FINDING THE SHORTEST PATH TO LAY CABLES ACROSS A CITY OR A GROUP OF CITIES

A MINOR PROJECT REPORT

Submitted by

DEVADITYA SINGH [Reg No: RA2011003011049]
KEERTHI KIRITI CHAKALI [Reg No: RA2011003011068]
ADITHYA DUTT KAMBHAMPATI [Reg No: RA2011003011072]

Faculty Name: S INIYAN

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

of

FACULTY OF ENGINEERING AND TECHNOLOGY

S.R.M. Nagar, Kattankulathur, Kancheepuram District

June 2022

DAA PROJECT

Finding the shortest path to lay cables across a city or group of cities.

Contents

Contribution table

Name	Reg. No.	Contribution
Adithya Dutt Kambhampati	RA2011003011072	Code and Algorithm
Chakali Murthy Gari Keerthi Kiriti	RA2011003011068	Analysis and testing of code
Devaditya Singh	RA2011003011049	Research

Problem definition

In it, we are using Kruskal's minimum spanning tree to find the most efficient way of laying the cable across a city or a group of cities

Problem Explanation with a diagram:

Let's simplify the map by converting it into a graph as below and naming important locations on the map with lettersand distance in meters (x 100).

Cannaregio	Ponte Scalzi	Santa Croce	Dell 'Orto	Ferrovia	Piazzale Roma	San Polo	Dorso Duro	San Marco	St. Mark Basilica	Castello	Arsenale
A	B	C	D	E	F	G	H	I	J	K	L

Let's understand how Kruskal's algorithm is used in the real-world example using the above map.

Step 1- Remove all loops and parallel edges So for the given map, we have a parallel edge running between Madonna dell Orto (D) to St. Mark Basilica (J), which is of length 2.4kms(2400mts).

We will remove the parallel road and keep the 1.8km (1800m) length for representation.

Step 2 – Arrange all the edges on the graph in ascending order. Kruskal's algorithm considers each group as a tree and applies disjoint sets to check how many of the vertices arepart of other trees.

B,C	1
I,J	1
B,E	2
C,G	2
G,I	2
C,D	2
K,L	3
E,F	4
A ,B	6
A,C	6
A,D	6
E,C	7
J,L	8
F,H	10
F,G	11
H,I	12
I,K	16
D,J	18
G,H	22
H,K	25

Step 3 –Add edges with least weight; we begin with the edges with least weight/cost. Hence, B, C is connected first considering their edge cost only 1.

I, J has cost 1; it is the edge connected next.

Then, we connect edges with weight = 2.

Similarly, we connect node K, Lwhich has an edge with weight = 3.

As given in the table above, all the edges are connected in ascending order, ensuring no loop or cycle is formed between 2 vertices.

Thisgives us the following graph, which is the minimum spanning tree forthe given problem.

Design technique used: - Greedy Approach

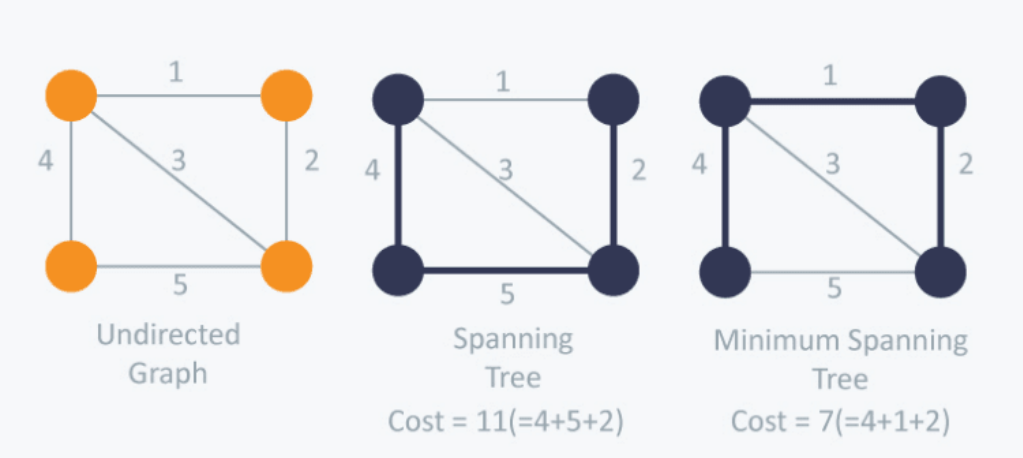
A greedy algorithm is an approach for solving a problem by selecting the best option available at the moment. It doesn't worry whether the current best result will bring the overall optimal result. The algorithm never reverses the earlier decision even if the choice is wrong. It works in a top-down approach.

An algorithm to construct a Minimum Spanning Tree for a connected weighted graph. It is a Greedy Algorithm. The Greedy Choice is to put the smallest weight edge that does not because a cycle in the MST constructed so far. If the graph is not linked, then it finds a Minimum Spanning Tree.

A spanning tree is the sum of weights of all the edges in a tree.

A minimum spanning tree (MST) is one which costs the least amongall spanning trees.

Here is an example of a minimum spanning tree.



Kruskal's algorithm uses the greedy approach for finding a minimum spanning tree.

Kruskal's algorithm treats every node as an independent tree and connects one with another only if it has the lowest cost compared to all other options available.

Algorithm of the problem

```
MST_Kruskal(Edges, V, E){
    e=0, i=0;
    sum=0;
    Sort(Edges);
    while(e<V-1){
        u=Edges[i].u;
        v=Edges[i].v;
        if(Adding edge {u, v} do not form cycle){
            Print(Adding edge {u, v} to MST);
            sum+=Edges[i].weight;
            e+=1;
        }
        i+=1;
    }
}
```

Explanation of Algorithm With an Example

The main idea of Kruskal's algorithm to find the Minimum Spanning Tree of a graph G with V vertices and E edges is:

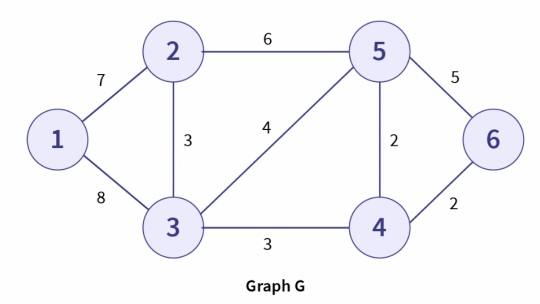
- To sort all the edges in ascending order according to their weights.
- Then select the first V-1 edges such that they do not form any cycle within them.

For any graph, G=(V,E), finding a Minimum Spanning Tree using Kruskal's algorithm involves the following steps -

- Sort all the edges of the graph in ascending order of their weights.
- Check the edge with minimum weight, if including it in the answer forms a cycle discard it, otherwise include it in the answer.
- Repeat the above step until we have chosen V-1 edges.

Example of Kruskal's Algorithm

Let's say we want to find the MST of the underlying connected, weighted, undirected graph with 6 vertices and 9 edges-



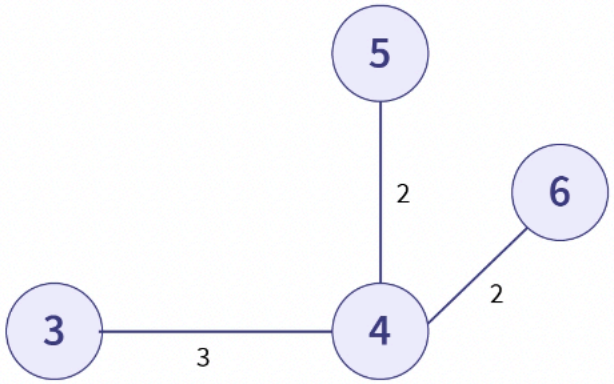
Graph G

Step 1 - Sort the edges in ascending order. Here is the list after sorting.

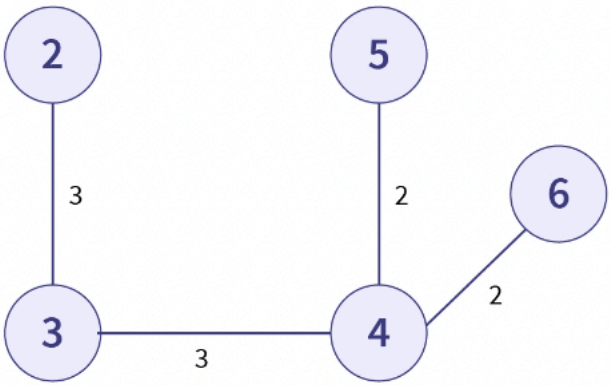
No.	u	v	Weight
1	4	5	2
2	4	6	2
3	3	4	3
4	2	3	3
5	3	5	4
6	5	6	5
7	2	5	6
8	1	2	7
9	1	3	8

Step 2 - Choose 5 edges from starting which do not form a cycle.

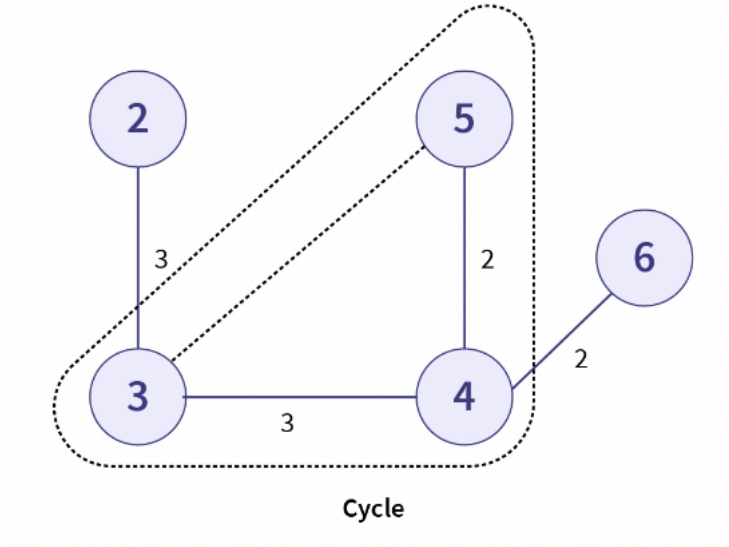
Step 3 -



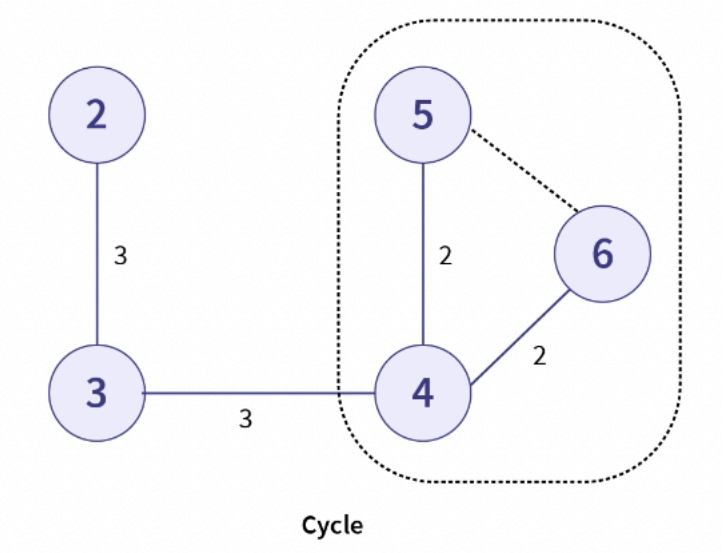
Step 4 -



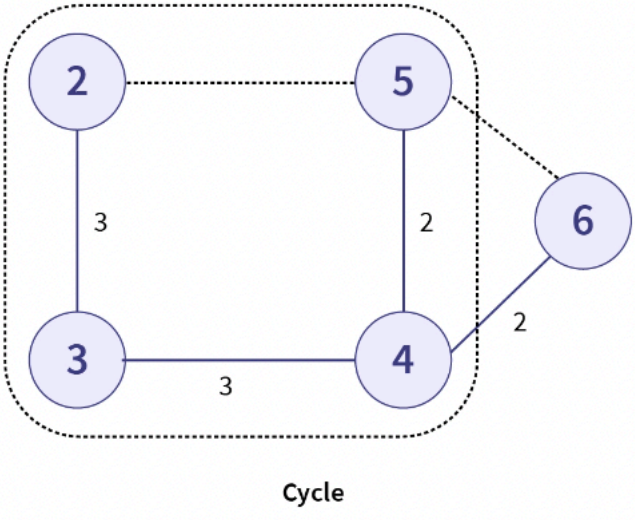
Step 5:



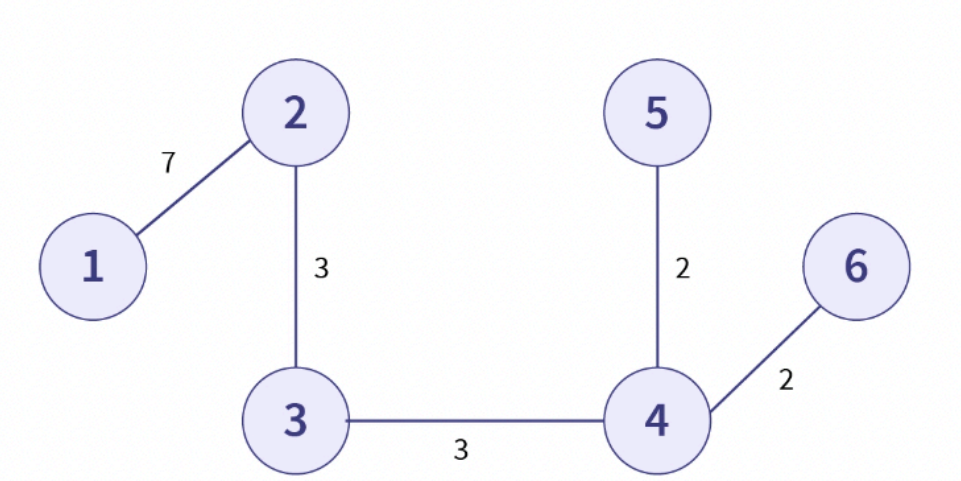
Step 6:



Step 7:



Step 8: After including all 5 edges, the MST will look like this:



So the weight of the MST can be calculated as :

7 + 3 + 3 + 2 + 2 = 17

Complexity Analysis

- Time Complexity-
  - Sorting of E edges costs us $O(E \times \log(E))$ time.
  - For each edge, we are using the Union-FInd algorithm which costs us $O(E \times a(V))$ time.
  - Hence, the overall time complexity is $O(E \times \log(E) + E) \simeq O(E \times \log(E))$

- Space Complexity-
  We are using a List/vector to store the E edges of the given graph so the space complexity is $O(E)$.

Conclusion:

We have successfully demonstrated Kruskal's algorithm using the greedy method to find the shortest path to lay wires in a particular city or a place. This project helps to reduce the cost of wiring across a place while maintaining the quality of work.

References:

- https://www.scaler.com/topics/data-structures/kruskal-algorithm/
- https://en.m.wikipedia.org/wiki/Kruskal's_algorithm
- https://www.tutorialspoint.com/data_structures_algorithms/kruskals_spanning_tree_algorithm.htm
- https://www.hackerearth.com/blog/developers/kruskals-minimum-spanning-tree-algorithm-example/