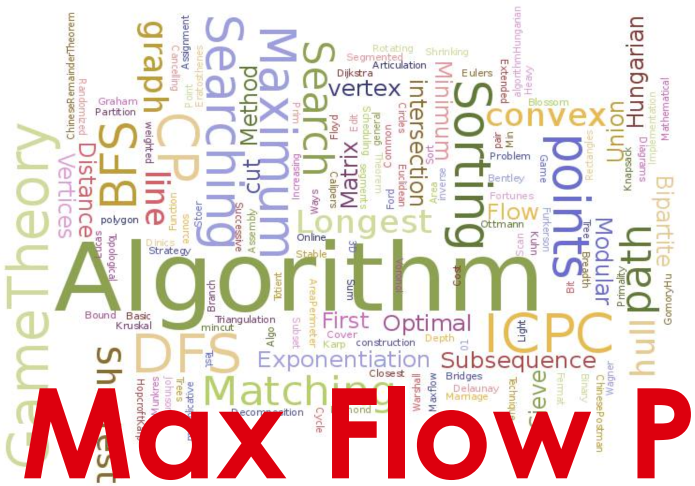


Max Flow Problem – Ford-Fulkerson Algorithm

Course: Algorithms

Faculty: Dr. Rajendra Prasath

Autumn 2018

Max Flow Problem
Ford – Fulkerson Algorithm

This lecture covers the max flow problem, more specifically Ford – Fulkerson Algorithm. In this lecture, we will explore how one can push the maximum flow into the network

Recap: Floyd – Warshall Algo

- For every pair (i, j) of vertices, there are two cases:
  - k is not an intermediate vertex in shortest path: i → j
    We keep the value of dist[i][j] unchanged

  - k is an intermediate vertex in shortest path: i → j
    Update the value of dist[i][j] as follows:

    if dist[i][j] > dist[i][k] + dist[k][j] then
        dist[i][j] = dist[i][k] + dist[k][j]

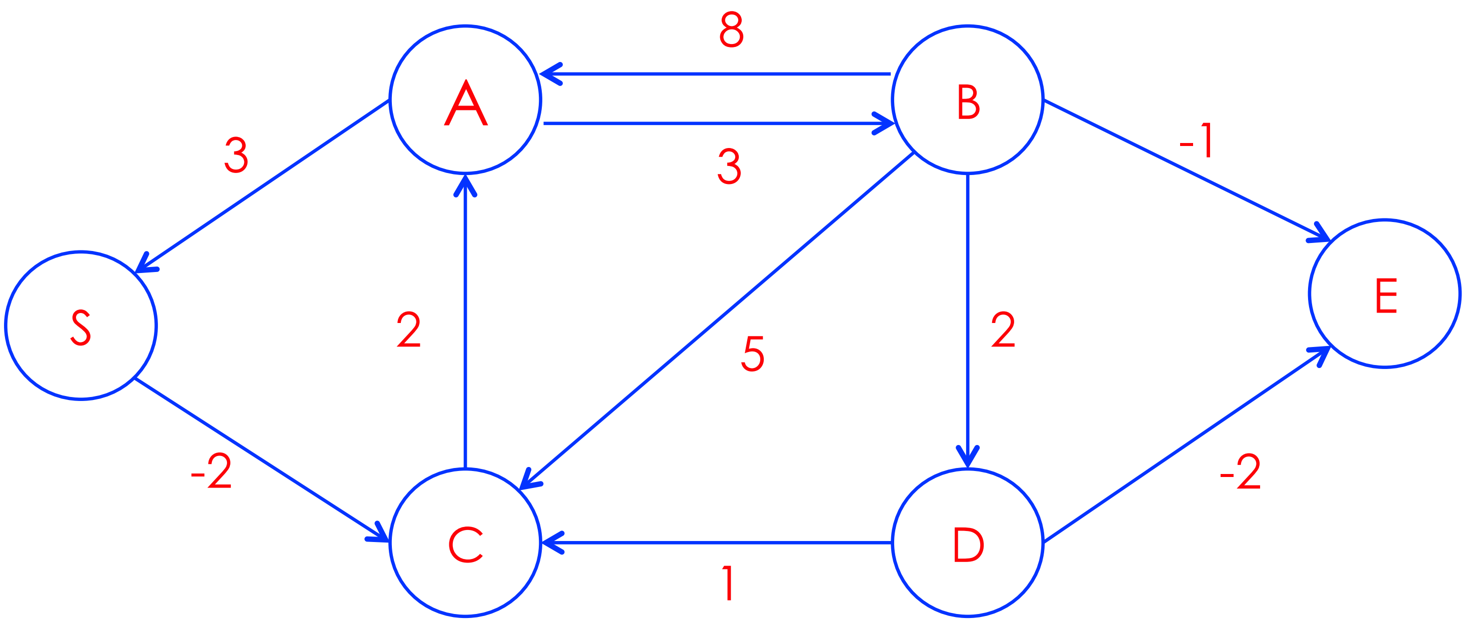
- Choose the **minimum** and store it in dist[i][j]

- Explore optimal substructure property in the all-pairs shortest path problem

Recap: Graph Algorithms

Graph Algorithms:

- Compute Single Source Shortest Path Algos
- Compute All Pairs Shortest Path Algorithms



Network Flow Algorithms

An Overview

- Numerous problems can be modeled as graph problems
  - Nodes
  - Edges with specific capacity associated with it over which commodities flow
- Variations of Linear Programming Problems
  - Optimization Problems
- Real-World problems

Network Flow Algorithms

Define a Flow Network

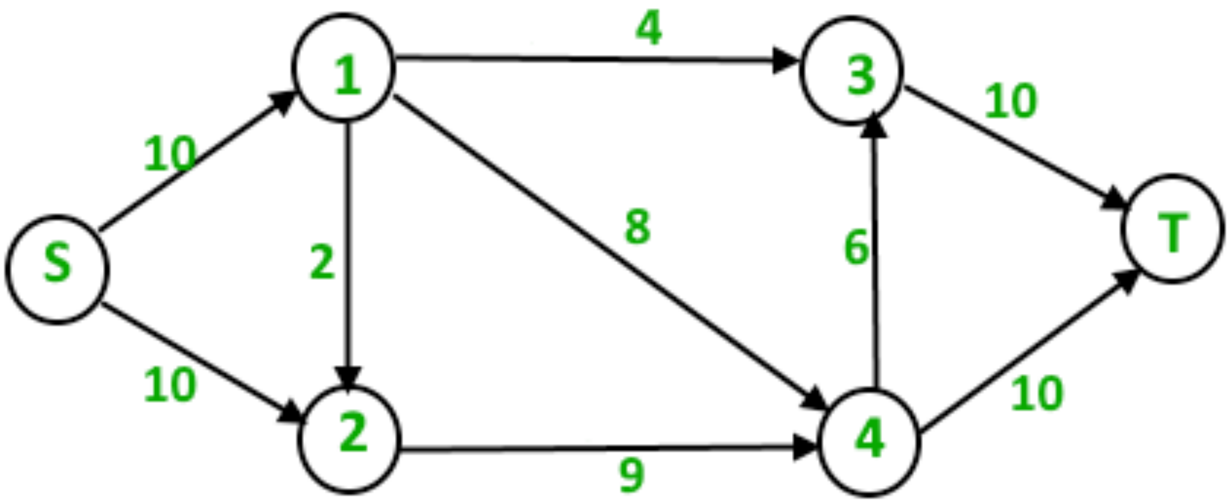
- A Flow Network is defined as a directed graph
  - **Source** Node
  - **Sink** Node and several other nodes connected with edges

- Each edge has an individual capacity
  - The maximum limit of flow that edge could allow

Network Flow Algorithms

An Example

- **S = Source** Node
- **T = Sink** Node
- Several other nodes connected with edges



Network Flow Algorithms

Maximum Flow Problem

- Given a network that shows the potential capacity over which goods can be shipped between two locations, compute the maximum flow supported by the network

Popular Algorithms:

- **Ford – Fulkerson Algorithm**
  - We focus on this algorithm with an example
- Dinic's algorithm

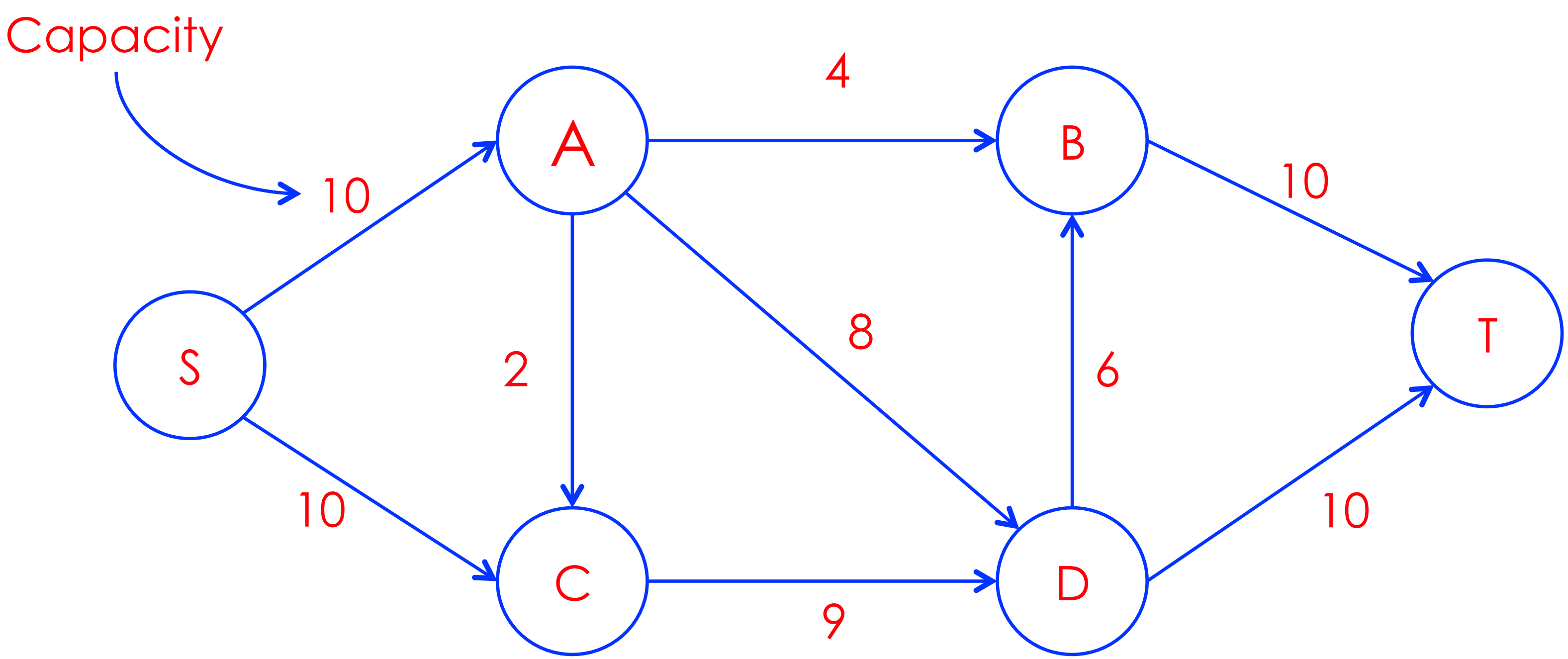
Network Flow - Examples

Maximum Flow Problem

- A type of network optimization problem
- Max Flow arises in many different contexts
  - **Networks:** routing as many packets as possible on a given network
  - **Transportation:** sending as many trucks as possible, where roads have limits on the number of trucks per unit time

Ford–Fulkerson Algo: Example

- How do we find the maximum possible flow from source s to sink t with given constraints?



Ford-Fulkerson Algorithm

Problem:

- Given a graph which represents a flow network where every edge has a capacity
- Given two vertices:
  - Source **s** and Sink **t** in the graph

Goal:

Find out the maximum possible flow from s to t with the following constraints:

a) Flow on an edge does not exceed the given capacity
b) In-flow is equal to out-flow for every vertex except s and t

Terminologies

Residual Graph:

- It is a graph, which indicates additional possible flow

- If there is such path from source s to sink t then there is a possibility to add flow

Residual Capacity:

= Original edge capacity – current flow

Terminologies (contd.)

Minimal Cut:

- Known as Bottleneck Capacity
- This decides the maximum possible flow from source s to sink t through an augmenting path

Augmenting Path:

Augmenting paths can be done in two ways:

a) Non-full forward edges

b) Non-empty backward edges

Ford-Fulkerson Algorithm

Basic Idea:

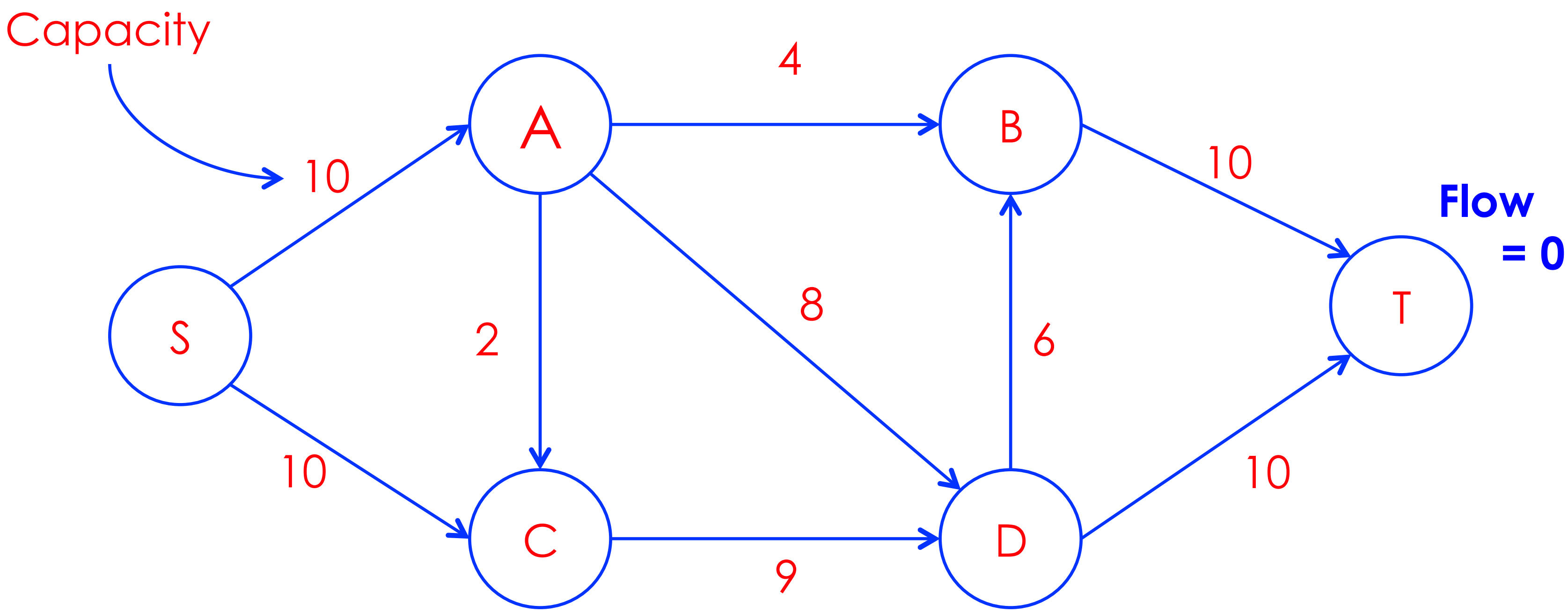
1) Start the algorithm with an initial flow as 0

2) While there is an augmenting path from source to sink then add this path-flow to flow

3) Return flow

Ford-Fulkerson – Illustration

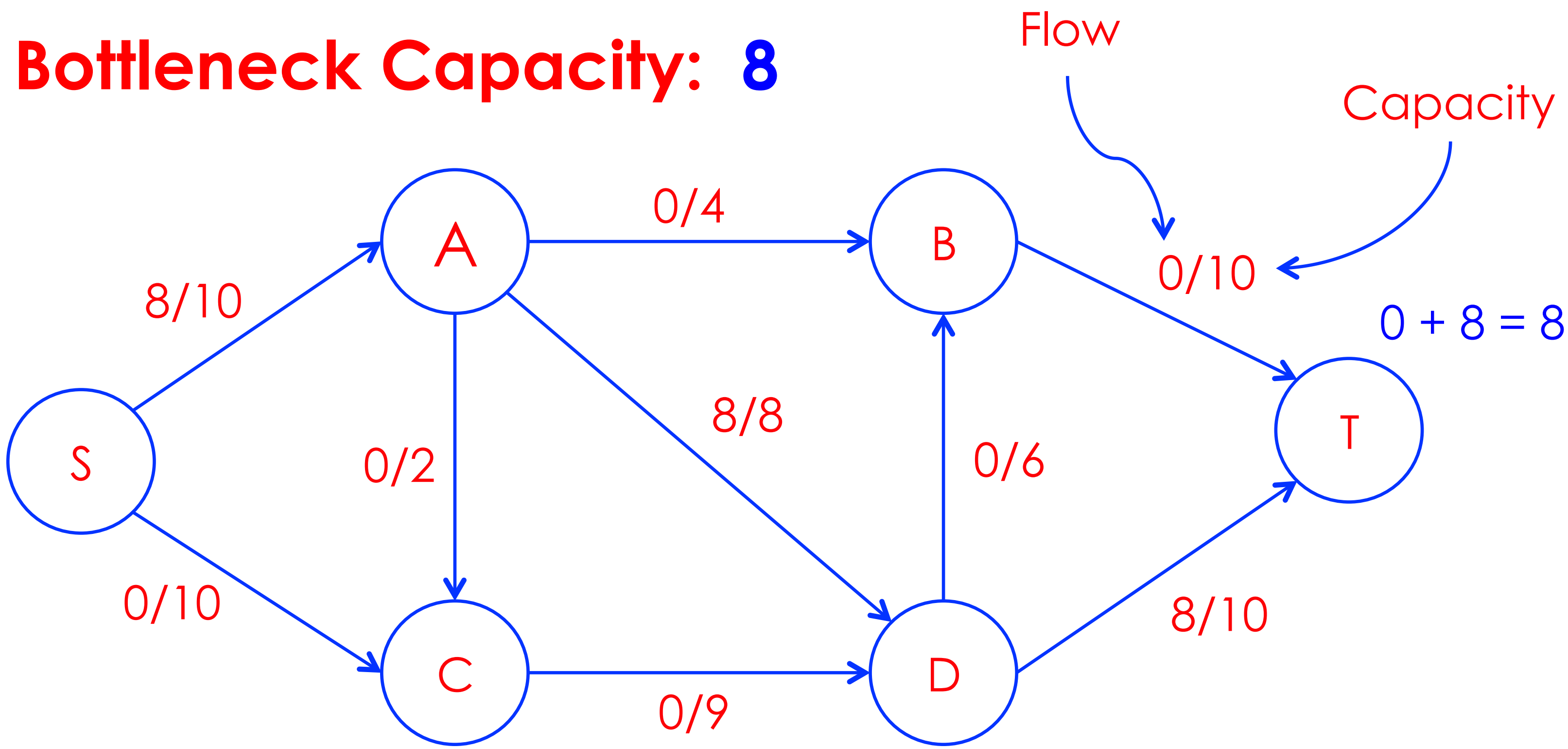
- **initial Graph:**



- **Maximum Flow = 0**

Ford-Fulkerson – Illustration

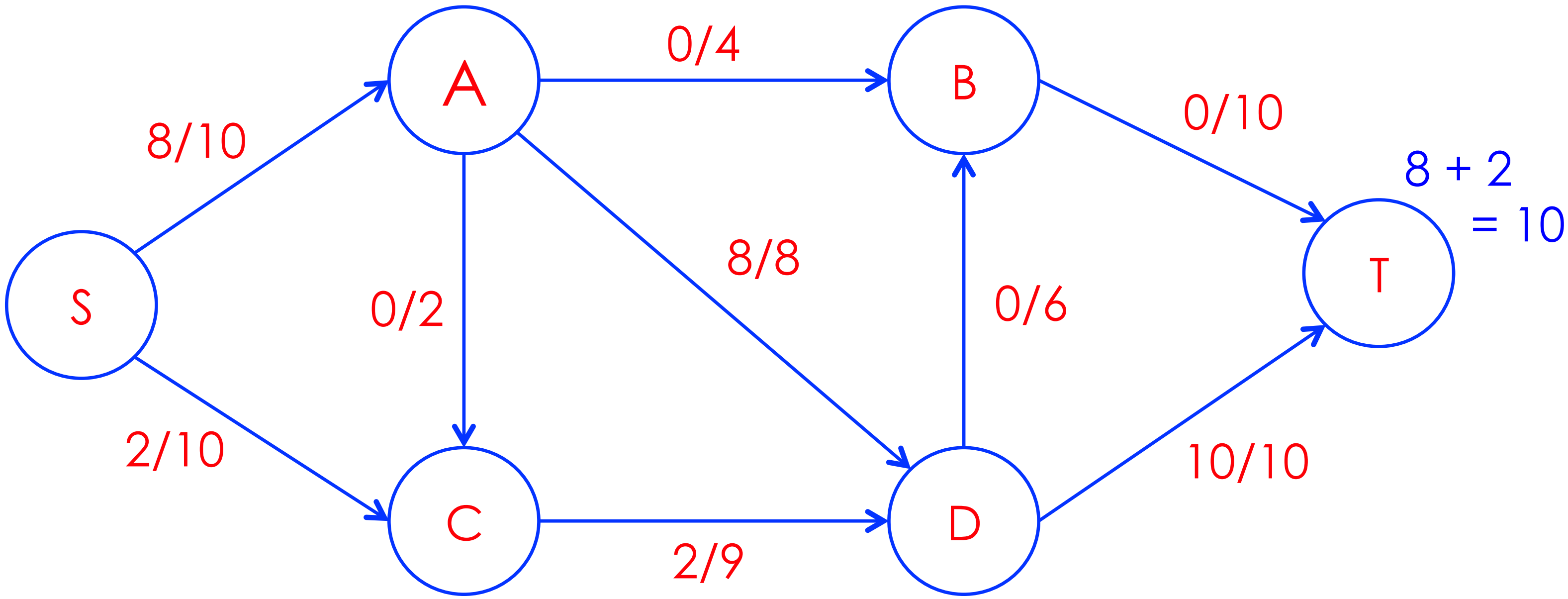
- **Path – 1: S → A → D → T**
- **Bottleneck Capacity: 8**



- **Maximum Flow: 0 + 8 = 8**

Ford-Fulkerson – Illustration

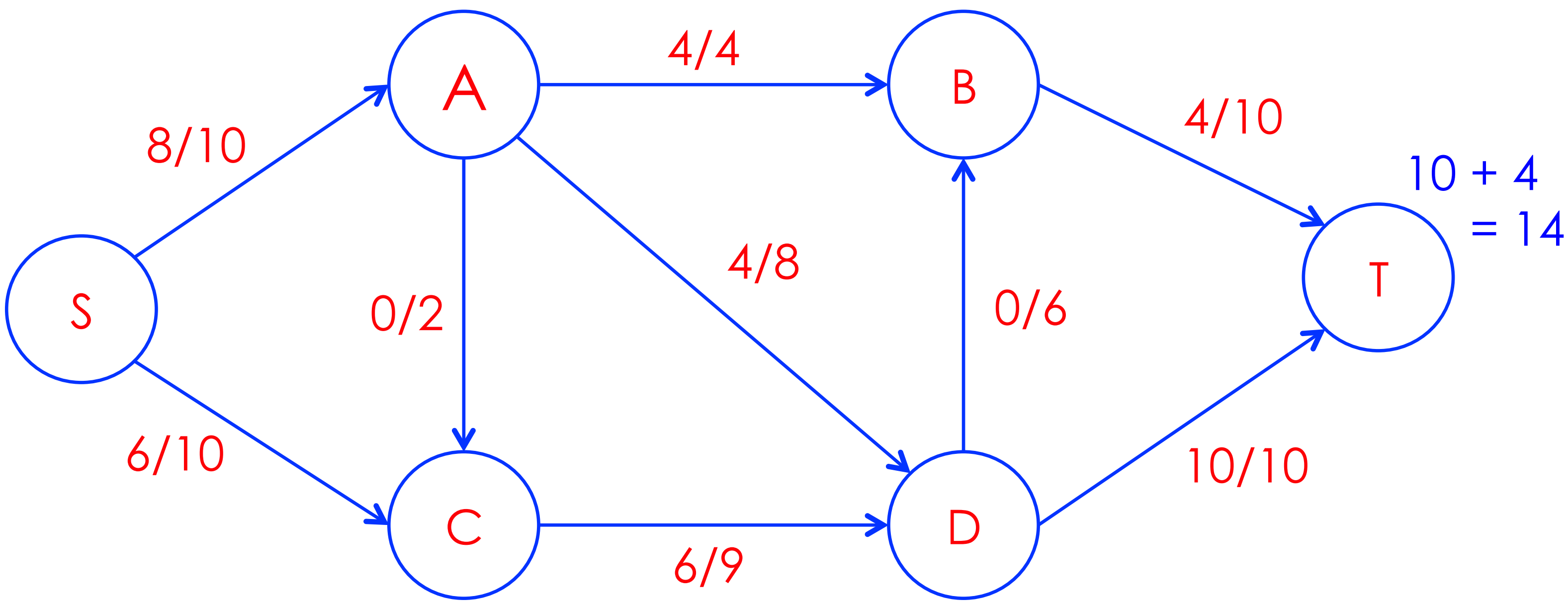
- **Path – 2: S → C → D → T**
- **Bottleneck Capacity: 2**



- **Maximum Flow: 0 + 8 = 8**

Ford-Fulkerson – Illustration

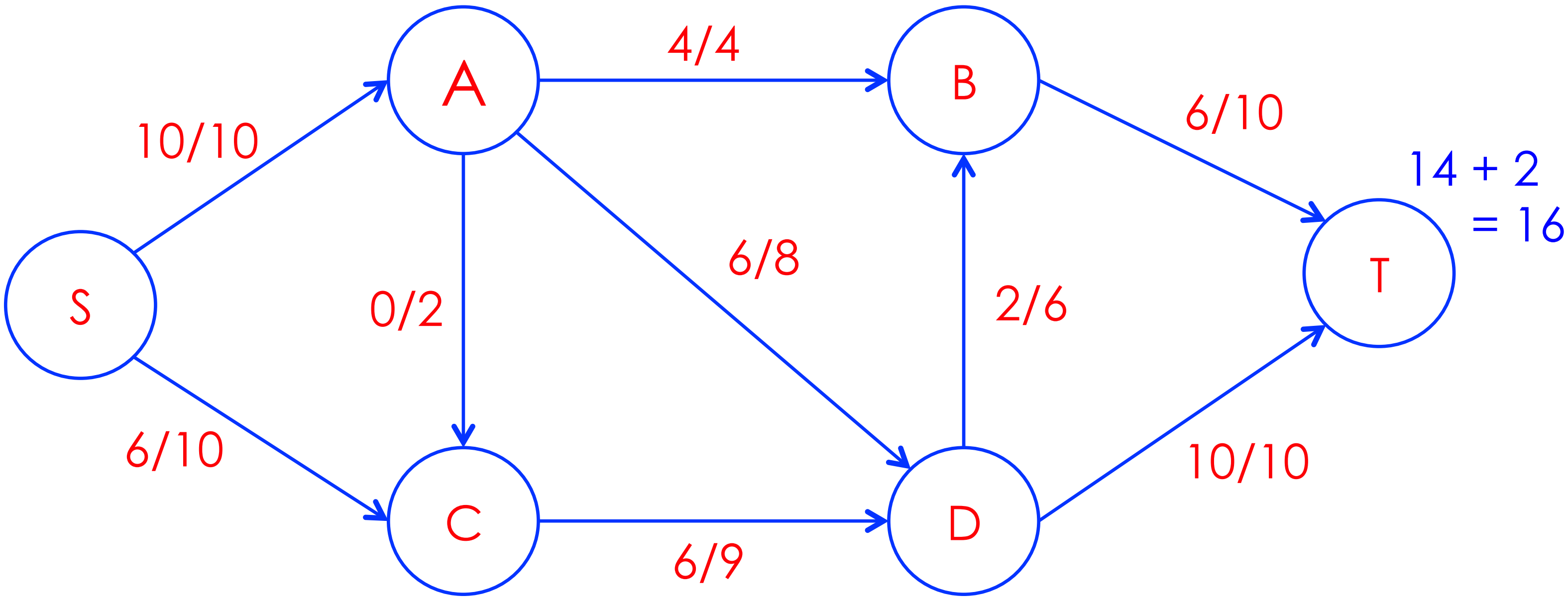
- **Path – 3: S → C → D → A → B → T**
- **Bottleneck Capacity: 4**



- **Maximum Flow: 10 + 4 = 14**

Ford-Fulkerson – Illustration

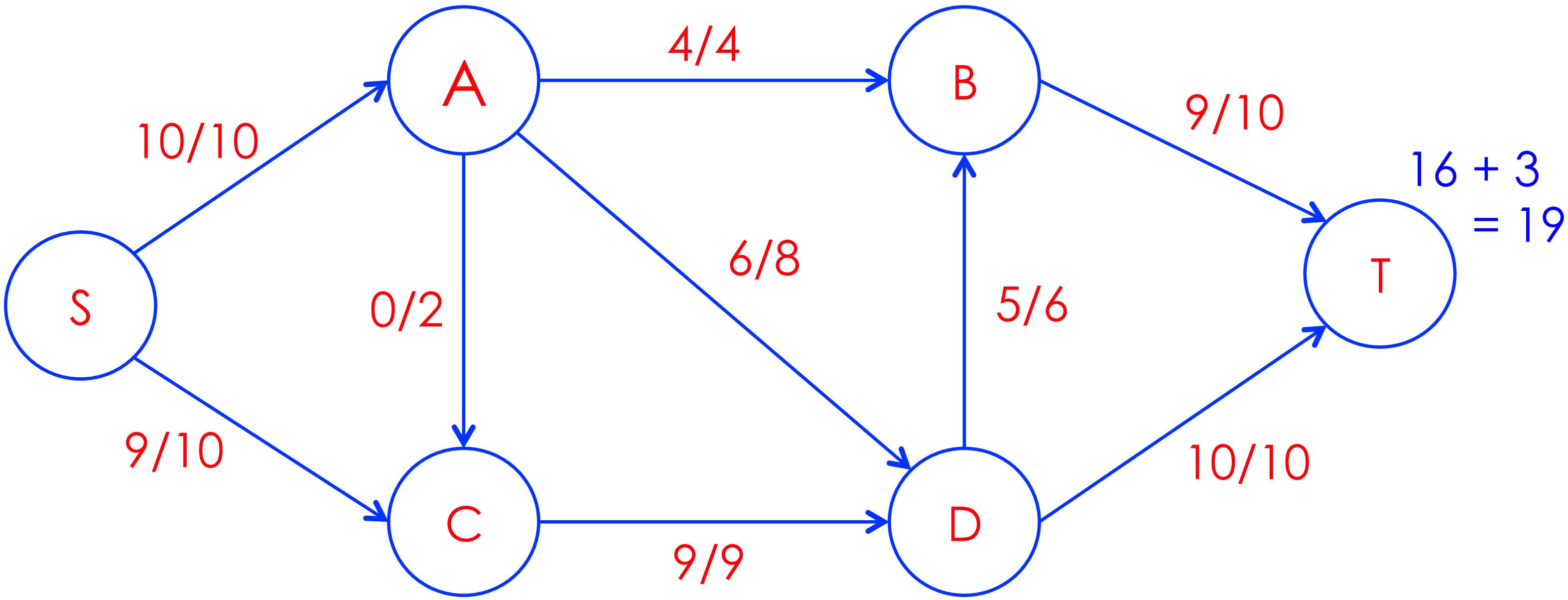
- **Path – 4: S → A → D → B → T**
- **Bottleneck Capacity: 2**



- **Maximum Flow: 14 + 2 = 16**

Ford-Fulkerson – Illustration

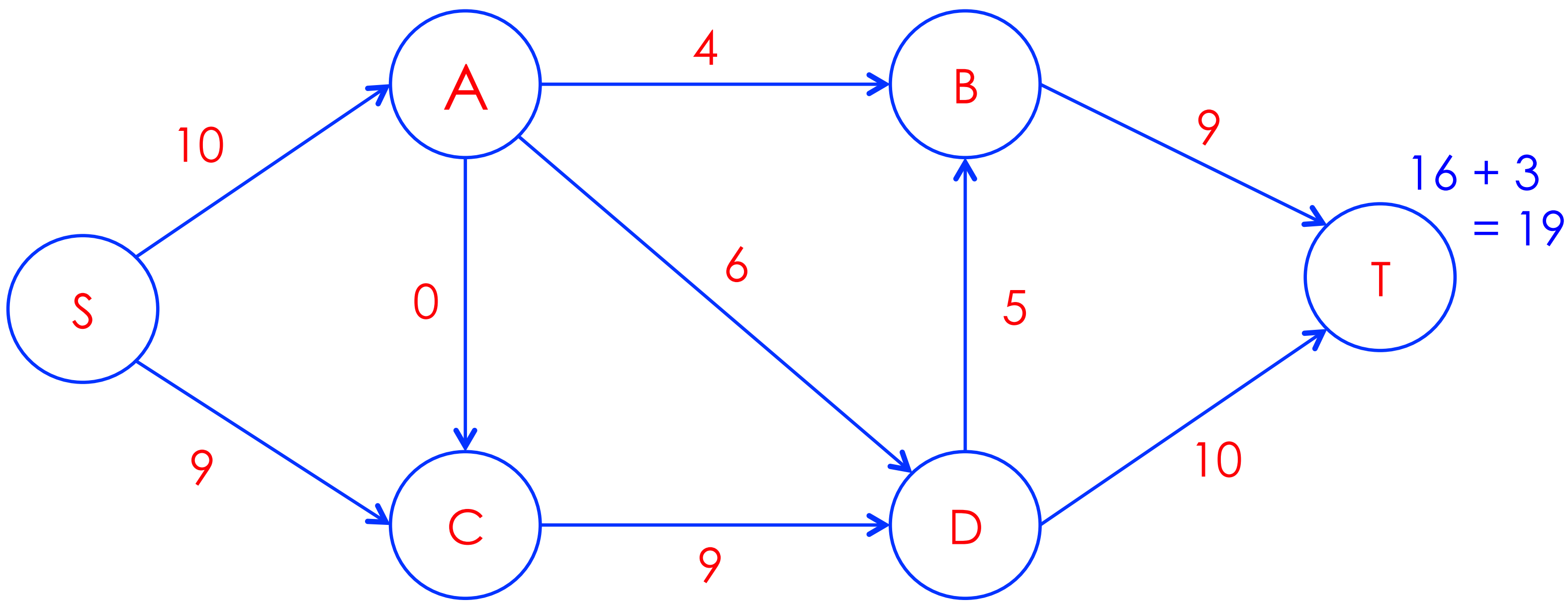
- **Path – 5: S → C → D → B → T**
- **Bottleneck Capacity: 3**



- **Maximum Flow: 16 + 3 = 19**

Ford-Fulkerson – Illustration

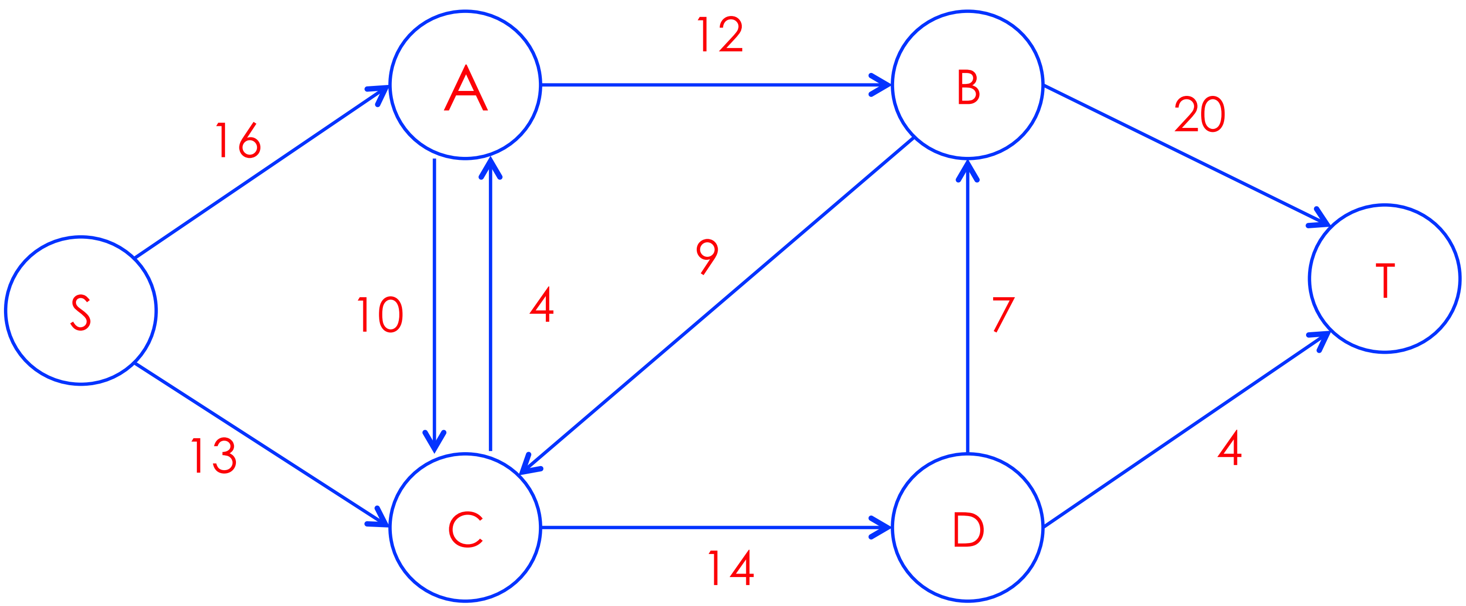
- **Final Max Flow Network**



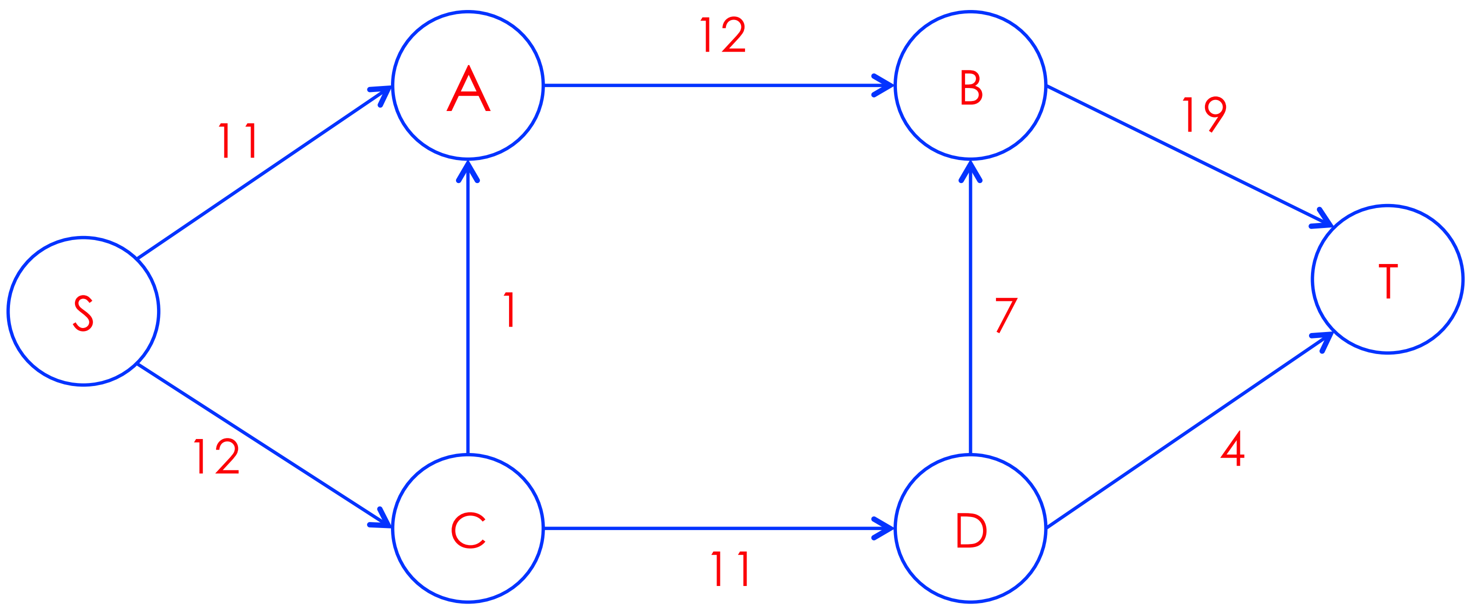
- **Maximum Flow = 19**

Ford-Fulkerson – Exercise

- Compute max flow of the following network:



Ford-Fulkerson – Exercise



- **Maximum Possible Flow = 23**

Ford-Fulkerson – Complexity

Computational Complexity:

O(f) $\left\{ \begin{array}{l} \text{1) Find an Augmenting path} \rightarrow O(E) \\ \text{2) Compute the bottleneck capacity} \\ \text{3) Augment each edge and the total flow} \end{array} \right.$

Time Complexity

= O(E f)

where E = # of edges

f = maximum flow

Next: Minimum Cost Flow

Overview:

- Supply Nodes (S) produce units shipped over a network of distribution nodes to be consumed at demand nodes (T)
- Each edge has the following:
  - (low, high) capacity
  - An Actual Flow
  - Associated cost per unit flowing over the edge

Goal:

  - Meet all demands and
  - Minimize the total cost of all edges

Help among Yourselves?

- **Perspective Students** (having CGPA above 8.5 and above)
- **Promising Students** (having CGPA above 6.5 and less than 8.5)

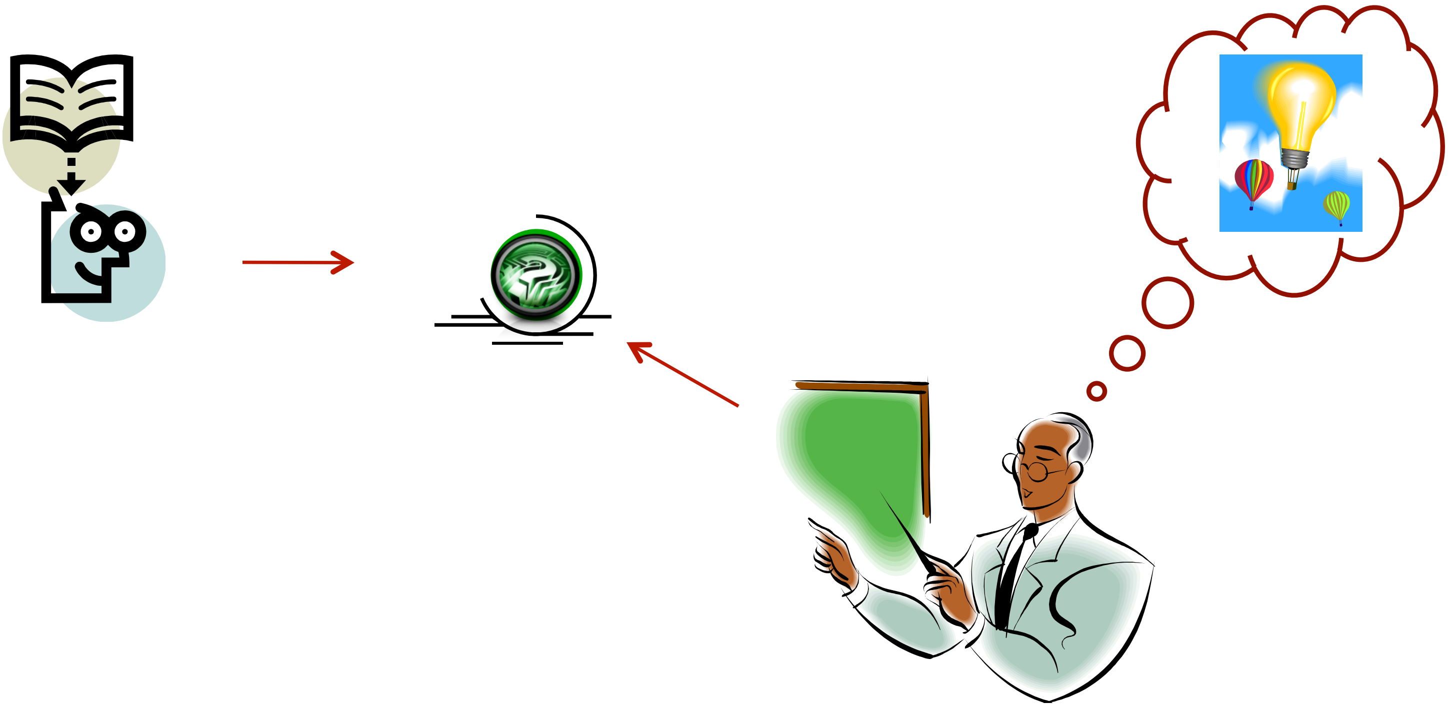
- **Needy Students** (having CGPA less than 6.5)
  - Can the above group help these students? (Your work will also be rewarded)

- You may grow a culture of **collaborative learning** by helping the needy students

Assistance

- You may post your questions to me at any time
- You may meet me in person on available time or with an appointment
- TA s would assist you to clear your doubts.
- You may leave me an email any time (email is the best way to reach me faster)

Thanks …



… Questions ???