Flow Networks and Flows

Flow Network is a directed graph that is used for modeling material Flow. There are two different vertices; one is a **source** which produces material at some steady rate, and another one is sink which consumes the content at the same constant speed. The flow of the material at any mark in the system is the rate at which the element moves.

Some real-life problems like the flow of liquids through pipes, the current through wires and delivery of goods can be modeled using flow networks.

Definition: A Flow Network is a directed graph G = (V, E) such that

1. For each edge (u, v) ∈ E, we associate a nonnegative weight capacity c (u, v) ≥ 0.If (u, v) ∉ E, we assume that c (u, v) = 0.
2. There are two distinguishing points, the source s, and the sink t;
3. For every vertex v ∈ V, there is a path from s to t containing v.

Let G = (V, E) be a flow network. Let s be the source of the network, and let t be the sink. A flow in G is a real-valued function f: V x V→R such that the following properties hold:

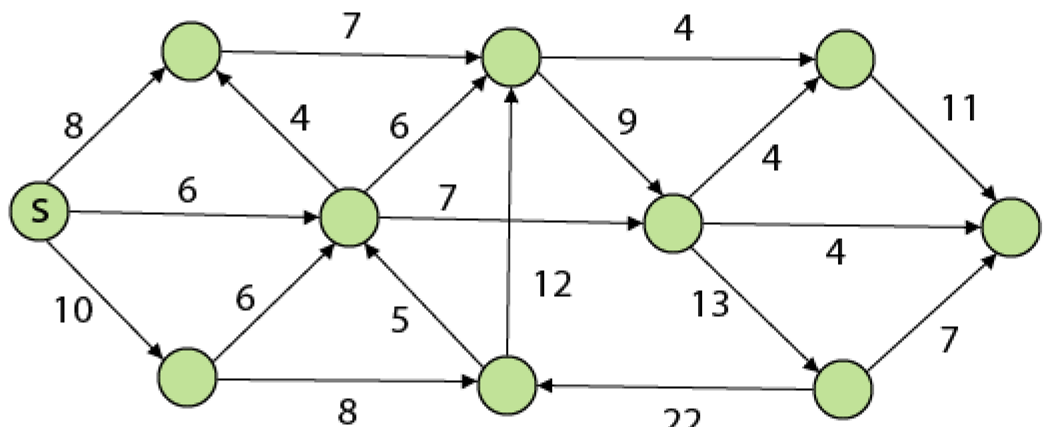
- **Capacity Constraint:** For all u, v ∈ V, we need f (u, v) ≤ c (u, v).
- **Skew Symmetry:** For all u, v ∈ V, we need f (u, v) = - f (u, v).
- **Flow Conservation:** For all u ∈ V-{s, t}, we need

$$\boxed{\sum_{v \in V} f(u, v) = \sum_{u \in V} f(u, v) = 0}$$

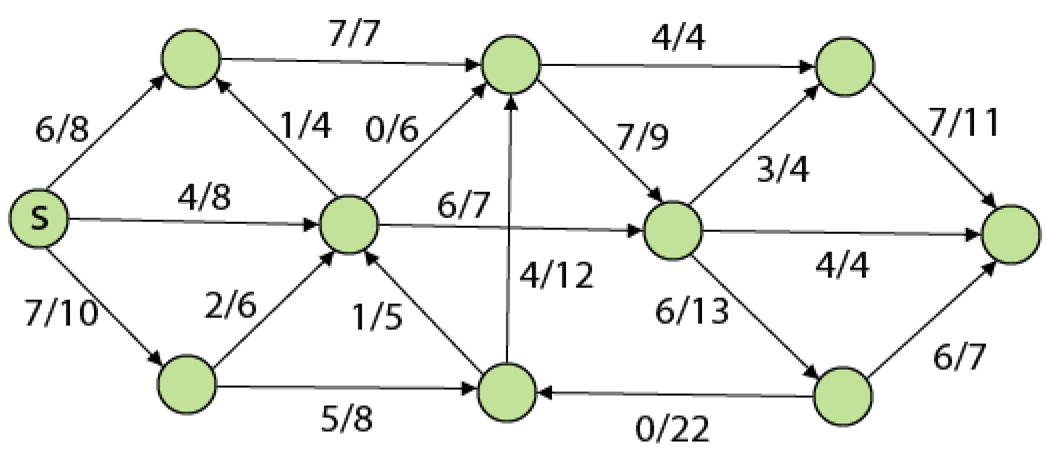
The quantity f (u, v), which can be positive or negative, is known as the net flow from vertex u to vertex v. In the **maximum-flow problem**, we are given a flow network G with source s and sink t, and we wish to find a flow of maximum value from s to t.

The three properties can be described as follows:

1. **Capacity Constraint** makes sure that the flow through each edge is not greater than the capacity.
2. **Skew Symmetry** means that the flow from u to v is the negative of the flow from v to u.
3. The flow-conservation property says that the total net flow out of a vertex other than the source or sink is 0. In other words, the amount of flow into a v is the same as the amount of flow out of v for every vertex v ∈ V - {s, t}



(a)



(b)

The value of the flow is the net flow from the source,

$$|f| = \sum_{v \in V} f(s, v)$$

The **positive net flow entering** a vertex v is described by

$$\sum_{\{u \in V : f(u,v) > 0\}} f(u, v)$$

The **positive net flow** leaving a vertex is described symmetrically. One interpretation of the Flow-Conservation Property is that the positive net flow entering a vertex other than the source or sink must equal the positive net flow leaving the vertex.

A flow f is said to be **integer-valued** if f (u, v) is an integer for all (u, v) ∈ E. Clearly, the value of the flow is an integer is an integer-valued flow.

Network Flow Problems

The most obvious flow network problem is the following:

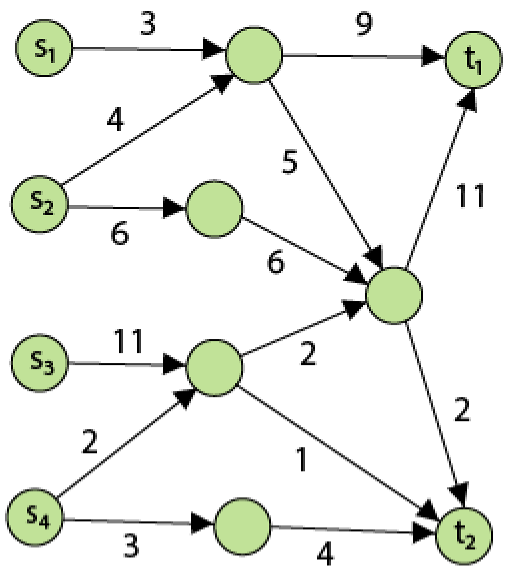
Problem1: Given a flow network G = (V, E), the maximum flow problem is to find a flow with maximum value.

Problem 2: The multiple source and sink maximum flow problem is similar to the maximum flow problem, except there is a set $\{s_1,s_2,s_3.......s_n\}$ of sources and a set $\{t_1,t_2,t_3..........t_n\}$ of sinks.

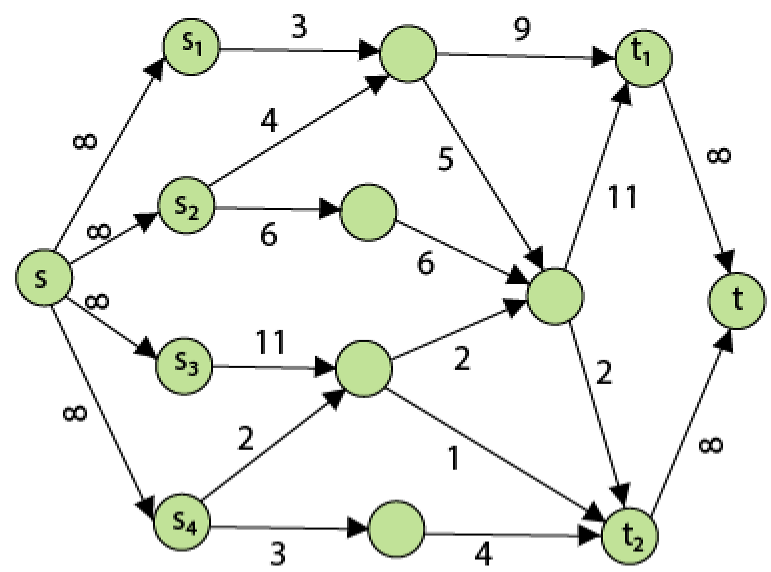
Fortunately, this problem is no solid than regular maximum flow. Given multiple sources and sink flow network G, we define a new flow network G' by adding

- A super source s,
- A super sink t,
- For each s_i, add edge (s, s_i) with capacity ∞, and
- For each t_i,add edge (t_i,t) with capacity ∞

Figure shows a multiple sources and sinks flow network and an equivalent single source and sink flow network



(a)



(b)

Residual Networks: The Residual Network consists of an edge that can admit more net flow. Suppose we have a flow network G = (V, E) with source s and sink t. Let f be a flow in G, and examine a pair of vertices u, v ∈ V. The sum of additional net flow we can push from u to v before exceeding the capacity c (u, v) is the residual capacity of (u, v) given by

$$c_f(u, v) = c\,(u, v) - f\,(u, v).$$

When the net flow f (u, v) is negative, the residual capacity c_f (u,v) is greater than the capacity c (u, v).

For Example: if c (u, v) = 16 and f (u, v) =16 and f (u, v) = -4, then the residual capacity c_f (u,v) is 20.

Given a flow network G = (V, E) and a flow f, the residual network of G induced by f is G_f = (V, E_f), where

$$E_f = \{(u, v) \in V \text{ x } V: C_f(u, v) \geq 0\}$$

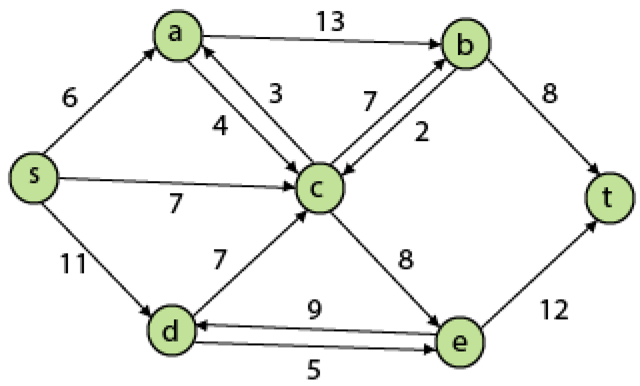
That is, each edge of the residual network, or residual edge, can admit a strictly positive net flow.

Augmenting Path: Given a flow network G = (V, E) and a flow f, an **augmenting path** p is a simple path from s to t in the residual networkG_f. By the solution of the residual network, each edge (u, v) on an augmenting path admits some additional positive net flow from u to v without violating the capacity constraint on the edge.

Let G = (V, E) be a flow network with flow f. The **residual capacity** of an augmenting path p is

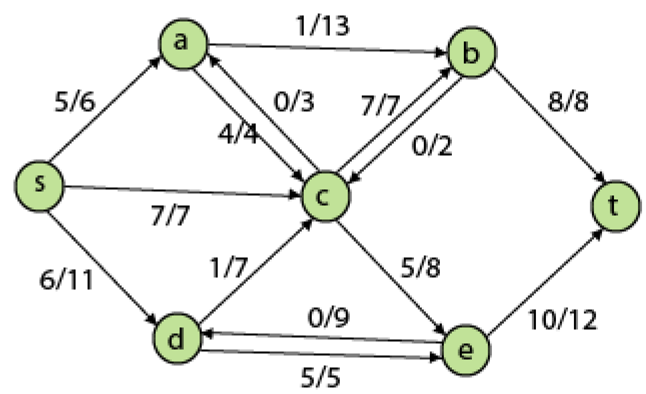
$$C_f(p) = \min\,\{C_f(u, v): (u, v) \text{is on p}\}$$

The residual capacity is the maximal amount of flow that can be pushed through the augmenting path. If there is an augmenting path, then each edge on it has a positive capacity. We will use this fact to compute a maximum flow in a flow network.



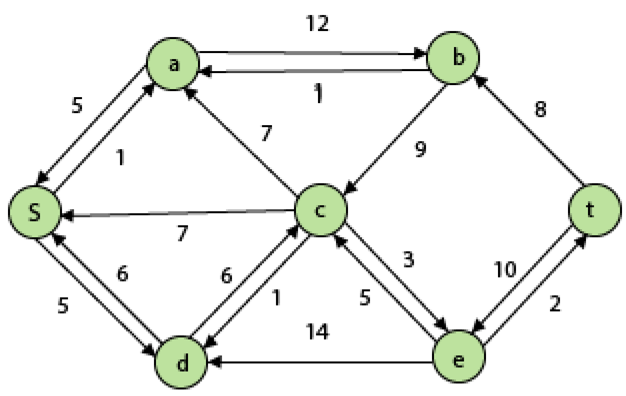
(a)

A flow network G= (V,E)

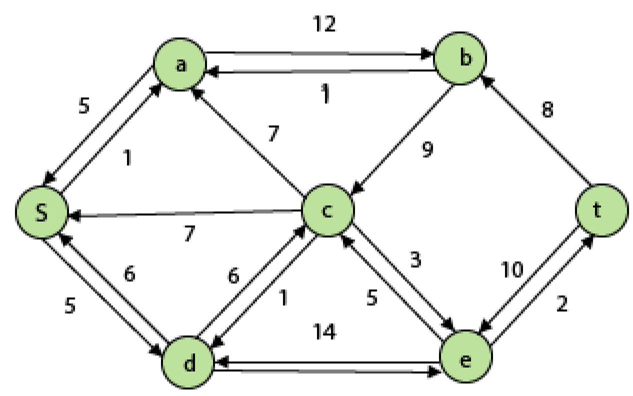


(b)

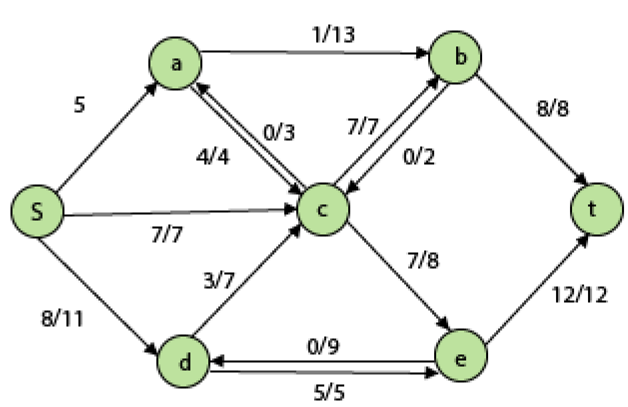
A flow in f in G



(c)
The residual network G_f

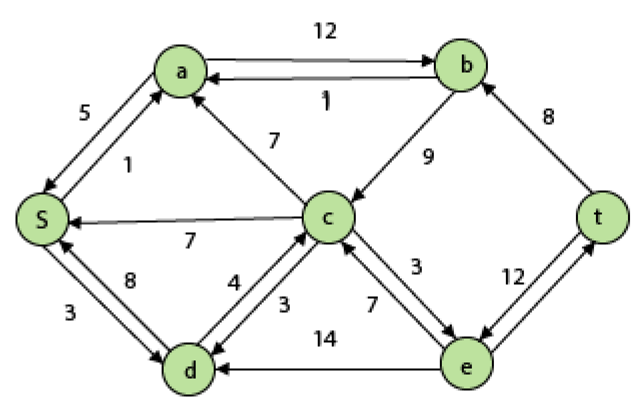


(d)
The grey edges form an augmenting path with capacity z.



(e)

A new flow $f' = f + f_p$



(f)

The residual network G_f

Ford-Fulkerson Algorithm

Initially, the flow of value is 0. Find some augmenting Path p and increase flow f on each edge of p by residual Capacity c_f (p). When no augmenting path exists, flow f is a maximum flow.

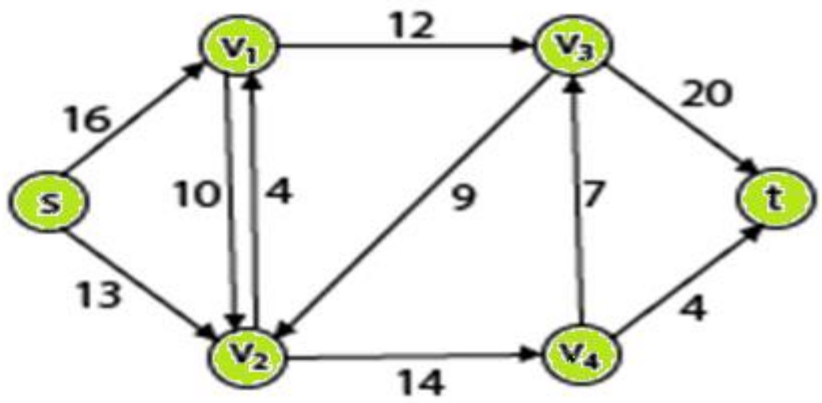
FORD-FULKERSON METHOD (G, s, t)

```
Initialize flow f to 0
while there exists an augmenting path p
do argument flow f along p
Return f
```

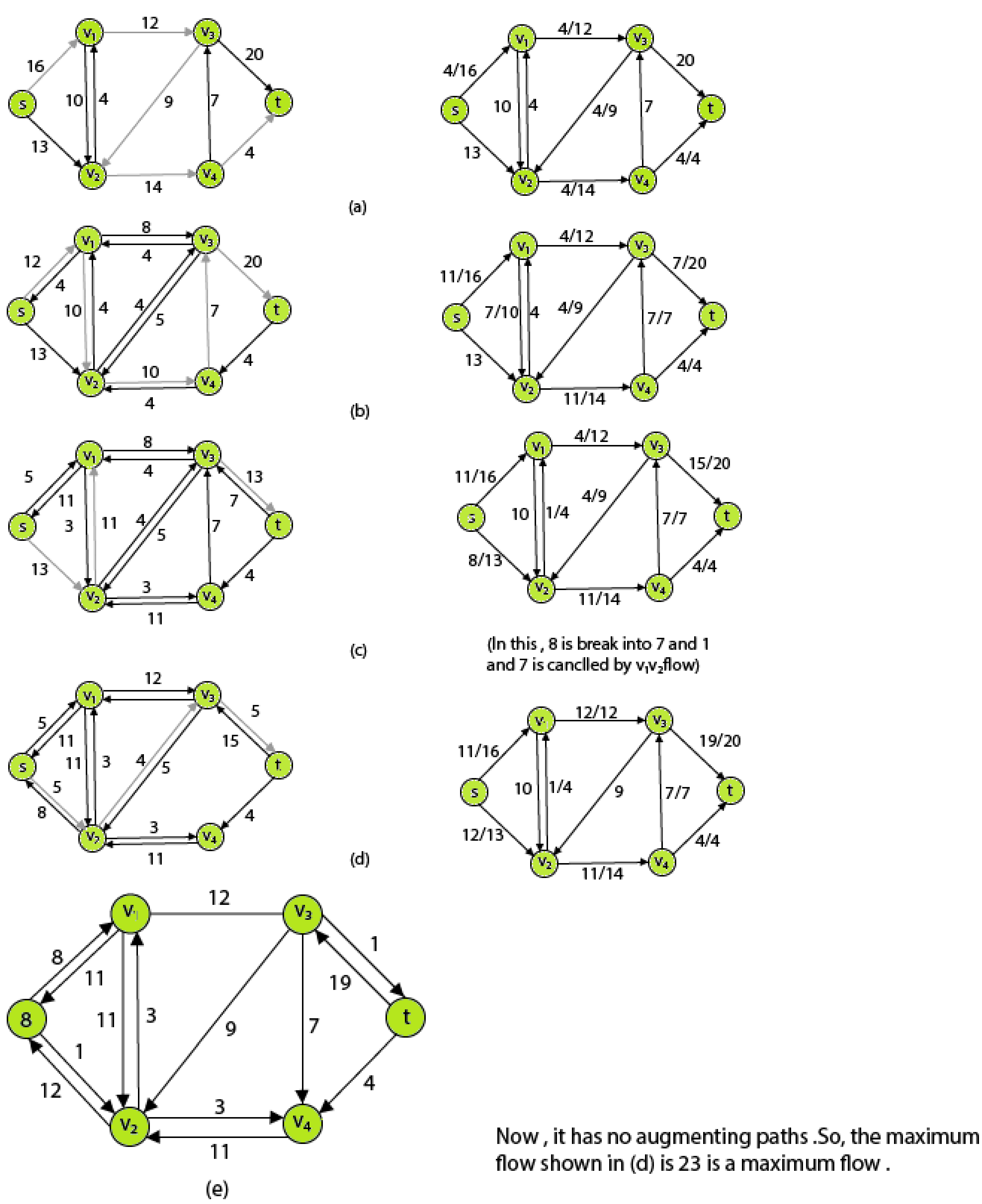
FORD-FULKERSON (G, s, t)

```
for each edge (u, v) ∈ E [G]
do f [u, v] ← 0
f [u, v] ← 0
while there exists a path p from s to t in the residual network Gf.
do cf (p)←min?{ Cf (u,v):(u,v)is on p}
for each edge (u, v) in p
do f [u, v] ← f [u, v] + cf  (p)
f [u, v] ←-f[u,v]
```

Example: Each Directed Edge is labeled with capacity. Use the Ford-Fulkerson algorithm to find the maximum flow.

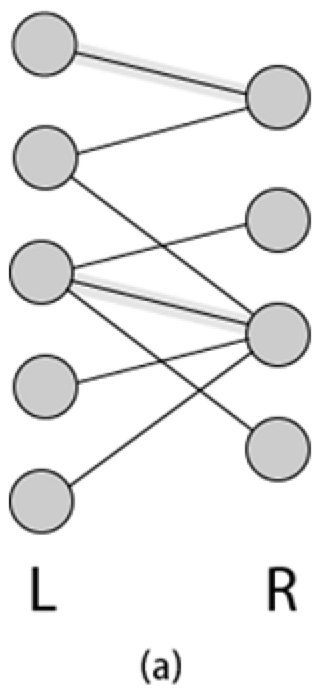


Solution: The left side of each part shows the residual network G_f with a shaded augmenting path p,and the right side of each part shows the net flow f.



Maximum Bipartite Matching

A Bipartite Graph is a graph whose vertices can be divided into two independent sets L and R such that every edge (u, v) either connect a vertex from L to R or a vertex from R to L. In other words, for every edge (u, v) either u ∈ L and v ∈ L. We can also say that no edge exists that connect vertices of the same set.



Matching is a Bipartite Graph is a set of edges chosen in such a way that no two edges share an endpoint. Given an undirected Graph G = (V, E), a Matching is a subset of edge M ⊆ E such that for all vertices v ∈ V, at most one edge of M is incident on v.

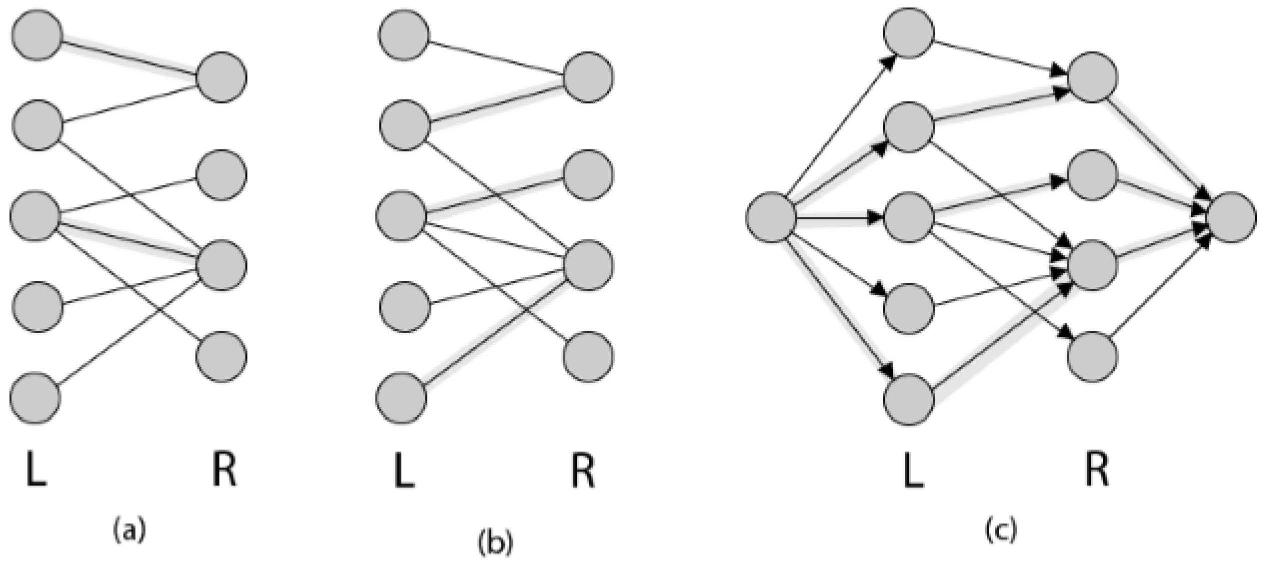
A Maximum matching is a matching of maximum cardinality, that is, a matching M such that for any matching M', we have|M|>|M' |.

Finding a maximum bipartite matching

We can use the Ford-Fulkerson method to find a maximum matching in an undirected bipartite graph G= (V, E) in time polynomial in |V| and |E|. The trick is to construct a flow network G= (V',E') for the bipartite graph G as follows. We let the source s and sink t be new vertices not in V, and we let V'=V ∪{s,t}.If the vertex partition of G is V = L∪R, the directed edges of G' are the edges of E, directed from L to R, along with |V| new directed edges:

$$\mathbf{E}' = \{(\mathbf{s},\mathbf{u}) : \mathbf{u} \in \mathbf{L}\} \cup \{(\mathbf{u},\mathbf{v}) : (\mathbf{u},\mathbf{v}) \in \mathbf{E}\} \cup \{(\mathbf{v},\mathbf{t}) : \mathbf{v} \in \mathbf{R}\}$$

Fig: A Bipartite Graph G = (V, E) with vertex partition V = L ∪ R.



Matching with Cardinality 2.

Matching with Cardinality 3.

The corresponding flow network G′ with a maximum flow shown. Each edge has unit Capacity. The shaded edges from L to R correspond to those in the maximum matching from (b).