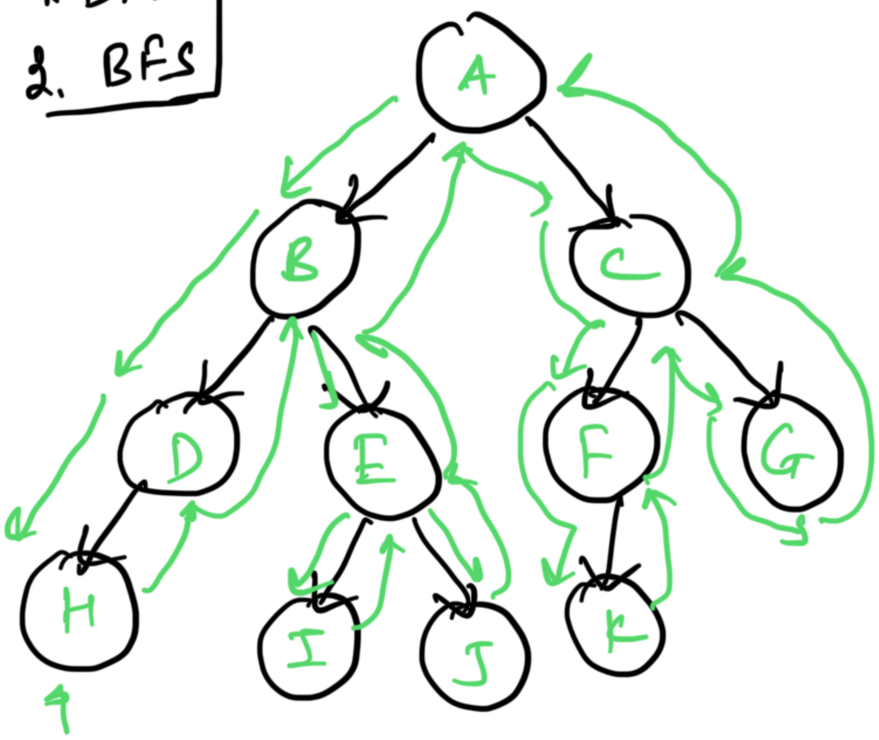
Graph Traversal { Node (Vertex), Edges }

1. DFS
2. BFS



Depth → preference

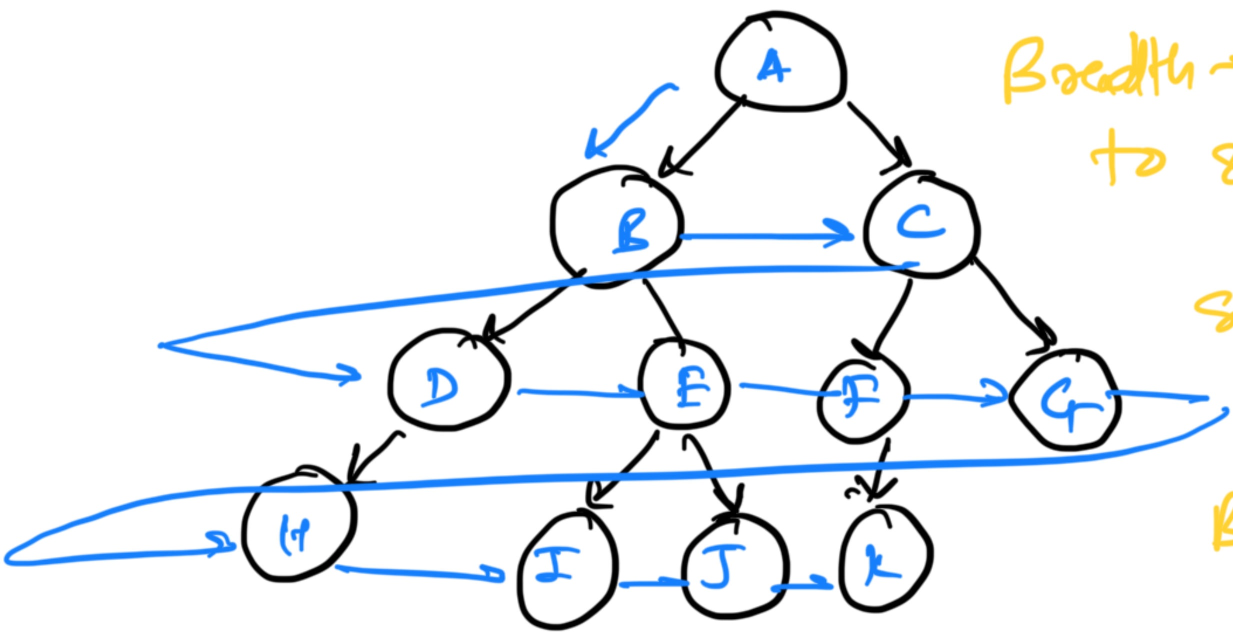
1 level down

check

↓

Depth First Traversal

DFS



Breadth → preference to siblings

↓

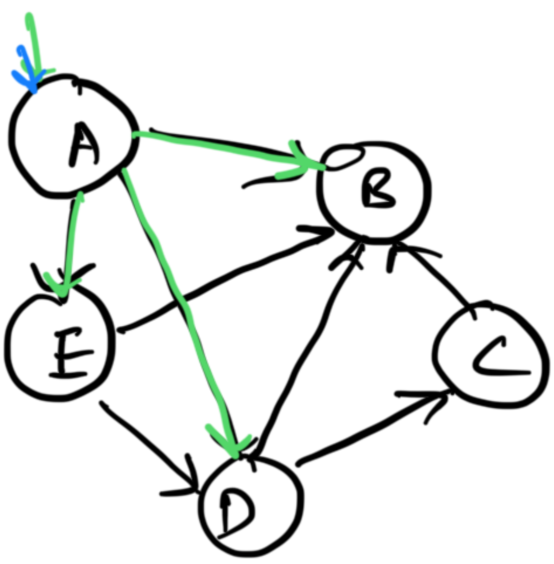
Side walk mode

↓

Breadth First Traversal

MCQ to design BFS & DFS

Ex



DFS :

A : ~~B~~, ~~D~~, E

~~B~~ : –

~~D~~ : ~~B~~, ~~C~~,

~~C~~ : ~~B~~,

E : B, D

breadth siblings

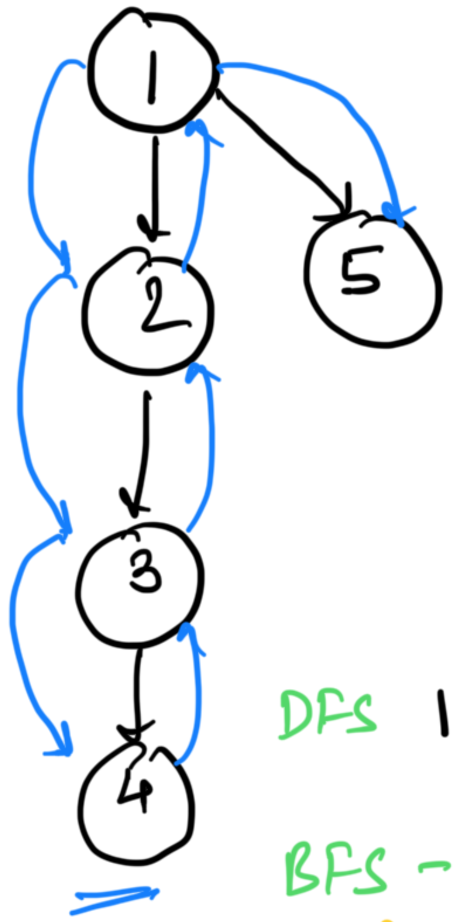
depth

DFS: A, B, D, C, E

BFS: A, B, D, E, C

~~Observed~~ order differ → A : B, D, E | A : D, E, B | A : E B D

Ex



DFS:

1 : ~~2~~, ~~5~~
2 : ~~3~~
3 : ~~4~~
4 : –
5 : –

1 → 2, 5 → 1
→ 5, 2 → 2

DFS 1, 2, 3, 4, 5 (LHS)

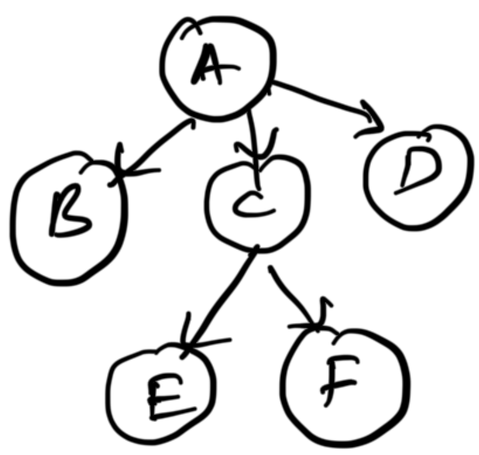
BFS – 1, 2, 5, 3, 4 (RHS)

LHS		RHS
1	→	2, 5
2	→	3
3	→	4
4	→	–
5	→	–

1, 2, 5, 3, 4 → BFS (RHS)

DFS (LHS)

Ex



visit A → B, C, D Explore

unvisited

visited B → –

C → E, F

E → –

F → –

D → –

DFS – A, B, C, E, F, D

BFS – A, B, C, D, E, F

BFS → Explore vertices level by level, starting from a given first vertex

→ data structure → $\boxed{\text{Queue}}$

→ shortest path → unweight graph

Appl^n

DFS → Explore vertices depth by depth starting from a given first node

→ data structure → $\boxed{\text{Stack}}$

→ Minimum Spanning Tree

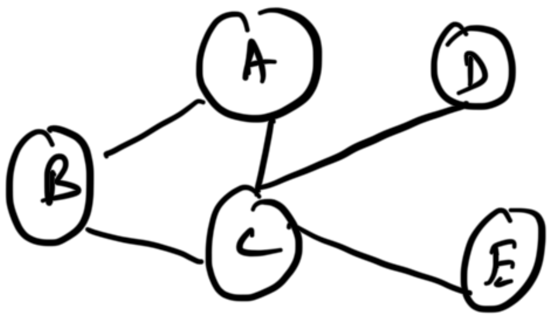
Appl^n

→ Kruskal Algo

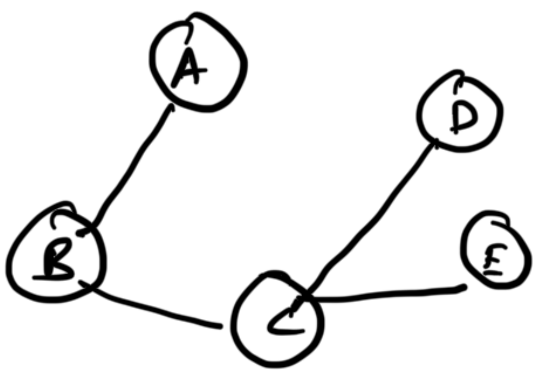
→ Prim's Algo

Diff BFS & DFS

Spanning Tree:

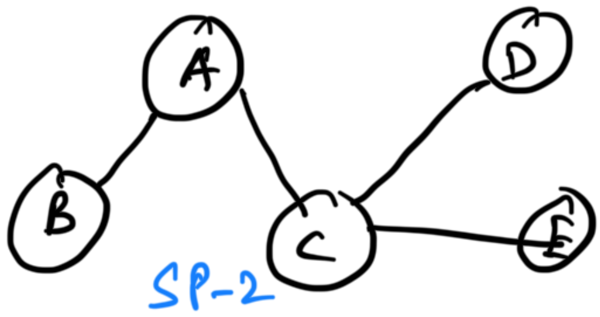


→



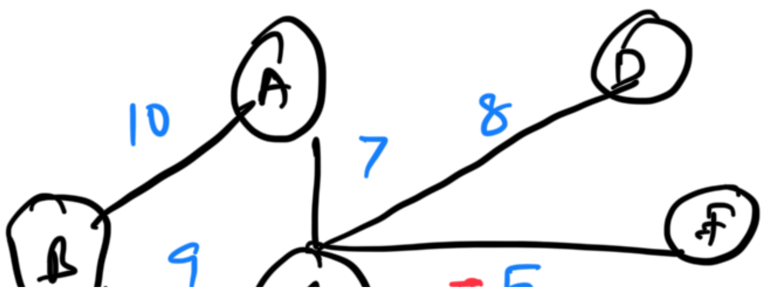
SPT-1

Graph



SP-2

– A subgraph T, G(V, E), undirected graph is a spanning tree, this graph is a tree



Weigh of a Graph

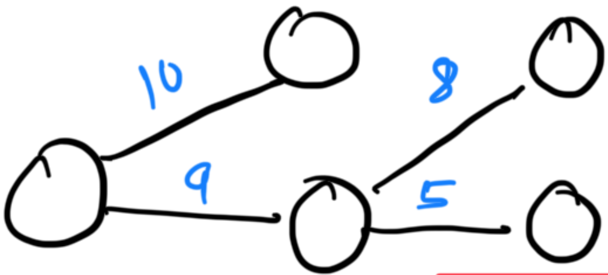
– The sum of

edge weight

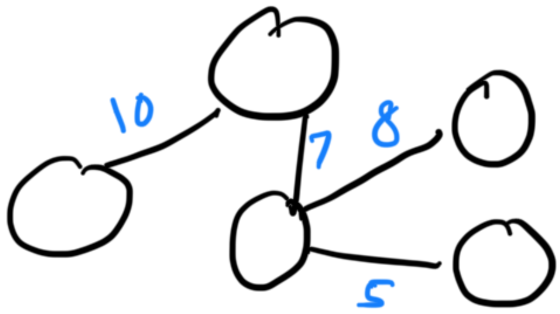
the weight of all edges

Weighted graph

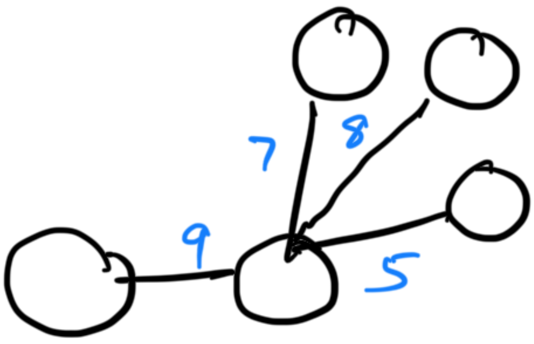
- weighted graph is a graph, In which each edge has a weight (real / int)



G1 ⇒ 32



G2 ⇒ 30

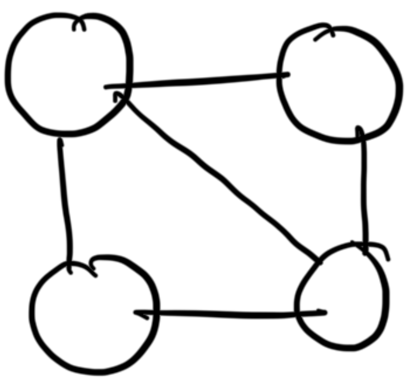


G3 ⇒ 29 → Minimum Spanning Tree (MST)

Minimum cost ↑

Minimum Spanning Tree -

It is subset of edges from a connected weighted graph that connects all the vertices without any cycles and with the minimum possible total edge weight



Greedy Algorithms -

Mumbai ———— Goa

Mini → Constraint

S1 S2 S3 S4 ... Sn

→ feasible sol^n

plane

Optimization

(Optimal Solution)

No guarantee

Greedy Technique

Top-Down Approach

(Max, or Min)

$\boxed{\text{Principle of optimality}}$

feasible sol^n

Dynamic Programming

Problem

S1 → optimal sol^n → Granted optimal sol^n

S2 → opt. sol^n

S3

Sn

Multiple sol^n

Dynamic Programming

Top Down Approach

$\boxed{\text{Memorization}}$

Recursion fn

fib(5)

fib(4) fib(3)

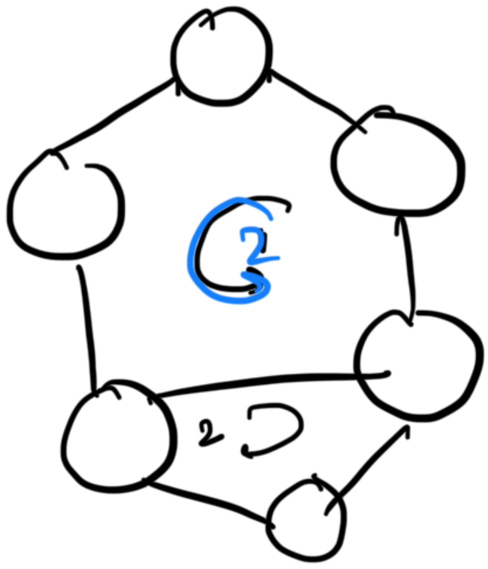
Bottom Up Approach

$\boxed{\text{Tabulation}}$

Iterations

Ex:

MCQ



$\rightarrow |E| = 7$ #2

no of cycles

$^{|E|}C_{|V|-1}$ $\quad {}^{7}C_{5} - 2$

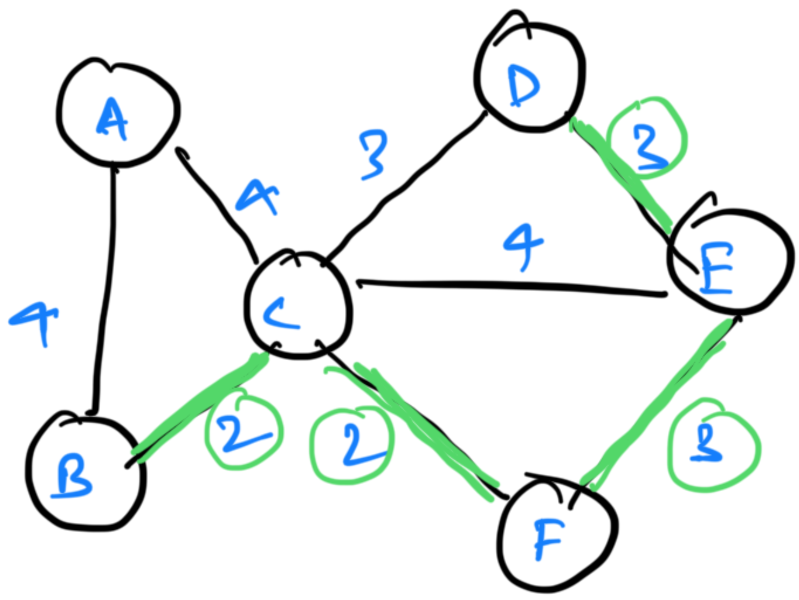
$$\boxed{{}^{|E|}C_{|V|-1} - \text{No of cycles}}$$

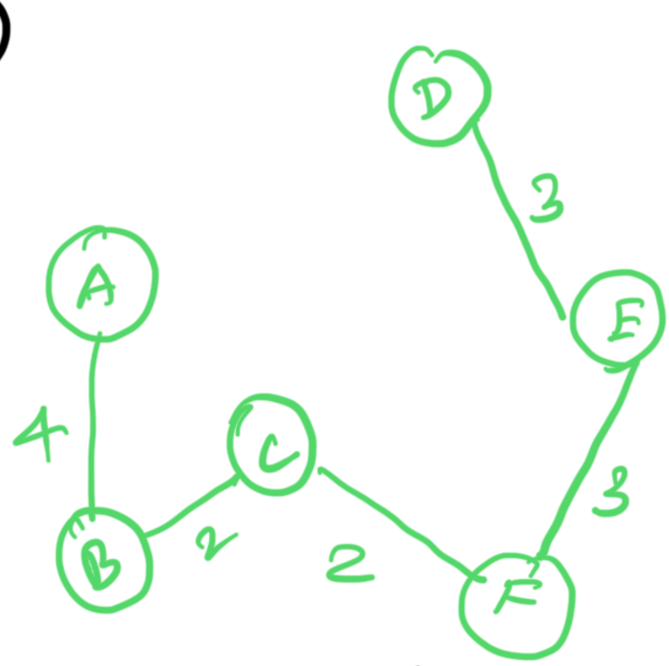
No. of spanning trees to be generated

1. Prim's
2. Kruskal's Algorithm

} to identify min spanning Tree

Imp Kruskals Algorithm





MST = 14

Kruskal Algo →

- Shortest weight in edge
- Sequence is not a problem
- Criteria shortest edge with smallest weight
- Greedy Algorithm

Greedy Algorithm

$\rightarrow \boxed{O(|V| \cdot |E|)}$

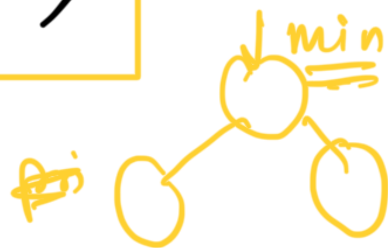
$O(n \cdot e)$

$\boxed{O(n^2)}$

Imp

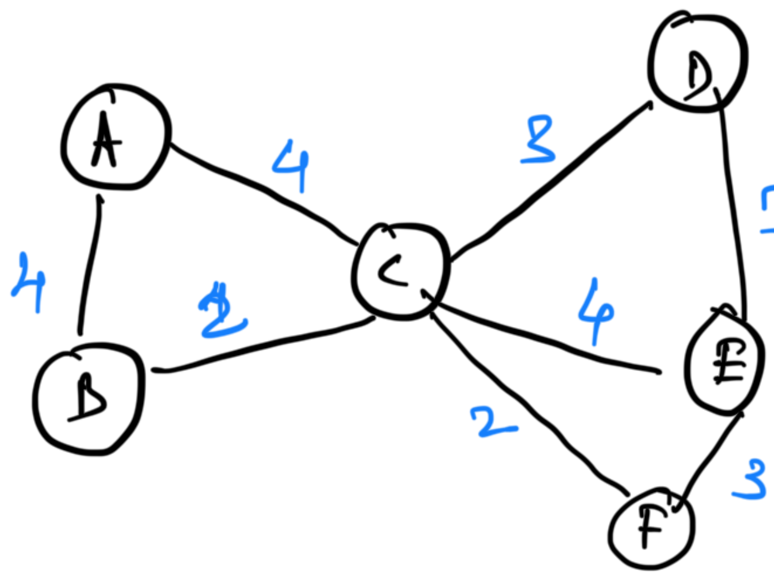
- Can we optimize ?

Minimum → priority ? →

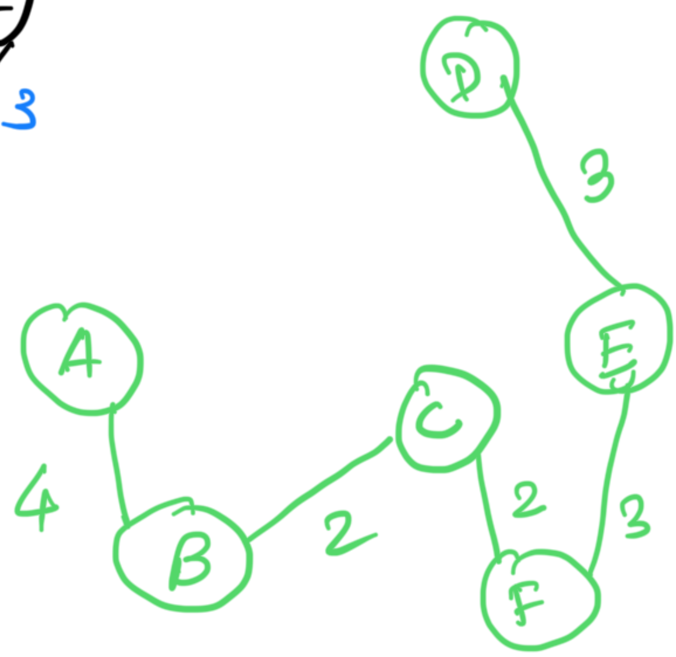


Minimum → $\boxed{\text{Min Heap}}$ → $O(\log n)$

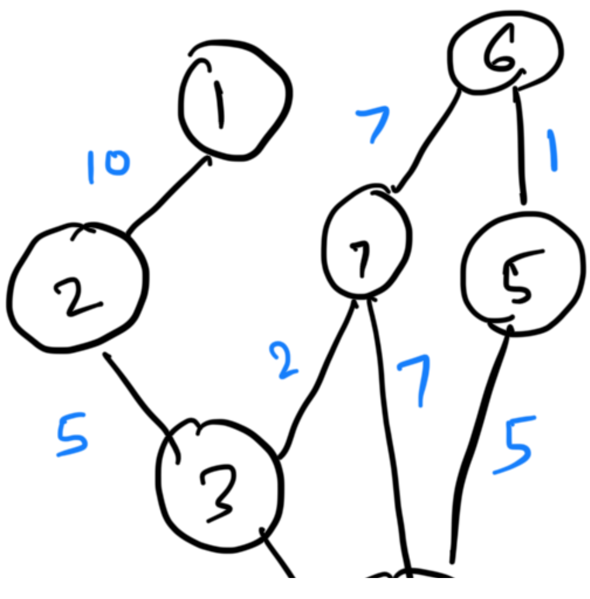
Prim's Algorithm



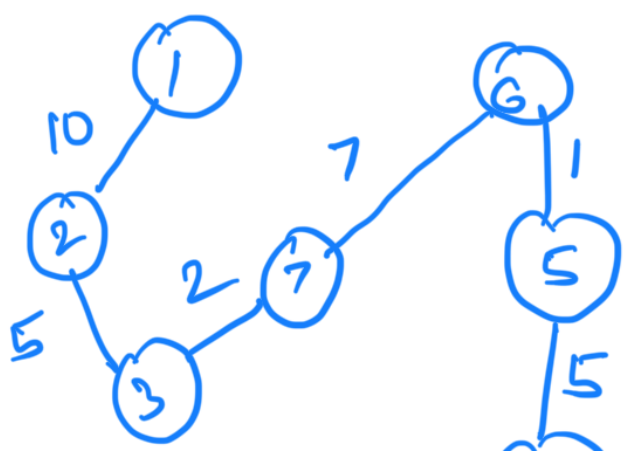
- Vertex
- shortest edge
- Greedy Algorithm

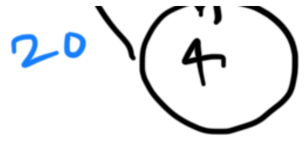


MST = 14



Kruskal's Algo

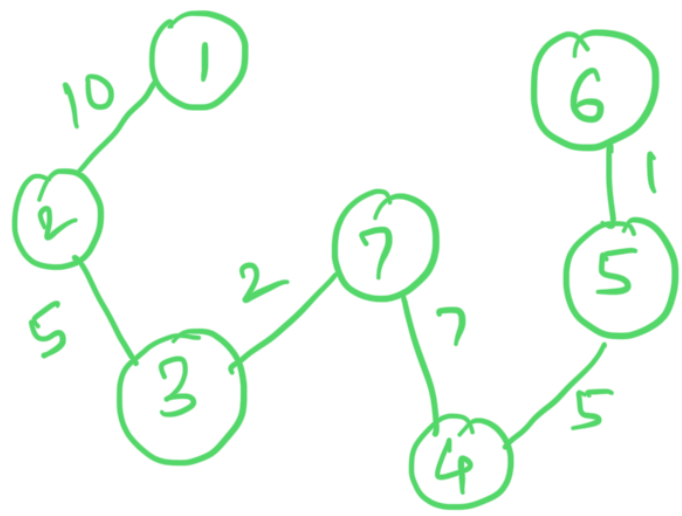




MST = 30

Input - ST

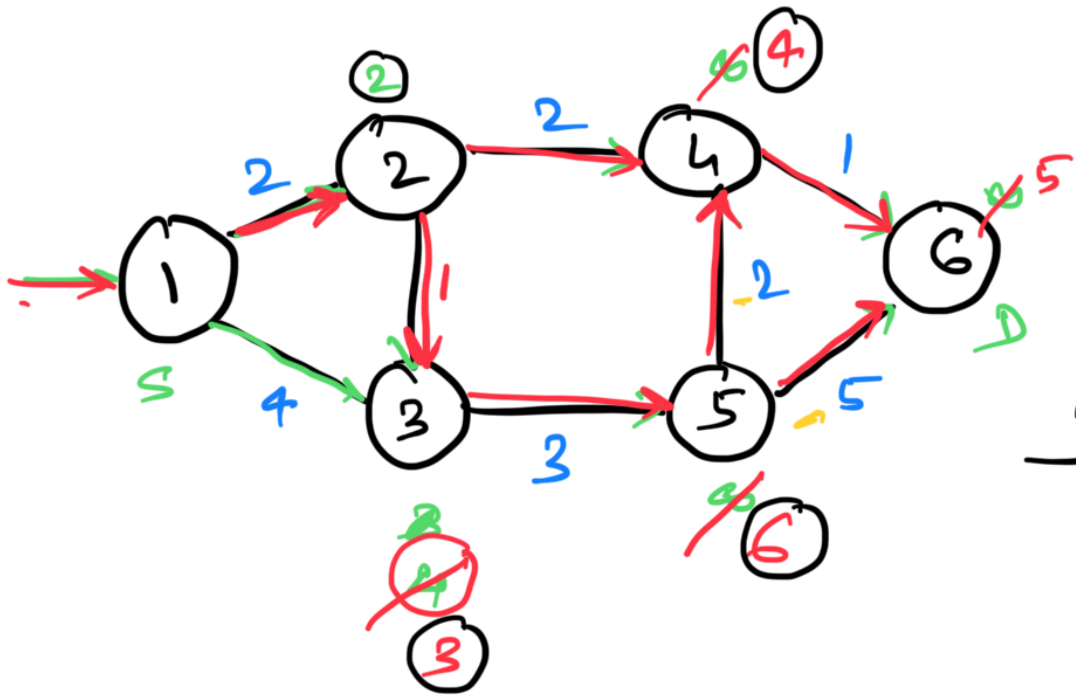
Prims Algo



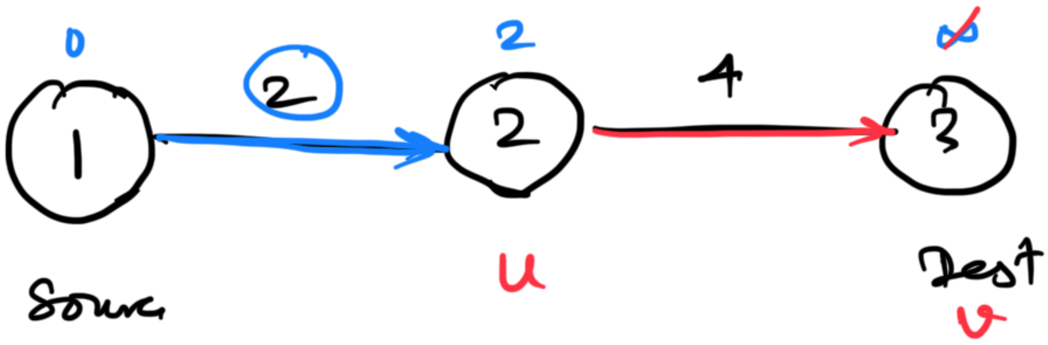
MST = 30

Shortest Path Algorithm

- Dijkstra Algorithm → Relaxation



V	d(v)
2	2
3	3
4	4
5	6
6	5



Relaxation $\left\{ \begin{array}{l} 2+4 \le \infty \\ 6 \le \infty \end{array} \right.$

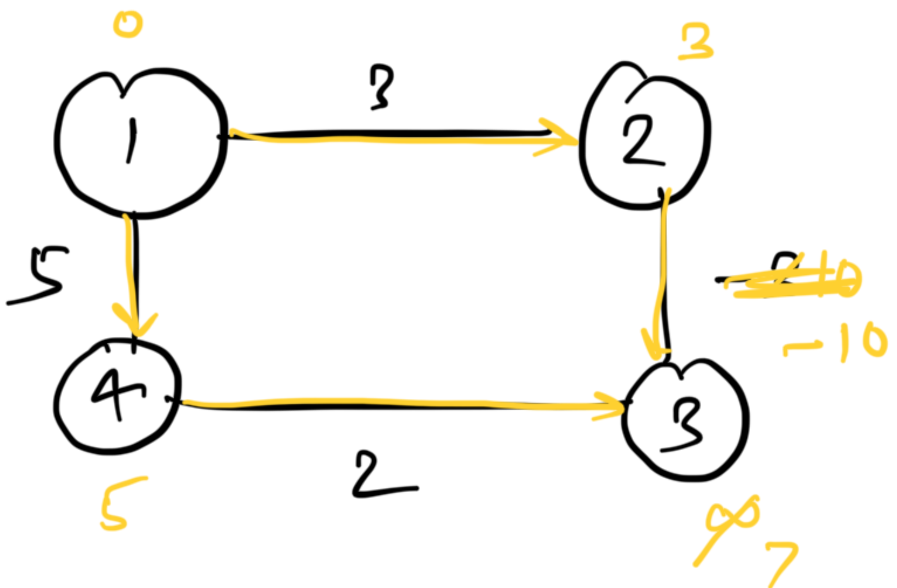
$$\text{if } (d[u] + [u,v] \le d[v])$$

$$d[v] = d[u] + c[u,v]$$

2km 4km

$\infty \le 6$ km

Ex



Dijkstra Algorithm

1) Vector $\rightarrow$ $O(|V| \cdot |V|) = O(|V|^2)$

$\approx O(n^2)$

2) Binary heap $\rightarrow$ $O(|V| + |E|)\log|V|$

$\approx O(V + |E| \log |V|)$

Imp

Dijkstra $\rightarrow$ Worst case $\rightarrow$ may or may not work for negative values

$O(V + E . \log(|V|))$

Bellman-Ford - Handle -ve values

$\hookrightarrow$ directed and undirected works with graph

$O(V \cdot E)$

Floyd Warshall — $O(|V|^3)$