جامعة الانبار

كلية العلوم

قسم الرياضيات التطبيقية

نظرية البيانات / الفصل الاول

Directed Graphs (II)

م. د. امين شامان امين

Lecture (9)

Directed Graphs

Dr. Ameen Sh. Ameen

Dept. of Applied Mathematics.

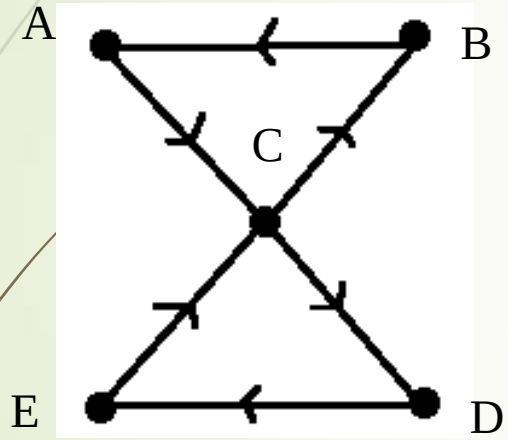
College of Science \ University of Anbar.

19 February 2021

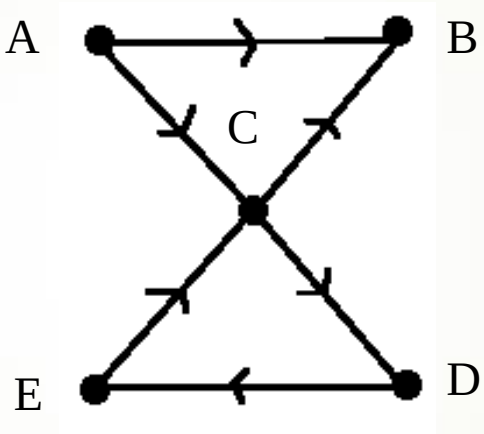
Outlines:

1) Eulerian Digraphs.
2) Hamiltonian Digraphs.
3) Flow Networks.

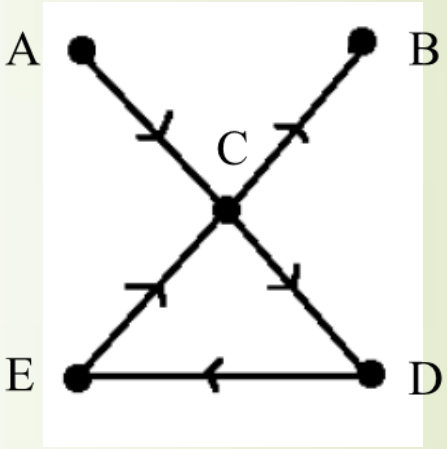
Eulerian Digraph: A digraph D is said to be Eulerian if it contains an Eulerian circuit (closed trial) which traverses every edge of D exactly once (such a trail is called Eulerian trail). A digraph D is said to be unicursal if it contains an open Eulerian trial .



Eulerian Digraph
Eulerian Circuit: ACDECBA

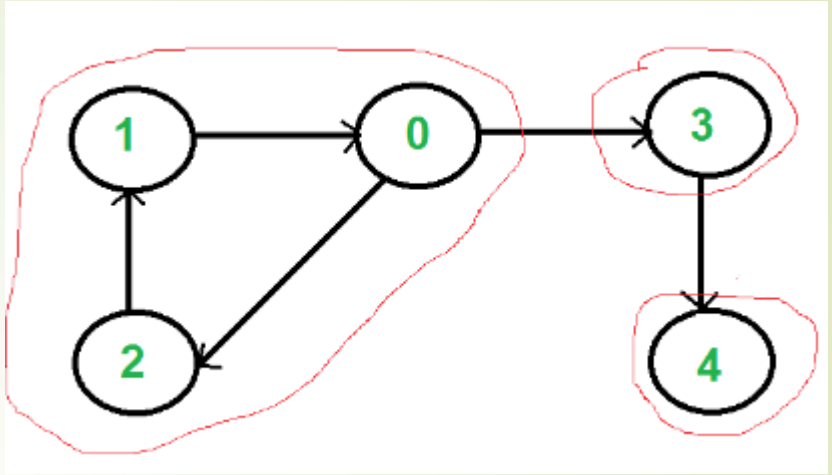


Not Eulerian Digraph

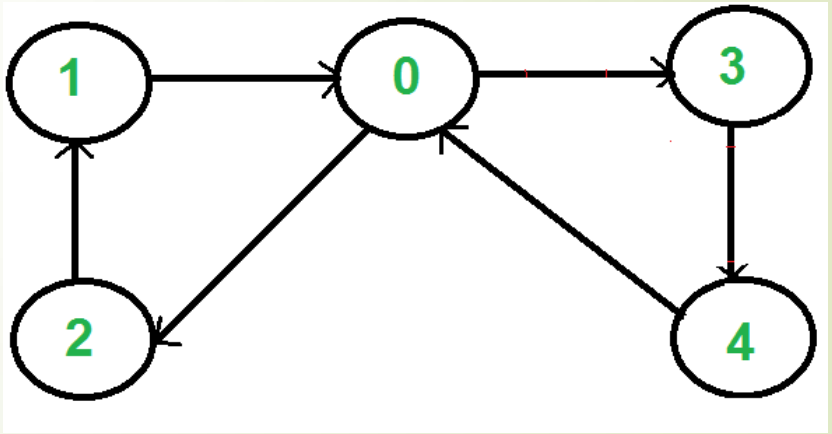


Unicursal Digraph
Eulerian Trial: ACDECB

A directed graph is strongly connected if there is a path between all pairs of vertices. A strongly connected component (**SCC**) of a directed graph is a maximal strongly connected subgraph. For example, there are 3 SCCs in the following graph.

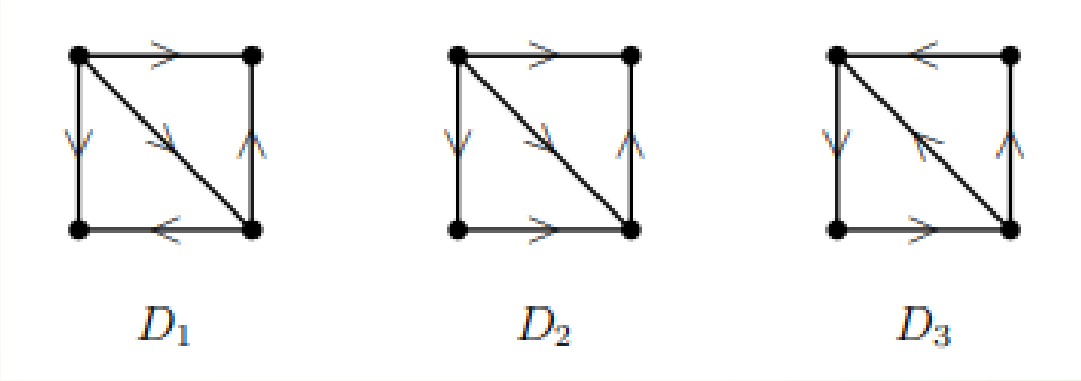


How to check if a directed graph is Eulerian?
A directed graph has an Eulerian circuit if following conditions are true
1) All vertices with nonzero degree belong to a single strongly connected component.
2) In degree is equal to the out degree for every vertex.



Definitions:

- ✓ A directed path in a digraph D is called a Hamiltonian directed path if it contains all the vertices of D.
- ✓ A directed cycle in D is called a Hamiltonian directed cycle if contains all the vertices of D.
- ✓ A directed graph D is called a Hamiltonian digraph if it contains a Hamiltonian directed cycle.



D_1 contains no Hamiltonian directed path. D_2 contains a Hamiltonian directed path but contains no Hamiltonian directed cycle. D_3 contains a Hamiltonian directed cycle and hence it is a Hamiltonian directed graph.

The following results give a sufficient conditions for a strongly connected digraph on n vertices to be Hamiltonian.

Theorem (Ghouila-Houri): If $d(u) \geq n$ for all vertices $u \in V(D)$, then D is Hamiltonian.

Theorem (Woodall): If $d^+(u) + d^-(v) \geq n$ for all pairs of non-adjacent vertices in D, then D is Hamiltonian.

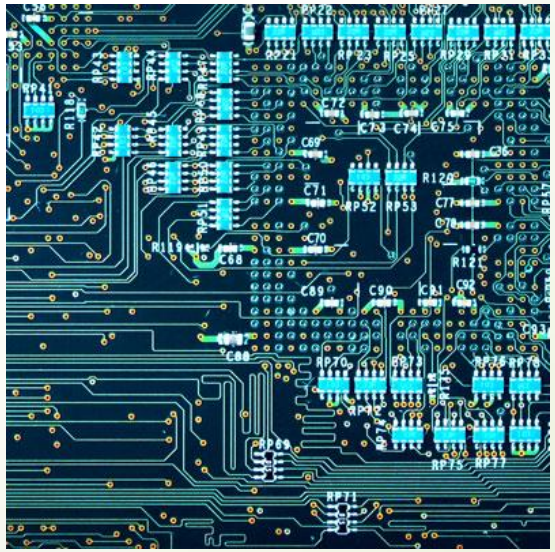
Theorem (Meyniel): If $d(u) + d(v) \geq 2n - 1$ for all pairs of non-adjacent vertices in D then, D is Hamiltonian.

Networks

Introduction:

Every day we are surrounded by countless connections and networks: roads and rail tracks, phone lines, the internet, electronic circuits and even molecular bonds. There are even social networks between friends and families.

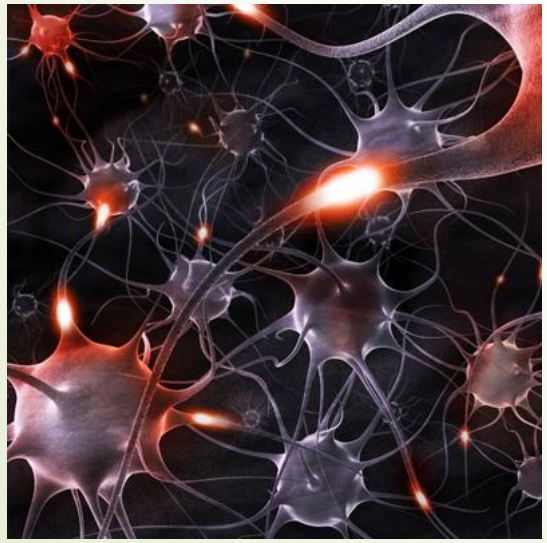
Road and Rail Networks



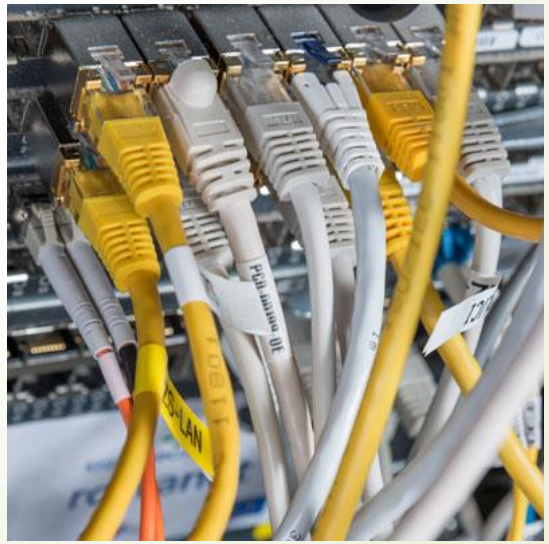
Computer Chips

Supply Chains

Friendships



Neural Connections

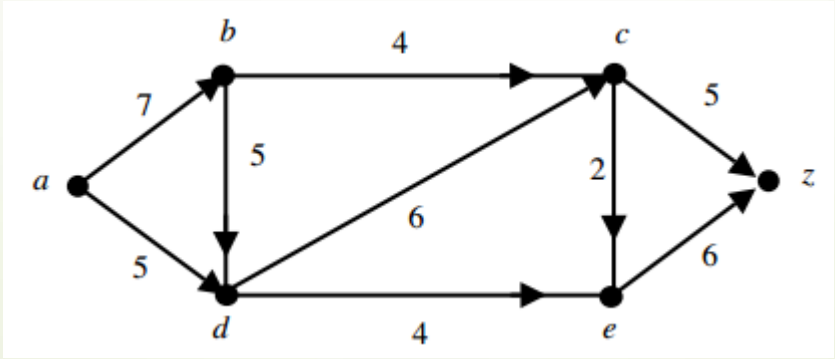


The Internet

What is the definition of a network in graph theory?

A **network** (flow network) is a directed graph $D(V, E)$ where each edge has a capacity $c(e)$ (the non-negative real number) and each edge receives a flow $f(e)$ for any edge e $\in E$. The amount of flow on an edge cannot exceed the capacity of the edge (i.e. $0 \leq f(e) \leq c(e)$). A flow must satisfy the restriction that the amount of flow into a node equals the amount of flow out of it, except when it is a source, which has more outgoing flow, or sink, which has more incoming flow.

i. e. A **network** (flow network) is a weighted directed graph.



Definition: In-flow and Out-flow in Networks. The inflow to a vertex v, denoted by $f_{in}(v)$, is defined by $f_{in}(v) = \sum_{uv \in E} f(uv)$ and the out-flow from v, denoted by $f_{out}(v)$, is defined by $f_{out}(v) = \sum_{vw \in E} f(vw)$. Note that $f_{in}(v) = f_{out}(v)$ for $v \in V(D)$.

The vertex set of a network can be partitioned into three types:

i) **Source:** A vertex with in-degree 0 is called a source vertex and is denoted by s and the set of all source vertices is called the source, and is denoted by S.

ii) **Sink:** A vertex with out-degree 0 is called a sink vertex and is denoted by t and the set of all sink vertices is called the sink, and is denoted by $\bar{S}$.

iii) A vertex that is neither a source nor a sink is called an **intermediate** vertex and the set of all the intermediate denoted by I.

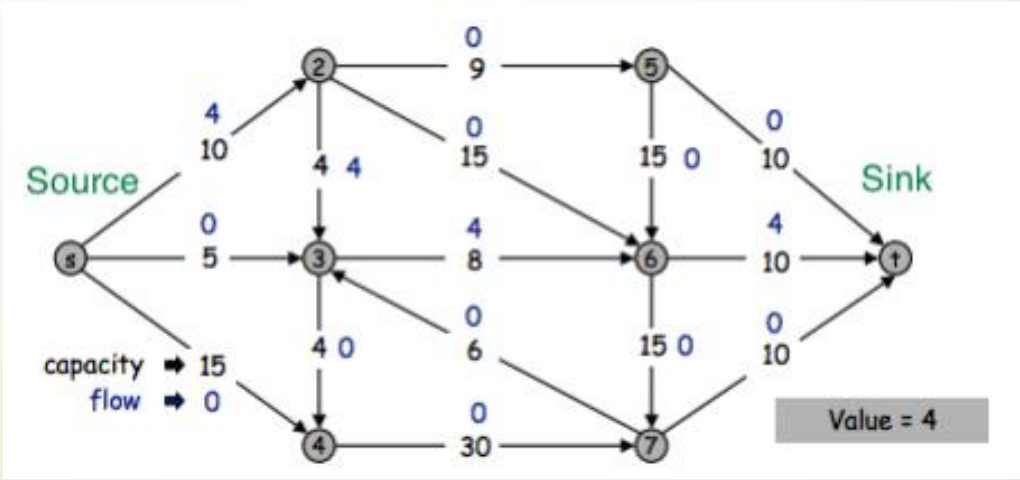
Note the following points:

i) The flow over any arc (edge) can be no more than the capacity of that arc .

ii) The in-flow on any intermediate vertex is equal to the out-flow of that vertex. That is, the flow is not obstructed or reduced at any intermediate vertices.

The value of a flow is the sum of the flow on all edges leaving the source s. We later show that this is equivalent to the sum of all the flow going into the sink t. The value of a flow represents how much we can transport from the source to the sink.

Definition: The value of a flow f is defined as $val\ f = \sum_{e\ out\ of\ s} f(e)$.



Ford-Fulkerson Algorithm for Maximum Flow Problem

Given a graph which represents a flow network where every edge has a capacity. Also given two vertices source s and sink t in the graph, find the maximum possible flow from s to t with following constraints:

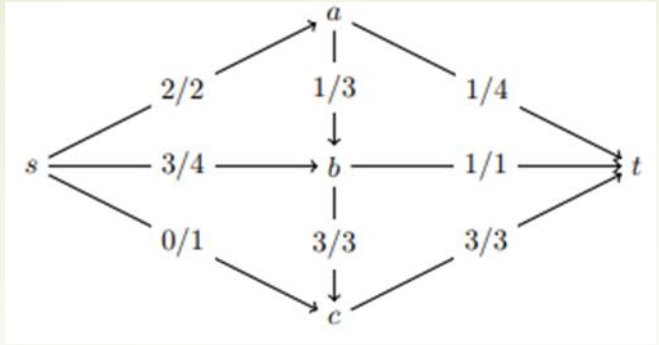
a) Flow on an edge doesn't exceed the given capacity of the edge.

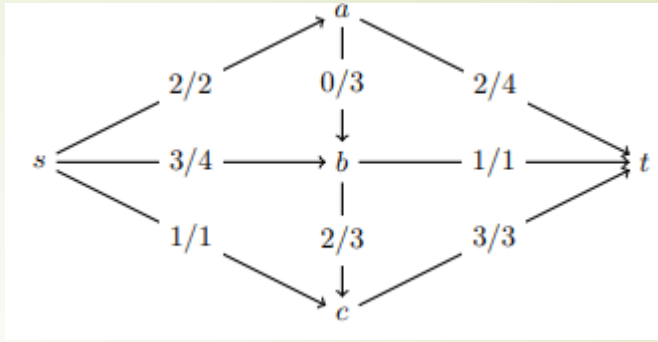
b) Incoming flow is equal to outgoing flow for every vertex except s and t.

Residual Graph of a flow network is a graph which indicates additional possible flow. If there is a path from source to sink in residual graph, then it is possible to add flow.

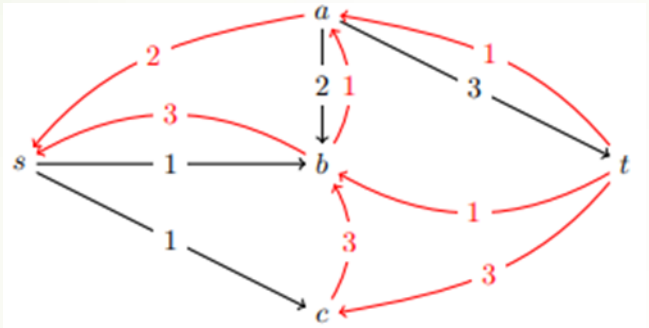
Residual Capacity which is equal to original capacity of the edge minus current flow. Residual capacity is basically the current capacity of the edge.

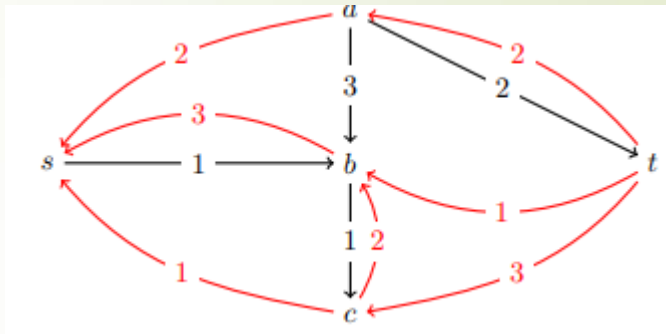
Original Graph





Residual Graph





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.

Minimal Cut: Which decides maximum possible flow from source to sink through an augmenting path.

Maximum Flow: (Max. Flow)

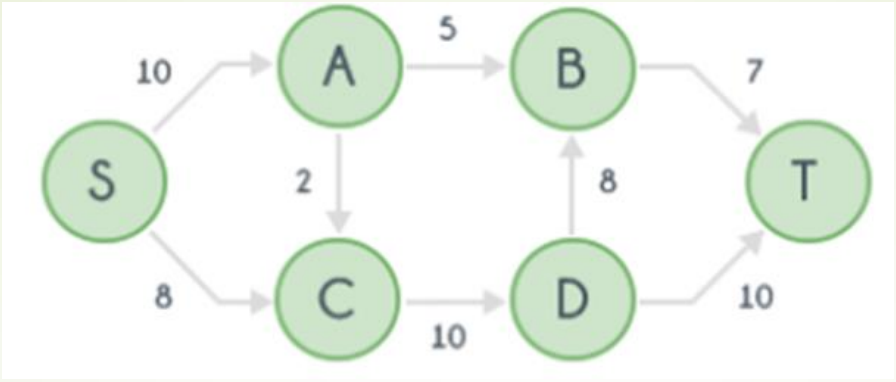
It is defined as the maximum amount of flow that the network would allow to flow from source to sink. Many algorithms exist in solving the maximum flow problem. The major algorithm to solve these kind of problems are Ford-Fulkerson algorithm is explained below.

The following is a simple idea of the algorithm:

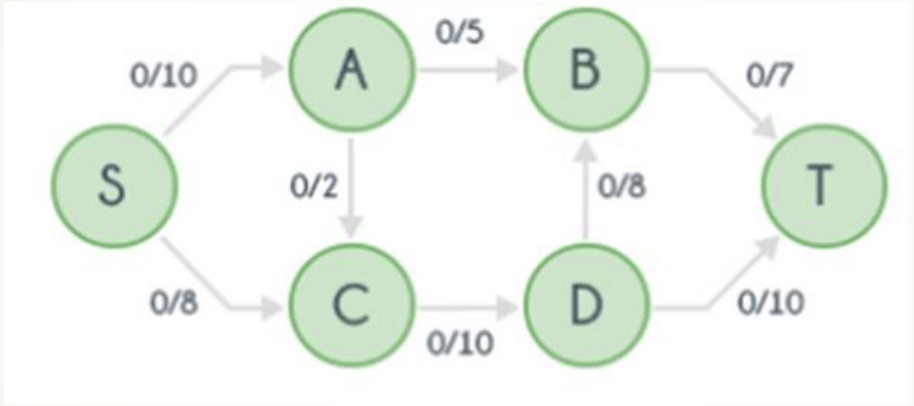
1) Start with a initial flow as 0.
2) While there is an augmenting path from source to sink add this path flow to flow.
3) Return flow

Now, we solve the following examples by using theorem above.

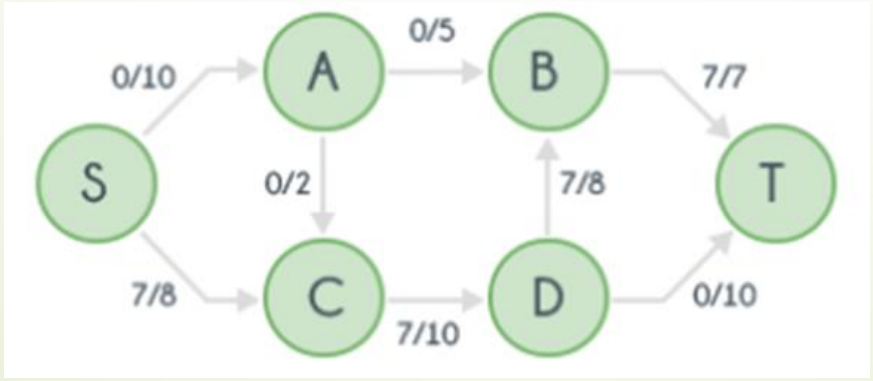
Example 1:



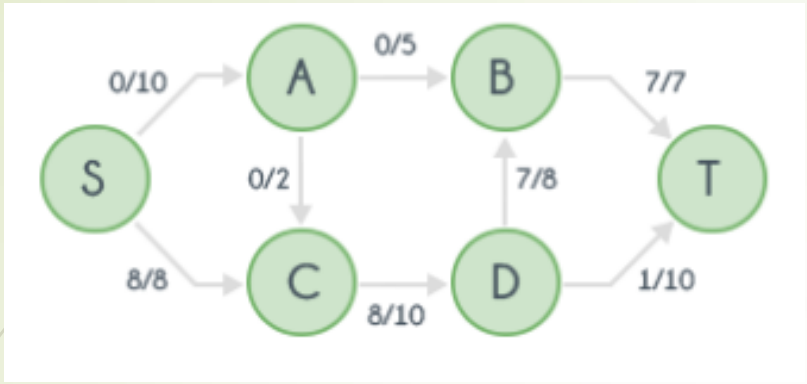
Flow/Capacity



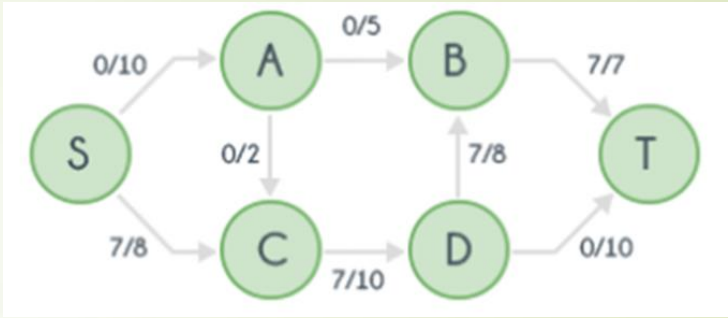
Flow = 0.

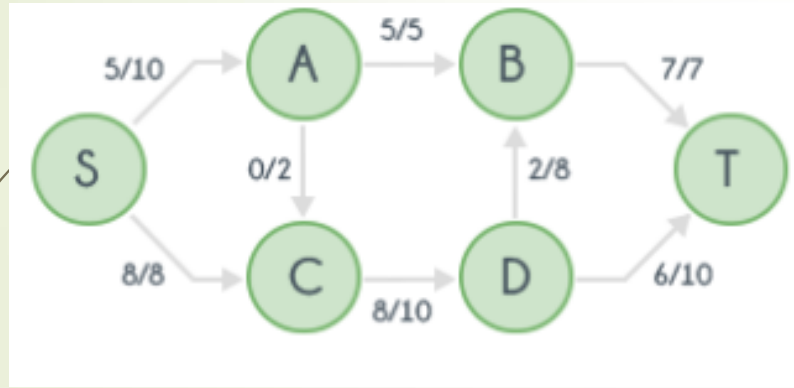


Path 1: $S - C - D - B - T$
Flow = 7.

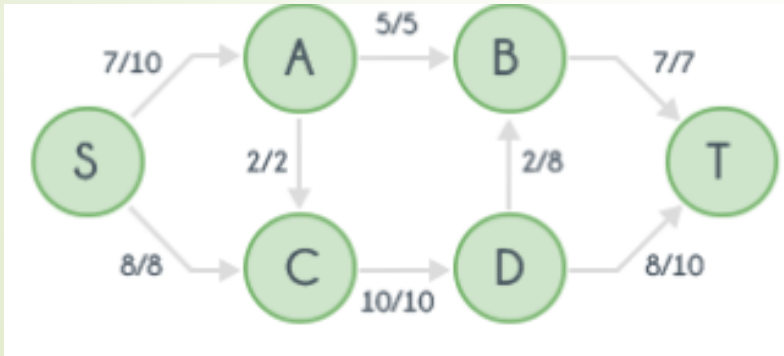


Path 2: $S - C - D - T$
Flow $= 1$.





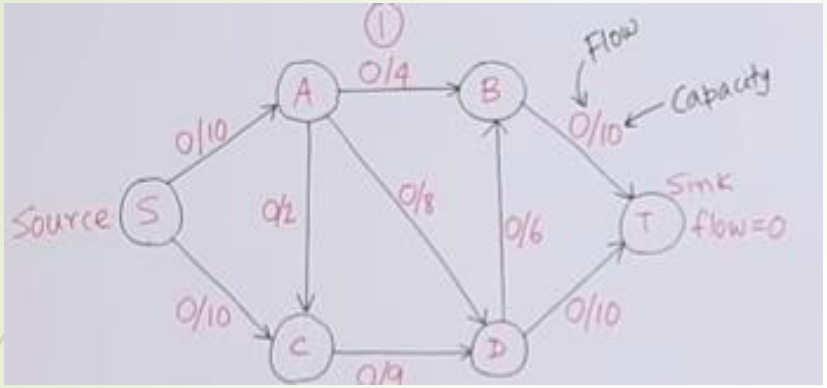
Path 3: $S - A - B - D - T$
Flow $= 5$.



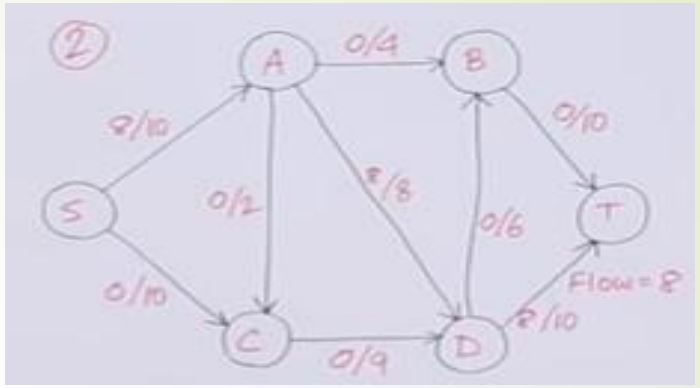
Path 4: $S - A - C - D - T$
Flow $= 2$.

No more paths left
Max Flow$= \ 7 + 1 + 5 + 2 = 15$

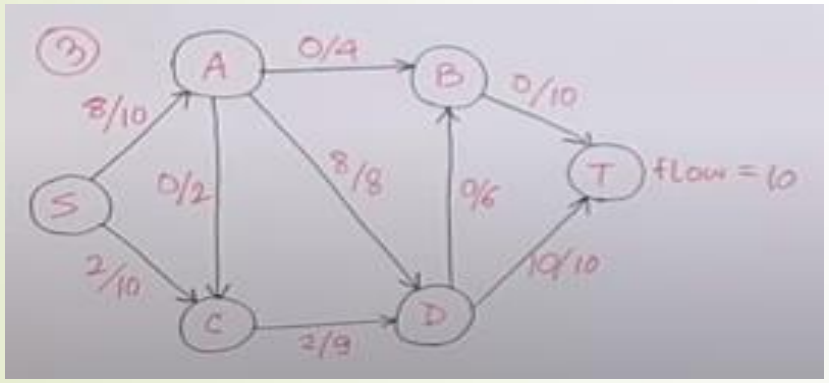
Example 2:



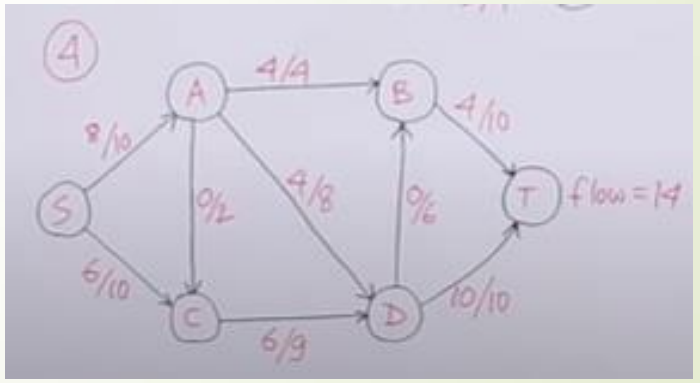
Flow = 0.



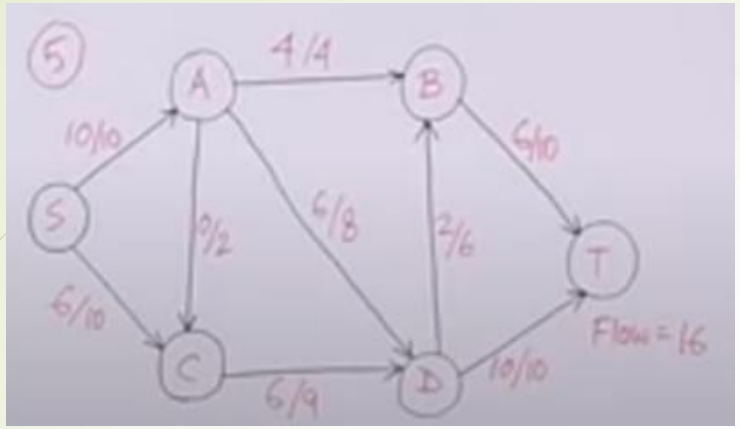
Path 1: $S - A - D - T$
Flow = 8.

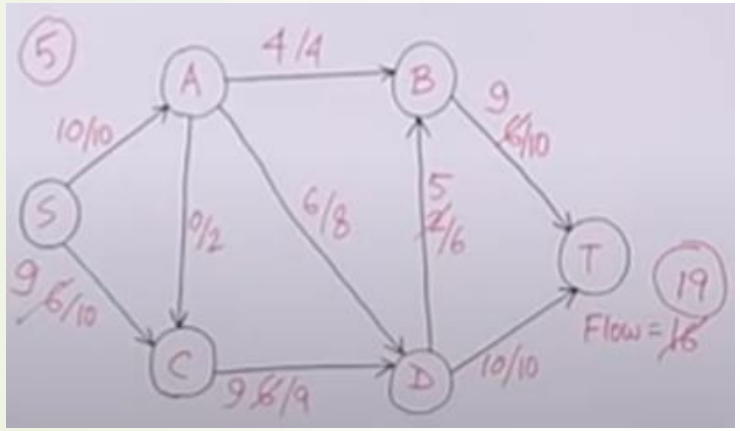


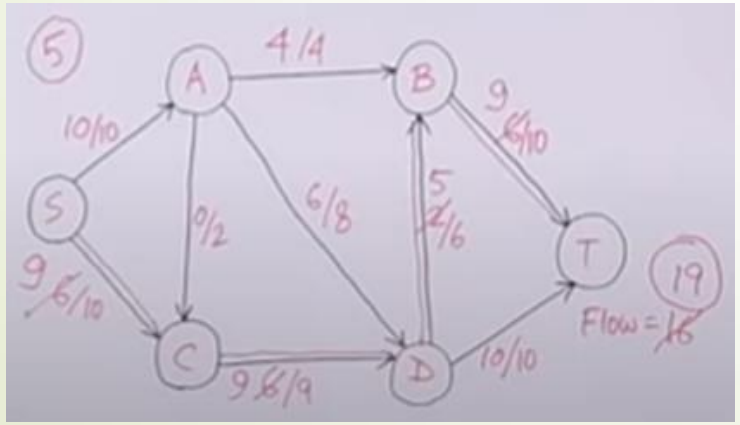
Path 2: $S - C - D - T$
Flow = 2.



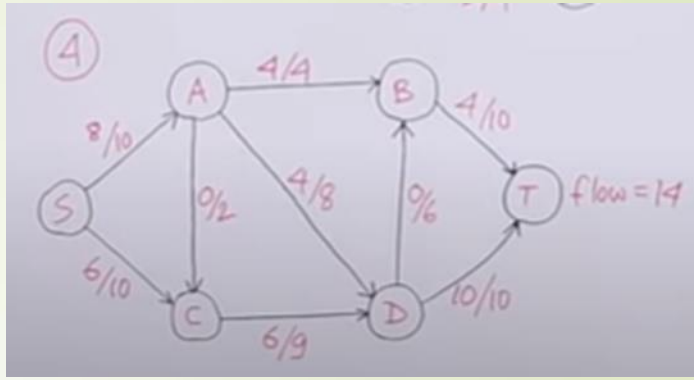
Path 3: $S - C - D - A - B - T$
Flow = 4.







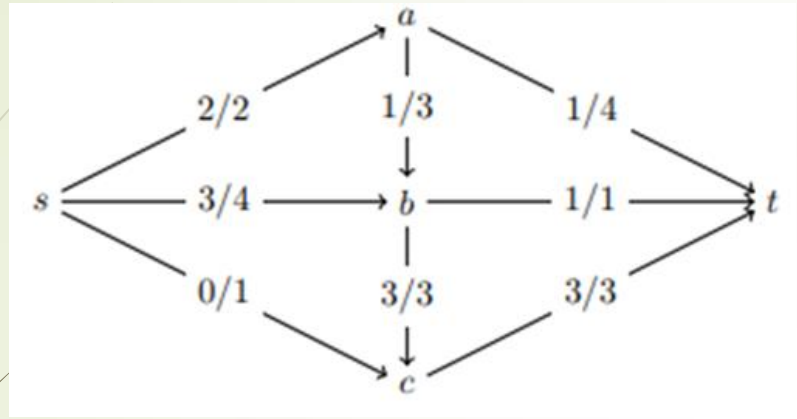
Path 4: $S - A - D - B - T$
Flow $= 2$.

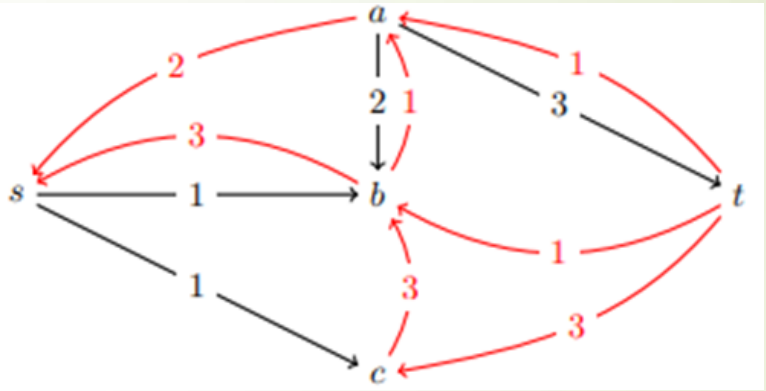


Path 5: $S - C - D - B - T$
Flow $= 3$.

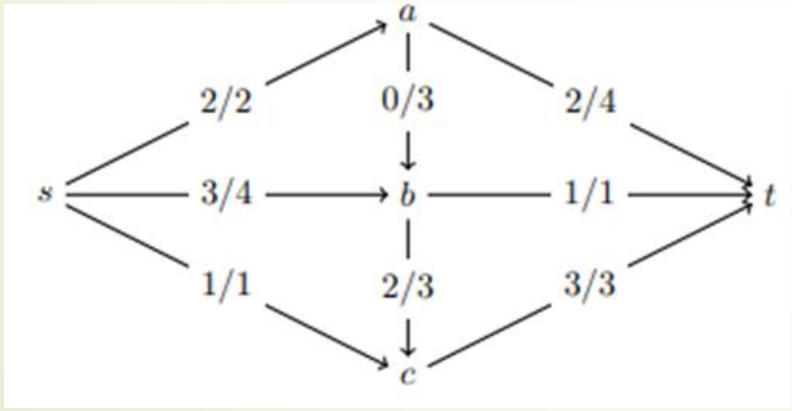
No more paths left
Max Flow$= 8 + 2 + 4 + 2 + 3 = 19$

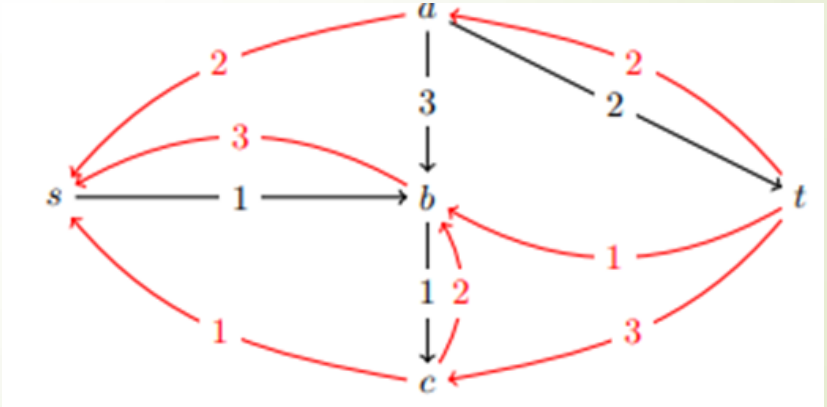
Example 3:





Max Flow= 1 + 1 + 3 = 5





No more paths left

Max Flow= 2 + 1 + 3 = 6

Thank You

References:

1. S. Arumugam and S. Ramachandran, (2015), Invitation to graph theory, Scitech Publ., Kolkata, India.
2. G. S. Singh, (2013). Graph theory, Prentice Hall of India, New Delhi.
3. R. Balakrishnan and K. Ranganathan, (2012). A textbook of graph theory, Springer, New York.
4. J.A. Bondy and U.S.R Murty, (2008). Graph theory, Springer.
5. G. Agnarsson and R. Greenlaw, (2007). Graph theory: Modeling, applications & algorithms, Pearson Education, New Delhi.
6. G. Chartrand and P. Zhang, (2005). Introduction to graph theory, McGraw-Hill Inc.
7. G. Sethuraman, R. Balakrishnan, and R.J. Wilson, (2004). Graph theory and its applications, Narosa Pub. House, New Delhi.
8. D.B. West, (2001). Introduction to graph theory, Pearson Education Inc., Delhi.
9. V.K. Balakrishnan, (1997). Graph theory, McGrawhill, New York.
10. G. Chartrand and L. Lesniak, (1996). Graphs and digraphs, CRC Press.
11. J.A. Bondy and U.S.R Murty, (1976). Graph theory with applications, North-Holland, New York.