Tutorial-6

1. What do you mean by minimum spanning tree? What are the applications of MST?

→ A minimum spanning tree (MST) is a subset of the edges of a connected, edge-weighted undirected graph that connects all the vertices together, without any cycles & with the minimum possible total edge weight.

Applications:

(i) Consider n stations are to be linked using a communication network & laying of communication link between any two stations involves a cost.
(ii) The ideal solution would be to extract a subgraph termed as minimum cost spanning tree.

(ii) Suppose you want to construct highways or railroads spanning several cities then we can use the concept of MST.

(iii) Designing LAN.

(iv) Laying pipelines connecting offshore drilling sites, refineries & consumer markets.

(v) Suppose you want to apply a set of houses with
- Electric power
- Water
- Telephone lines
- Sewage lines.

2. Please analyse the time and space complexity of Prim, Kruskal, Dijkstra and Bellman ford algorithm.

→ Prim's →

Time Complexity: $O(|E| \log |V|)$
Space Complexity: $O(|V|)$

Kruskal's →

Time Complexity: $O(|E| \log |E|)$
Space Complexity: $O(|V|)$

Dijkstra →

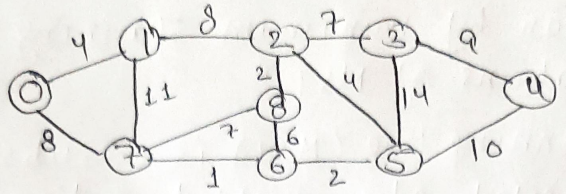
Time Complexity: $O(V^2)$
Space Complexity: $O(V^2)$

Bellman Ford →

Time Complexity: $O(VE)$
Space Complexity: $O(E)$

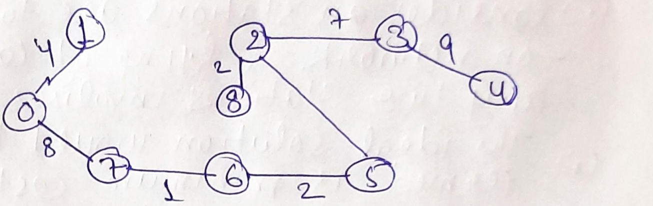
3. Apply Kruskal and Prim's algo on given graph to compute MST and its weight.

→



Kruskal's algorithm

(source) O	(destination) V	W (weight)	
6	7	1	✓
5	6	2	✓
2	8	2	✓
0	1	4	✓
2	5	4	✓
6	8	6	×
2	3	7	✓
7	8	7	×
0	7	8	✓
1	2	8	×
4	3	9	✓
4	5	10	×
1	7	11	×
3	5	14	×



weight = 1+2+2+4+4+7+8+9
= 37

Prim's algorithm:

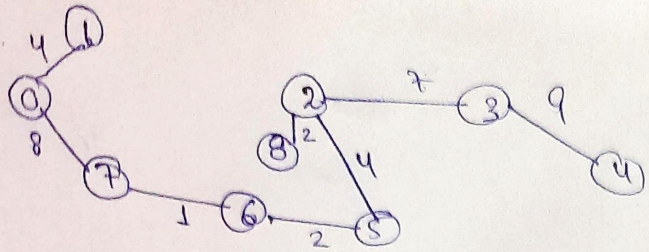
Consider the graph in question

Weight

0	1	2	3	4	5	6	7	8
0 ~~∞~~	∞	∞	∞	∞	∞	∞	∞	∞
							$\boxed{8}$	
	$\boxed{4}$	$\boxed{8}$						7
	11					$\boxed{1}$		
			7		4			$\boxed{2}$
					$\boxed{2}$			6
		$\boxed{4}$	14	10				
			$\boxed{7}$					
				$\boxed{9}$				

Parent :-

0	1	2	3	4	5	6	7	8
-1	~~-1~~ 0	~~-1~~ 1	-1	-1	-1	~~-1~~ 1	~~-1~~ 0	-1



Weight = 4+8+1+2+4+2+7+9 = 37

4. Given a directed graph (weighted). You are also given the shortest path from a source vertex 's' to a destination vertex 't'. Does the shortest path remain same in the modified graph in following cases.

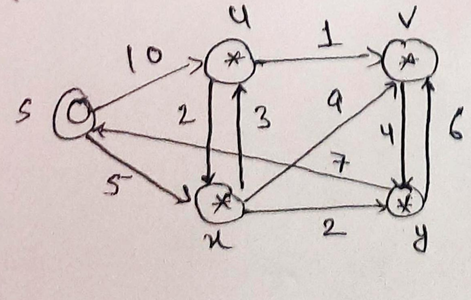
(i) If weight of every edge is increased by 10 units.

(ii) If weight of every edge is multiplied by 10 units.

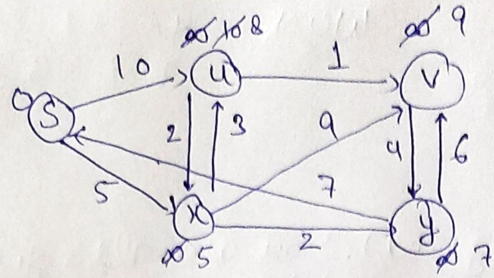
→ (i) The shortest path may change. The reason is there may be different number of edges in different paths from 's' to 't'. e.g. shortest path be of weigh 15 and has 5 edges. Let there be another path with 2 edges & total weight is 25. The weight of shortest path is increased by 5*10 & becomes 15+50. Weight of other path is increased by 2*10 & becomes 25+20. So the shortest path changes to the other path that has weight 45.

(ii) If we multiply all edges weight by 10, the shortest path doesn't change. The reason is simple weights of all paths from 's' to 't' get multiplied by some amount. The no. of edges on a path doesn't matter. It is like changing units of weights.

5. Apply Dijkstra and Bellman algorithm on graph to compute shortest path to all nodes from node s.



→ Dijkstra algorithm :



node	shortest dist from source node
u	8
x	5
v	9
y	7

Bellman ford algorithm

1st →	S 0	u ~~10~~ ~~∞~~	v ∞	x ~~∞~~ 5	y ∞
2nd →	S 0	u 10	v ~~∞~~ 11	x 5	y ∞
3rd →	S 0	u ~~10~~ 8	v ~~11~~ 9	x 5	y ~~∞~~ 7
4th →	S 0	u 8	v 9	x 5	y 7

graph doesn't have -ve cycle.

final graph :-