
        v = get_shortest_node();
        visited[v] = 1;

        for (j = 0; j < N; j++) {
            if (shortest[j] > shortest[v] + distance[v][j])
                shortest[j] = shortest[v] + distance[v][j];
        }
    }

    for (i = 0; i < N; i++) {
        printf("From %d, To %d = %d\n", start, i, shortest[i]);
    }
    return 0;
}
```

26강 보고서 돌아보기

- 플로이드-워셜 알고리즘의 시간복잡도: $O(V^3)$

```
int main() {
    int i, j, k;

    for (i = 0; i < N; i++) for (j = 0; j < N; j++) w[i][j] = INF;

    set_edge(0, 1, 2); set_edge(0, 2, 10); set_edge(1, 2, 1); set_edge(1, 3, 3);
    set_edge(2, 3, 4); set_edge(2, 4, 8); set_edge(3, 4, 2); set_edge(3, 5, 7); set_edge(4, 5, 6);

    for (i = 0; i < N; i++) for (j = 0; j < N; j++) d[i][j] = w[i][j];

    for (k = 0; k < N; k++)
        for (i = 0; i < N; i++)
            for (j = 0; j < N; j++)
                if (d[i][j] > d[i][k] + d[k][j])
                    d[i][j] = d[i][k] + d[k][j];

    for (i = 0; i < N; i++) {
        for (j = 0; j < N; j++) {
            if (i != j) printf("From %d, To %d = %d\n", i, j, d[i][j]);
        }
    }
    return 0;
}
```

26강 보고서 돌아보기

- 다익스트라 알고리즘에서는
  - 방문하지 않은 정점 중 거리가 가장 짧은 정점의 대한 거리를 확정하는 단계와, 이 정점으로 부터 나온 가중치를 반영하여 거리를 업데이트 하는 단계가 있다.
  - 이를 힙으로 한다면?
  - O(E log V)

```
for (i = 0; i < N; i++) {
    v = get_shortest_node();
    visited[v] = 1;
    for (j = 0; j < N; j++) {
        if (shortest[j] > shortest[v] + distance[v][j])
            shortest[j] = shortest[v] + distance[v][j];
    }
}
```

컴퓨터로 풀 수 없는 문제들

(이후로 시험 범위 아님)

컴퓨터로 풀 수 없는 문제들?

- 본 강의에서 다양한 문제를 자료구조와 알고리즘을 이용하여 풀어보았다.

- 컴퓨터로 풀지 못하는 문제는 무엇이 있을까?

- 먼저 떠오르는 예
  - 데이터가 엄청 크다면?
  - 무한히 크다면?
  - 간단한 문제도 데이터가 무한정 크면 유한한 메모리를 가지는 컴퓨터로는 풀 수 없다!

컴퓨터로 풀 수 없는 문제들?

- 데이터가 무한정 큰 문제를 풀 수 없다는 것은 자명하다. 다른 예는 없을까?

- 정지 문제: 프로그램의 코드와 입력이 주어졌을 때 프로그램을 실행하고 입력을 넣으면 유한 시간 내에 프로그램이 종료될 것인가? (halting problem)

- 아주 쉬울 것 같은데 실제로는 **계산 불가능한 문제**

컴퓨터로 풀 수 없는 문제들?

- 프로그램 a에 대해 입력 i를 넣었을 때 유한시간 내에 종료되는지를 알려주는 halt(a, i) 함수가 있다고 가정하자.

```
int trouble(s) {
    if(halt(s, s) == false) { return true; }
    else while(1) { }
}
```

- trouble(trouble)의 실행 값은?
- 만약, true라면, halt(trouble, trouble)이 false라는 뜻인데 이는 trouble(trouble)이 멈추지 않는다는 것을 의미한다. 근데 true가 방금 반환되었으니 멈추었다. -> 모순

컴퓨터로 풀 수 없는 문제들?

- 프로그램 a에 대해 입력 i를 넣었을 때 유한시간 내에 종료되는지를 알려주는 halt(a, i) 함수가 있다고 가정하자.

```
int trouble(s) {
    if(halt(s, s) == false) { return true; }
    else while(1) { }
}
```

- trouble(trouble)의 실행 값은?
- 만약, true라면, halt(trouble, trouble)이 false라는 뜻인데 이는 trouble(trouble)이 멈추지 않는다는 것을 의미한다. 근데 true가 방금 반환되었으니 멈추었다. -> 모순

컴퓨터로 풀 수 없는 문제들?

- 프로그램 a에 대해 입력 i를 넣었을 때 유한시간 내에 종료되는지를 알려주는 halt(a, i) 함수가 있다고 가정하자.

```
int trouble(s) {
    if(halt(s, s) == false) { return true; }
    else while(1) { }
}
```

- trouble(trouble)의 실행 값은?
- 만약, 무한 루프를 돈다면, halt(trouble, trouble)이 true라는 뜻인데, 이는 trouble(trouble)이 멈춘다는 것을 의미한다. 근데 현재 무한루프를 돌고 있다. -> 모순

정지 문제

- 따라서, 임의의 프로그램과 입력에 대해 정지 여부를 판단할 수 있는 알고리즘은 존재하지 않는다.

- 물론, 특수한 경우는 가능할 수 있다.

- 여러분들이 제출한 과제 코드에 입력을 넣었는데,
  - 멈춘다? 정답 여부를 판별 가능
  - 멈추지 않는다? 시간이 지나고 정답을 출력하고 끝날 수도 있다! 어떻게 채점하지?

풀기 어려운 문제?

- 무한한 저장 공간이 필요한 문제나, 정지 문제는 컴퓨터로 풀 수 없다.

- 컴퓨터로 풀기 어려운 문제는 없을까?

- “어렵다”가 무슨 뜻?
  - 다항시간 내에 해결할 수 없는 문제들
  - 쉬운 문제: $O(N), O(N^2), O(N \log N), O(N^{100})$
  - 어려운 문제: $O(2^N), O(N!)$

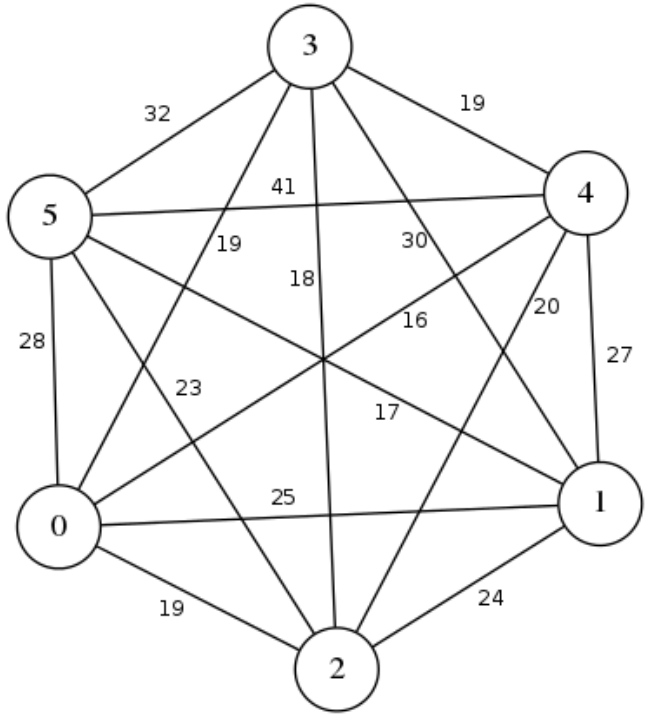
P문제와 NP문제

- P 문제: 다항 시간내에 답을 찾을 수 있는 결정 문제들
  - 우리가 배운 알고리즘들이 해결하는 문제: 정렬, 크루스칼, 다익스트라

- NP 문제: 결정론적 튜링 머신(우리가 아는 컴퓨터)으로 다항 시간 내에 풀어낼 수 없는 결정 문제
  - 풀지는 못하더라도 다항 시간 내에 정답인지는 확인할 수 있다!

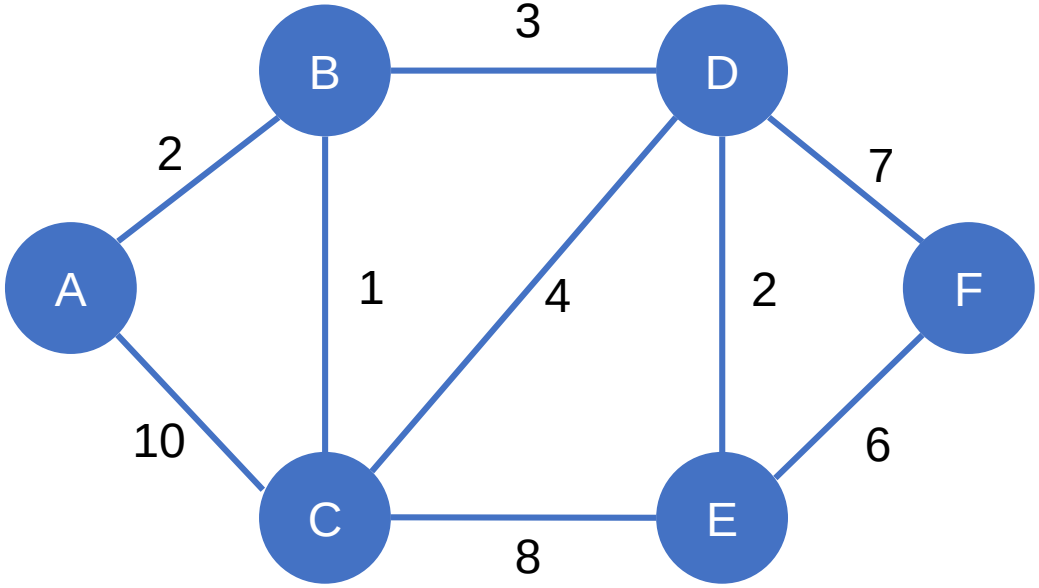
NP 문제의 예제

- Travelling Salesmen Problem (TSP)
  - 그래프의 모든 정점을 방문하고 돌아올 때 가중치의 합이 X이하인 경로가 있는가?
    - = 최소 가중치의 합이 얼마인가?



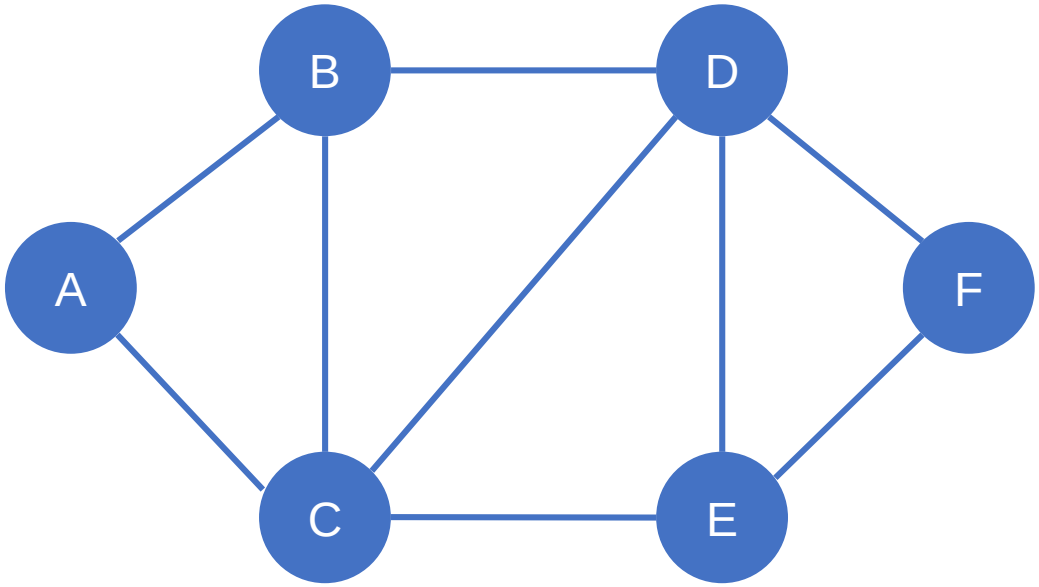
NP 문제의 예제

- Longest Path Problem
  - 주어진 두 정점 사이의 경로 중 가중치의 합이 X 이상인 경로가 있는가? = 최대 가중치의 합이 얼마인가?
  - 최단 경로는 다익스트라로 해결 가능했는데 최장 경로는 NP 문제
  - 가중치를 음으로 바꾸어서 하면 안되나?



NP 문제의 예제

- 3-Coloring Problem
  - 평면 그래프의 정점을 3가지의 색으로 색칠하려고 한다. 인접한 정점은 색이 달라야 한다고 할 때 가능한가?
  - 4가지 색은 가능하다고 했는데.. (4색정리)



P = NP?

- P문제와 NP문제들의 집합이 동일할까? (P = NP?)
  - 다항 시간내에 정답을 확인할 수 있는 문제들은 다항 시간 내에 풀어낼 수도 있는가?

- 컴퓨터과학의 최고 지성들이 연구했지만 아직 풀어내지 못한 문제
  - 해결하면 백만 달러를 수여하는 밀레니얼 문제 중에 하나

강의 정리

- **자료구조** 및 **알고리즘**

- 총 28강 동안,
  - 자료구조: 배열, 리스트, 스택, 큐, 트리, 힙, 해쉬, 그래프
  - 알고리즘: 시간 및 공간복잡도, 재귀 호출, 이진 탐색, 정렬, 전/중/후위 순회, 너비 우선 탐색 및 깊이 우선 탐색, 크루스칼 알고리즘, 다익스트라 알고리즘, 플로이드-워셜 알고리즘, 계산 가능성 등

자료구조를 기억할 때

- 이 자료구조는 데이터를 어떤 형태로 저장하는지?
- 실생활의 어떤 데이터를 모델링하기 위해 사용되는지?
- 삽입/삭제/탐색의 최악의 경우 시간복잡도/공간복잡도가 얼마인지?
- + 각 자료구조의 특이한 성질들

알고리즘을 기억할 때

- 알고리즘의 목적이 무엇인지?
- 알고리즘의 입력과 출력이 무엇인지?
  - 입력과 출력에 어떤 제약사항이 있는지?
- 알고리즘의 최악의 경우 시간복잡도/공간복잡도가 얼마인지?

남은 일정

- 6월 15일 (월): 기말 고사
- 6월 18일 (목): 과제 3 마감

- 만약, 보강 수업이 필요하다면 추후 공지하겠습니다.

- 한 학기 동안 고생하셨습니다!