Data Structure & Algorithms

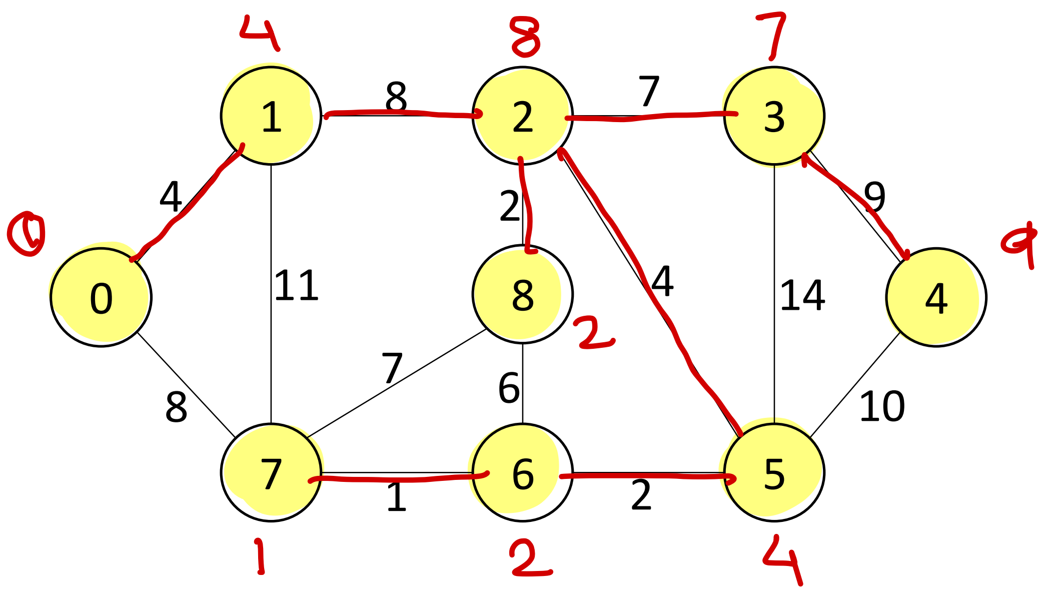
Sunbeam Infotech

Prim's MST

1. Create a set mstSet that keeps track of vertices already included in MST.
2. Assign a key value to all vertices in the input graph. Initialize all key values as INFINITE. Assign key value as 0 for the first vertex so that it is picked first.
3. While mstSet doesn't include all vertices
   i. Pick a vertex u which is not there in mstSet and has minimum key value.
   ii. Include u to mstSet.
   iii. Update key value of all adjacent vertices of u. For every adjacent vertex v, if weight of edge u-v is less than the previous key value of v, update the key value as weight of u-v.

Parent

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



mst

0	T
1	F
2	T
3	F
⋮	
8	

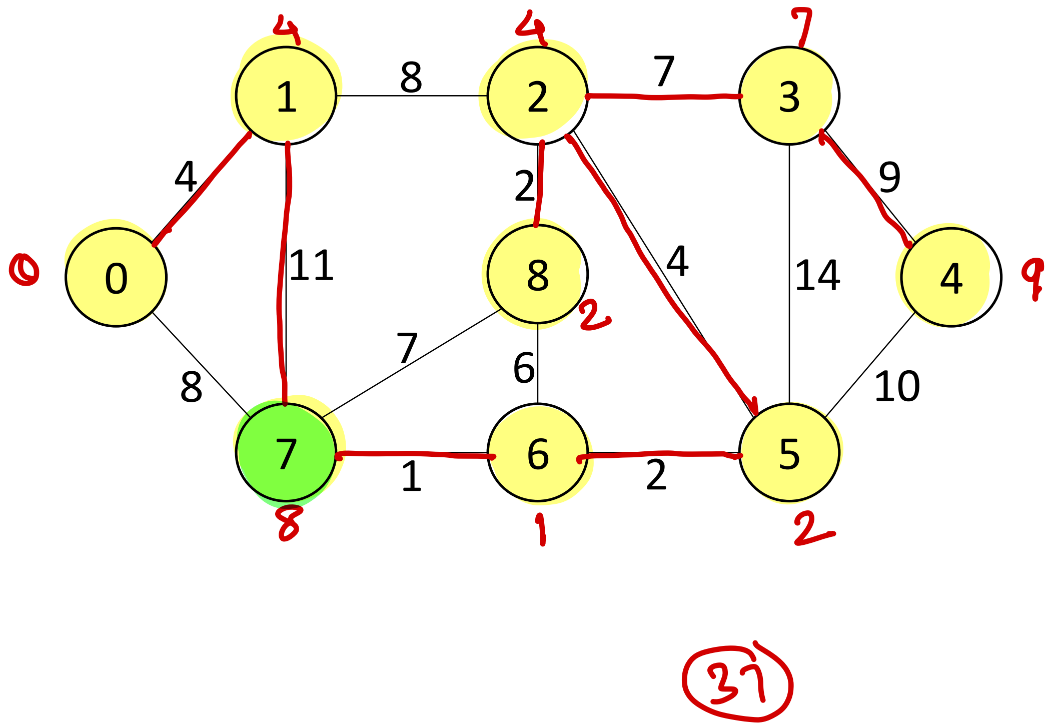
keys

0	0
1	∞
2	∞
3	∞
⋮	
8	∞

Prim’s MST

1. Create a set mstSet that keeps track of vertices already included in MST.
2. Assign a key value to all vertices in the input graph. Initialize all key values as INFINITE. Assign key value as 0 for the first vertex so that it is picked first.
3. While mstSet doesn’t include all vertices
   i. Pick a vertex u which is not there in mstSet and has minimum key value.
   ii. Include u to mstSet.
   iii. Update key value of all adjacent vertices of u. For every adjacent vertex v, if weight of edge u-v is less than the previous key value of v, update the key value as weight of u-v.

0	
1	
2	
3	
4	
5	
6	
7	
8	



Prim's algo :

Time Complexity
$= O(V^2)$

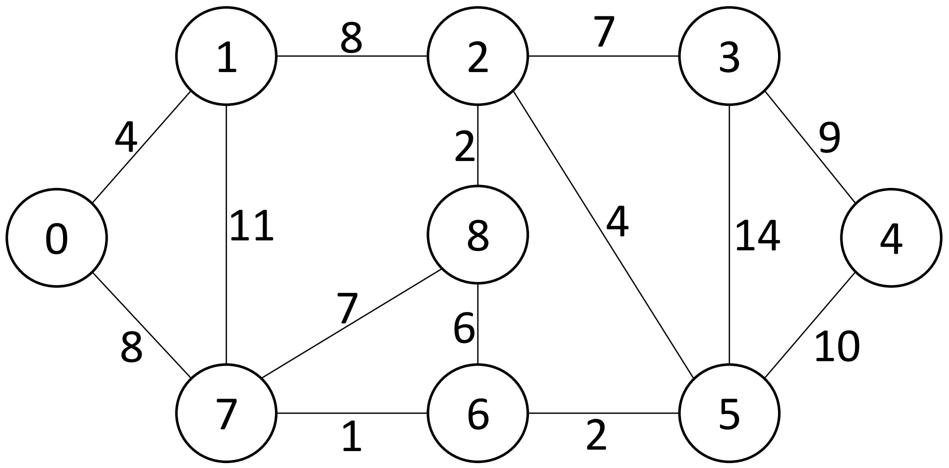
$V \to$ all V vertices
$\hookrightarrow V \to$ find min key vertex

Time Complexity
$= O(V \log V)$

$V \to$ all vertices
$\hookrightarrow O(\log V)$ find min key vertex using min heap

Union Find Algorithm - detect cycle in the graph

1. Consider all vertices as disjoint sets (parent = -1).
2. For each edge in the graph
   1. Find set of first vertex.
   2. Find set of second vertex.
   3. If both are in same set, cycle is detected.
   4. Otherwise, add second vertex set in the set of first. - union both sets



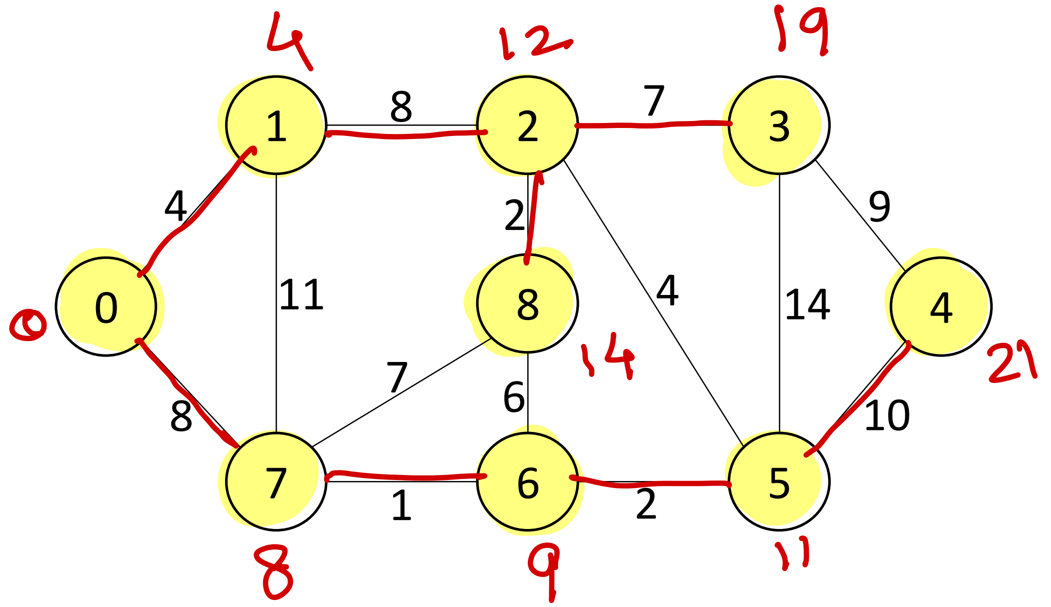
wt	src	des
1	7	6
2	8	2
2	6	5
4	0	1
4	2	5
6	8	6
7	2	3
7	7	8
8	0	7
8	1	2
9	3	4
10	5	4
11	1	7
14	3	5

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

Dijkstra’s Algorithm

- 1) Create a set sptSet (shortest path tree set) that keeps track of vertices included in shortest path tree, i.e., whose minimum distance from source is calculated and finalized. Initially, this set is empty.
- 2) Assign a distance value to all vertices in the input graph. Initialize all distance values as INFINITE. Assign distance value as 0 for the source vertex so that it is picked first.
- 3) While sptSet doesn’t include all vertices
- ….a) Pick a vertex u which is not there in sptSet and has minimum distance value.
- ….b) Include u to sptSet.
- ….c) Update distance value of all adjacent vertices of u. To update the distance values.

0	
1	
2	
3	
4	
5	
6	
7	
8	

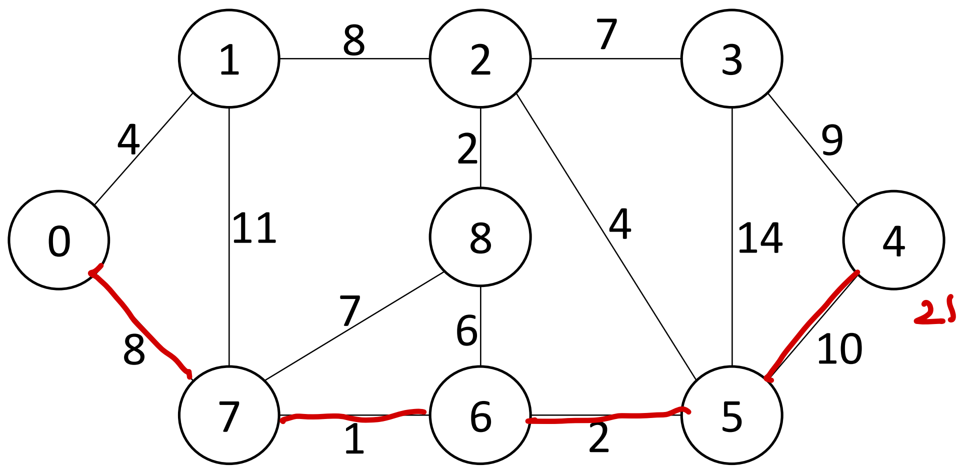


dist[v] = dist[u] + weight u-v

Dijkstra’s Algorithm

- 1) Create a set sptSet (shortest path tree set) that keeps track of vertices included in shortest path tree, i.e., whose minimum distance from source is calculated and finalized. Initially, this set is empty.
- 2) Assign a distance value to all vertices in the input graph. Initialize all distance values as INFINITE. Assign distance value as 0 for the source vertex so that it is picked first.
- 3) While sptSet doesn’t include all vertices
- ….a) Pick a vertex u which is not there in sptSet and has minimum distance value.
- ….b) Include u to sptSet.
- ….c) Update distance value of all adjacent vertices of u. To update the distance values.

0	
1	
2	
3	
4	
5	
6	
7	
8	



SUNBEAM

Thank you!

Nilesh Ghule <nilesh@sunbeaminfo.com>