Take a graph and find the max-flow between two vertices

Akhila Jetty (irm2014006)
Arun Kumar Reddy (irm2014005)
Swarnima (iwm2014003)

INTRODUCTION

- Maximum flow problems involve finding a feasible flow through a single-source, single-sink flow network that is maximum.
- Each edge is labeled with a capacity, which acts as the maximum flow of that edge.
- We have to find maximum flow between a given source and sink.
- For this we have to consider all the path between source and sink.
- There are many real life applications of max flow problem.
    - Airlines scheduling .
    - Water supply to different households .
    - Baseball elimination .

Problem Definition

The goal is to find the maximum flow from the vertex s(source) to the vertex t(sink).

- The flow in an edge does not exceed the maximum flow of that edge.
- The Incoming flow is equal to the outgoing flow for all the vertices except the source and the sink.

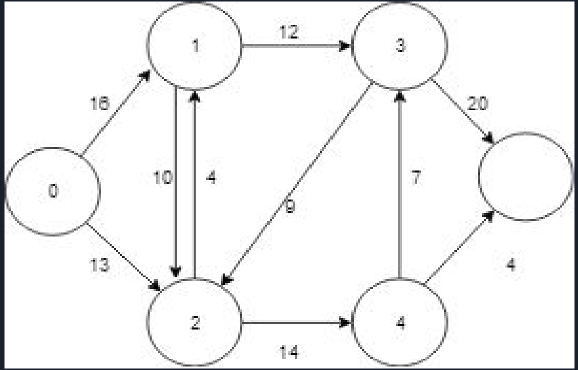


Fig 1. A simple directed graph

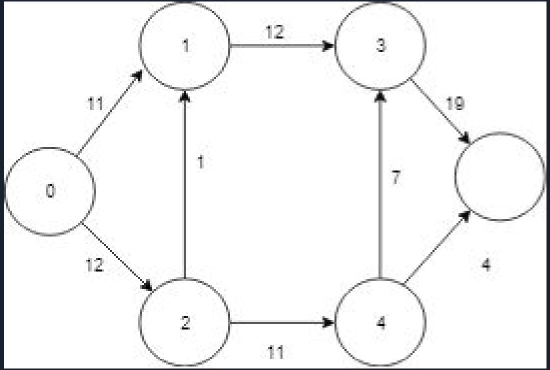


Fig 2. Max flow between source 0 and sink 5

Terminology Used:

- Augmented Path : An augmenting path can be defined as a simple path which does not contain any cycles and consisting of only edges which have a positive capacity from the source to the sink.
- Residual capacity : The amount of additional flow we can push from u to v before exceeding the capacity c(u, v) of the edge from u to v is the residual capacity of (u, v) given by $c_f(u, v) = c(u, v) - f(u, v)$.
- Residual Network : The residual network of G induced by f is a $G_f = (V, E_f)$, where $E_f = \{(u, v) \in V \times V : c_f(u, v) > 0\}$.

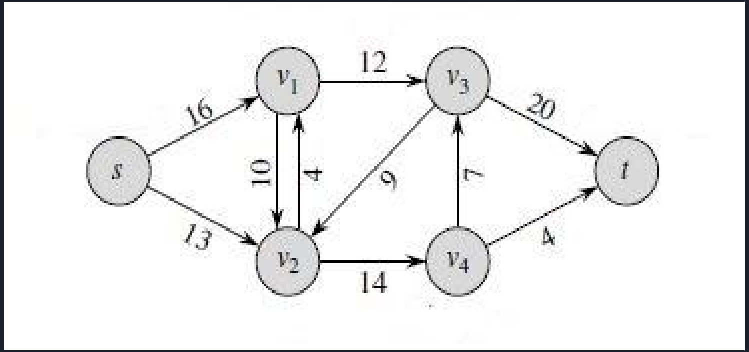


Fig 6. Simple graph

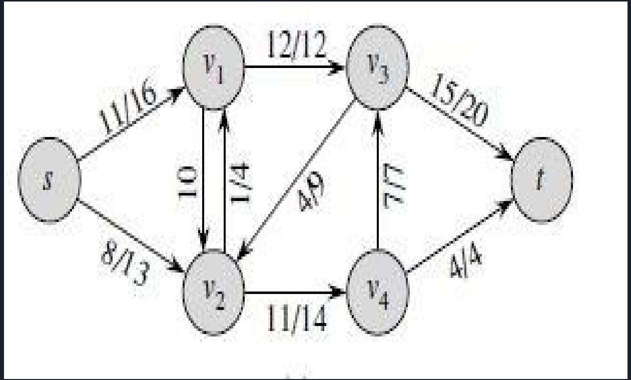


Fig 4. Flow graph

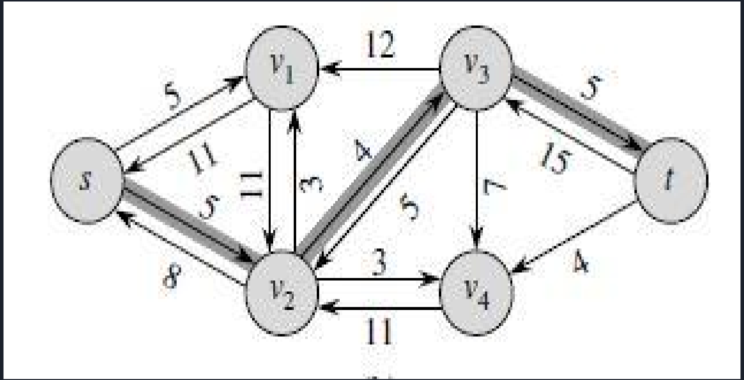
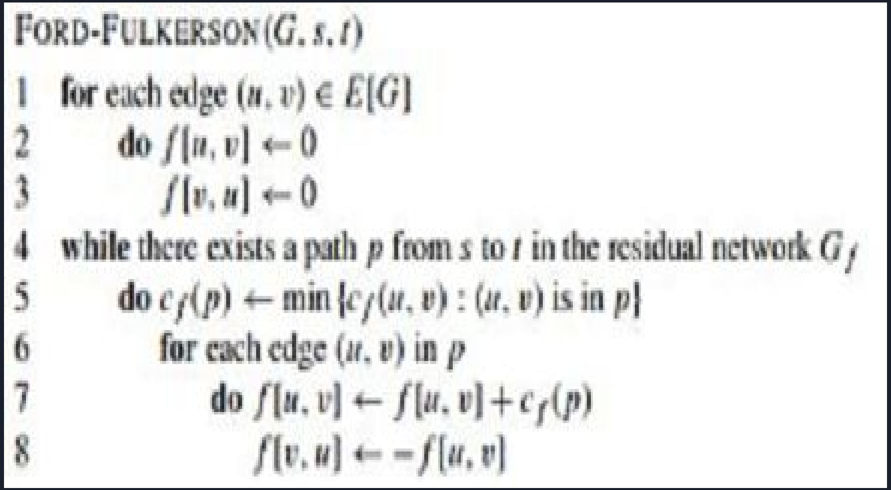


Fig 5. The residual network with augmenting path p shaded

Approach : Ford-Fulkerson Method

- Start with initial flow as 0.
- While there is an augmented path between source and sink
  - Add this path-flow to flow.
- Return flow.



```
FORD-FULKERSON(G, s, t)
1  for each edge $(u, v) \in E[G]$
2       do $f[u, v] \leftarrow 0$
3          $f[v, u] \leftarrow 0$
4  while there exists a path $p$ from $s$ to $t$ in the residual network $G_f$
5       do $c_f(p) \leftarrow \min\{c_f(u, v) : (u, v) \text{ is in } p\}$
6          for each edge $(u, v)$ in $p$
7              do $f[u, v] \leftarrow f[u, v] + c_f(p)$
8                 $f[v, u] \leftarrow -f[u, v]$
```

Modification of ford-fulkerson Algorithm

- Adjacency matrix is a highly inefficient form of representation. so, a different form of representation of the graph might prove to be more beneficial.
- The representation must support the following conditions to be efficient for this algorithm
  - For a BFS to be efficient, the adjacent vertices of a given vertex must be accessible in linear time with respect to the number of neighbours of that vertex.
  - To update the flow of the residual graph in both directions, the edge capacities must be accessible in constant time
- A logical idea would to be use an adjacency list in place of the adjacency matrix. The adjacency list satisfies the first condition but fails for the second one .
- Therefore we can represent the graph as an array of maps/dictionaries.
- Each map in the array contains the neighbours and edge capacities of the vertex represented by its index in the array.

Analysis

- The path flow of every augmented path is determined by the critical edge or the edge with the least capacity in that path. Once the residual graph is updated, this edge disappears. This edge can become critical again only if the flow of the edge is decreased due to an update in the reverse direction,
- BFS always gives the shortest path to a vertex. So when the edge becomes critical for a second time, the path is always longer than in the previous instance when the edge was critical.
- So this way it can be proved that the number of times, an edge can actually become critical is O(V/2-1). As there are E edges in the network, the total number of possible critical edges are O(VE).
- Each augmenting path will have at least one critical edge, therefore the number of possible augmenting paths is O(VE)

TIME COMPLEXITY :-

- Approach 1 :
  - The flow graph is represented as an adjacency matrix with the capacities of each edge as the edge weights.
  - Here we use BFS to find a path with the minimum edges and the flow of the path is added to the total flow and the residual graph is updated.
  - The above step is repeated as long there is a path from the source to the sink vertex.
  - The worst case complexity of Breadth first traversal is O(V^2) as the flow graph is represented as an adjacency matrix .
  - In the worst case, the flow added from each augmented path is as low as possible i.e 1. The number of paths will be the value of the maximum flow of that network in the worst case. Therefore, the time complexity is of the order O(EV^3) .

- Approach 2:
  - So, we have satisfied the above two conditions and optimized the algorithm. The time taken for BFS is now O(E) as a result of the new form of representation.
  - The number of paths will be the value of the maximum flow of that network in the worst case.
  - The overall complexity is therefore reduced to $O(V*E^2)$.