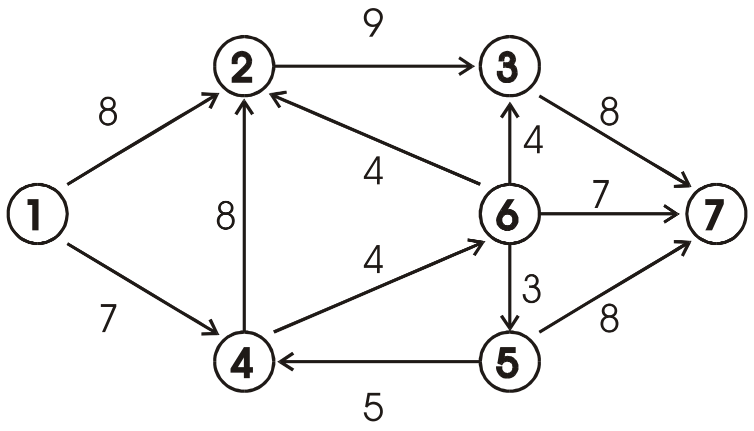
The Maximal Flow Problem

The Maximal flow problem

We are given a digraph – that is a graph with directed edges connecting one vertex to another. Along each edge (or corresponding to each edge) there is an integer-valued quantity that represents the capacity of that edge.

One vertex is defined to be the source, and another vertex is defined to be the sink. We imagine that a fluid is flowing from the source to the sink along the edges, which we may picture as pipes having the designated maximum capacity. The source is the source of the fluid, and the sink is its ultimate destination.

For example, the digraph

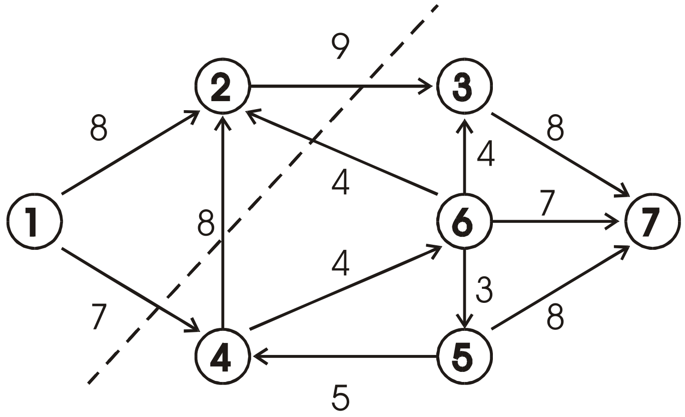


defines a capacitative network where vertex 1 is source and vertex 7 is the sink.

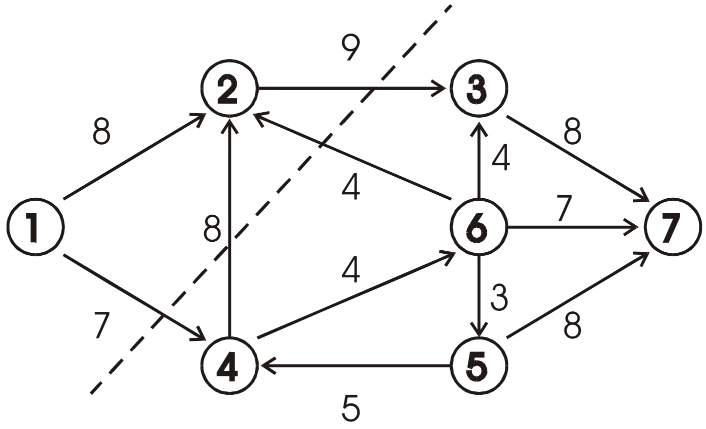
A flow is feasible if the inflow to it does not exceed the outflow from it. (This is called the conservation condition).

The problem is to find the maximum flow in the network – that is the value of the flow that makes the feasible flow as large as possible.

This is found by a process of making “cuts” in the digraph. To illustrate this first



The cut here is shown by the dashed line. Along the cut we evaluate the maximum flow. Any flow in the direction of source to sink (forward flows) is counted at a maximum, and the sum of all forward flows is found. Any flows form sink to source – backflows - are set to zero, and are effectively ignored.



$7 + 9 = 16$

Forward flows are $1 \longrightarrow 4$ and $2 \longrightarrow 3$ with maximum flow 7 + 9 = 16; back flows are $4 \longrightarrow 2$ and $6 \longrightarrow 2$ and we ignore the flows along these backflows when finding the value of the cut.

Thus, technically, a cut is a partition of the digraph into two sets and a computation of the maximum forward flow along the edges joining vertices in one set to vertices in the other. In a cut the source must be in one of the sets and the sink in the other.

Here, our cut partitions the digraph into sets.

S = { 1,2 }
T = { 3,4,5,6,7 }

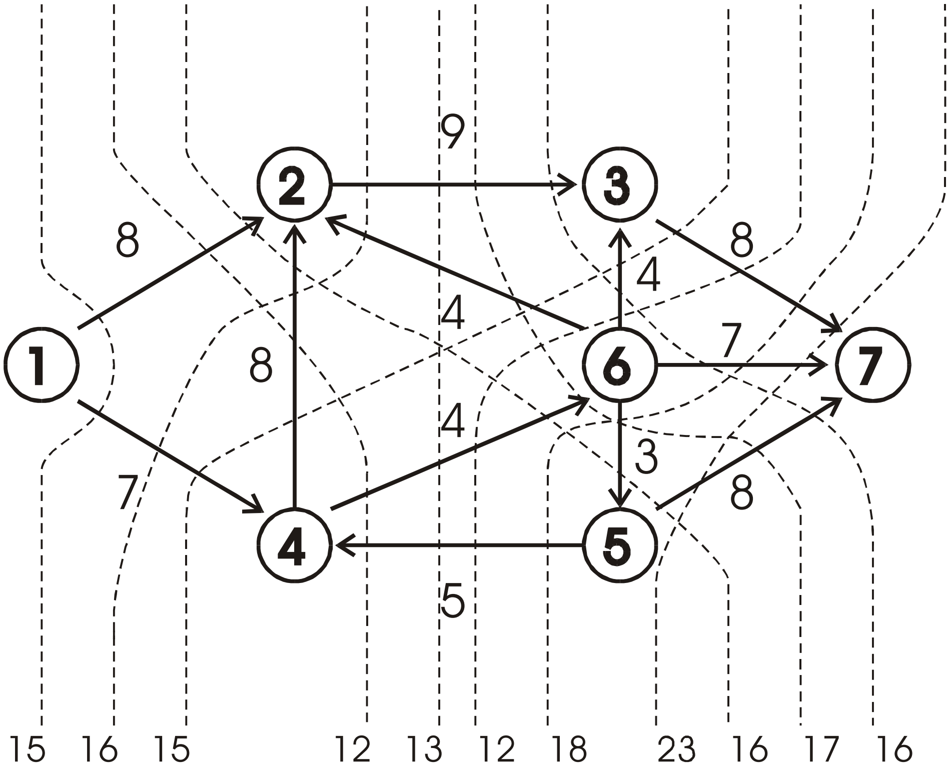
With forward flows $1 \xrightarrow{7} 4$, $2 \xrightarrow{9} 3$ and backflows $4 \xrightarrow{8} 2$, $6 \xrightarrow{4} 2$

The sum of the forward flows is 7 + 9 = 16 as before.

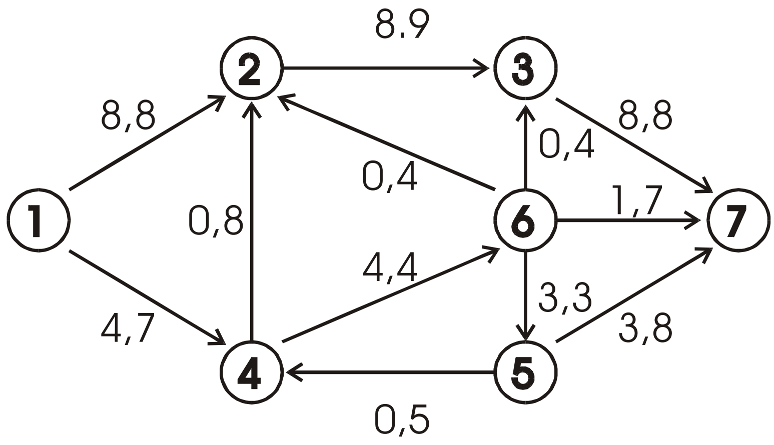
(This technical language might be important for theoretical work, and also if we were considering programming a computer to find the maximum flow.)

The solution to the maximum flow problem now follows the max-flow, min-cut theorem (the Ford-Fulterson theorem): In a capacitated network, the value of a maximum flow is equal to the capacity of a minimum cut.

Thus, to solve the problem we simply evaluate all possible cuts – the maximum flow will correspond to the value of the minimum cut.



This diagram shows all the cuts in the capacitative network. The minimum value of the cut is 12, and this is the maximum flow from source (1) to sink (7). Along the minimum cuts the assignment of flows must equal the capacity of the edge; elsewhere the assignment must be such that the flow through the cut is equal to the maximum flow. One possible assignment of flows in this example is:



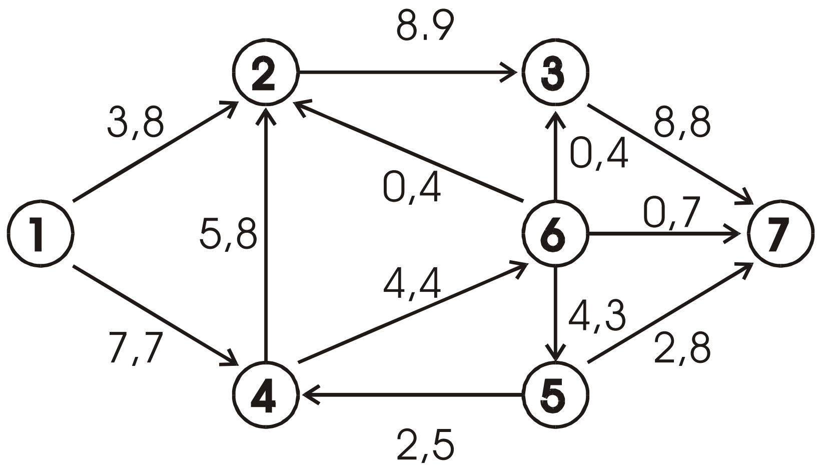
On this diagram the first number along the edge represents the actual flow along that edge; the second represents the capacity of the edge.

The process of finding all the cuts in a network could be tedious, especially is the network was large. Additionally, we may start with a feasible flow and ask the question can the flow be improved so that we obtain an increased or maximum flow rate.

We seek an algorithm that starts with a feasible flow and augments it until we obtain a maximum flow.

This is provided by the Edmonds-Karp algorithm to solve the maximum flow problem.

We will illustrate this process by using it on our example. A feasible flow is given by:

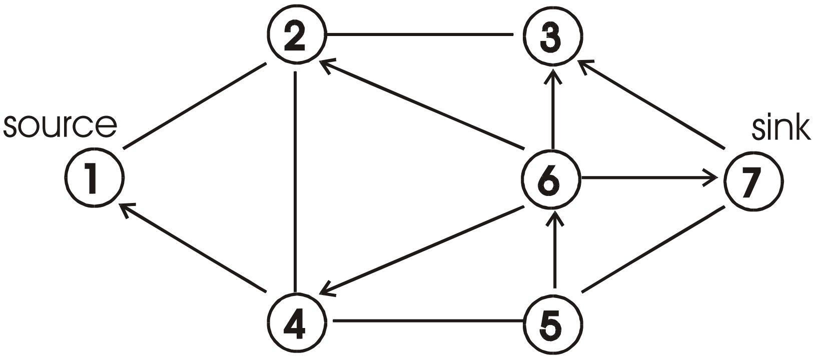


We now construct another digraph derived from this one, with the rules

1. If an edge is flow saturated, replace it by an edge with the arrow pointing in the opposite direction. An edge is flow saturated if its flow is equal to its capacity. Here $1 \longrightarrow 4$, $4 \longrightarrow 6$, and $3 \longrightarrow 7$ are flow saturated and will be replaced by edges with arrows pointing in the opposite direction.

2. If an edge has flow zero leave it unchanged – this applies to edges $6 \longrightarrow 2$, $6 \longrightarrow 3$ and $6 \longrightarrow 7$.

3. If an edge has a flow that is not zero but less than its capacity then replace it by an edge without any direction (arrows). This applies to edges $1 \longrightarrow 2$, $4 \longrightarrow 2$, $2 \longrightarrow 3$, $5 \longrightarrow 4$, $6 \longrightarrow 5$ and $5 \longrightarrow 7$.



The next stage is to ask whether it is possible to connect the source to the sink along edges that are either undirected or are directed with arrows pointing from source to sink.

Here, there is a path 1 – 2 – 4 – 5 – 7; there is also a second path 1 – 2 – 4 – 5 – 6 – 7. The backwards-pointing arrows prevent any other path.

Since there is a path, it is possible to augment the flow along this path. To augment the path, decrease any backflow along it as much as possible and increase the forward flow as much as possible. The increase or decrease along the path must be the same along all edges.

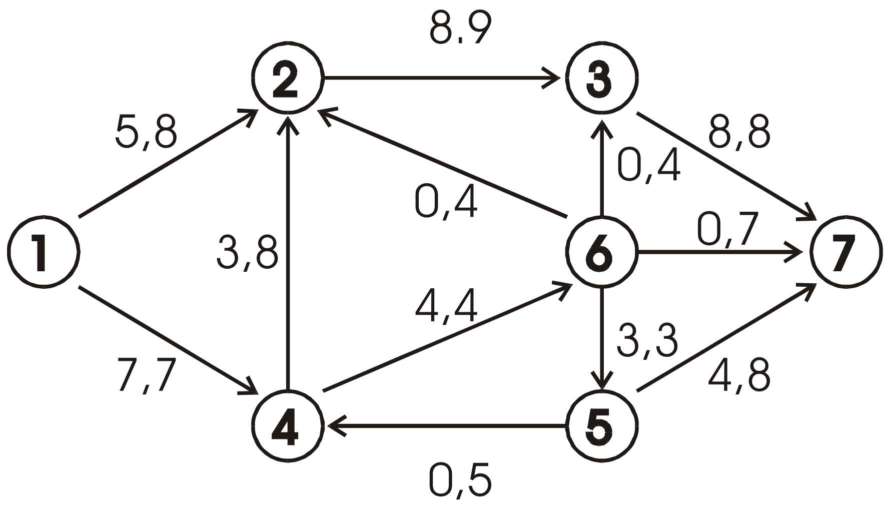
The existing flow is

$$1 \xrightarrow{3,8} 2 \xleftarrow{5,8} 4 \xleftarrow{2,5} 5 \xrightarrow{2,8} 7$$

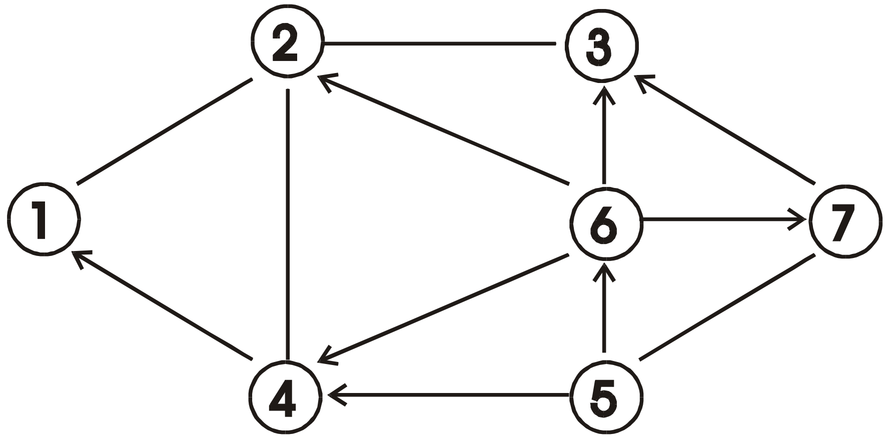
Increasing by +2 units along this path, we obtain the flow

$$1 \xrightarrow{5,8} 2 \xleftarrow{3,8} 4 - \overset{0,5}{-} - 5 \xrightarrow{4,8} 7$$

We obtain the following revised assignment:



We repeat the process of constructing the derived digraph.



It is now not possible to construct a path in the derived digraph connecting the source to the sink. This tells us that the augmented flow is a maximal flow and that we have solved the problem.