2022.11.09

KRUSKAL ALGORITHM

SCV Junior

소프트웨어학과 201921017 이지우
jwo2955@gmail.com

001. 트리

- 그래프와 트리
- 신장 트리(ST)
- 최소 신장 트리(MST)

002. 크루스칼 알고리즘

- 작동 방식
- 프로그래밍적 구현

003. Advanced

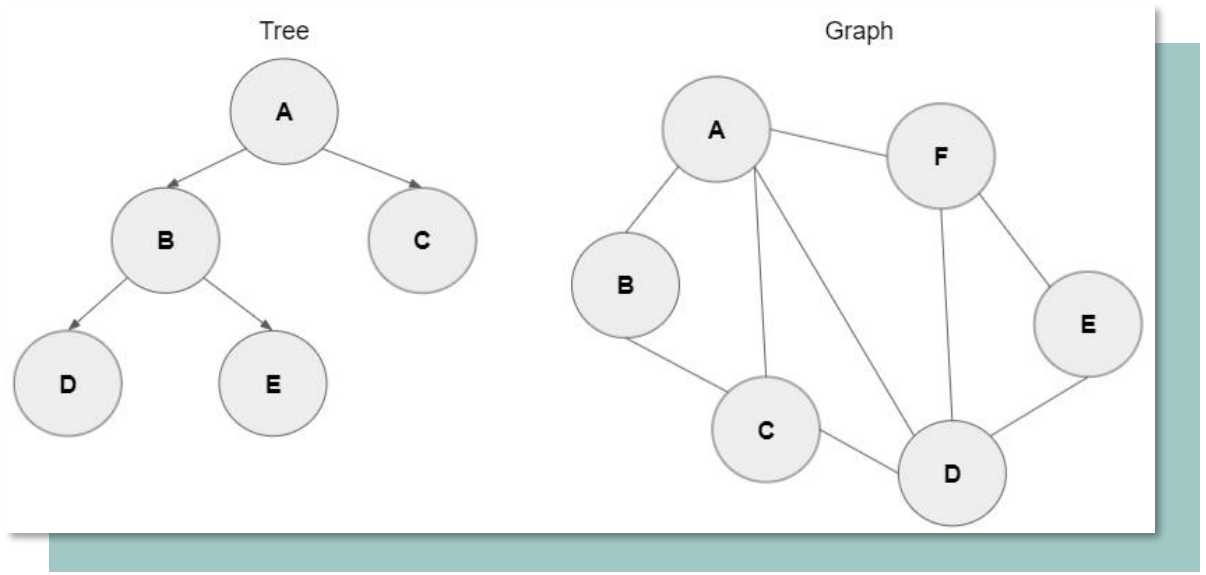
- 문제 풀기
- 프림 알고리즘

001. 트리

001. 트리

그래프와 트리

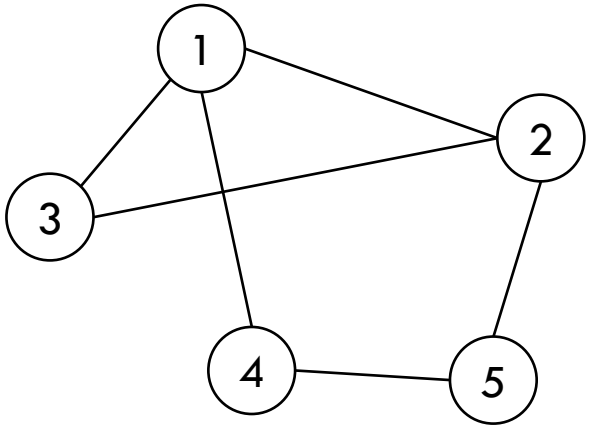
트리는 N개의 정점을 N-1개의 간선으로 모두 연결한 **그래프**
즉, 그래프는 트리를 포함하는 개념



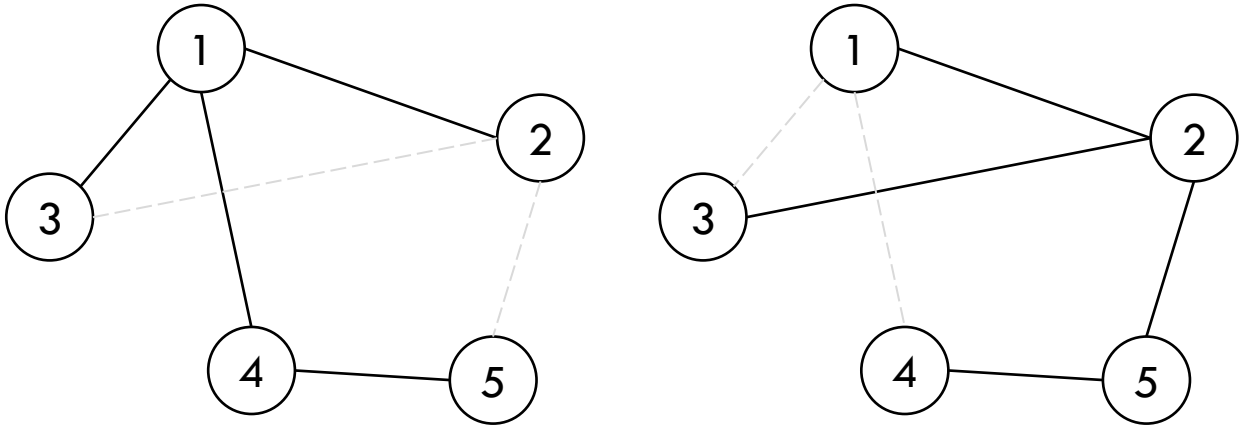
001. 트리

신장 트리

신장 트리(Spanning Tree)는 **기존 그래프의 모든 노드를 포함**하지만
사이클이 존재하지 않는 그래프 == 그래프로부터 뽑아낸 트리



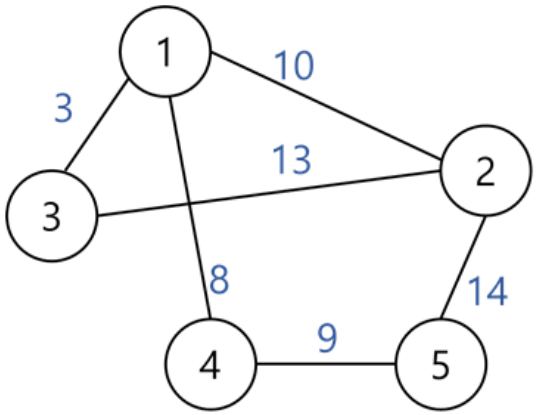
기존 그래프



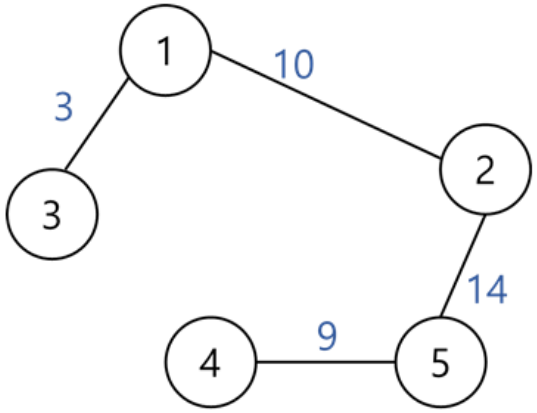
신장 트리

001. 트리

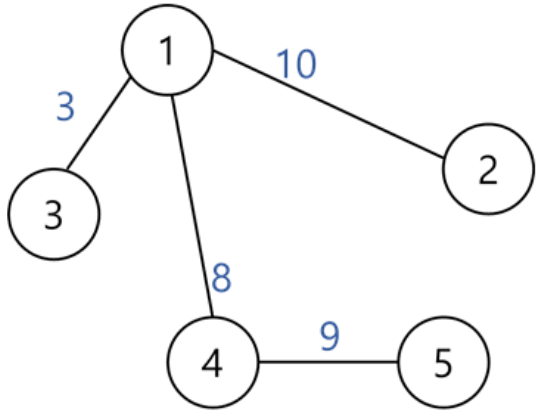
최소 신장 트리



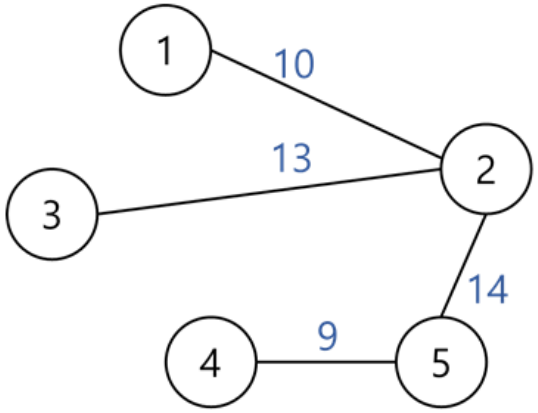
최소 신장 트리(Minimum Spanning Tree)는 신장 트리 중 가중치의 합이 최소인 신장 트리



가중치 합: 36



가중치 합: 30



가중치 합: 46

그래프가 주어졌을 때,
최소 신장 트리를 구하는 방법(알고리즘)은?

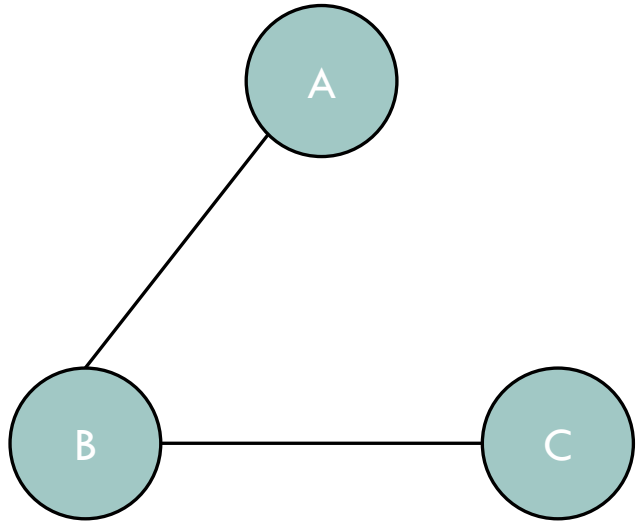
002. 크루스칼 알고리즘

002. 크루스칼 알고리즘

작동 방식

(신장) **트리**는

그래프의 **모든 N개의 정점들을 N-1개의 간선으로 연결한** 연결 그래프



N개의 정점들이 N-1개로 연결되었다 == 사이클을 이루지 않는다

그래프로부터 트리를 만들 때,

사이클을 이루지 않게 하려면

간선으로 연결된 두 정점의 **부모가 다른지 확인**

002. 크루스칼 알고리즘

작동 방식

1. 모든 간선들의 가중치를 **오름차순으로 정렬**
2. 가중치가 가장 작은 간선을 선택
3. 선택한 간선이 사이클을 발생시키는지 확인
    ① 사이클이 발생하지 않는 경우, union
    ② 사이클을 발생하는 경우, pass
4. 2~3을 반복한다

002. 크루스칼 알고리즘

프로그래밍적 구현 – Union-Find

사이클 존재 여부 확인 알고리즘

Union-Find(Disjoing Set, 서로소 집합)

union()

두 노드가 같은 조상을 가지도록 합치는 기능

find()

두 노드의 조상 노드가 같은지를 검사하는 기능

```cpp
int getRoot(vector<int>& parent, int x)
{
    if (parent[x] == x) return x;
    return parent[x] = getRoot(parent, parent[x]);
}

void union(vector<int>& parent, int a, int b)
{
    a = getRoot(parent, a);
    b = getRoot(parent, b);
    if(a < b) parent[b] = a;
    else parent[a] = b;
}

bool find(vector<int>& parent, int a, int b)
{
    a = getRoot(parent, a);
    b = getRoot(parent, b);
    if(a == b) return true;
    else return false;
}
```

002. 크루스칼 알고리즘

프로그래밍적 구현 – Kruskal

```
class Edge{
    int node[2];
    int distance;
    Edge(int a, int b, int distance){
        this->node[0] = a;
        this->node[1] = b;
        this->distance = distance;
    }

    //오름차순 정렬 기준
    bool operator<(Edge &edge){
        return this->distance < edge.distance;
    }
};
```

```
void kruskal(vector<Edge> &graph, int n)
{

    sort(graph.begin(), graph.end());

    vector<int> parent(n+1);

    for(int i=1; i<=n; i++){
        parent[i] = i;
    }

    for(int i=0; i<graph.size(); i++){
        //싸이클 여부 확인
        if(!find(graph[i].node[0], graph[i].node[1])){
            union(graph[i].node[0], graph[i].node[1]);
        }
    }
}
```

003. Advanced

003. Advanced

문제 풀기

1922번: 네트워크 연결 (acmicpc.net)

그래프가 주어졌을 때,
최소 신장 트리를 구하는 방법(알고리즘)은? : 크루스칼 알고리즘 또는 프림 알고리즘