MST

Minimum Spanning Tree

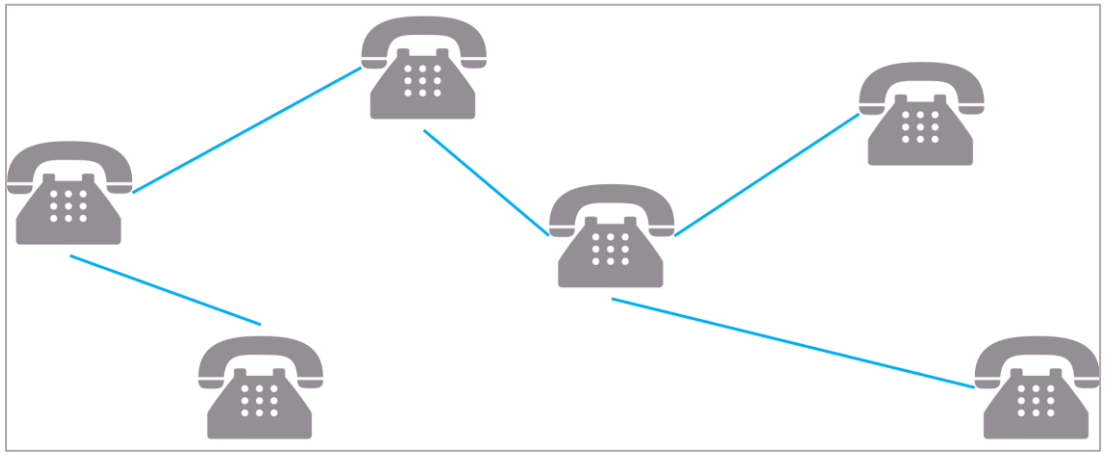
2020.08.09

Minimum Spanning Tree

[Minimum Spanning Tree]

- **그래프**의 간선에 저장된 가중치로 만들어진 **관계 트리** –

어떤 일의 최소한의 비용, 최소한의 거리, 스케쥴링, 최소 대기시간 등등으로 실생활에서 낭비방지, 최소소비, 가성비를 구하는데에 쓰일 수 있다.



대표적으로 통신에 자주 사용된다.

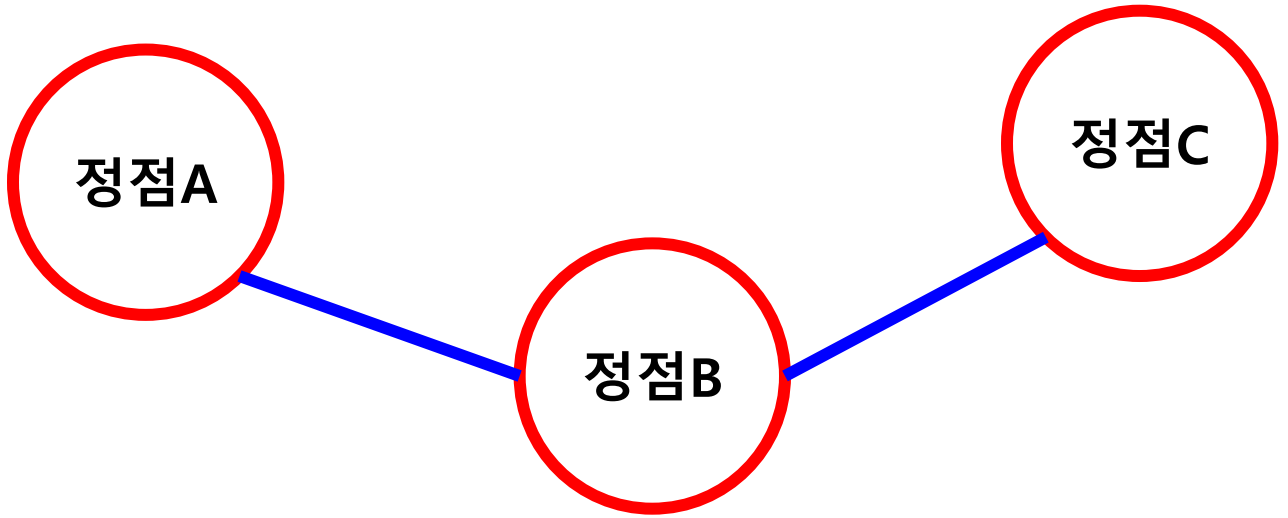
CONTENS

- Minimum Spanning Tree
  - Graph
    - Prim’s Algorithm
    - Kruskal’s Algorithm

Graph

- 사물 간의 관계, 데이터와 데이터의 관계, 순서를 표현할 때 유용한 자료구조
- 정점 (vertex) : 위치
- 간선 (edge) : 정점과 정점을 잇는 선

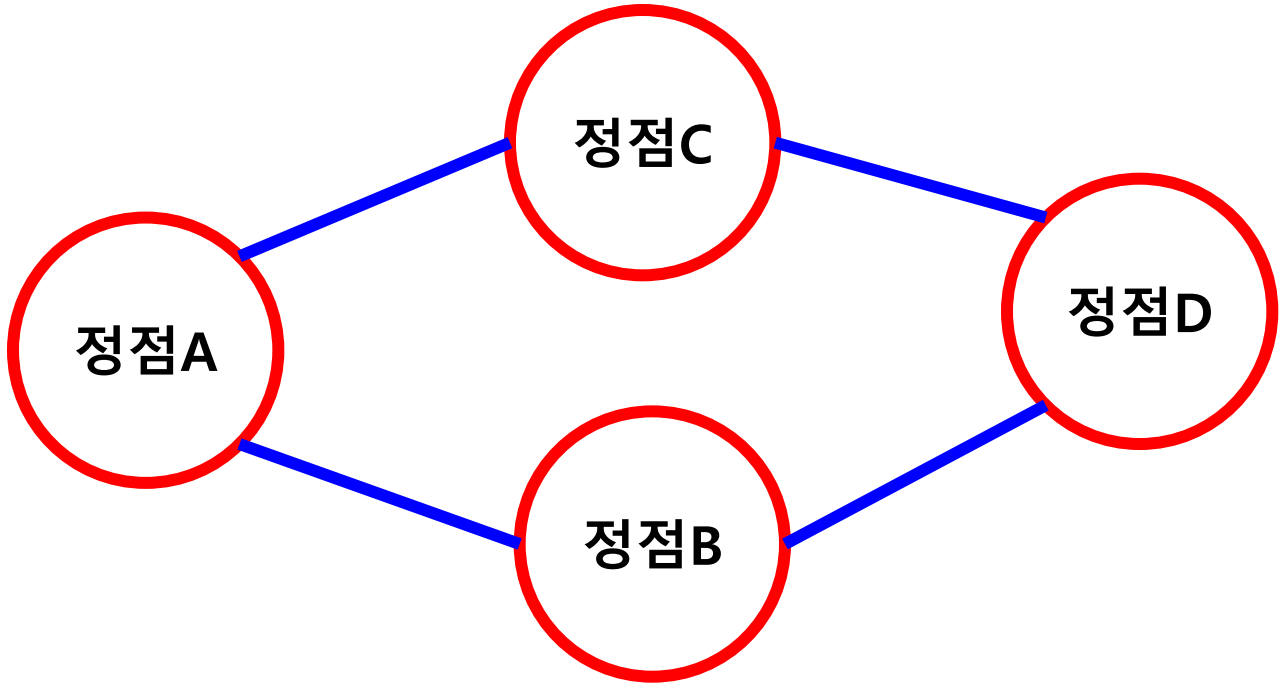
[정의] 정점들의 집합 V, 간선들의 집합 E가 있고
그래프가 G라고 할 때, G = (V , E)이다



Graph

[그래프의 특징]

1. **인접** : 한 정점이 간선으로 연결된 다른 정점들을 “인접되어 있다” 또는인접한 관계”, “이웃”이라고 한다.
2. **경로** : 한 정점에서 다른 한 정점으로 가는 길
3. **경로의 길이** : 한 정점에서 다른 한 정점에 도착할 수 있는 경로의 개수
4. **사이클** : 한 정점에서 다시 그 정점으로 돌아올 수 있는 경로

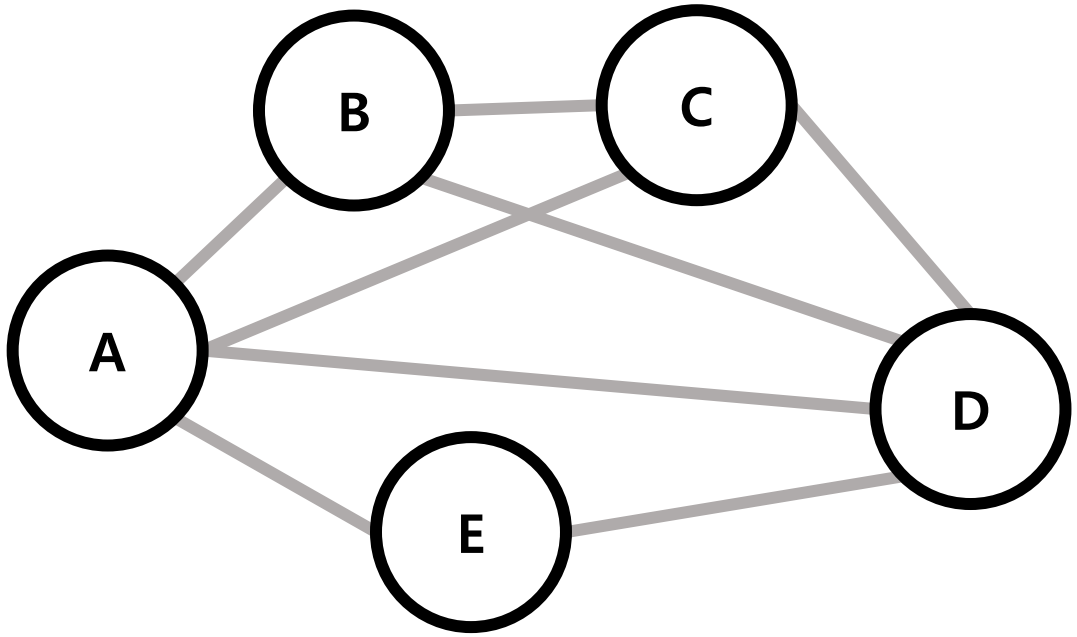


Graph

[관계표현1]

인접행렬 : 정점의 인접관계를 나타낸 행렬

1. 한 그래프의 정점의 수가 N이라 할 때 N x N의 행렬을 만든다.
2. 한 정점과 다른 정점이 인접되어 있는 경우 1 아니면 0으로 표시한다.



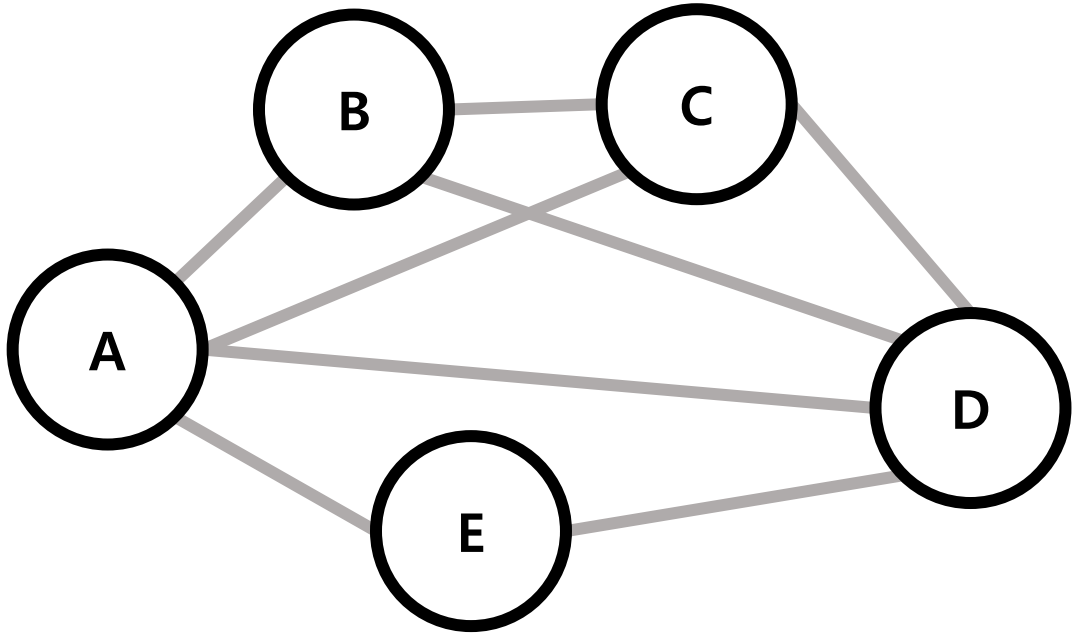
-	A	B	C	D	E
A	0	1	1	1	1
B	1	0	1	0	1
C	1	1	0	0	0
D	1	0	0	0	1
E	1	1	0	1	0

Graph

[관계표현2]

인접리스트 : 정점의 인접관계를 나타낸 리스트

1. 한 그래프의 정점의 수가 N이라 할 때 리스트 N개를 만든다.
2. 한 정점의 인접된 정점들을 해당 정점의 인접리스트에 추가한다.



LIST	Vertex
A	B, C, D, E
B	A, C, D
C	A, B, D
D	A, B, C, E
E	A, C, E

Graph

[행렬 vs 리스트]

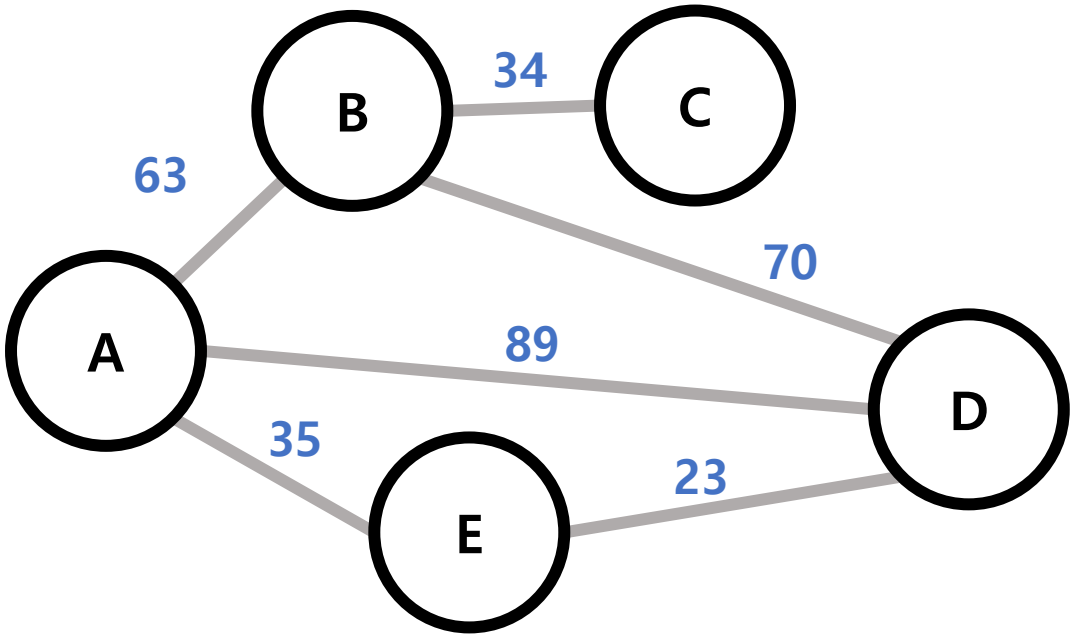
행렬 : 정점의 크기에 따른 메모리(N^2) 소비가 크지만
탐색이 빠름[O(1)]

리스트 : 정점의 추가와 삭제이 가능하고 메모리 관리가 용이하다.
하지만 순차탐색[O(n)]으로 관계를 알 수 있다.

Graph

[간선의 가중치]

가중치 : 간선 노드에 저장되어 있는 데이터이며 쓰임에 따라
예시로 지하철 거리나 비용, 우선순위가 될 수 있다.



Minimum Spanning Tree

[Minimum Spanning Tree = Minimum Weight Spanning Tree]

- **그래프**에 있는 간선의 가중치로 만든 관계 트리 –

< 만드는 방법 > 그래프를 트리로 만드는 방법

프림 알고리즘

크루스칼 알고리즘

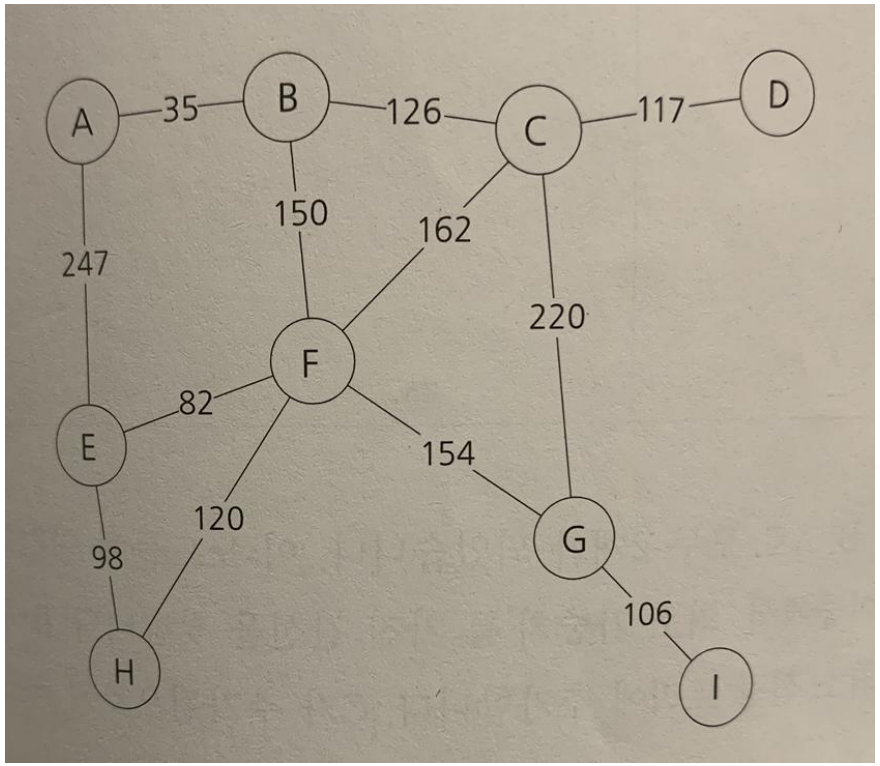
Prim's Algorithm

한 정점에서 이웃 정점들과 연결된 간선의 가중치가
가장 낮은 정점을 자신의 자식으로 두어 트리를 형성하는 알고리즘

단, 사이클을 형성되지 않아야 한다.

1. 그래프 내에 임의의 한 정점을 선택한다.
2. 선택한 정점에 이웃과 연결된 간선들 중 가장 낮은 가중치를 자신의 자식으로 둔다.
3. 2를 반복하며 알고리즘 종료

[사이클 방지방법 : 방문했던 노드를 무시한다.]



Kruskal's Algorithm

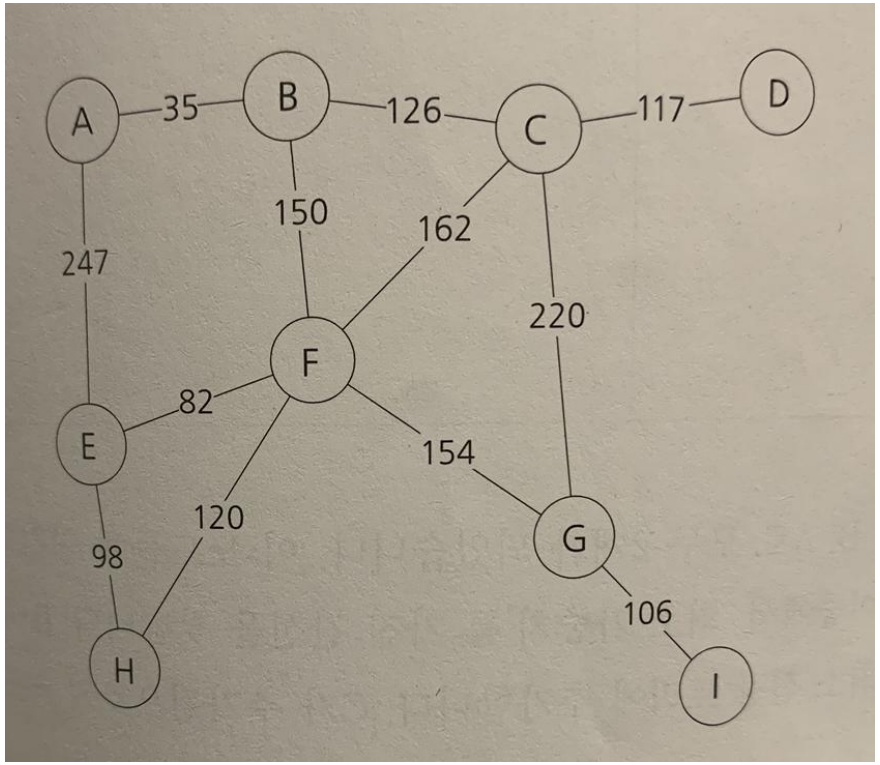
그래프 내에서 연결된 두 정점의 간선들 중에 가중치가 가장 낮은 것으로 트리를 형성하는 알고리즘

단, 사이클을 형성되지 않아야 한다.

1. 그래프 내에 있는 두 정점이 연결된 모든 간선의 가중치 오름차순으로 정리한다.
2. 가중치가 낮은 집합부터 형성해 나가면서 서로가 분리 집합이라면 합집합을 하여 트리를 형성한다.
3. 2을 반복하며 알고리즘 종료

[사이클 방지 : 한 집합 내에서 연결될 경우 무시한다.]

A – B : 35
E – F : 82
E – H : 98
G – I : 106
C – D : 117
F – H : 120
B – C : 126
B – F : 150
F – G : 154
C – F : 162
C – G : 220
A – E : 247



Prim's Algorithm VS Kruskal's Algorithm

크루스칼 알고리즘 시간 복잡도 :: **O(Elog2E)**

프림 알고리즘 시간 복잡도 :: **O(Elog2V)**

간선의 개수가 작은 경우에는 **크루스칼 알고리즘**을
간선의 개수가 많은 경우에는 **프림 알고리즘**이 좋다.

E = edge
V = vertex

Source of Reference

[자료구조] 그래프(Graph)란
https://victorydntmd.tistory.com/102

[알고리즘] MST(3) - 프림 알고리즘 (Prim's Algorithm)
https://gmlwjd9405.github.io/2018/08/13/data-structure-graph.html