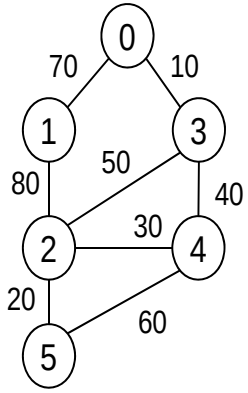
최소비용신장트리

최소비용신장트리: Kruskal's 알고리즘

Reference: https://en.wikipedia.org/wiki/Kruskal's_algorithm, CC-BY-SA
Reference: p.291 in (Horowitz et al., 1993)

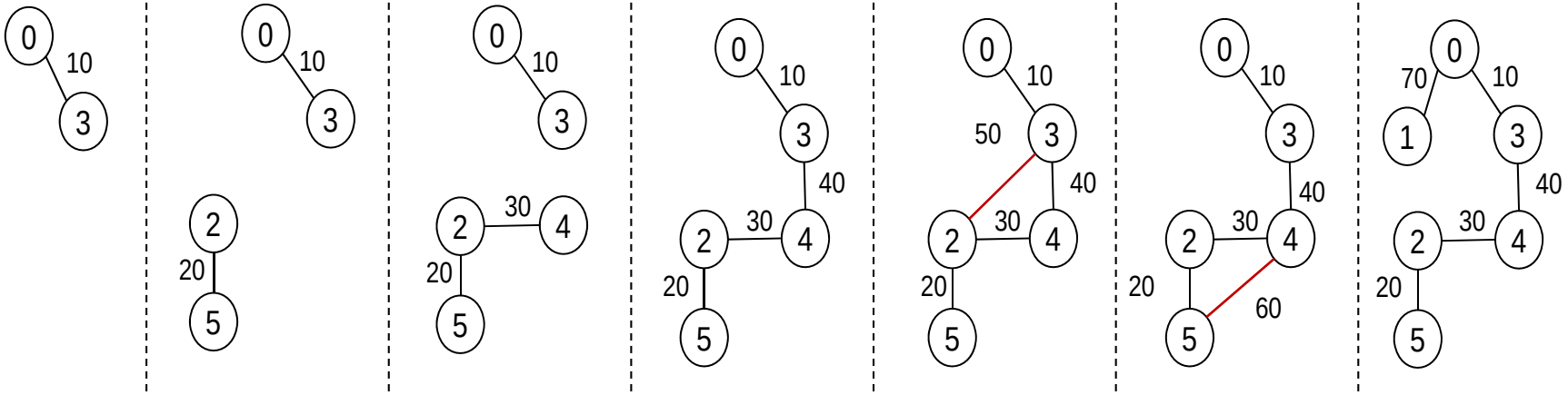


Kruskal's MST(minimum spanning tree, 최소비용신장트리) algorithm

- 입력 그래프의 총 노드 수는 n으로 가정
- T는 MST를 구성하는 간선들의 집합으로 최초 공집합
- E는 입력 그래프 내 모든 간선들의 집합

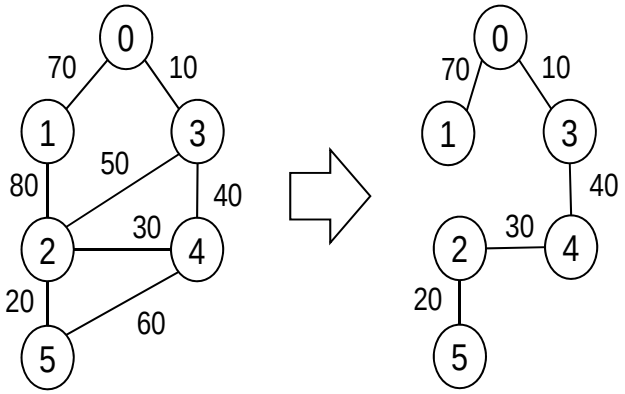
- T의 크기가 n-1미만이거나 E가 공백이 아닌 동안 다음 작업을 반복한다.
  (1) 입력 그래프로부터 최소비용간선 e를 제거
  (2) e를 T에 추가했을 때 사이클이 발생하지 않으면 e를 T에 추가

- T의 크기가 n-1 미만이면 MST 발견 불가



최소비용신장트리: Kruskal's 알고리즘

```
static class Edge implements Comparable<Edge>{
        int     v1,v2;
        double          weight;
        public Edge(int v1, int v2, double weight) {
                this.v1=v1; this.v2=v2; this.weight=weight;
        }
        @Override
        public int compareTo(Edge that) {
                return this.weight<that.weight? -1 : this.weight>that.weight? 1 : 0;
        }
        @Override
        public String toString() { return "("+v1+","+v2+","+weight+")"; }
}
private static LinkedList<Edge> KruskalMST(LinkedList<Edge>[] adjList, int V) {
        LinkedList<Edge>  mst=new LinkedList<>();
        LinkedList<Edge>  edges=new LinkedList<>();
        for (int i = 0; i < adjList.length; i++) edges.addAll(adjList[i]);
        PriorityQueue<Edge>       minPQ=new PriorityQueue<>(edges);
        UF      uf=new UF(V);
        while(mst.size()<V-1 && minPQ.size()>0){
                Edge  e=minPQ.remove();
                if(uf.find(e.v1)==uf.find(e.v2)) continue;
                mst.add(e);
                uf.union(e.v1, e.v2);
        }
        return mst;
}
```



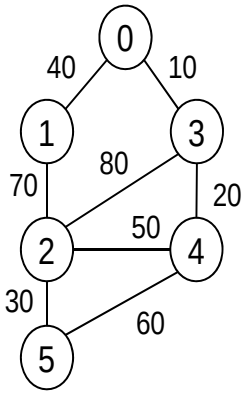
System.out.println(KruskalMST(adjList, V));

[(0,3,10.0), (2,5,20.0), (2,4,30.0), (3,4,40.0), (0,1,70.0)]

Reference: https://en.wikipedia.org/wiki/Kruskal's_algorithm, CC-BY-SA
Reference: p.291 in (Horowitz et al., 1993)
https://algs4.cs.princeton.edu/43mst/KruskalMST.java.html, GPLv3

최소비용신장트리: Prim's 알고리즘

Reference: https://en.wikipedia.org/wiki/Prim's_algorithm, CC-BY-SA
Reference: p.293 in (Horowitz et al., 1993)

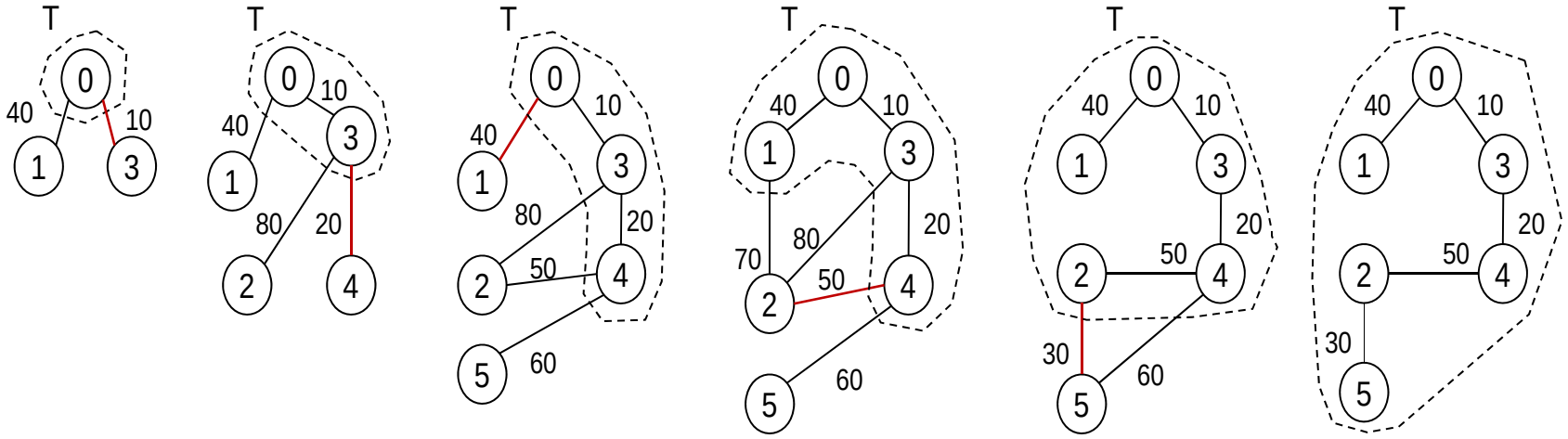


Prim's MST(minimum spanning tree, 최소비용신장트리) algorithm

- 입력 그래프의 총 노드 수는 n으로 가정
- T는 MST를 구성하는 간선들의 집합으로 최초 공집합
- TV는 MST에 속해 있는 정점들의 집합으로 최초 임의 정점 포함. TV={0}

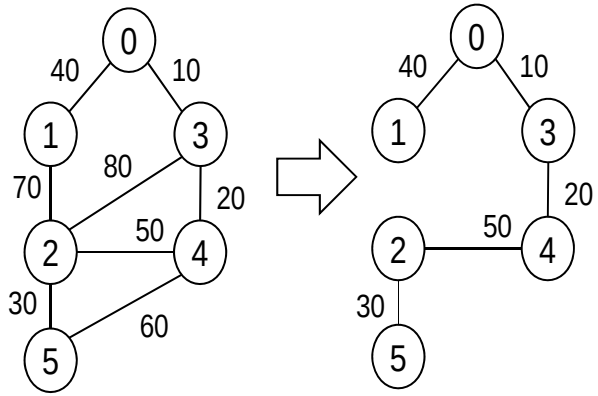
- T의 크기가 n-1 미만인 동안 다음 작업을 반복한다.
  (1) u∈TV 이고 v∉TV인 최소비용간선 (u, v)를 T에 추가하고 v를 TV에 추가
  (2) 그러한 최소비용간선 미발견시 break

- T의 크기가 n-1 미만이면 MST 발견 불가



최소비용신장트리: Prim's 알고리즘

```
static class Edge implements Comparable<Edge>{
        int     v1,v2;
        double          weight;
        public Edge(int v1, int v2, double weight) { this.v1=v1; this.v2=v2; this.weight=weight; }
        @Override
        public String toString() { return "("+v1+","+v2+","+weight+")"; }
        @Override
        public int compareTo(Edge that) {
                return this.weight<that.weight? -1 : this.weight>that.weight? 1 : 0;
        }
}
private static LinkedList<Edge> PrimMST(LinkedList<Edge>[] adjList, int V) {
        LinkedList<Edge>  mst=new LinkedList<>();
        PriorityQueue<Edge>       minPQ=new PriorityQueue<>(adjList[0]);
        boolean         mstV[]=new boolean[V];
        mstV[0]=true;
        while(mst.size()<V-1 && minPQ.size()>0){
                Edge  e=minPQ.remove();
                if(mstV[e.v2]) continue;
                mst.add(e);
                mstV[e.v2]=true;
                for (Edge outEdge : adjList[e.v2]) if(mstV[outEdge.v2]==false) minPQ.add(outEdge);
        }
        return mst;
}
```



System.out.println(PrimMST(adjList, V));

[(0,3,10.0), (3,4,20.0), (0,1,40.0), (4,2,50.0), (2,5,30.0)]

Reference: https://en.wikipedia.org/wiki/Prim's_algorithm, CC-BY-SA
Reference: p.293 in (Horowitz et al., 1993)
https://algs4.cs.princeton.edu/43mst/PrimMST.java.html, GPLv3

References

- C로 쓴 자료구조론 (Fundamentals of Data Structures in C, Horowitz et al.). 이석호 역. 사이텍미디어. 1993.
- 쉽게 배우는 알고리즘: 관계 중심의 사고법. 문병로. 한빛아카데미. 2013.
- C언어로 쉽게 풀어 쓴 자료구조. 천인국 외 2인. 생능출판사. 2017.
- 프로그래밍 콘테스트 챌린징, Akiba 등 공저, 로드북, 2011.
- https://introcs.cs.princeton.edu/
- Introduction to Algorithms, Cormen et al., 3rd Edition (The MIT Press)