Advanced Algorithms & Data Structures

Department of CSE

ADVANCED ALGORITHMS AND DATA STRUCTURES
23CS03HF

Topic:

The Ford-Fulkerson Method

Session - 26

AIM OF THE SESSION

To familiarize students with the concept of the Ford-Fulkerson Method.

INSTRUCTIONAL OBJECTIVES

This Session is designed to:

1. Demonstrate :- the Ford-Fulkerson Method.
2. Describe :- Network flow and the Ford-Fulkerson maximum flow.

LEARNING OUTCOMES

At the end of this session, you should be able to:

1. Define :- the Ford-Fulkerson Method.
2. Describe :- Network flow and the Ford-Fulkerson maximum flow
3. Summarize:- Ford-Fulkerson maximum flow.

Maximum Flow

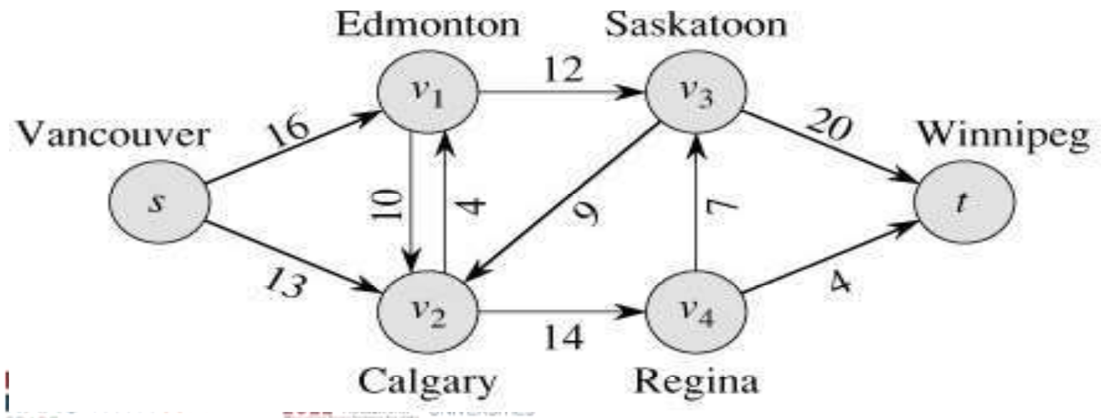
It is defined as the maximum amount of flow that the network would allow to flow from source to sink.

Multiple algorithms exist in solving the maximum flow problem.

Two major algorithms to solve these kind of problems are Ford-Fulkerson algorithm and Dinic's Algorithm.

What is Network Flowing

- In graph theory, a flow network is defined as a directed graph involving a source(S) and a sink(T) and several other nodes connected with edges.
- Each edge has an individual capacity c(u,v)>=0 which is the maximum limit of flow that edge could allow.
- If (u,v) is not in E assume c(u,v)=0.
- Flow in a network is an integer-valued function f defined on the edges of G satisfying 0≤f(u,v)≤c(u,v), for every edge (u,v) in E.
- Assume that every vertex v in V is on some path from s to t.
- Following is an illustration of a network flow:



Initially,
c(s,v1)=16
c(v1,s)=0
c(v2,s)=0 …

Conditions for network flow

For each edge (u,v) in E, the flow f(u,v) is a real valued function that must satisfy following 3 conditions :

- Capacity Constraint : $\forall\ u,v \in V,\ \ f(u,v) \leq c(u,v)$
- Skew Symmetry : $\forall\ u,v \in V,\ \ f(u,v) = -f(v,u)$
- Flow Conservation: $\forall\ u \in V - \{s,t\}\ \ \sum_{v \in V} f(s,v) = 0$

Skew symmetry condition implies that f(u,u)=0.

The value of a flow

The value of a flow is given by :

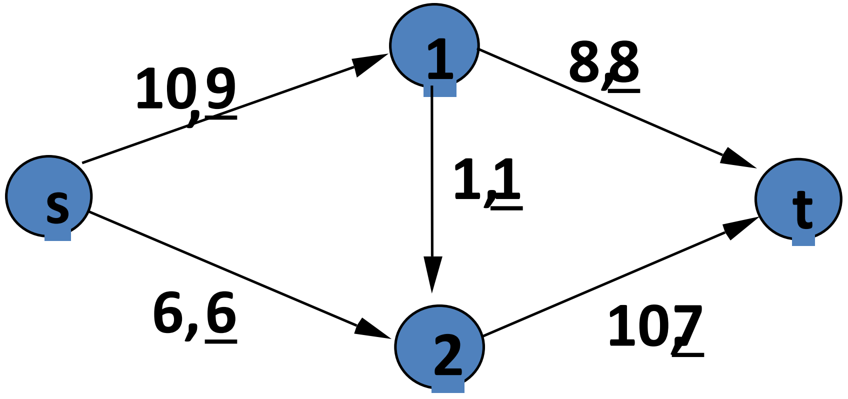
$$|f| = \sum_{v \in V} f(s,v) = \sum_{v \in V} f(v,t)$$

The flow into the node is same as flow going out from the node and thus the flow is conserved. Also the total amount of flow from source s = total amount of flow into the sink t.

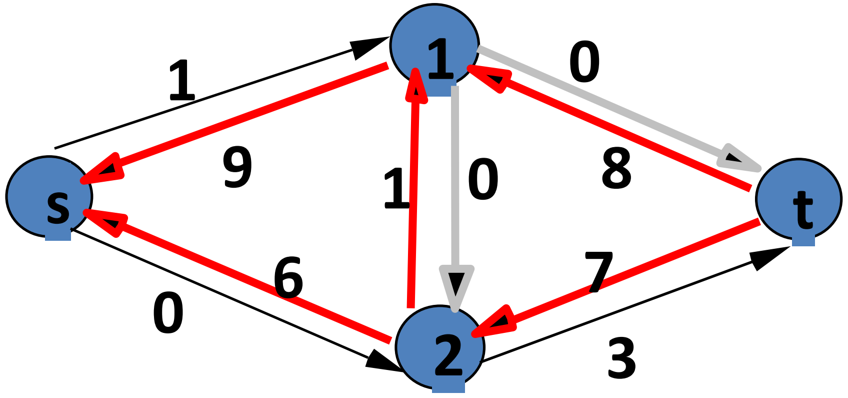
The Ford-Fulkerson Method

- It was developed by L. R. Ford, Jr. and D. R. Fulkerson in 1956.
- A residual network graph indicates how much more flow is allowed in each edge in the network graph.
- The residual capacity of the network with a flow f is given by:
- The residual capacity (rc) of an edge (i,j) equals c(i,j) – f(i,j) when (i,j) is a forward edge, and equals f(i,j) when (i,j) is a backward edge. Moreover the residual capacity of an edge is always non-negative

$$c_f(u,v) = c(u,v) - f(u,v)$$



Original Network



Residual Network

Augmenting Paths

- An augmenting path is a simple path from source to sink which do not include any cycles and that pass only through positive weighted edges.
- If there are no augmenting paths possible from S to T, then the flow is maximum.
- 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.

Implementation

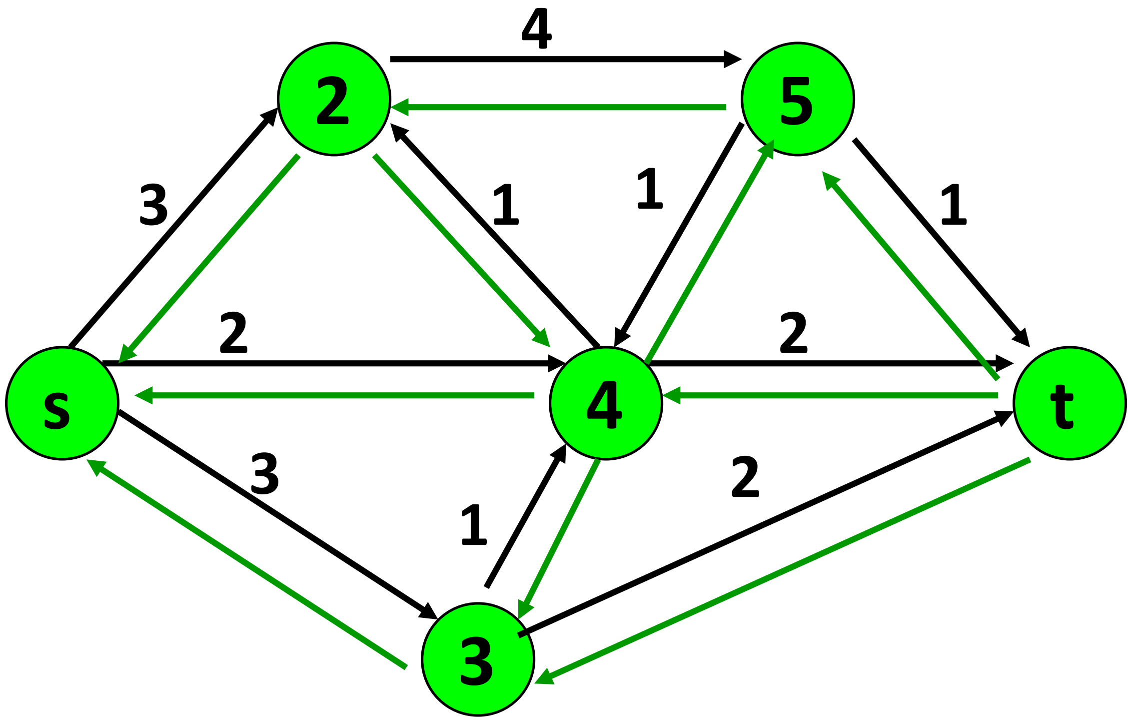
- An augmenting path in residual graph can be found using DFS or BFS.
- Updating residual graph includes following steps: (refer the diagrams for better understanding)
  - For every edge in the augmenting path, a value of minimum capacity in the path is subtracted from all the edges of that path.
  - An edge of equal amount is added to edges in reverse direction for every successive nodes in the augmenting path.

The Ford-Fulkerson's Algorithm

```
FORDFULKERSON(G,E,s,t)
FOREACH e ∈ E
   f(e) ← 0
G_f ← residual graph
WHILE (there exists augmenting path P)
   f ← augment(f, P)
   update G_f
ENDWHILE
RETURN f
```

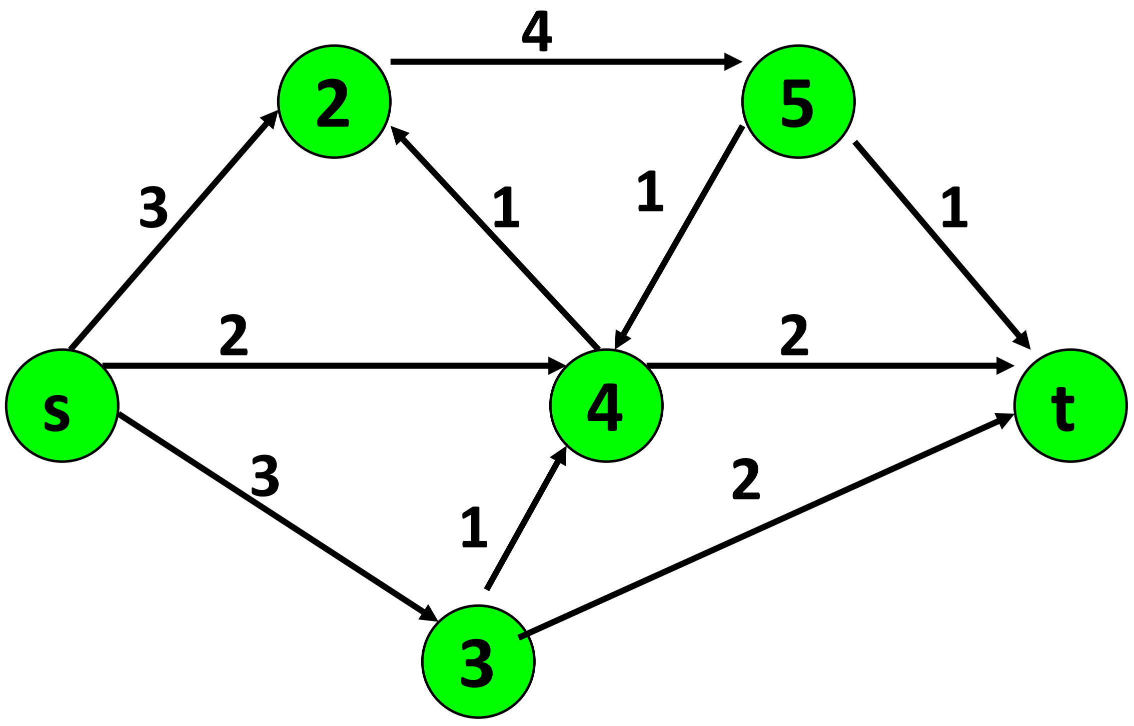
```
AUGMENT(f,P)
b ← bottleneck(P)
FOREACH e ∈ P
   IF (e ∈ E)
      // backwards arc
      f(e) ← f(e) + b
   ELSE
      // forward arc
      f(e^R) ← f(e) - b
RETURN f
```

The Ford-Fulkerson Max Flow



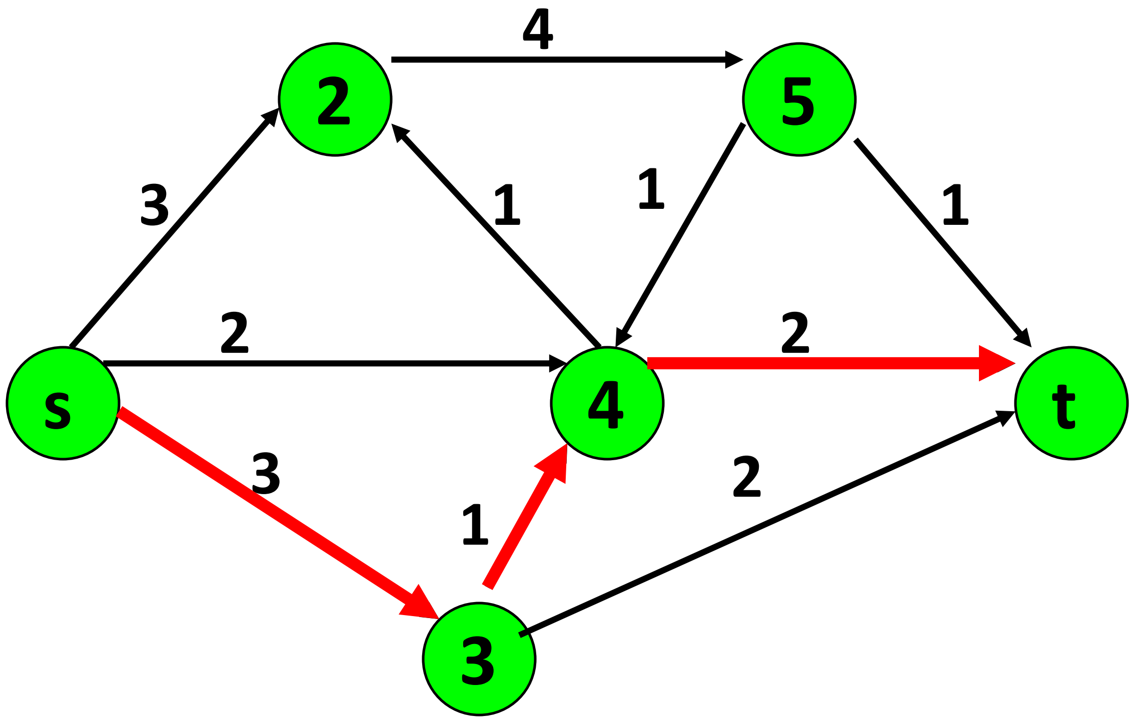
This is the original network, plus reversals of the arcs.

The Ford-Fulkerson Max Flow



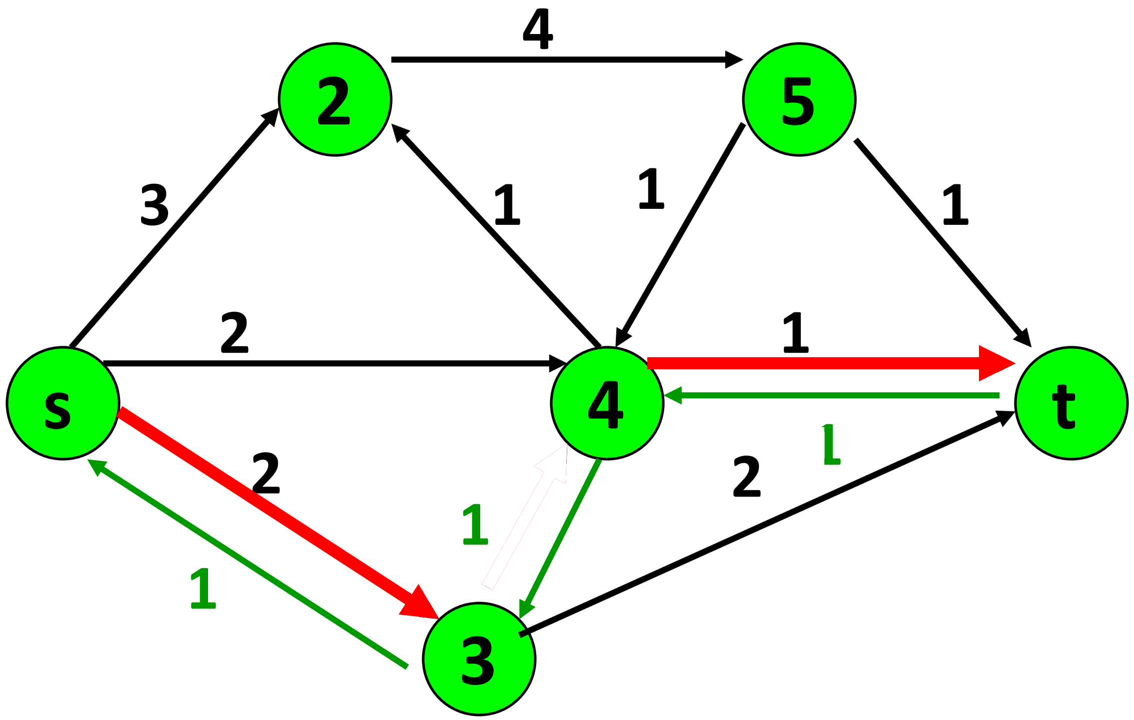
This is the original network, and the original residual network.

The Ford-Fulkerson Max Flow



Find any s-t path in G(x)

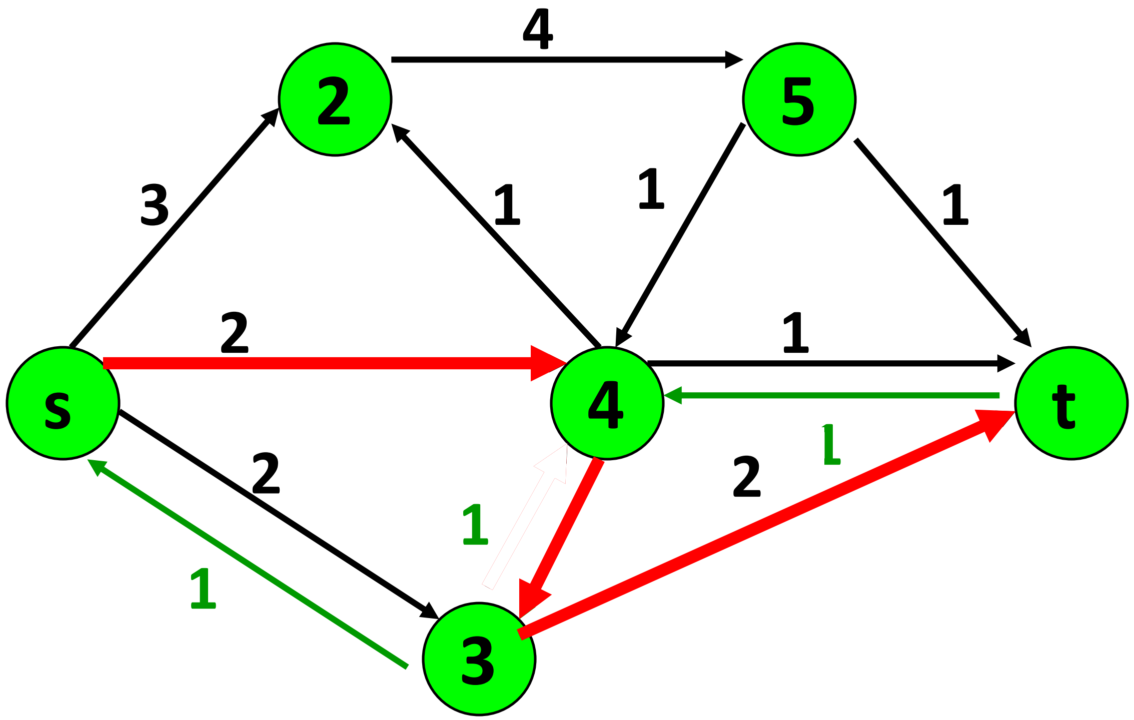
The Ford-Fulkerson Max Flow



Determine the capacity Δ of the path.

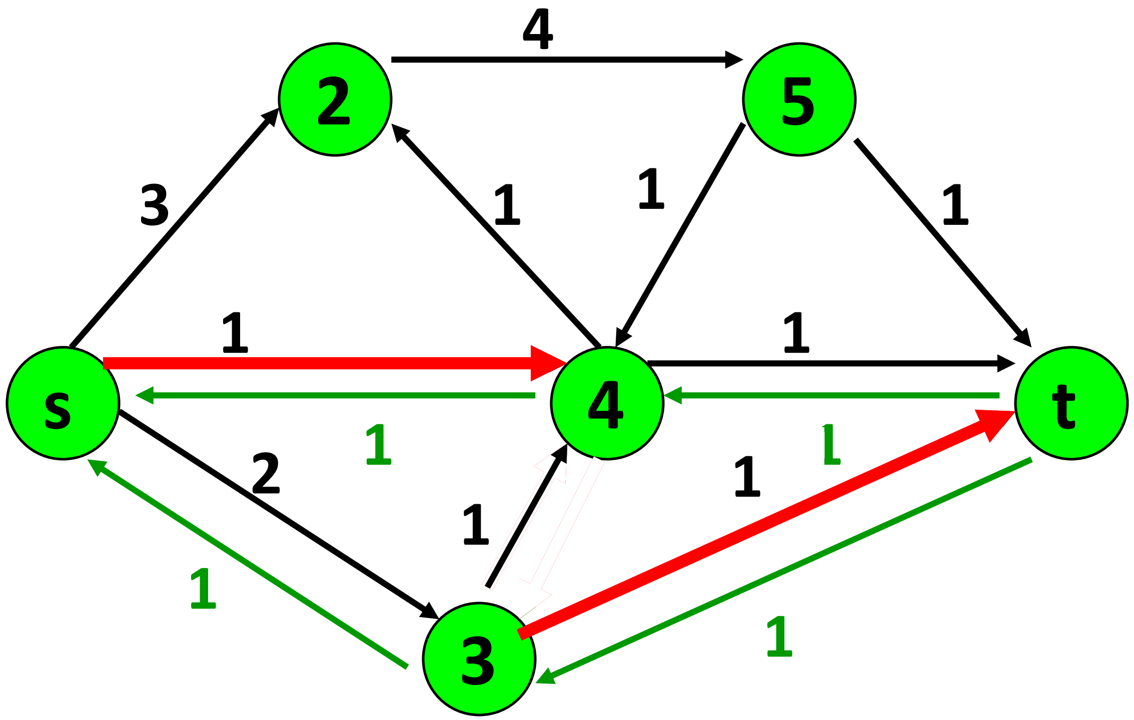
Send Δ units of flow in the path.
Update residual capacities.

The Ford-Fulkerson Max Flow



Find any s-t path

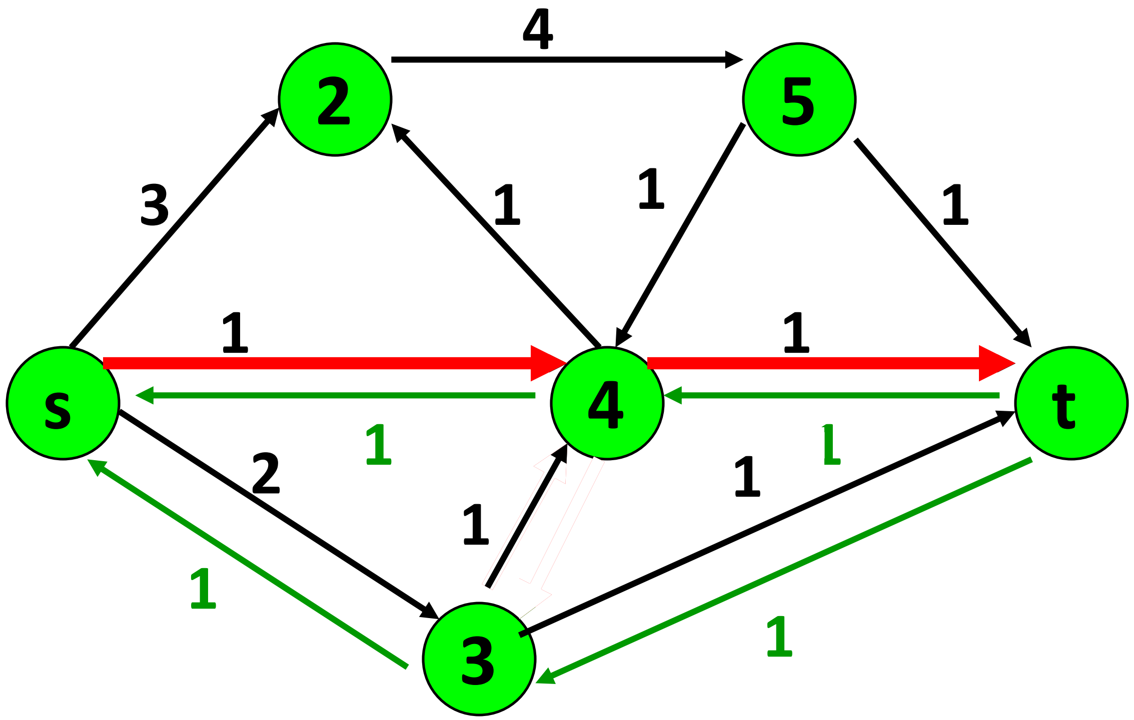
The Ford-Fulkerson Max Flow



Determine the capacity Δ of the path.

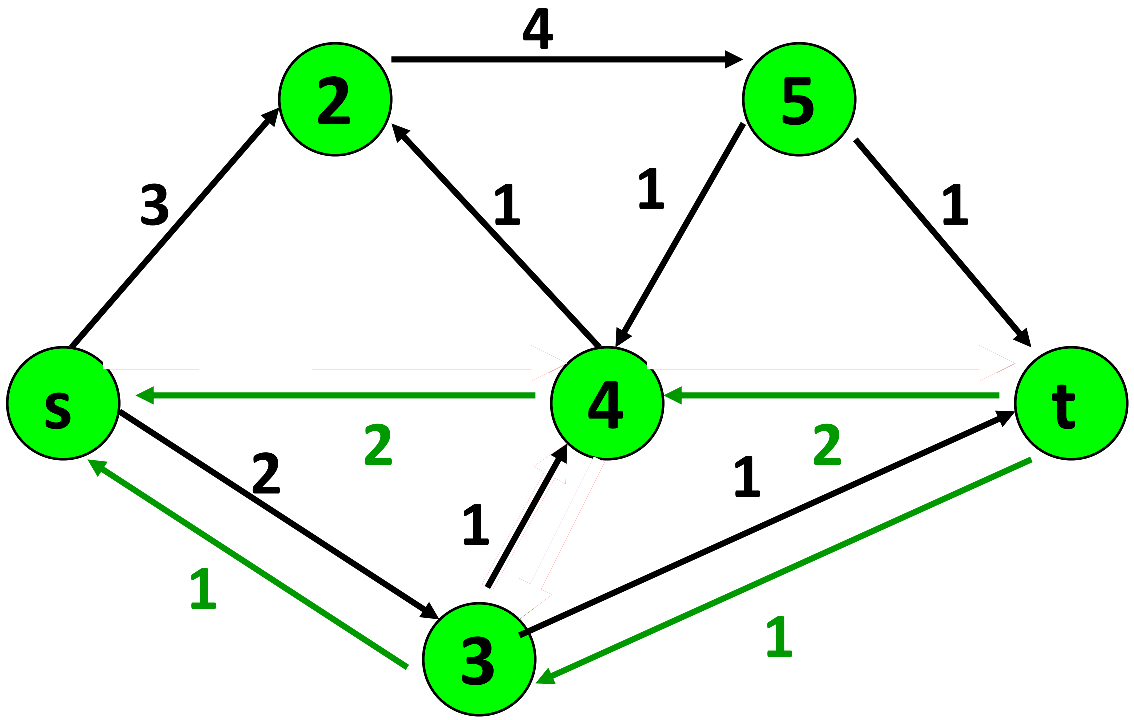
Send Δ units of flow in the path.
Update residual capacities.

The Ford-Fulkerson Max Flow



Find any s-t path

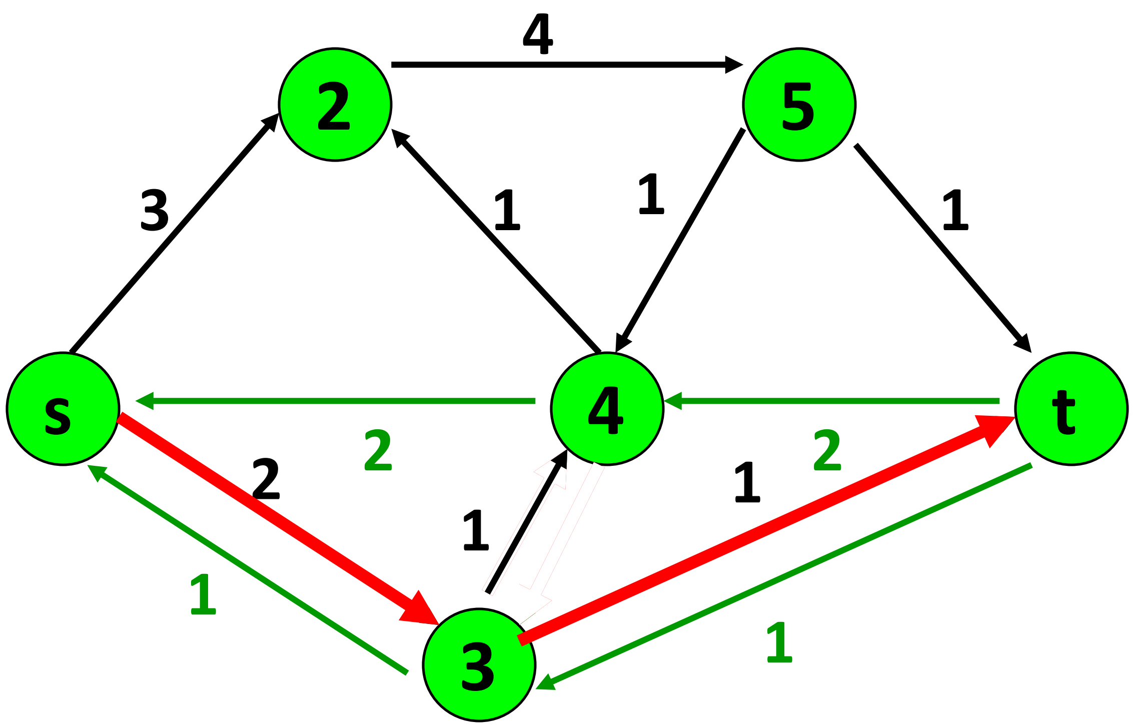
The Ford-Fulkerson Max Flow



Determine the capacity Δ of the path.

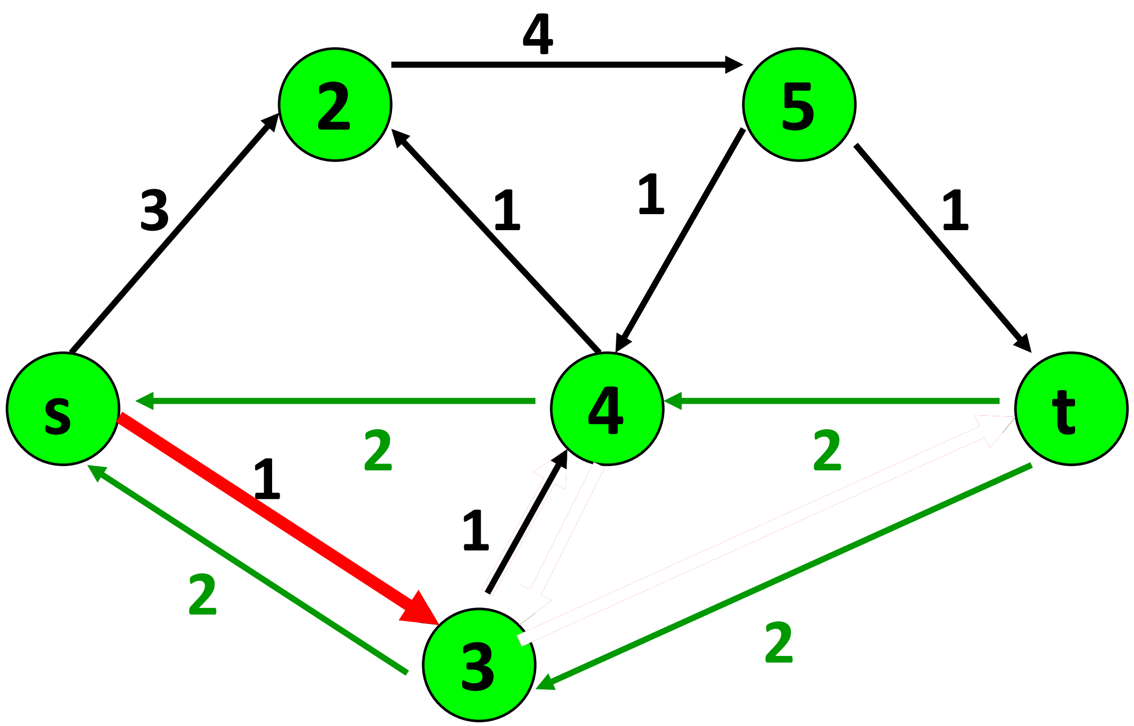
Send Δ units of flow in the path.
Update residual capacities.

The Ford-Fulkerson Max Flow



Find any s-t path

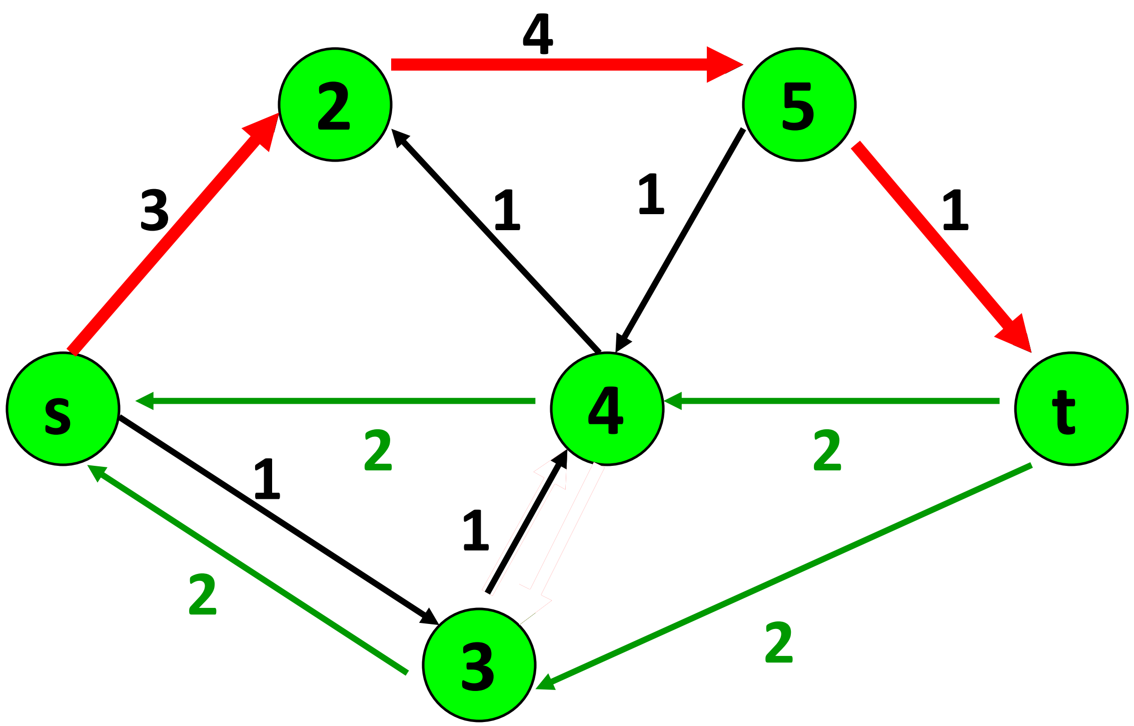
The Ford-Fulkerson Max Flow



Determine the capacity Δ of the path.

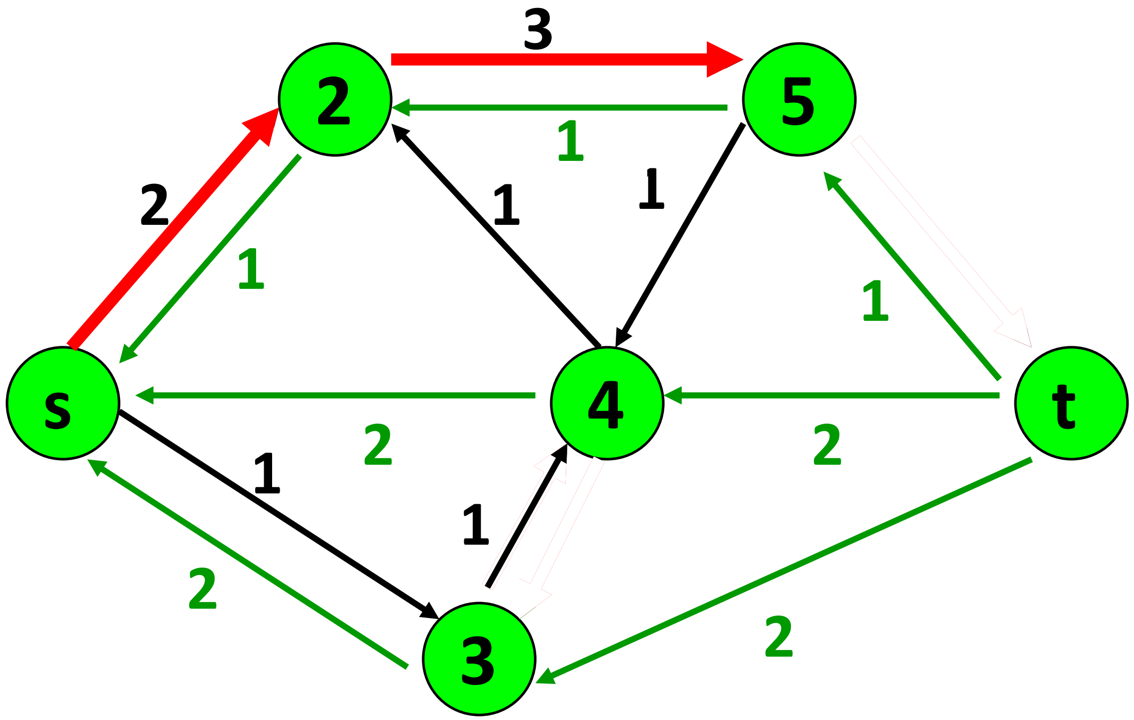
Send Δ units of flow in the path.
Update residual capacities.

The Ford-Fulkerson Max Flow



Find any s-t path

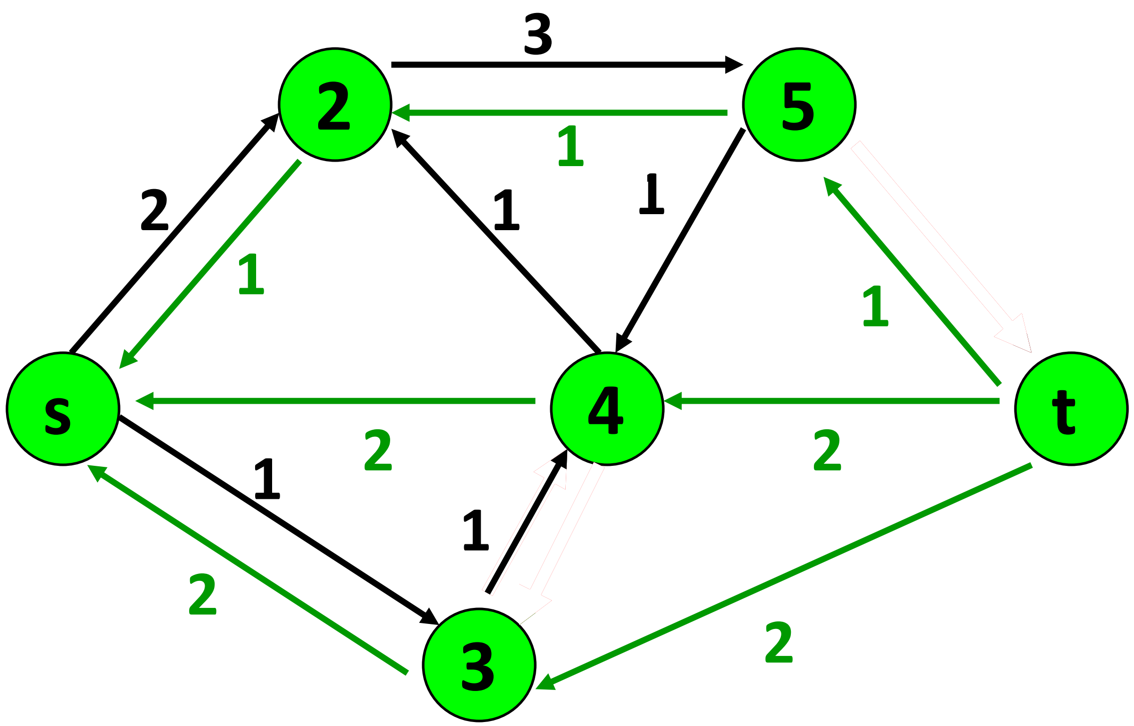
The Ford-Fulkerson Max Flow



Determine the capacity Δ of the path.

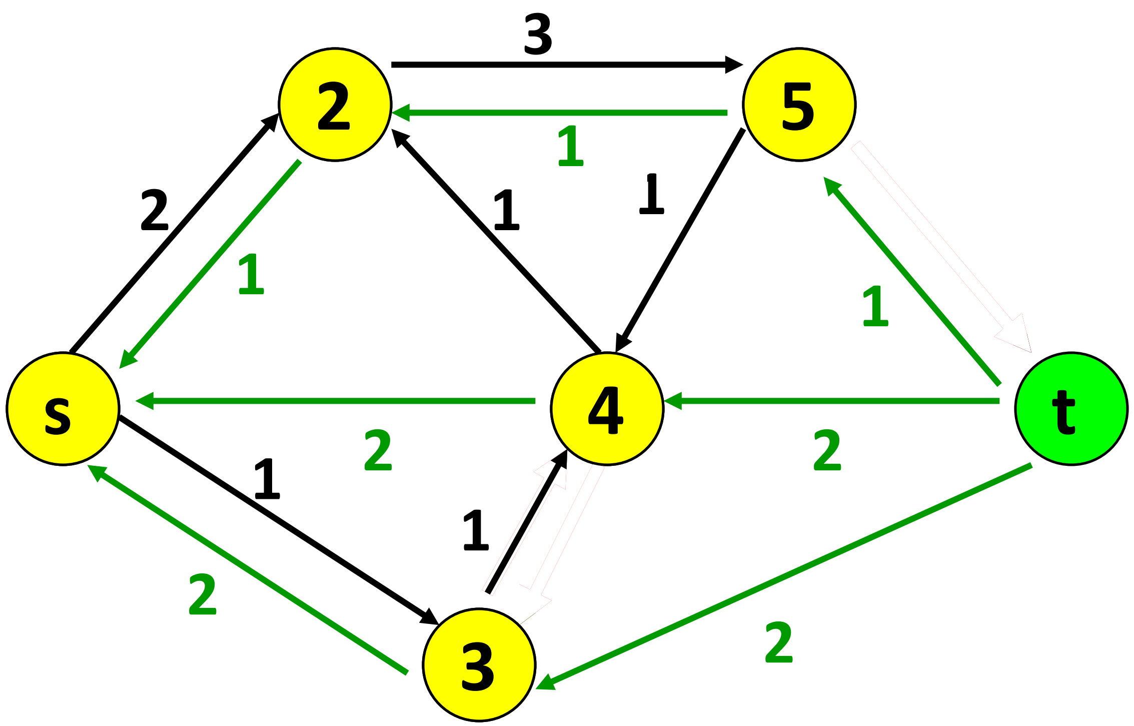
Send Δ units of flow in the path.
Update residual capacities.

The Ford-Fulkerson Max Flow



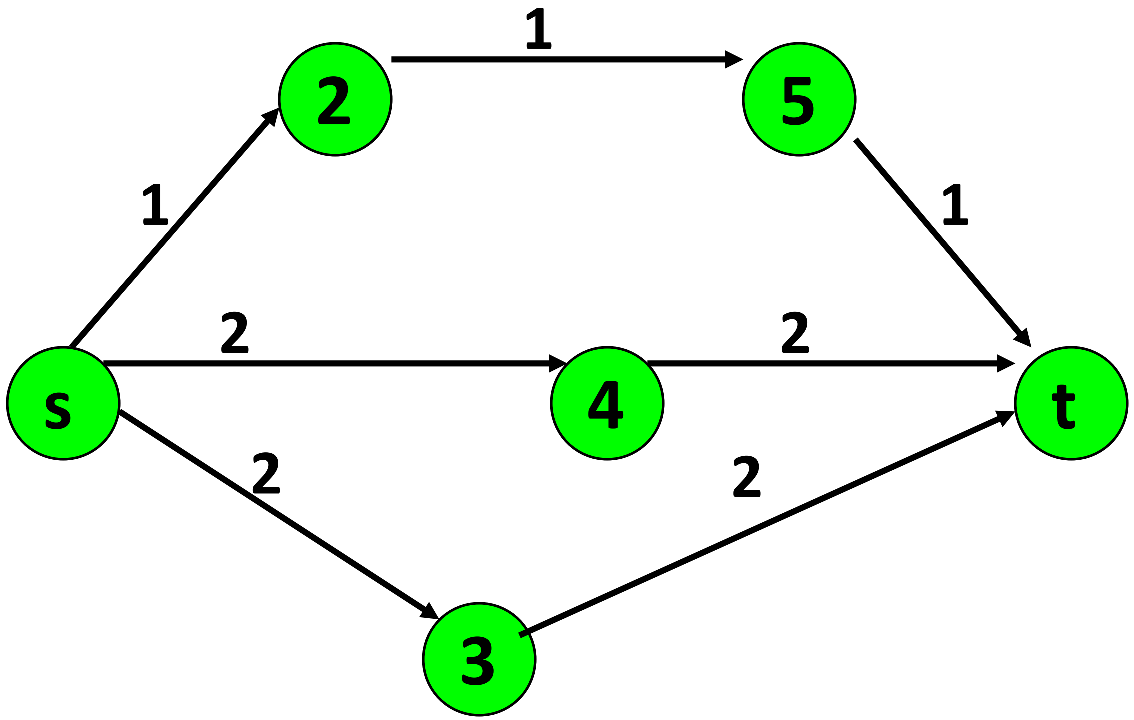
There is no s-t path in the residual network.
This flow is optimal

The Ford-Fulkerson Max Flow



These are the nodes that are reachable from node s.

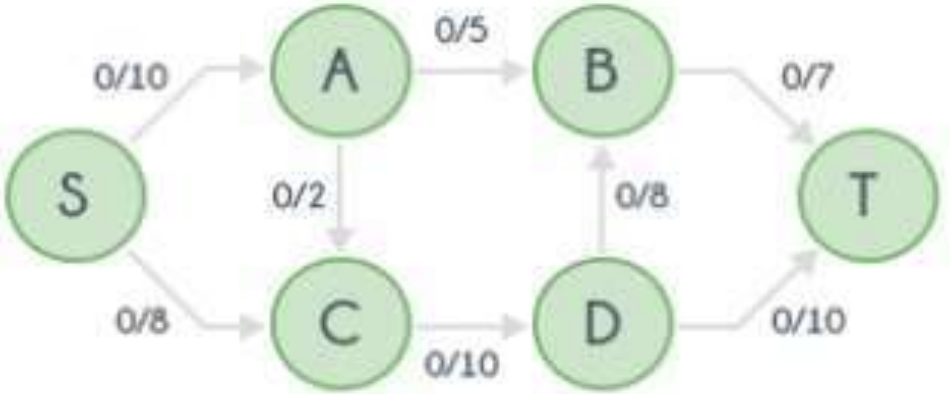
The Ford-Fulkerson Max Flow



Here is the optimal flow=1+2+2 = 5

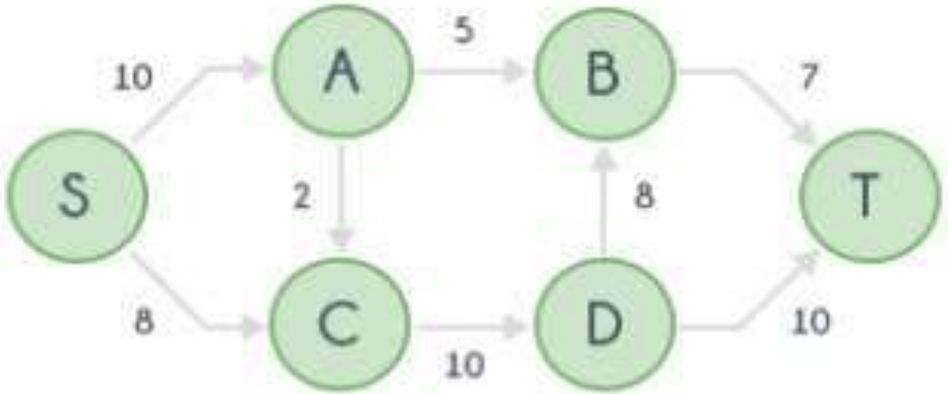
A demonstration of working of Ford-Fulkerson algorithm is shown below with the help of diagrams.

Network (G)

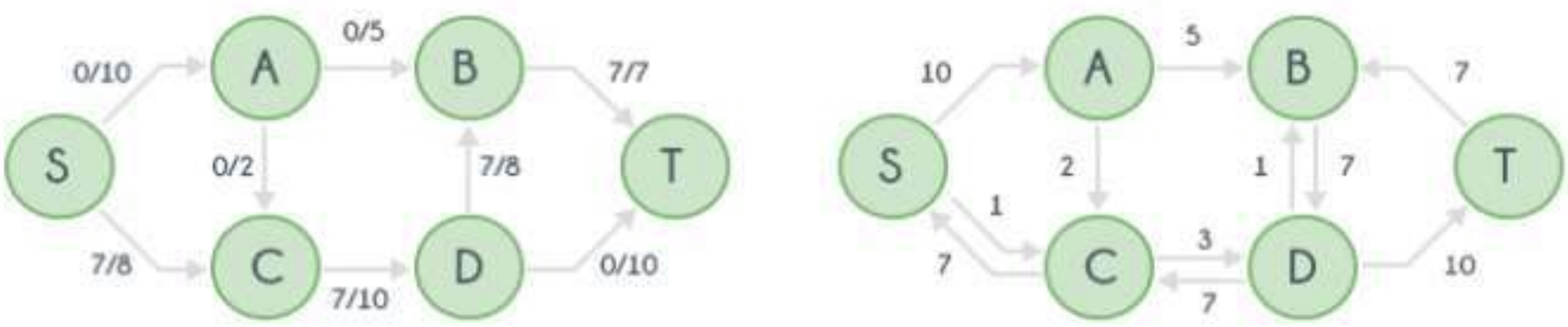


Flow = 0

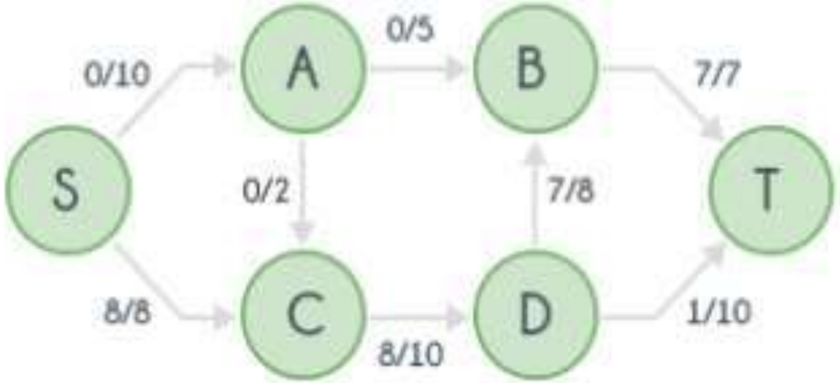
Residual Graph (G_R)

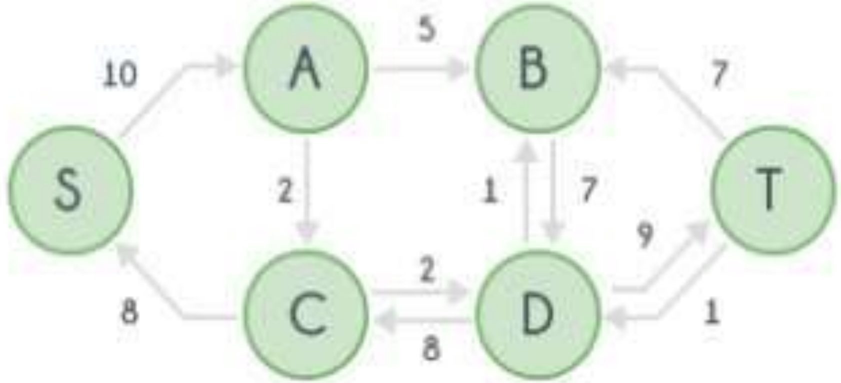


Path 1: S - C - D - B - T ⟶ Flow = Flow + 7

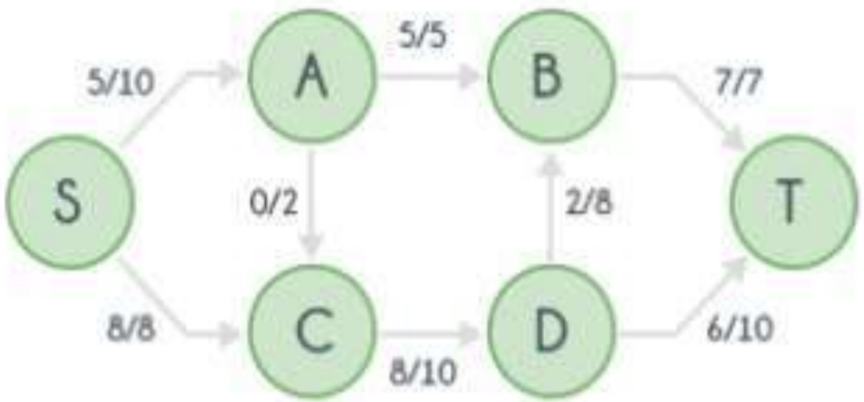


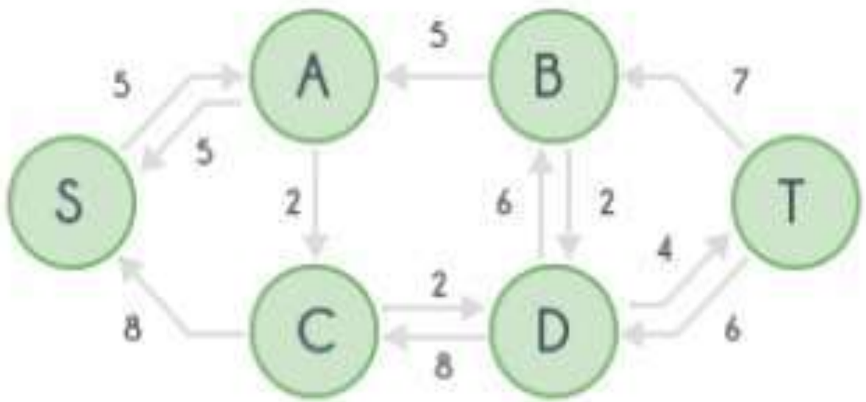
Path 2: S - C - D - T → Flow = Flow + 1



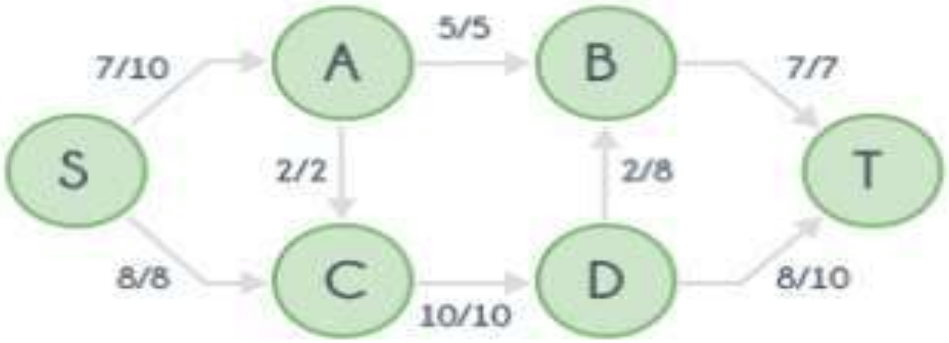


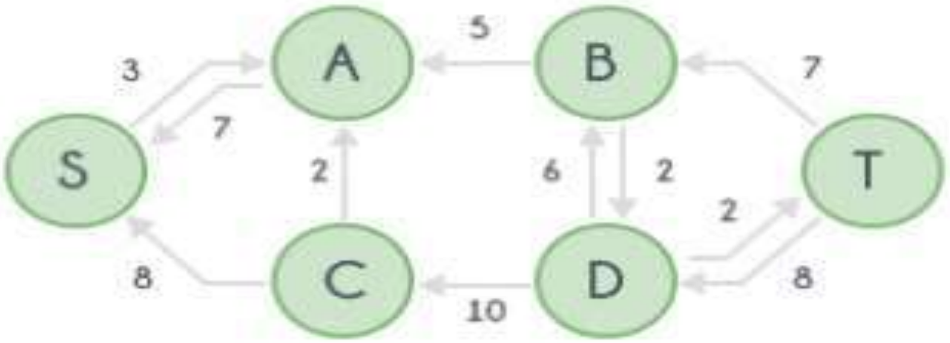
Path 3: S - A - B - T → Flow = Flow + 5





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





No More Paths Left

Max Flow = 15

KL UNIVERSITY (DEEMED TO BE UNIVERSITY)

SUMMARY

- Multiple algorithms exist in solving the maximum flow problem.
- Two major algorithms to solve this kind of problems are the Ford-Fulkerson algorithm and Dinic's Algorithm.

SELF-ASSESSMENT QUESTIONS

- Which data structure is commonly used to find augmenting paths in the Ford-Fulkerson algorithm?

A. Stack

B. Queue

C. Binary Search Tree

D. Priority Queue

- What is the time complexity of the Ford-Fulkerson algorithm in the worst case if implemented with the breadth-first search (BFS)?

A. (O(V + E))

B. (O(V^2E))

C. (O(VE^2))

D. (O(V^3))

TERMINAL QUESTIONS

1. **What is the primary objective of the Ford-Fulkerson algorithm?**
2. **What is the stopping criterion for the Ford-Fulkerson algorithm?**
3. **What does the Ford-Fulkerson algorithm use to find the augmenting path?**

REFERENCES FOR FURTHER LEARNING OF THE SESSION

Reference Books :

1Introduction to Algorithms, Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein., 3rd, 2009, The MIT Press.

2 Algorithm Design Manual, Steven S. Skiena., 2nd, 2008, Springer.

3 Data Structures and Algorithms in Python, Michael T. Goodrich, Roberto Tamassia, and Michael H. Goldwasser., 2nd, 2013, Wiley.

4 The Art of Computer Programming, Donald E. Knuth, 3rd, 1997, Addison-Wesley Professiona.

MOOCS :

1. https://www.coursera.org/specializations/algorithms?=

2.https://www.coursera.org/learn/dynamic-programming-greedy-algorithms#modules

THANK YOU

THANK YOU