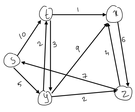
* Dijkstras Algorithm *



Nodes	t	y	x	z
s	∞	∞	∞	∞
s,y	10	5	9	∞
s,y,t	7		14	7
s,y,t,t_2			8	7
			8	

* Floyd warshall Algorithm *

Methodology:

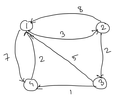
1> write down adjacency matrix (A^0).

2> Then apply this general formula to calculate values:

- first write down the values cost[i,k] & c(k,j) as it is.
- for the rest values,

A^k : cost[i,j] = min [A^{k-1} cost[i,j] , cost[i,k] + cost[k,j]]

k ∈ vertexes of graph



A^0 =

	1	2	3	4
1	0	3	∞	7
2	8	0	2	∞
3	5	∞	0	1
4	2	∞	∞	0

calculating via 1,

A^1 =

	1	2	3	4
1	0	3	∞	7
2	8	0	2	15
3	5	8	0	1
4	2	5	∞	0

now via 2,

A^2 =

	1	2	3	4
1	0	3	5	7
2	8	0	2	15
3	5	8	0	1
4	2	5	7	0

now via 3,

A^3 =

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

now via 4,

A^4 =

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

* Ford Fulkerson algo *

Algorithm: Initialize Overall Flow (OF) = 0

while (Augmenting path) {

- list AP with its residual capacities
- Pick the least one as the bottleneck capacity
- subtract this BC with flow of every edge
- OF = OF + BC;

}

Return OF;

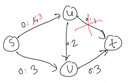
Example:

AP : s → u → t

RC : 4 1

BC = 1

OF = 0+1 = 1

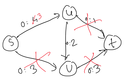


AP : s → v → t

RC : 3 3

BC = 3

OF = 1+3 = 4

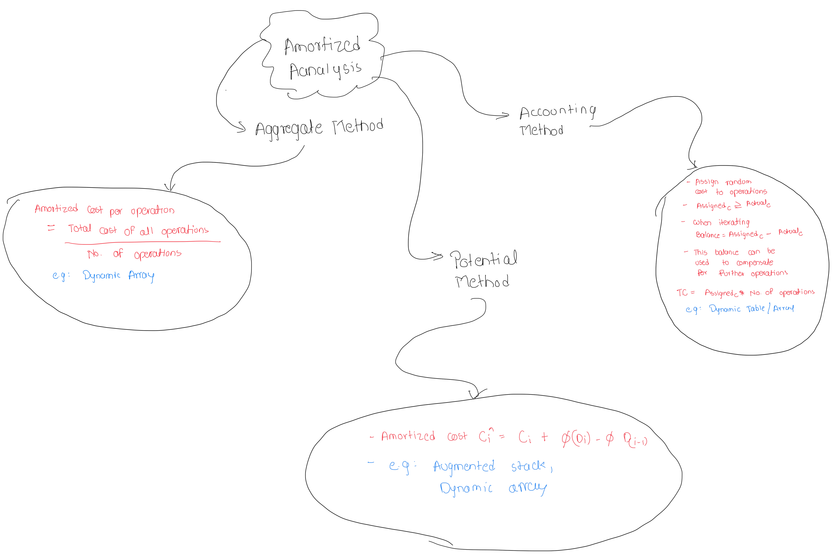


NO AP exists

∴ Overall flow = 4

* A* Algo *

$$f(N) = g(N) + h(N)$$

* Amortized Analysis *

- does not depend on input value