

Network Flows

Dr. P. Gayathri

Professor

SCOPE

VIT Vellore

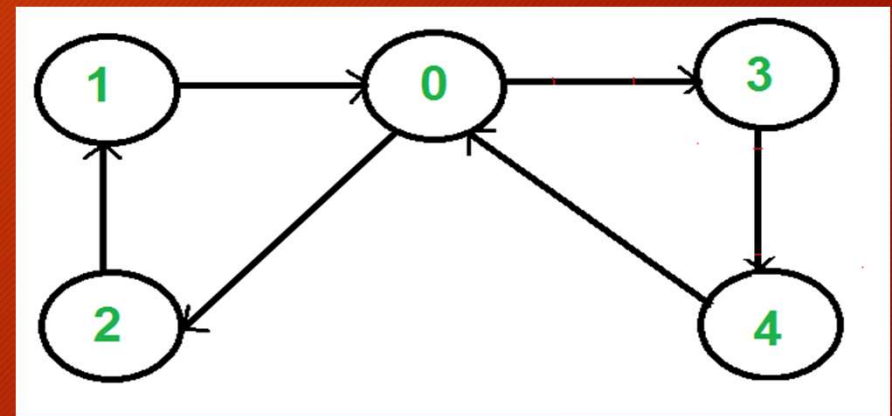
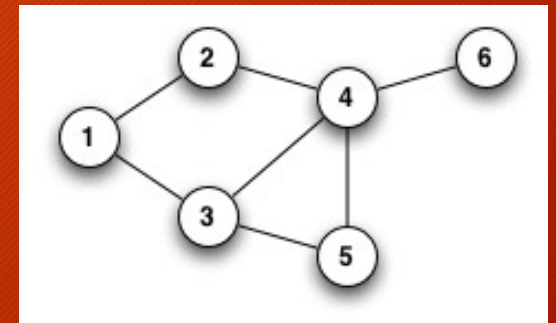
Graph

2

- $G = \{V, E\}$
- Types - Directed (digraph) and undirected
- Network flow - involves directed graph
- Incoming edges and outgoing edges
- Indegree and outdegree
- Example digraph:

$V = \{0, 1, 2, 3, 4\}$

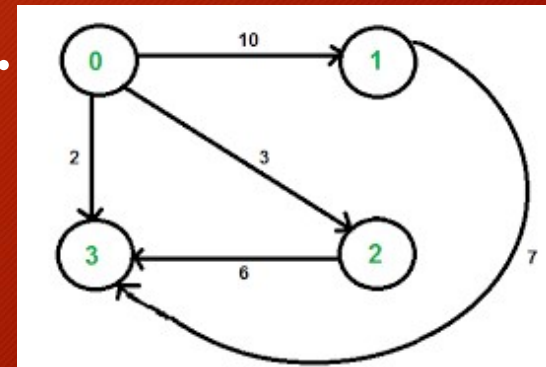
$E = (0, 2) (0, 3) (1, 0) (2, 1) (3, 4) (4, 0)$



Network Flow

3

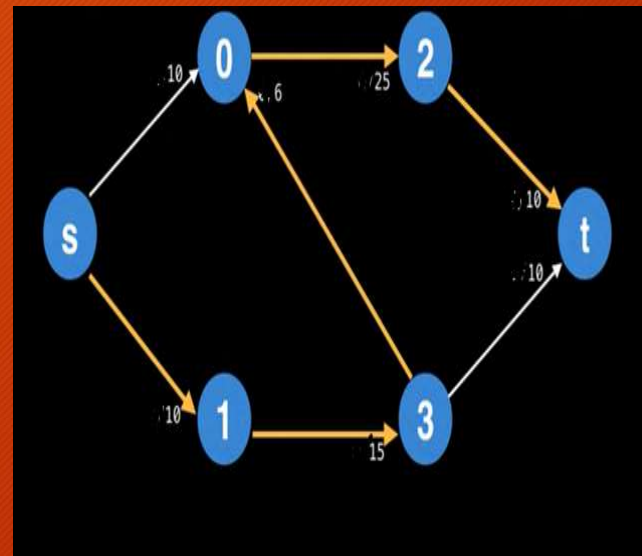
- In graph theory, a **flow network** / **network flow** is defined as a directed graph.
- Network flow involves a source vertex, sink vertex and several other vertices / nodes connected with edges.
- Each edge has an individual capacity which is the maximum limit of flow that edge could allow.
- **Source vertex** is a vertex with indegree zero.
- **Sink vertex** is a vertex with outdegree zero.



Network Flow

4

- **Network flow problems** are a class of computational problems in which the input is a flow network (a graph with numerical capacities on its edges), and the goal is to construct a flow.



Network Flow Conditions

5

- For any non-source and non-sink node, the input flow is equal to output flow.
- For any edge(E_i) in the network, $0 \leq \text{flow}(E_i) \leq \text{Capacity}(E_i)$.
- Total flow out of the source node is equal to total flow in to the sink node.

Maximum Flow

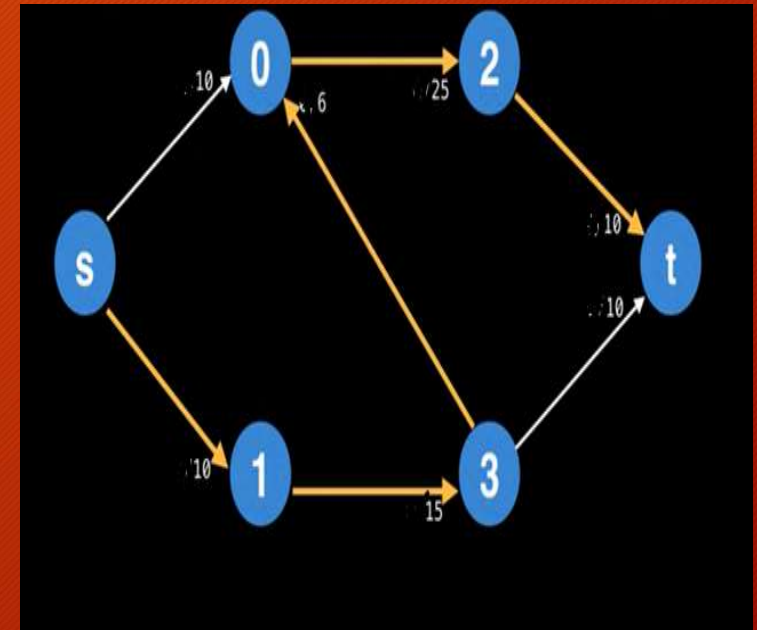
6

- It is defined as the maximum amount of flow that the network would allow to flow from source to sink.
- Each edge has certain capacity. Flow should not exceed the capacity.

Max Flow - Applications

7

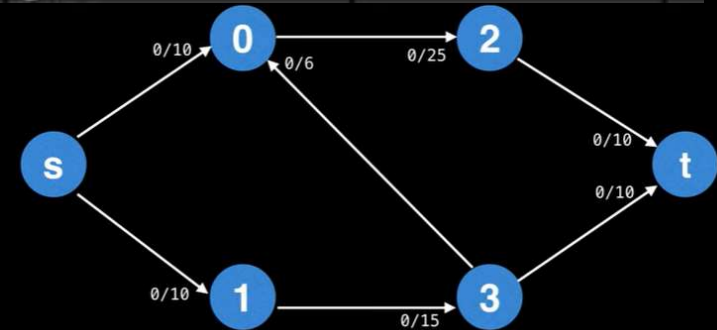
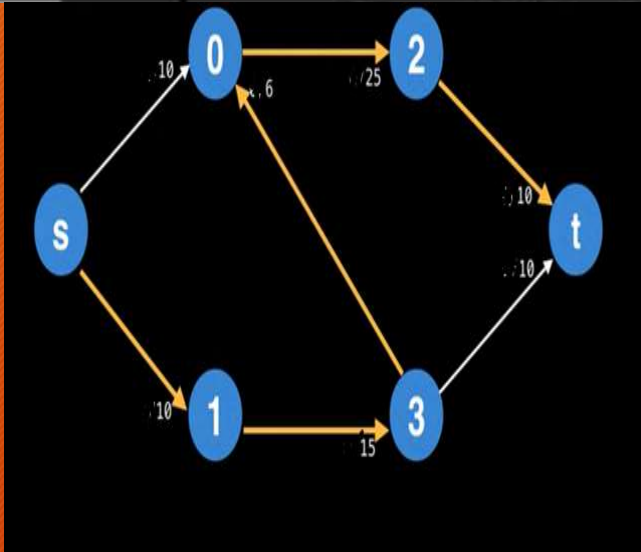
- Obvious applications of network flow involve physical situations.
- Suppose edges are roads with cars, pipes with water or wires with electric current. Flow represents the volume of water allowed to flow through the pipes, the number of cars the road can sustain in traffic or net electric current.



Flow Graph / Flow Network

8

A **flow graph** (flow network) is a directed graph where each edge (also called an arc) has a certain **capacity** which can receive a certain amount of **flow**. The flow running through an edge must be less than or equal to the capacity.



Each edge in the flow graph has a certain flow and capacity specified by the fraction adjacent to each edge. Initially, the flow through each edge is 0 and the capacity is a non negative value.

Max Flow

9

- Multiple algorithms exist in solving the maximum flow problem.
- Major algorithm to solve these kind of problems is Ford-Fulkerson algorithm

Ford-Fulkerson Algorithm

10

- It was developed by L. R. Ford, Jr. and D. R. Fulkerson in 1956.
- Inputs required are network flow graph G , source node S and sink node T .

Ford-Fulkerson Algorithm

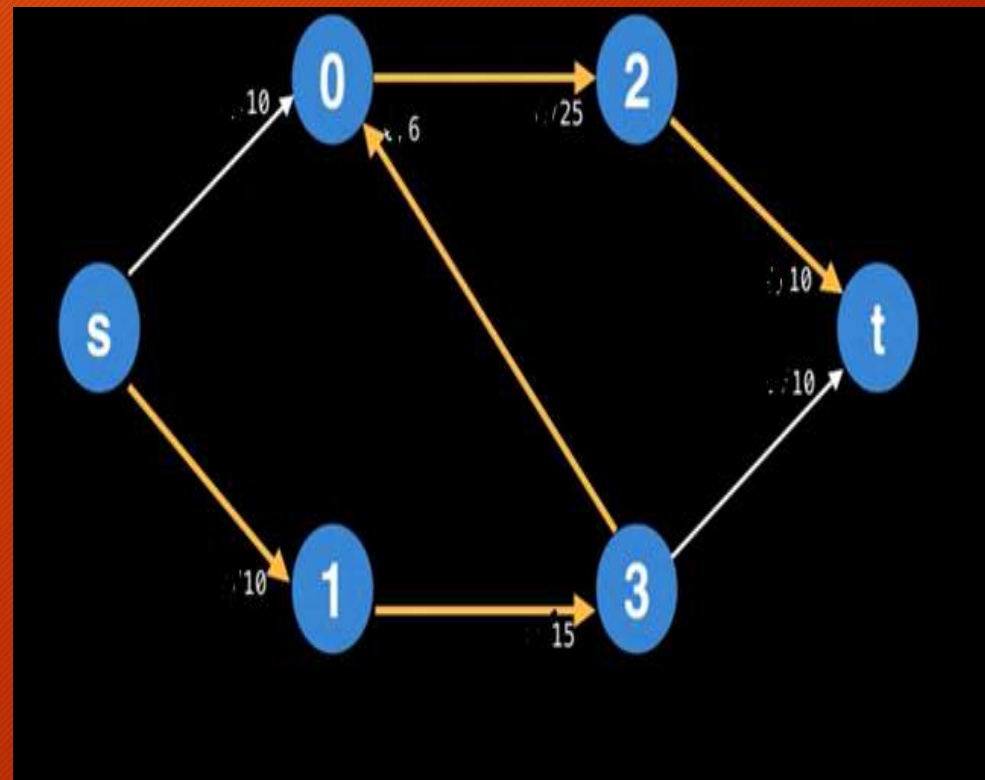
11

- To find the maximum flow, the ford Fulkerson algorithm repeatedly finds augmenting paths through the residual graph and augments the flow until no more augmenting paths can be found.

Augmenting path

12

- Augmenting path - Simple path from source to sink which do not include any cycles and that pass only through positive weighted edges.

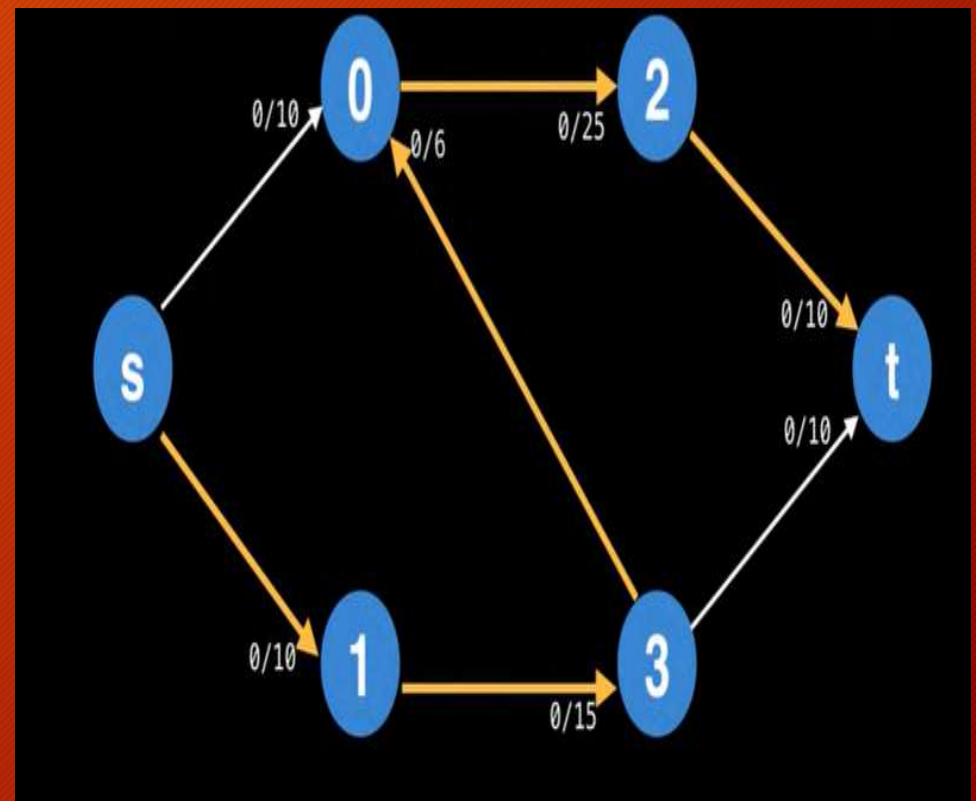


Residual network graph

13

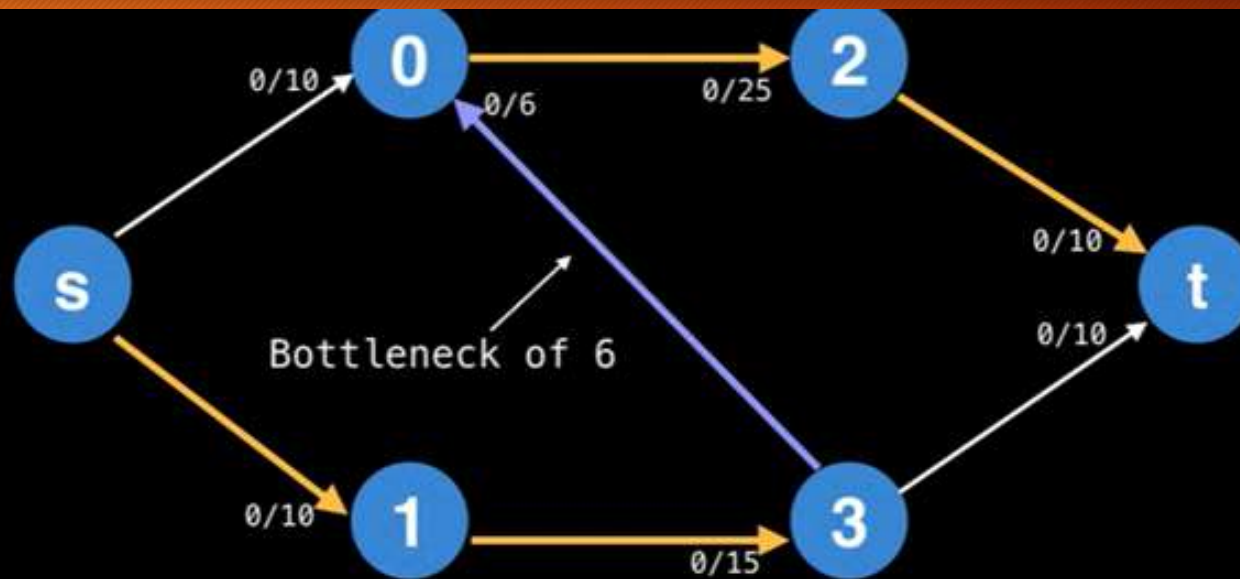
- A residual network graph indicates how much more flow could be allowed in each edge in the network graph i.e., unused flow of each edge.
- Initially flow is 0.

Flow graph



Bottleneck value

14



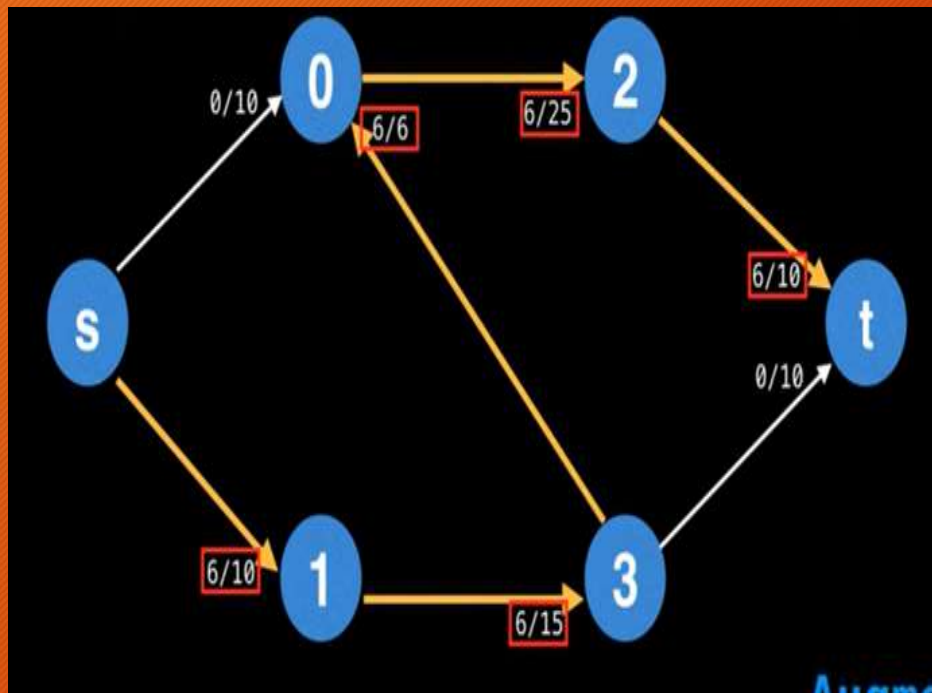
In the augmenting path above, the **bottleneck** is the “smallest” edge on the path. We can use the bottleneck value to **augment the flow** along the path.

Augmenting the flow

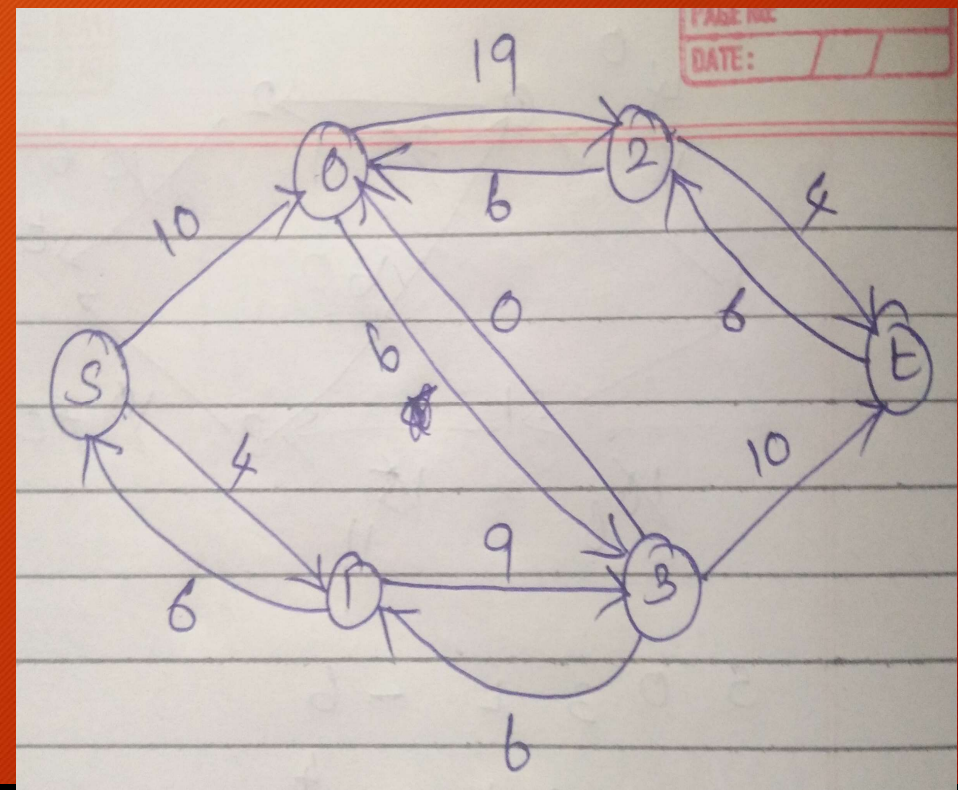
15

Increase the flow by bottleneck value.
Therefore, Flow = 0 + 6 = 6

Flow graph



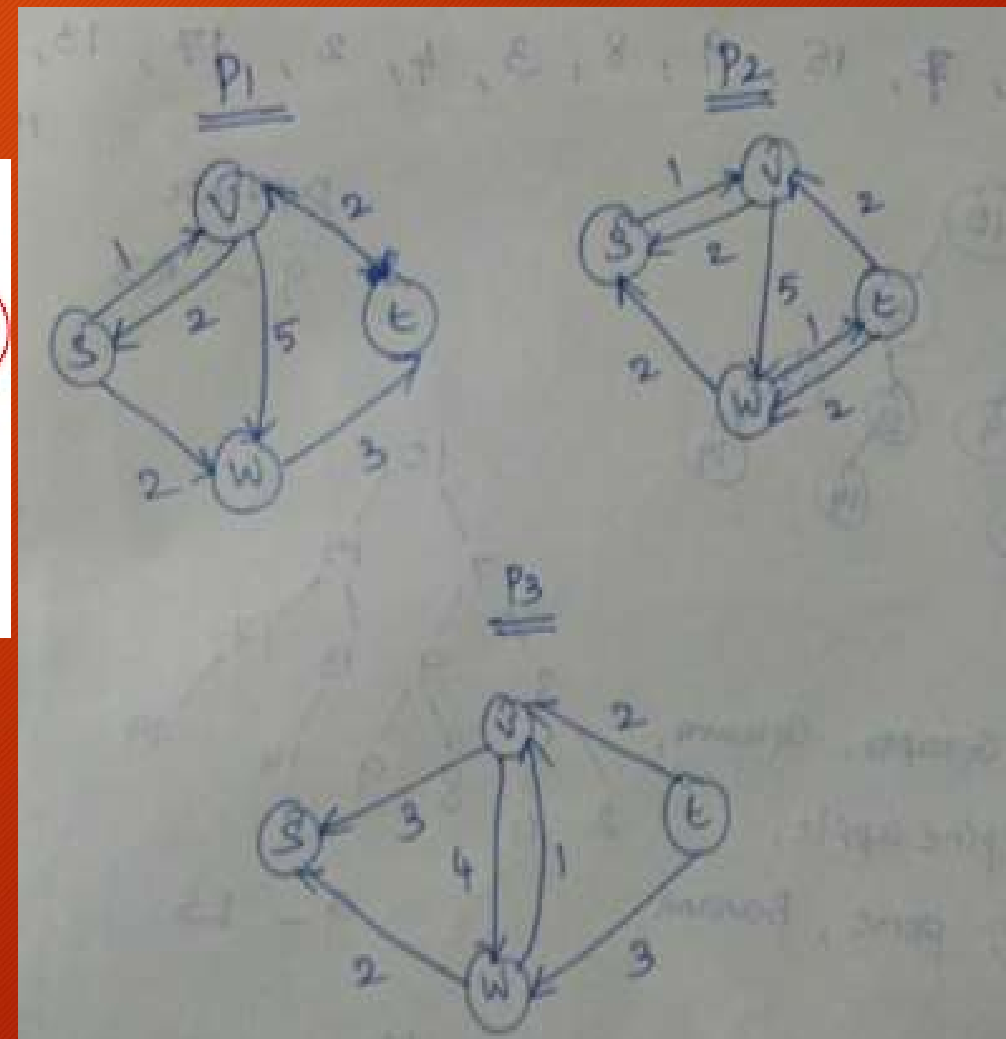
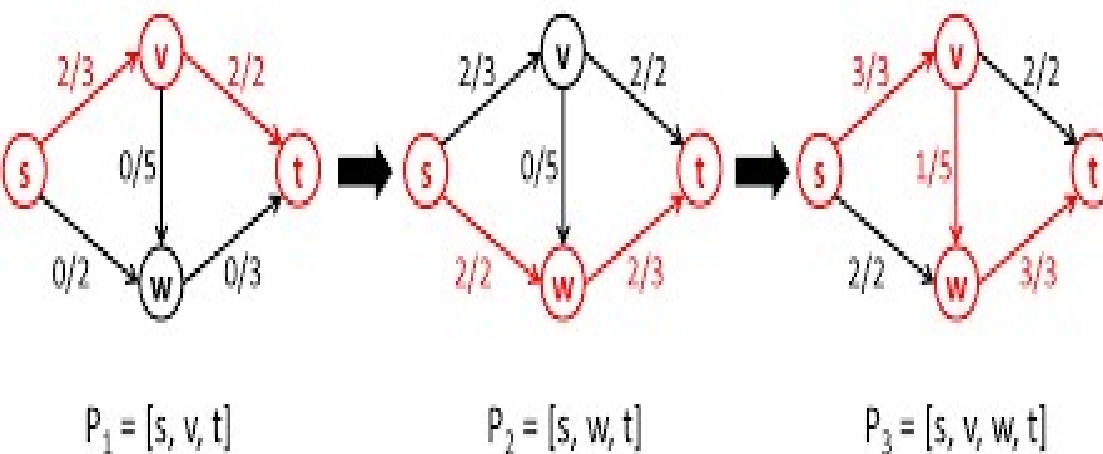
Residual network graph



Augmenting the flow means **updating** the flow values of the edges along the augmenting path.

Augmenting paths and Residual graphs

16



Ford-Fulkerson Algorithm

17

- As per algorithm, if there are no augmenting paths possible from S to T, then the flow is maximum.
- If there is a path from source to sink in residual graph, then it is possible to add flow
- The result i.e. the maximum flow will be the total flow out of source node which is also equal to total flow in to the sink node.

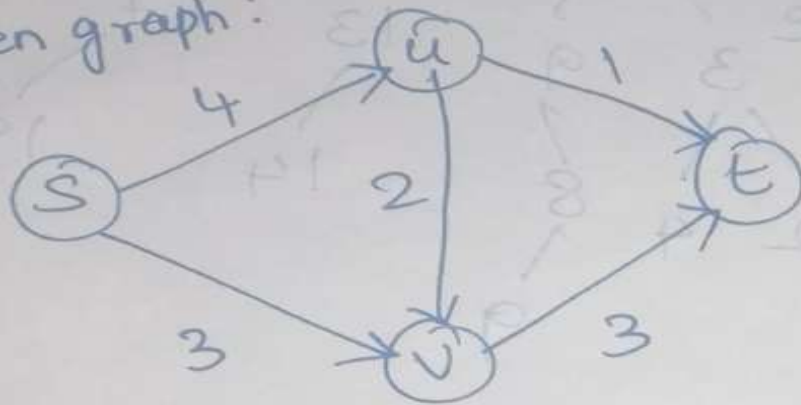
Ford-Fulkerson Algorithm

18

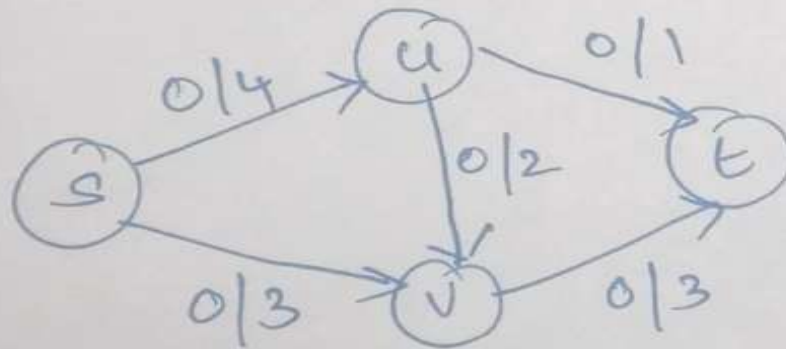
FordFulkerson (Graph G , Node S , Node T):

1. Initialize flow in all edges to 0
2. while (there exists an augmenting path(P) between S and T in residual network graph)
 - {
 Augment flow between S to T along the path P
 Update residual network graph
}
3. Return flow

Given graph:



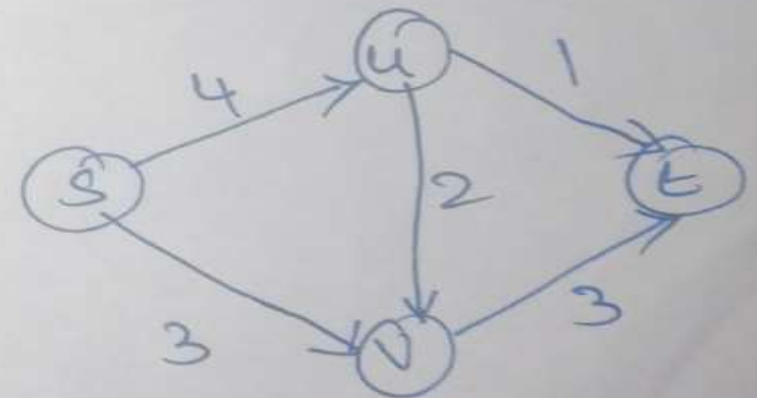
Step 1 : Initialize flow = 0.



Source = s

Sink = t

Residual graph



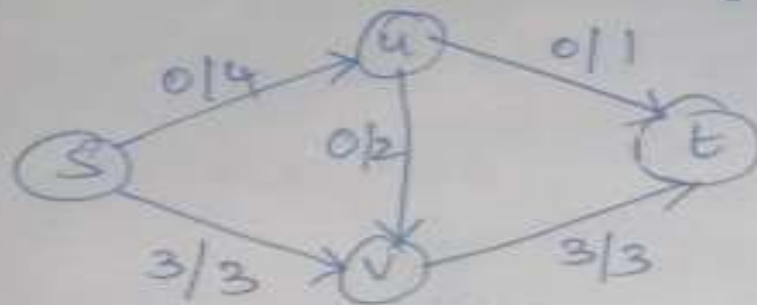
Example Problem - 1

20

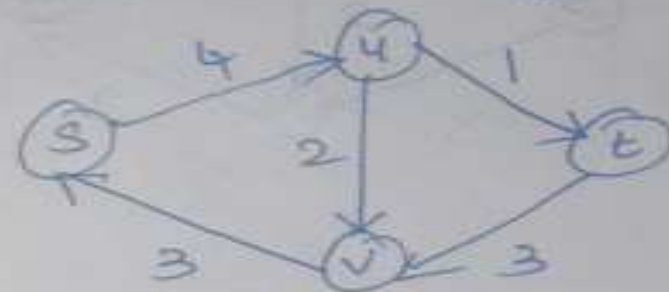
Step 2 : Identity augmenting paths in residual graph.

$$P_1 = [S, v, t]$$

Flow = 3
N/w Flow graph.

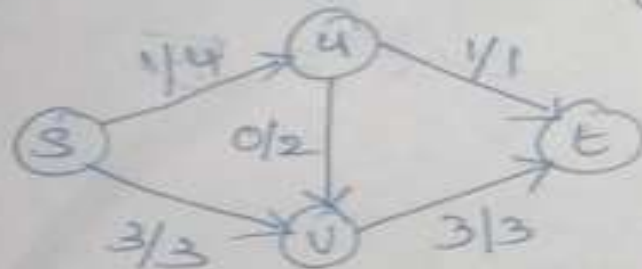


Residual graph

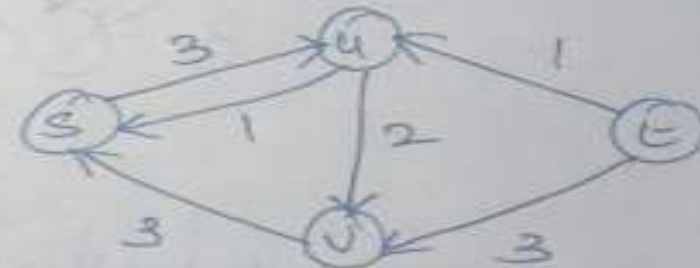


$$P_2 = [S, u, t]$$

Flow = 1
N/w flow graph



Residual graph



Example Problem - 1

21

↳ At this stage, no augmenting paths possible from s to t in residual graph.

↳ Algorithm terminates.

Maximum flow = total flow out of source node

$$= 1 + 3$$

$$= 4$$

= total flow into the sink node.

Time Complexity

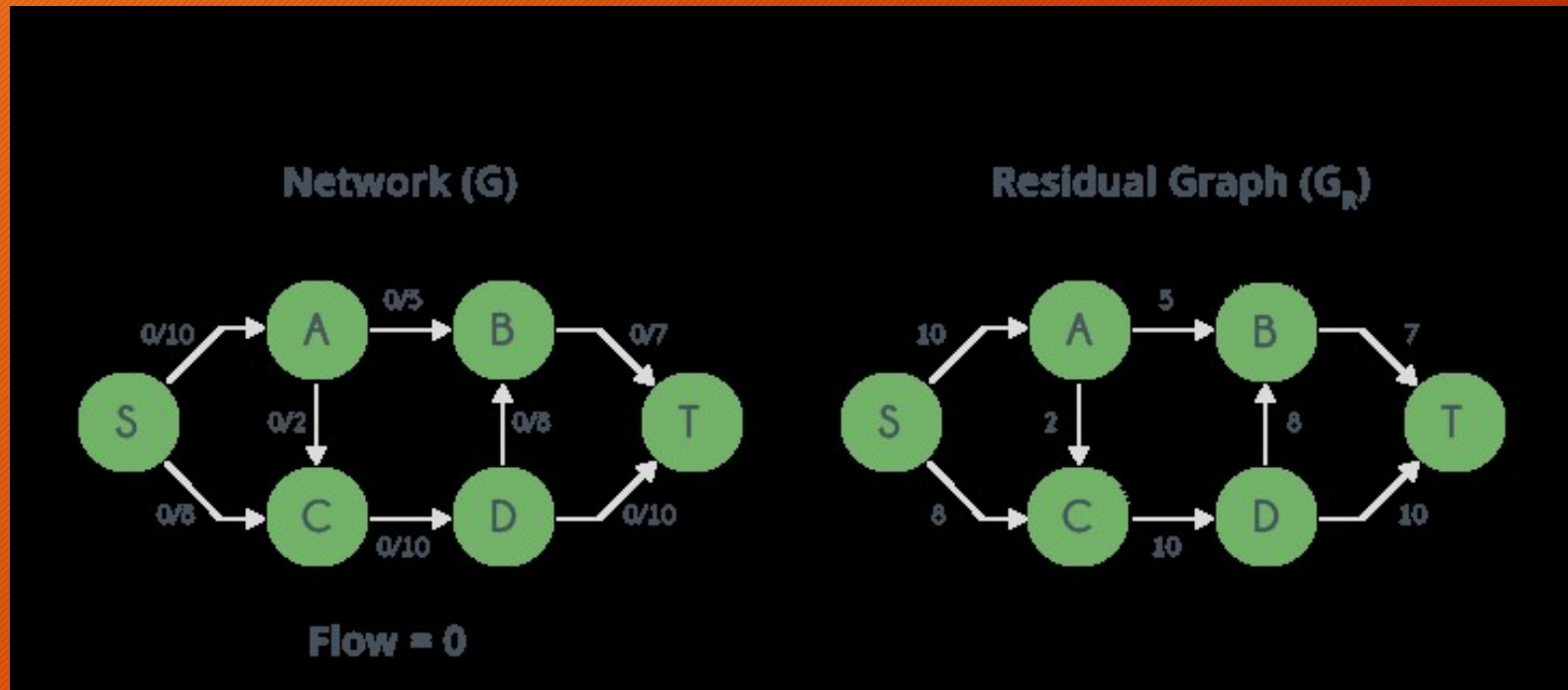
22

- Time Complexity = $O(\text{Max Flow} * E)$

We run a loop while there is an augmenting path. In worst case, we may add 1 unit flow in every iteration. Therefore the time complexity becomes $O(\text{max_flow} * E)$.

Example Problem - 2

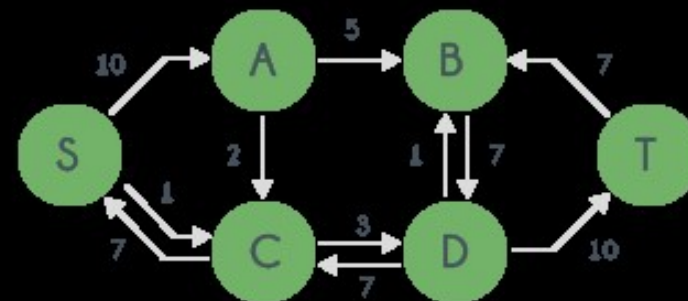
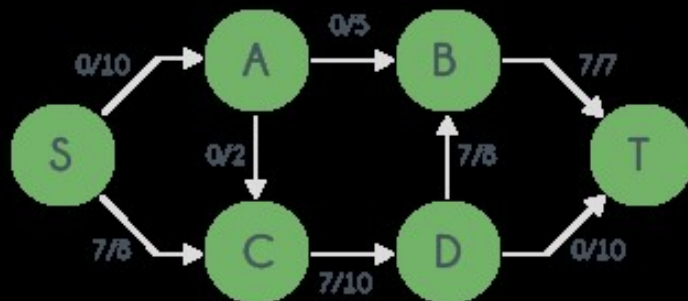
23



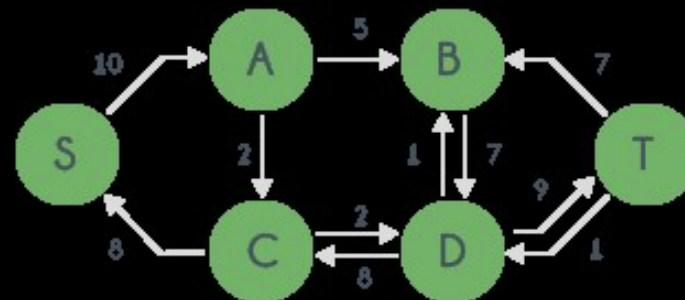
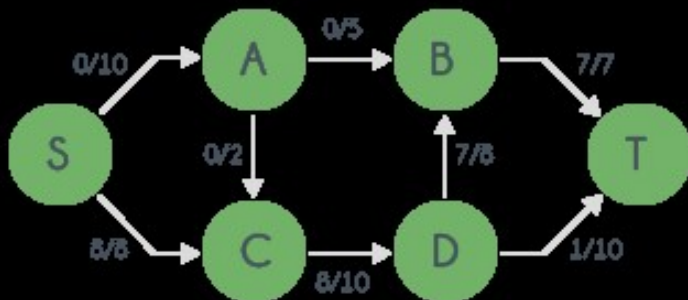
Example Problem - 2

24

Path 1: S - C - D - B - T $\rightarrow$ Flow = Flow + 7



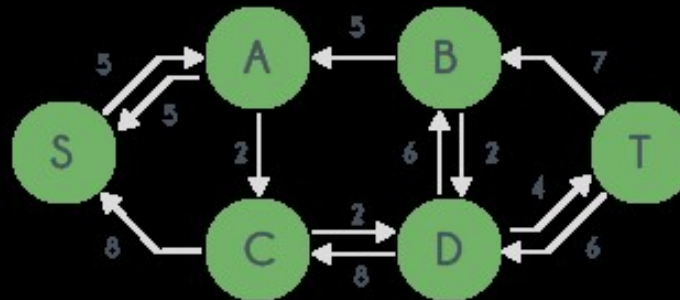
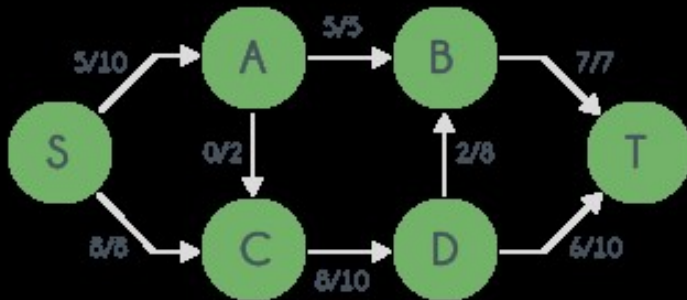
Path 2: S - C - D - T $\rightarrow$ Flow = Flow + 1



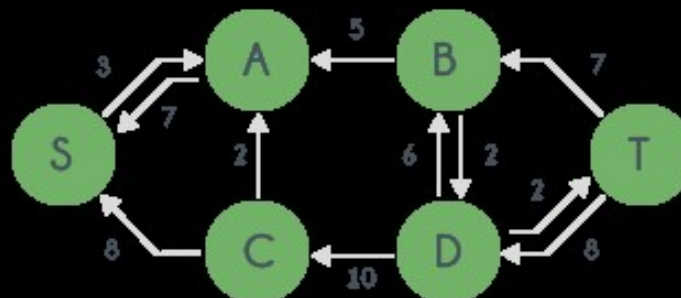
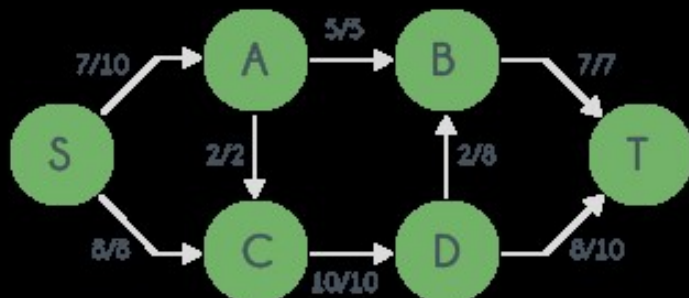
Example Problem - 2

25

Path 3: S - A - B - T → Flow = Flow + 5



Path 4: S - A - C - D - T → Flow = Flow + 2



No More Paths Left
Max Flow = 15