MST

- Prims
- Kruskal

Single source shortest path (SSSP)

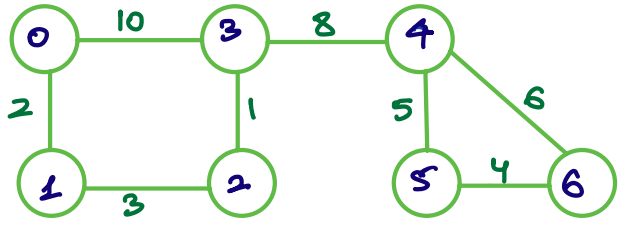
- Dijkstra
- Bellman ford

DIJKSTRA:

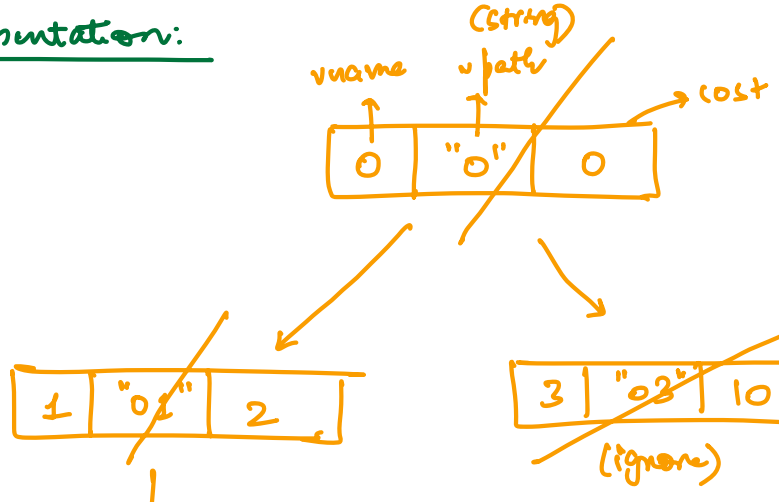
- Greedy Algo (Similar to Prims)
- Priority Queue
- Source Node

Src node = 0

0 → 1 : 2
0 → 2 : 5
0 → 3
0 → 4
0 → 5
0 → 6



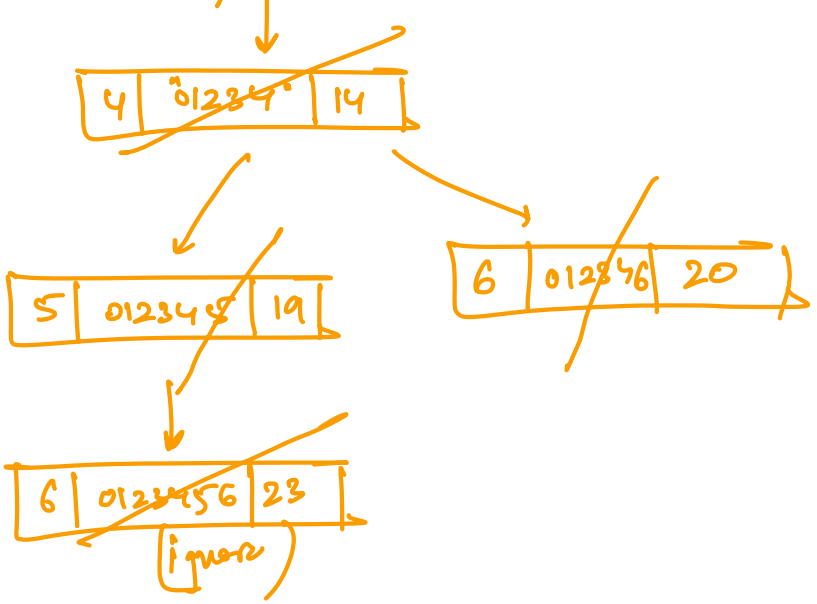
Implementation:



0 ✓
1 ✓
2 ✓
3 ✓
4 ✓
5 ✓
6 ✓

1. Remove
2. visited
2. Print
4. mark unvisited

0 via 0 @ 0
1 via 01 @ 2
2 via 012 @ 5
3 via 0123 @ 6
4 via 01234 @ 14
5 via 012345 @ 19



6 via 012346 @ 20

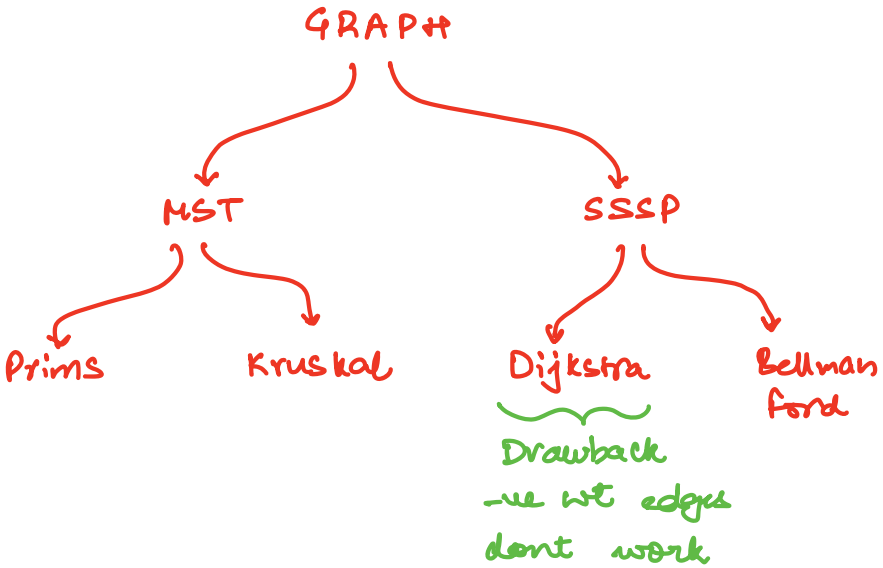
$E(1+\log E+1++1)+2E$

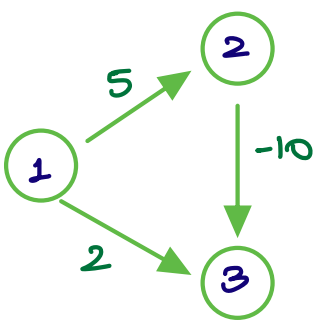
$E\log E+2E$

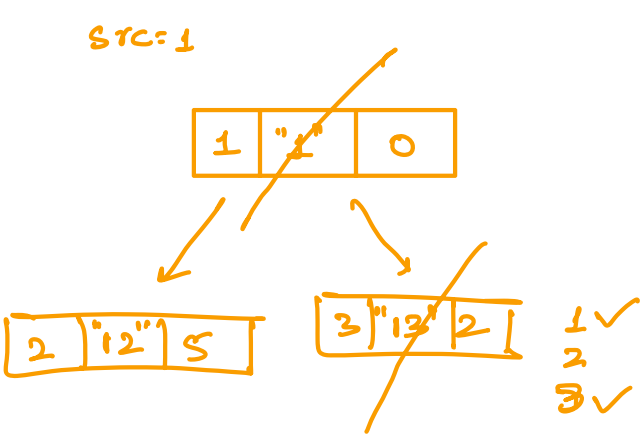
$E\log E$ $\quad E=V^2$

$E\log V^2$

$O(E\log V)$





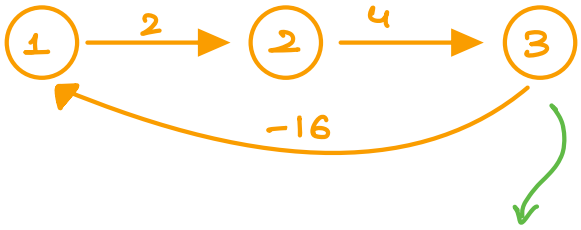


→ queue
→ visited
→ print
→ values

1 via "1" @ 0
3 via "12" @ 2
wrong answer

Bellman Ford:

- -ve wt edge ✓
- -ve wt cycle ✗
  SSSP ✗



127 127 127 : -4-10 = -14

123 123 : 6-16+6 = -4

123 : 6

SSSP ⟶ no -ve wt cycle ✗

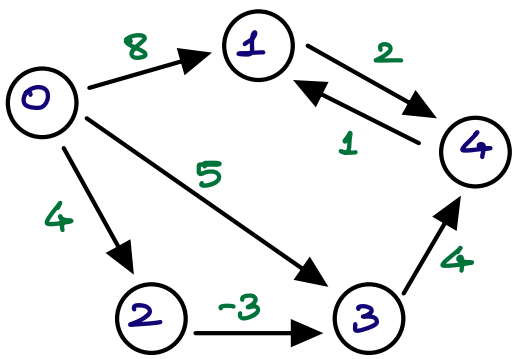
⟶ -ve wt edge but no -ve wt cycle : Bellman ford

⟶ +ve wt edges: dijkstra, bf

Example:

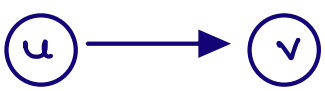
Edges
0 → 2 : 4
0 → 1 : 8
0 → 3 : 5
2 → 3 : -5
1 → 4 : 2
4 → 1 : 1
3 → 4 : 4

Cost
0 → 0
1 → ∞
2 → ∞
3 → ∞
4 → ∞



Relax every edge

V-1 times



edge (u → v)

old cost — new cost

if (cost[v] > cost[u] + cost edge u→v)

cost[v] = cost[u] + cost edge u→v

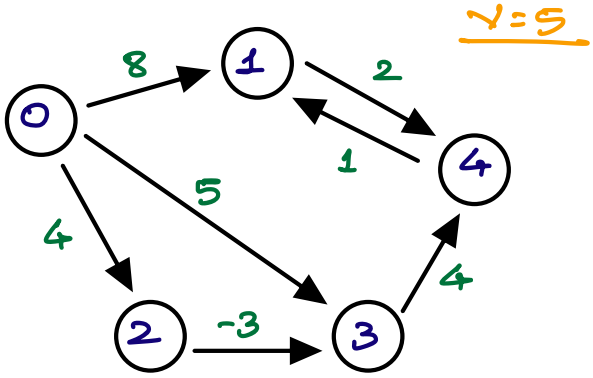
Edges

0 → 2 : 4
0 → 1 : 8
0 → 3 : 5
2 → 3 : -3
1 → 4 : 2
4 → 1 : 1
3 → 4 : 4

Cost

src = 0

Vertex	Cost
0	0
1	~~∞~~ ~~8~~ 6
2	~~∞~~ 4
3	~~∞~~ ~~8~~ 1
4	~~∞~~ ~~10~~ 5



1st time

0 → 2
∞ > 0+4
oc nc
✓

0 → 1
∞ > 0+8
nc
✓

0 → 3
∞ > 0+5
✓

2 → 2
5 4-2
5 > +1
✓

1 → 4
∞ > 8+2
✓

4 → 1
8 10+1
8 > 11
✗

3 → 4
10 +1+4
10 > 5

2nd time

0 → 2
4 0+4
✗

0 → 1
8 0+8
✗

0 → 3
1 0+5
✗

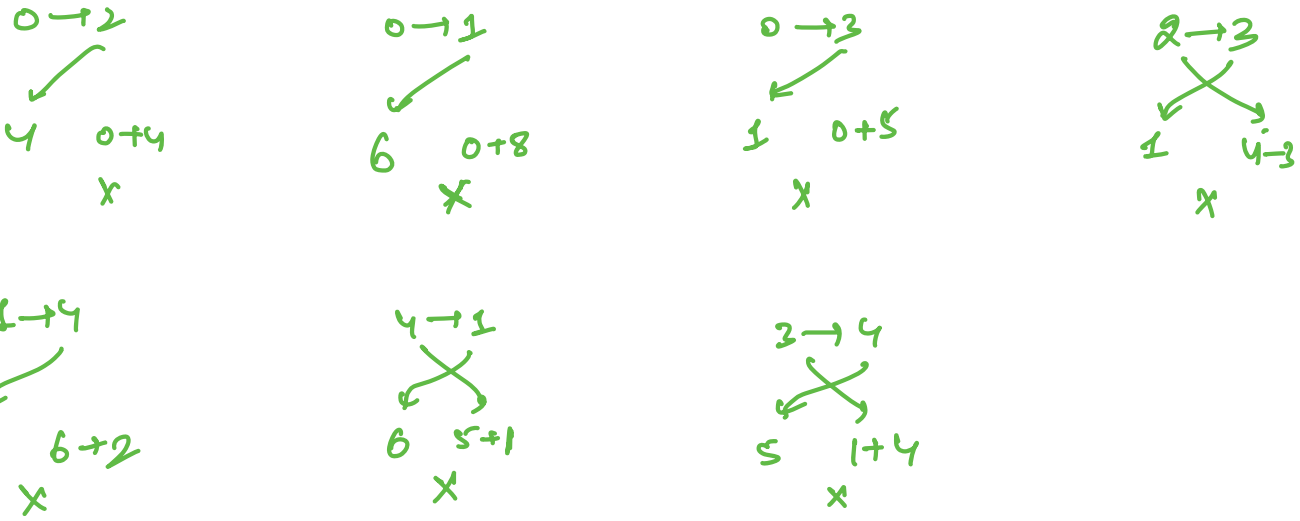
2 → 3
1 4-3
✗

1 → 4
5 8+2
✗

4 → 1
8 5+1
8 > 6
✓

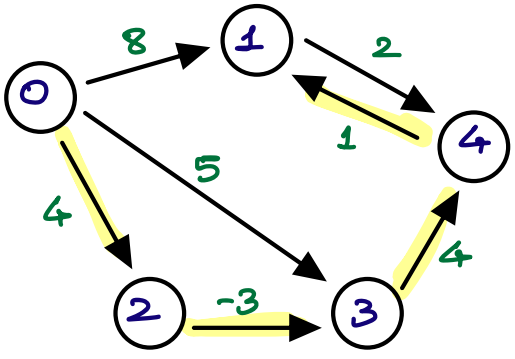
3 → 4
5 1+4
✗

2nd time



Stop no changes

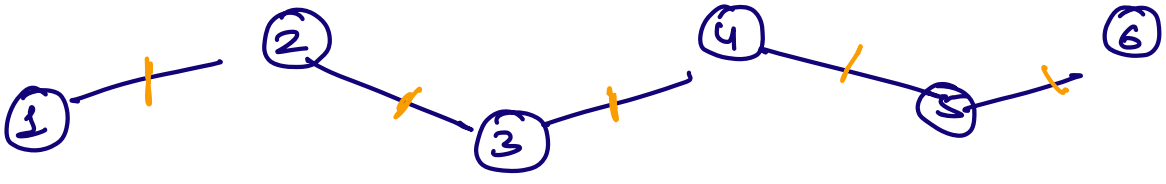
Optimize



Vertex → Cost

0 → 0
1 → ~~∞~~ ~~8~~ 6
2 → ~~∞~~ 4
3 → ~~∞~~ ~~8~~ 1
4 → ~~∞~~ ~~10~~ 5

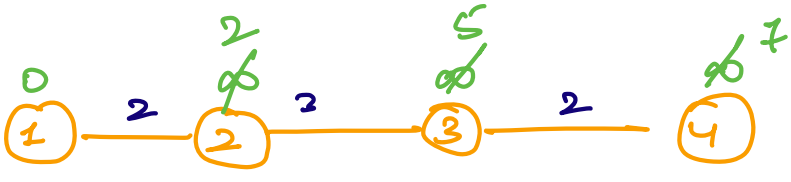
Why V-1 ?



V=6

V-1 is the max distance possible b/w any 2 nodes

Eg:

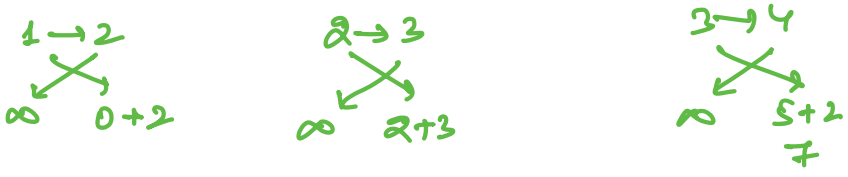


V=4

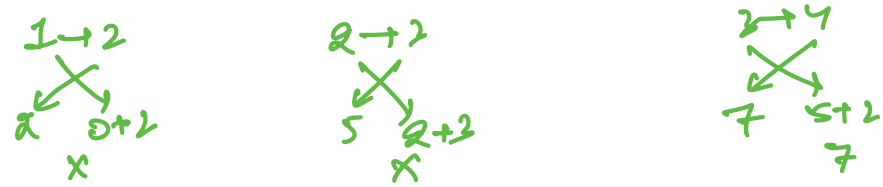
Edges:

1→2 : 2
2→3 : 3
3→4 : 2

1st time:

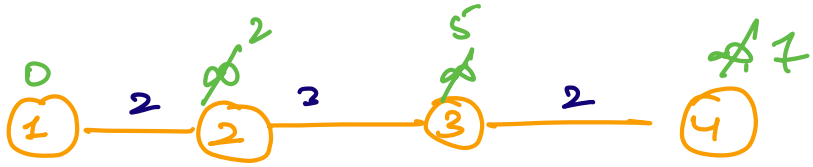


2nd time:



3rd time:

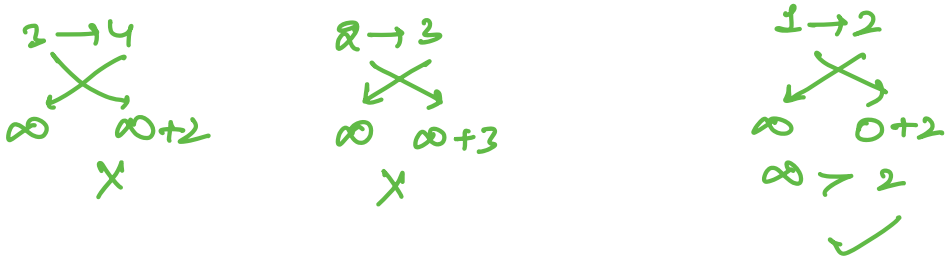
no change



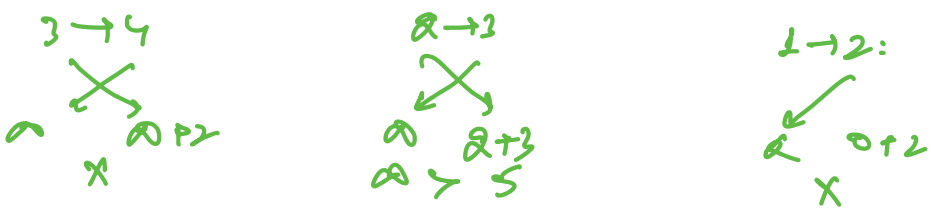
Edges:

3→4 : 2
2→3 : 3
1→2 : 2

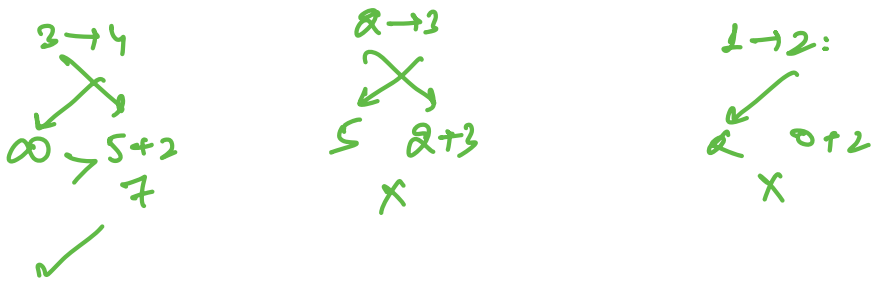
1st time



2nd time



3rd time

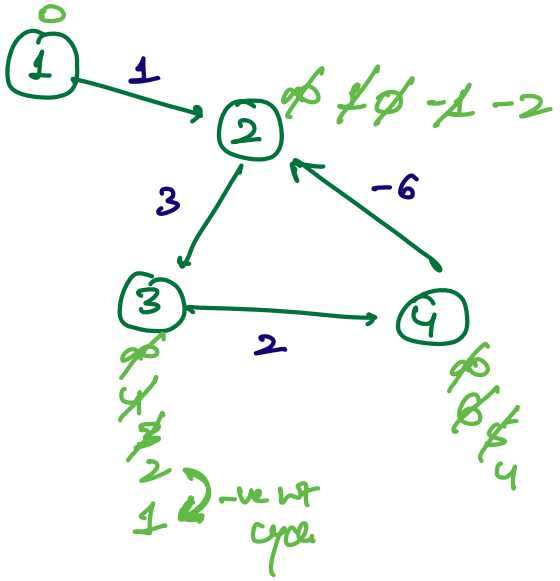


Bellman ford → -ve wt edge ✓

-ve wt cycle

V-1 times relax

Vth time : vertex cost update ⇒ -ve wt cycle



1st relax ✓

2nd relax ✓

2nd relax ✓

4th relax ⇒ changes