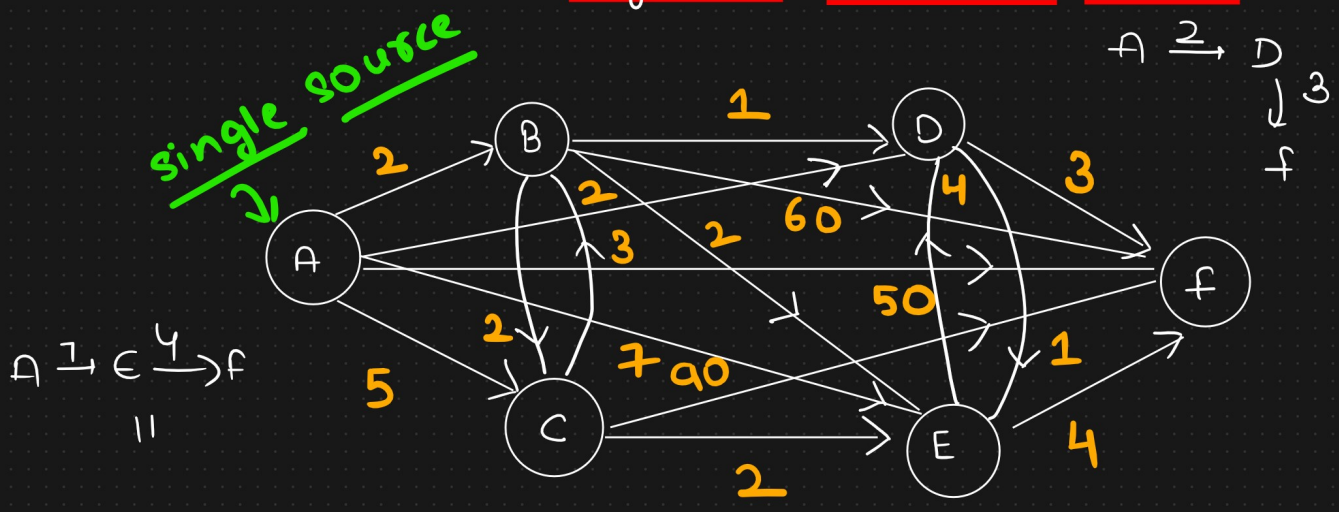
Dijkstra's Algorithm

↳ Single source shortest path



Graph

A → { B:2, C:5, D:2, E:7, F:50 }

B → { C:2, D:1, E:2, f:60 }

C → { B:3, E:2, f:90 }

D → { E:1, f:3 }

E → { D:4, f:4 }

f → { }

A → B = 2

A → C = 5

A → B → C = 2 + 3 = 5

A → B → f = 2 + 60 = 62

A → B → D = 2 + 1 = 3

A → B → E = 2 + 2 = 4

A → D → E = 3 + 1 = 4

E update the key to some decrement value in a minheap

⇓

Decrease key

↳ $\Theta(\log n)$

2 < inf

Always starting Point = A

	A	B	C	D	E	f	V
	0 N	∞ N	∞ N	∞ N	∞ N	∞ N	
A		2 A	5 A	2 A	7 A	50 A	$5\log n + 5$
B			4 B	2 A	4 B	50 A	$2\log n + 4$
D			4 B		3 D	5 D	$2\log n + 2$
E			4 B			5 D	$O\log n + 2$
C						5 D	$O\log n + 3$
f							

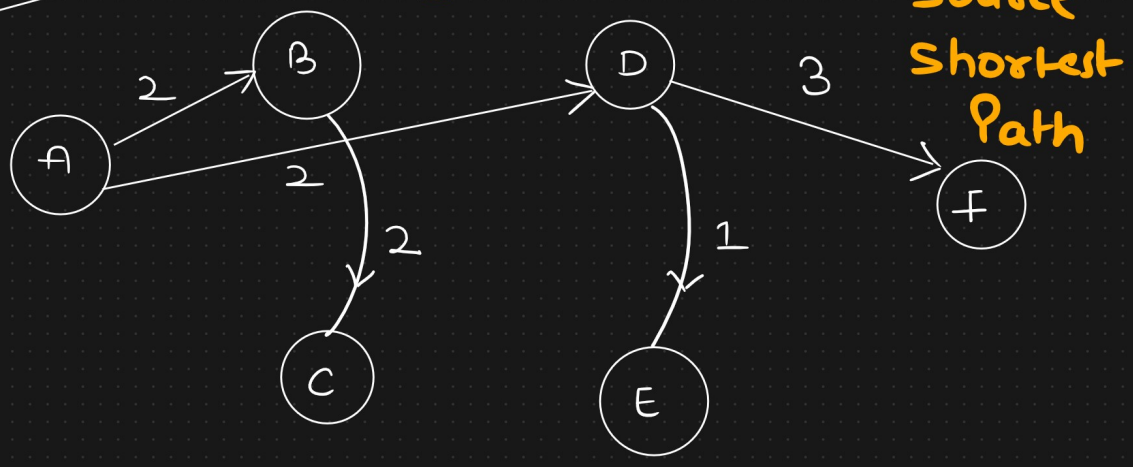
Vlog V

Result

- A - B = 2
- A - C = 4
- A - D = 2
- A - E = 3
- A - f = 5

↳ single Source Shortest Path

Dijkstra's Algorithm



Time complexity

↳ Graph Data using Adjacency List

Time complexity

$\Rightarrow V + V\log V + E\log V + 2E$

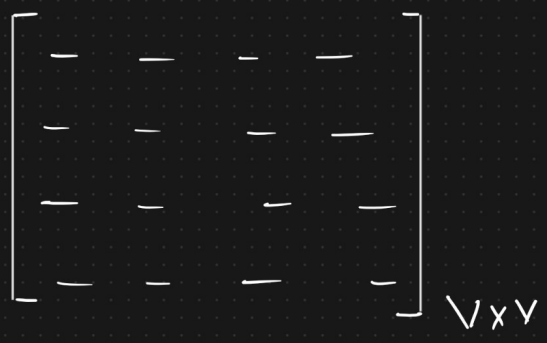
$\Rightarrow \Theta((V+E)\log V)$

$\log E \approx \log V$

Graph Data using Adjacency Matrix ↗ $V \times V$

$\Rightarrow V + V\log V + E\log V + V^2$

$\Rightarrow \Theta(V^2)$

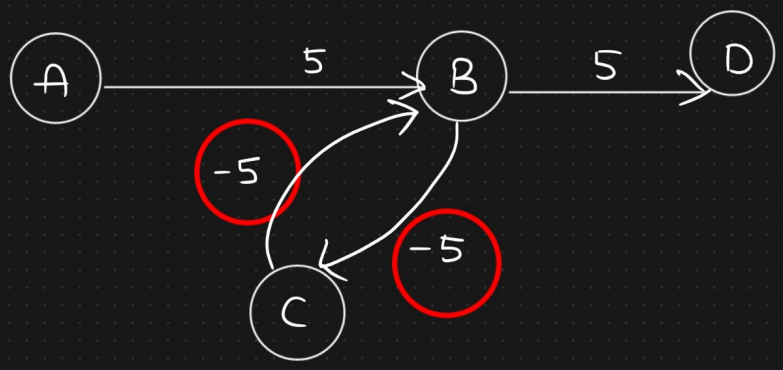


Drawback

↳ Not preferrable when we have negative edge weights.

Real life situations

(At what situations, do you think we get -ve edge weights)



Bellman ford

↳ single source shortest path

Dynamic Programming

covers both -ve & +ve edge weights

Optimization Problem

DP takes more time to generate the results > Greedy algorithms time generation

↳ optimized solution

4 Path

10 Paths to solve any problem

Explore all the possible paths

↓

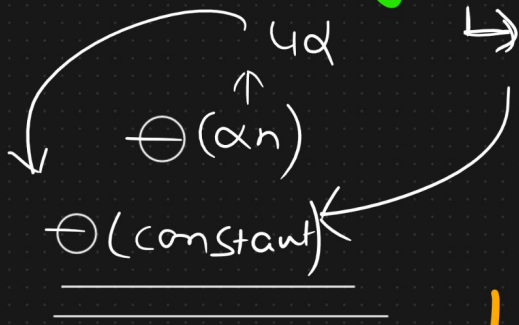
Optimized result

Kruskal's Algorithm

↳ cycle detection algorithm

↳ Union by Rank

↳ Path Compression



Proper In depth Discussion

Reference:

https://www.geeksforgeeks.org/dijkstras-shortest-path-algorithm-greedy-algo-7/