CS101

PROJECT

FORD-FULKERSON ALGORITHM FOR MAXIMUM FLOW PROBLEM

INTRODUCTION

The Ford-Fulkerson algorithm (FFA), sometimes known as the Ford-Fulkerson method, computes the maximum flow in a flow network. Due to the lack of complete specification of the method for locating augmenting pathways in a residual graph, it is frequently referred to as a "method" rather than a "algorithm".

It was released in 1956 by D. R. Fulkerson and L. R. Ford Jr.

The method works as follows: we transmit flow via one of the paths as long as there is a path from the source (start node) to the sink (end node), with available capacity on all edges in the path. We then choose another route, and so forth. A path that has additional capacity is referred to as augmenting path.

MAXIMUM FLOW PROBLEM

A weighted directed graph is presented to us as a representation of a flow network, where each edge has a specific capacity. The graph has two specific vertices, source "s" which is the start node and sink "t", the end node.

The maximum flow from s to t must be determined with the ensuing restrictions:

- Flow on an edge does not exceed the edge's specified capacity.
- For all but s and t, the inflow and outflow are equal.
- Total flow into s and total flow out t are equal.

Flow network: It is defined as a weighted directed graph involving a source, a sink and several other nodes that connected with edges having a certain capacity which indicates the maximum flow possible through that edge.

Maximum flow: It is defined as the maximum amount of flow that can go into sink in the network.

IMPORTANT TERMINOLOGIES

- Residual Graph: It's a graph giving additional flow possible. If there is such path from source to sink then there is possibility to add flow.
- Residual capacity: For each edge the original capacity minus flow gives us the residual capacity for that edge.
- Minimal cut: Also referred to as bottleneck capacity, it determines the greatest flow from source to sink through an augmented path.
- Augmenting path: A path can be augmenting in two ways-
  i. It has non-full forward edges
  ii. It has non- empty backward edges

THE ALGORITHM

A SIMPLE IDEA OF THE FORD-FULKERSON ALGORITHM

INPUT: A flow network G(V, E) with a source s, sink t, and a flow capacity for every edge in the given network.

STEPS:

- Start with initial flow as zero.
- While there is an augmenting path from source to sink add path flow to the flow value.
- Return flow.

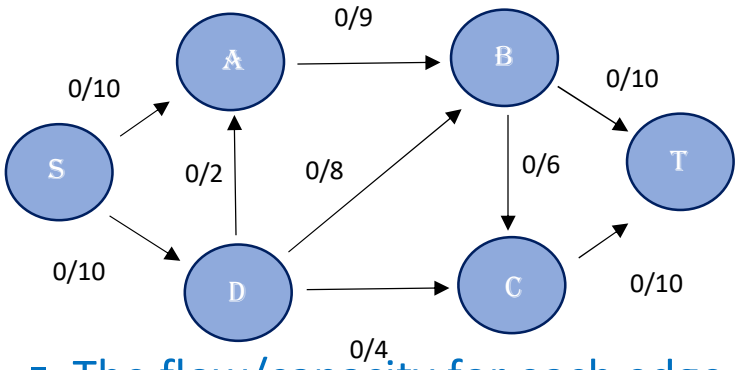
The following is kept after each algorithmic step:

- Capacity restrictions: An edge's capacity cannot be exceeded by the flow along it.
- Flow conservation which states that, excluding the sink and source, there is no net flow to a node.
- The flow entering at t must be equal to the flow departing from s.

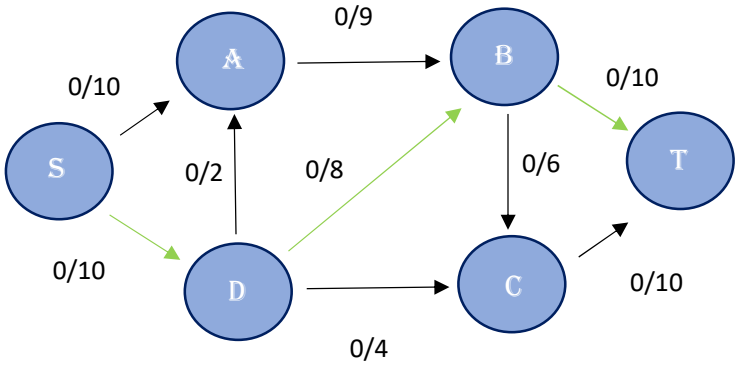
EXAMPLE

Let us take an example in order to understand the algorithm

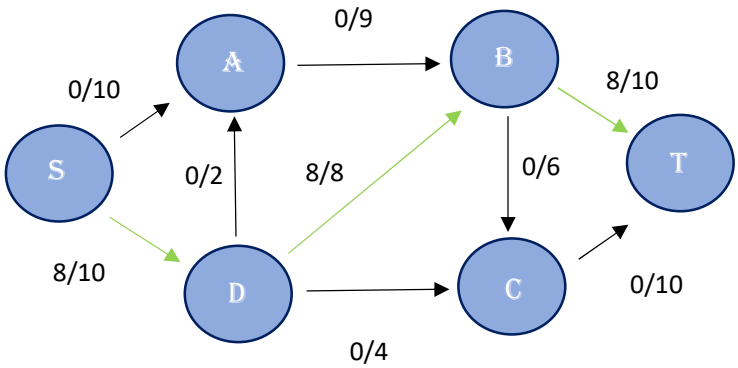
Consider the graph below:



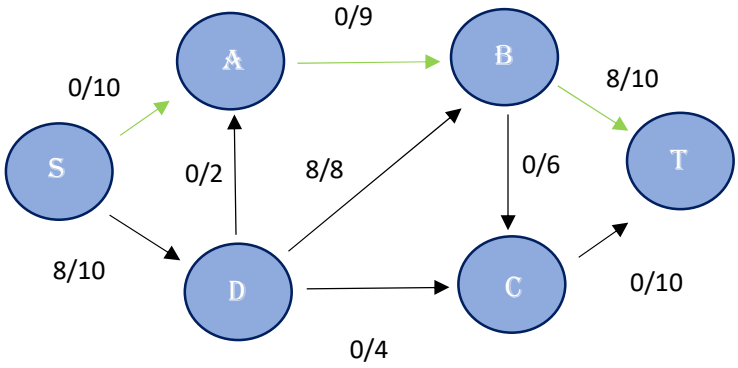
- The flow/capacity for each edge is given. Initially flow is zero for every edge. Now, select any arbitrary path from S to T. In this step we have selected path S-D-B-T.



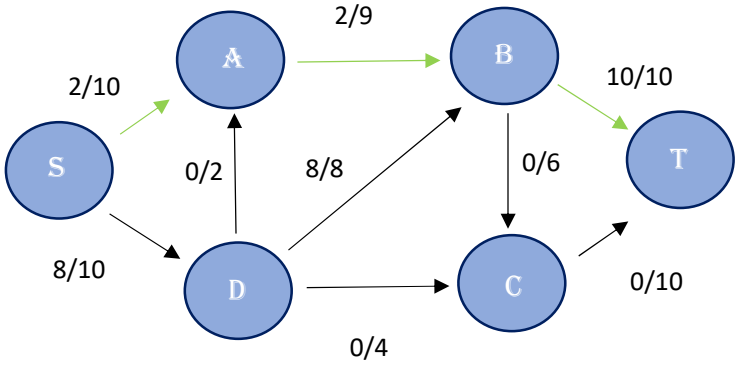
- The minimum capacity among the three edges of the path selected is 8(D-B). This is our bottleneck capacity. Based on this, we update the flow/capacity for each path.



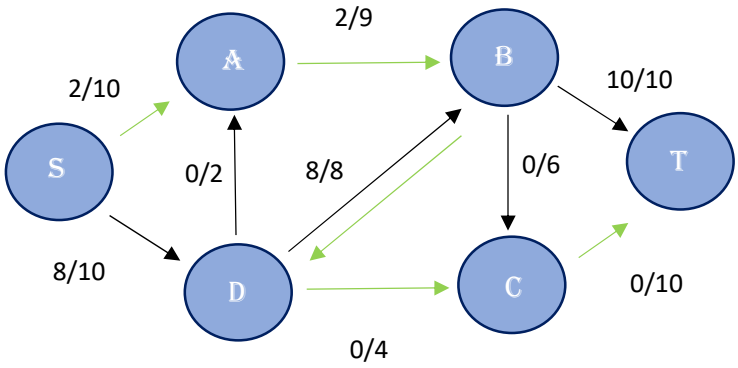
- Select another path S-A-B-T. The minimum capacity among these edges is 2 (B-T).



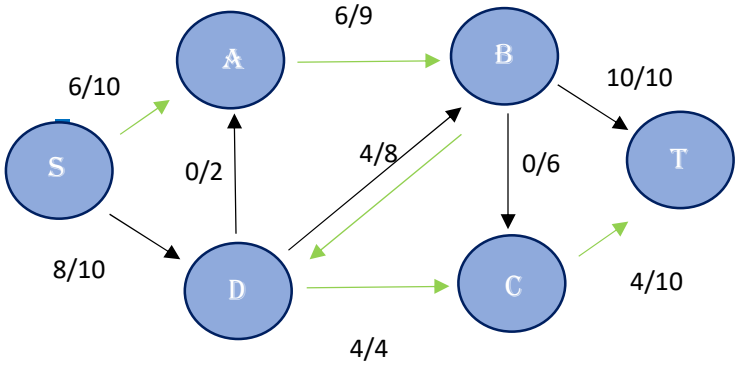
- Update the flow/capacities according to this.



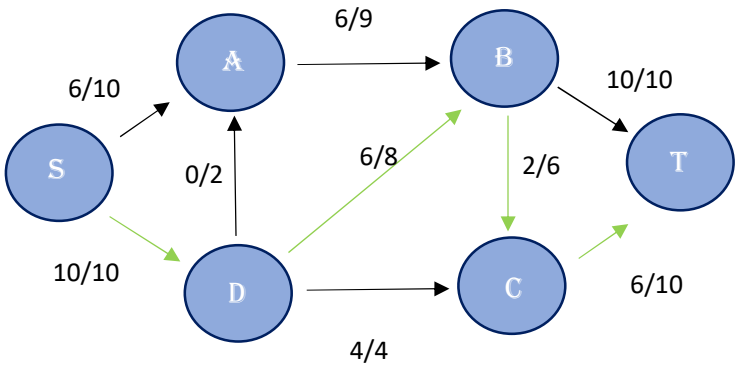
- Next, we select the path S-A-B-D-C-T. The bottleneck capacity for this augmented path is 4(D-C). Here we have considered the non-empty backward edge B-D.



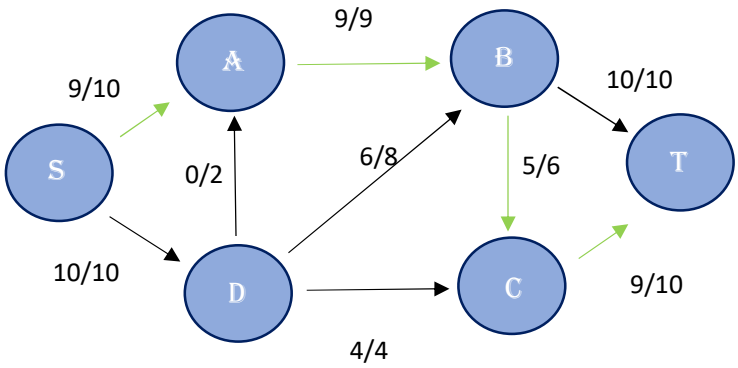
- Now, we update the flow/capacity for each edge.



- Next, we select S-D-B-C-T. The bottleneck capacity for this path is 2(S-D). The updated flow/capacity for this path are:



- Finally we choose the path S-A-B-C-T. The bottleneck capacity for this path is 3. Updating the flow/capacity we get:



- Note that there are no more augmenting paths left in the graph. The maximum flow, thus, is 8+2+4+2+3=19.
- We can find the maximum flow for a given graph in this way using Ford-Fulkerson algorithm.

COMPLEXITY OF THE ALGORITHM

We know that the Ford-Fulkerson algorithm involves the following steps:

1. We find an augmenting path.
2. We compute bottleneck capacity.
3. We augment each edge and total flow.

The complexity of finding an augmenting path is O(E) where E denotes the number of in the network.

Also, the complexity for running the above steps for a chosen augmenting path is O(F) where F is the maximum flow.

Thus, the complexity of the whole algorithm is O(F*E).

APPLICATIONS AND CONCLUSION

Some applications of maximum flow problems and Ford-Fulkerson algorithm are:

- Edge-disjoint paths: The algorithm can be used to compute maximum number of edge-disjoint paths between two vertices say s and t. Edge disjoint paths refer to the set of paths in a graph for which an edge in the graph appears in at most one graph.
- Bipartite matching: Another application is to find a matching (a subgraph in which every vertex has degree at most one) with maximum number of edges in a bipartite graph.
- Exam-scheduling: It is a real-life application of maximum flow problems and Ford-Fulkerson algorithm. We find the maximum number of exams possible for n different classes, r rooms where one room is available for one class, and t different time slots and each exam must be overseen by one of the p proctors.

The Ford-Fulkerson algorithm is thus an easy and useful approach towards many real-life maximum flow problems.