Maximum Flow Problem

Maximum Flow problem/
Ford-Fulkerson method/
Edmonds-Karp method

Maximum Flow Problem

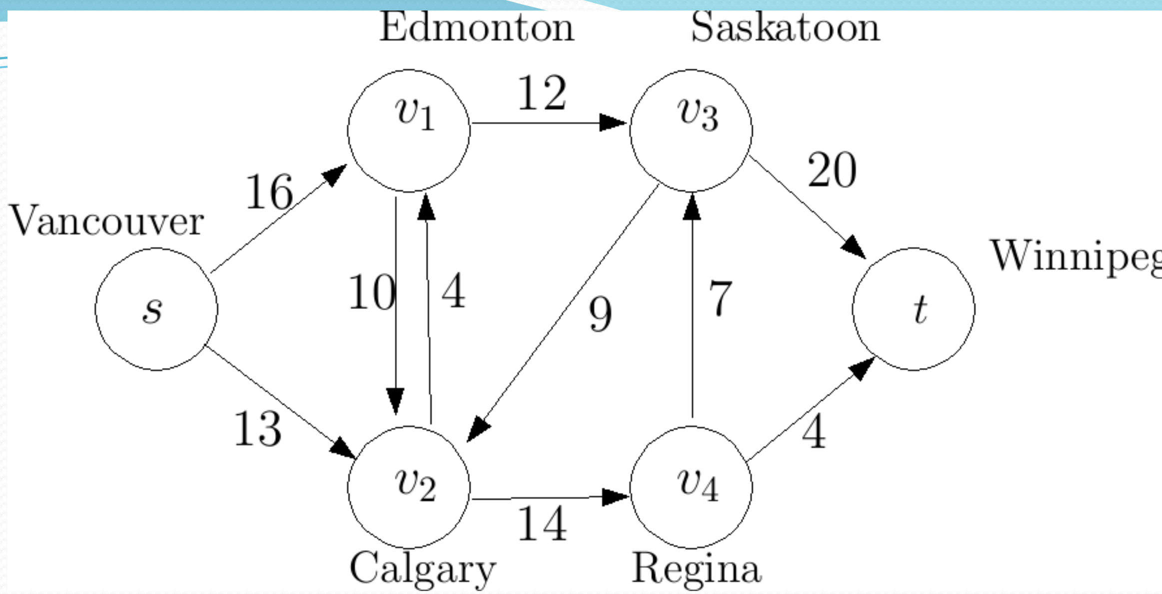
Given a flow network as a directed graph in which

- fluid is flowing along the edges of the graph
- each edge has maximum capacity that it can carry

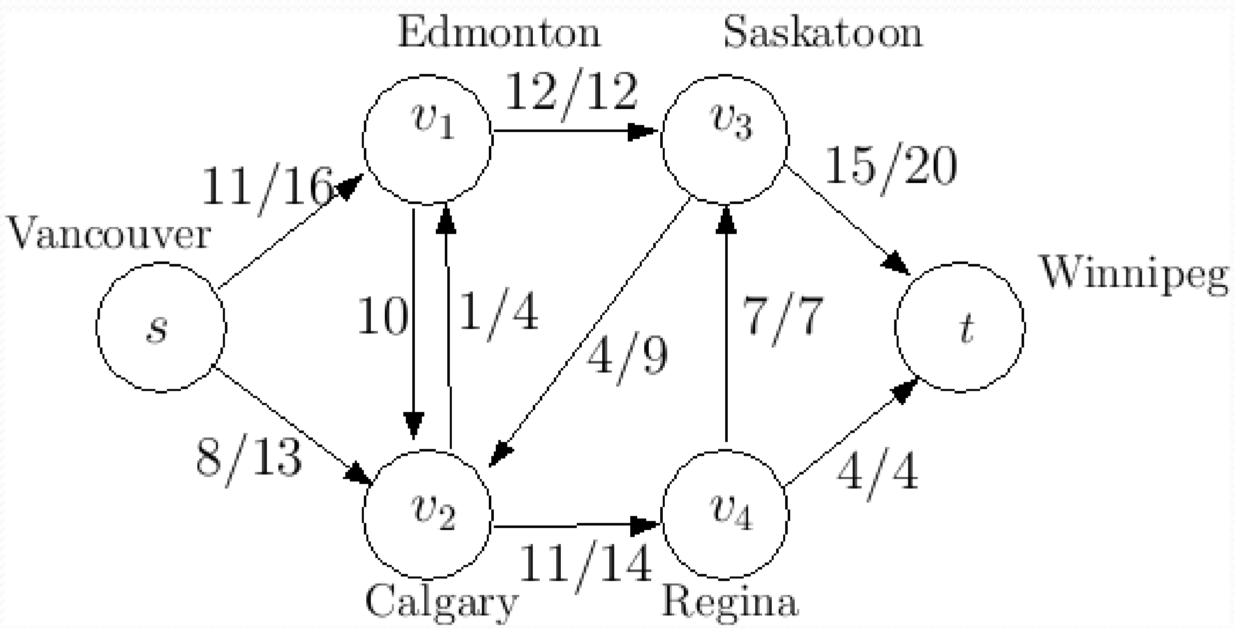
How much flow we can push from the source to the sink?

A flow network

G



A flow for G



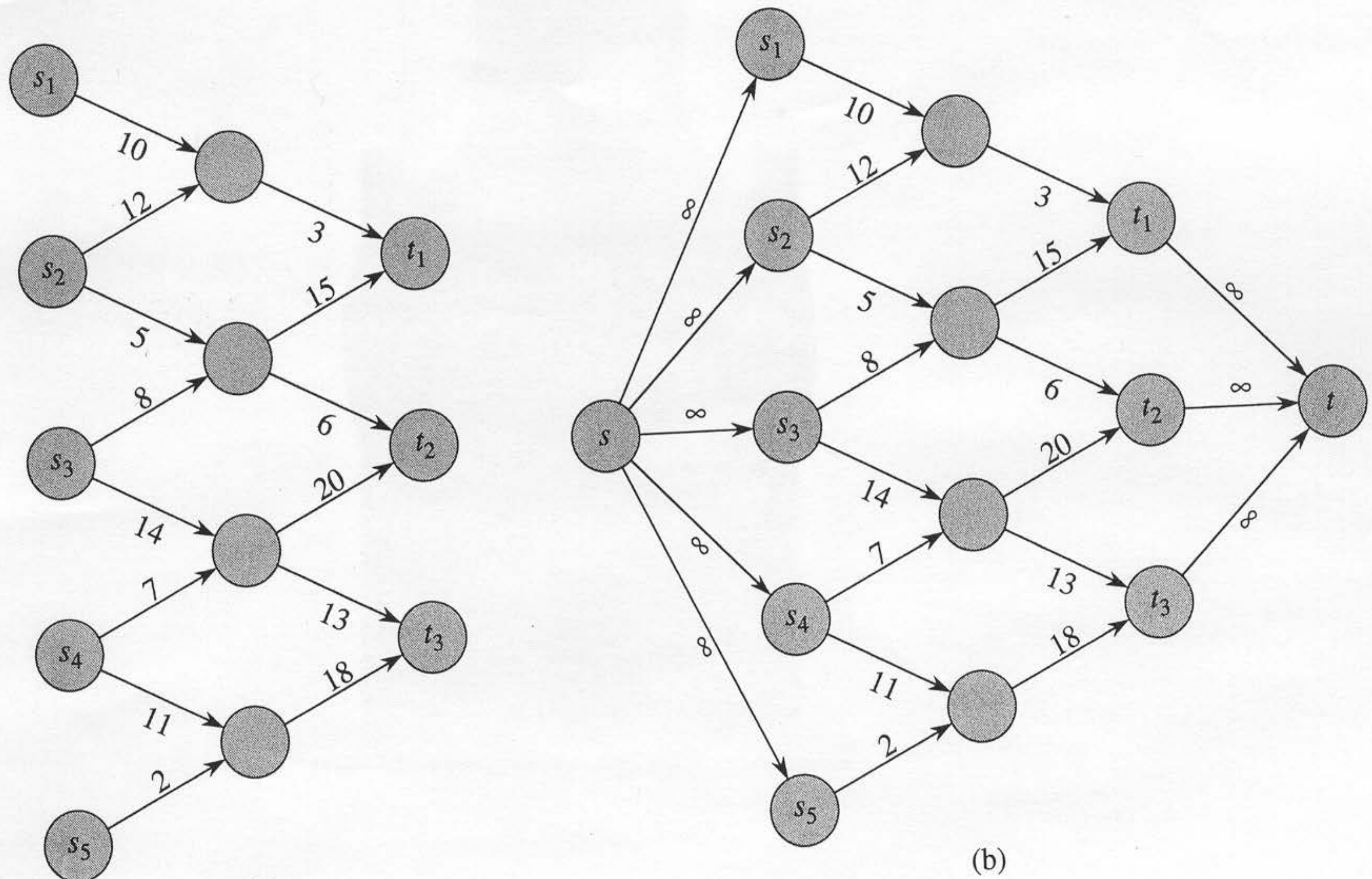
Definitions

- Flow network G=(V,E): directed graph with
  - Edge (u,v) has a nonnegative capacity c(u,v) >=0
  - If (u,v) not in E, then c(u,v)=0
  - Every vertex lies on some path from the source s to the sink t
- Flow f : a function V X V -> R satisfying
  - Capacity constraint: f(u,v) <= c(u,v)
  - Skew symmetry: f(u,v) = - f(v,u)
  - Flow conservation: for all u in V-{s,t},

    sum(f(u,v) : v in V) = 0
- Value of the flow f = |f| := sum(f(s,v) : v in V)

Multi-source Multi-sink Flow Problem

- Many sources si and Many sinks tj
- Convert to Single-source Single-sink flow problem by
  - a supersource s' and a supersink t'
  - Attach s' to all the si with infinite capacity
  - Attach all the tj to t‘ with infinite capacity



(a) (b)

Set notation and Lemmas

- $f(X,Y) = \text{sum}_{x \in X} \text{sum}_{y \in Y} f(x,y)$
- Lemma
  - $f(X,Y) = -f(Y,X)$
  - $f(X,X)=0$
  - If X and Y are disjoint, then

    $f(X \cup Y, Z) = f(X,Z) + f(Y,Z)$

    $f(Z, X \cup Y) = f(Z,X) + f(Z,Y)$

Properties

- $|f| = f(s,V) = f(V,t)$
  - $f(u,V)=0$ for all u in V-s-t, so $f(V\text{-}s\text{-}t,V)=0$ and $f(V,V\text{-}s\text{-}t)=0$
  - $|f|=f(s,V)=f(V,V)-f(V\text{-}s,V)=f(V,V\text{-}s)$

    $= f(V,V\text{-}s\text{-}t)+f(V,t)= f(V,t)$
- Cut (S,T) of G(V,E): a partition of V into S and T=V-S such that s in S, t in T
- f(S,T) : Net flow across cut (S,T)
- c(S,T) : Capacity of cut (S,T)

Properties (cont'd)

- **Minimum cut** of G: a cut having minimum capacity over all cuts of G
- **For any cut (S,T), $f(S,T)=|f|$**
  - $f(S-s,V)=0$
  - $f(S,T)=f(S,V)-f(S,S)=f(S,V)=f(S-s,V)+f(s,V)=|f|$
- $|f| <= c(S,T)$ for any cut (S,T)
  - $|f|=f(S,T) <= c(S,T)$
- **Maximum flow $|f| <=$ capacity of minimum cut**

Residual Network

- Given a flow network G and a flow f, residual capacity of (u,v) is defined by

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

  - If (u,v) in E & $c(u,v) > f(u,v) > 0$

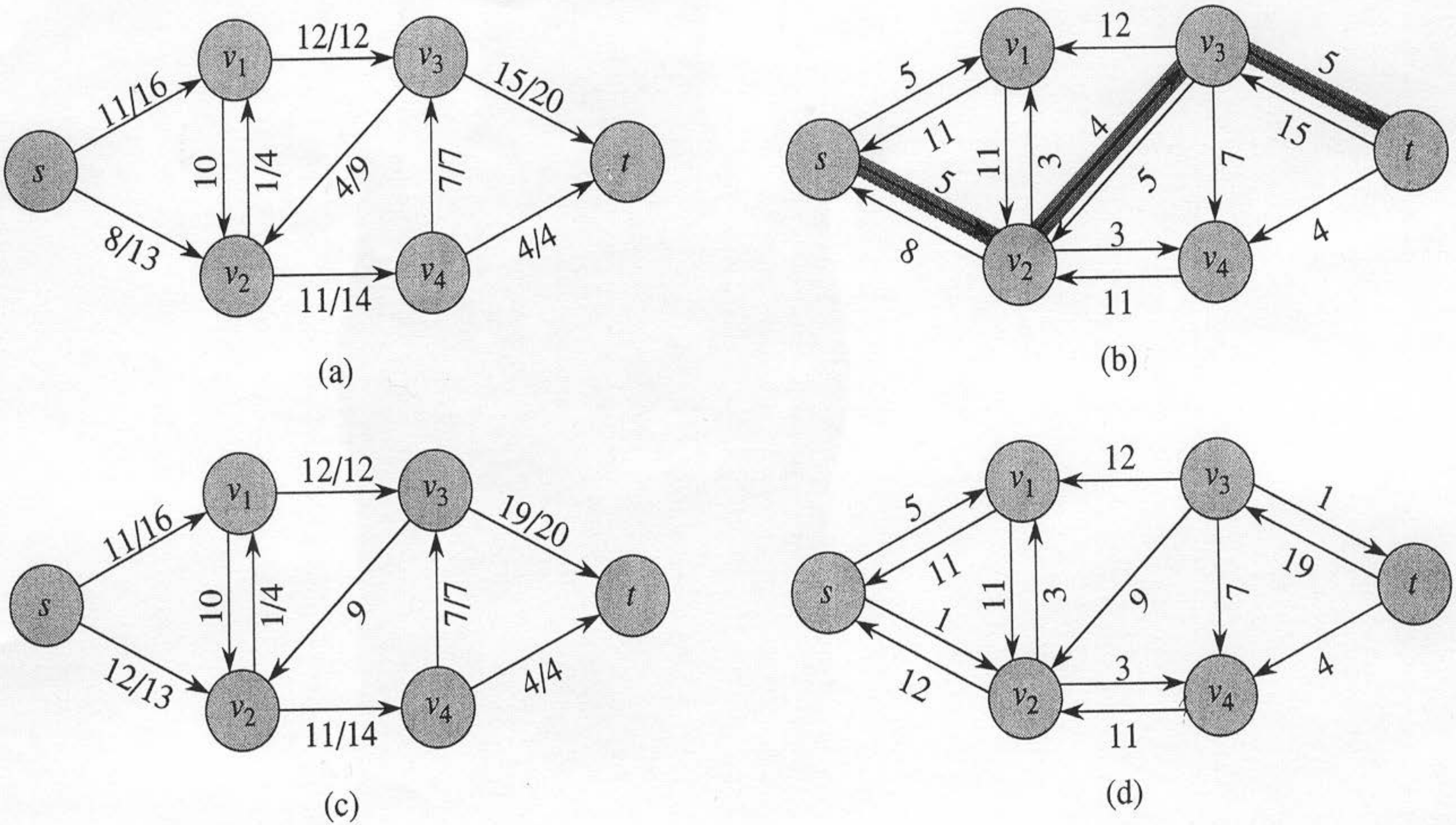
    $0 < c_f(u,v) < c(u,v)$

  - If $f(u,v) < 0$

    $0 \leq c(u,v) < c_f(u,v)$

- Residual network of G induced by f is

  $G_f = (V,E_f)$ where $E_f=\{(u,v) : c_f(u,v) > 0\}$



s
t
v_1
v_2
v_3
v_4
11/16
12/12
15/20
10
1/4
4/9
7/7
8/13
11/14
4/4
(a)
12
5
11
11
3
4
5
15
7
5
8
3
11
4
(b)
11/16
12/12
19/20
10
1/4
9
7/7
12/13
11/14
4/4
(c)
12
5
1
11
11
3
9
7
19
1
12
3
11
4
(d)

Augmenting paths

- augmenting path: a simple path from s to t in G_f
- residual capacity of path p

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

- Max-Flow Min-cut Theorem

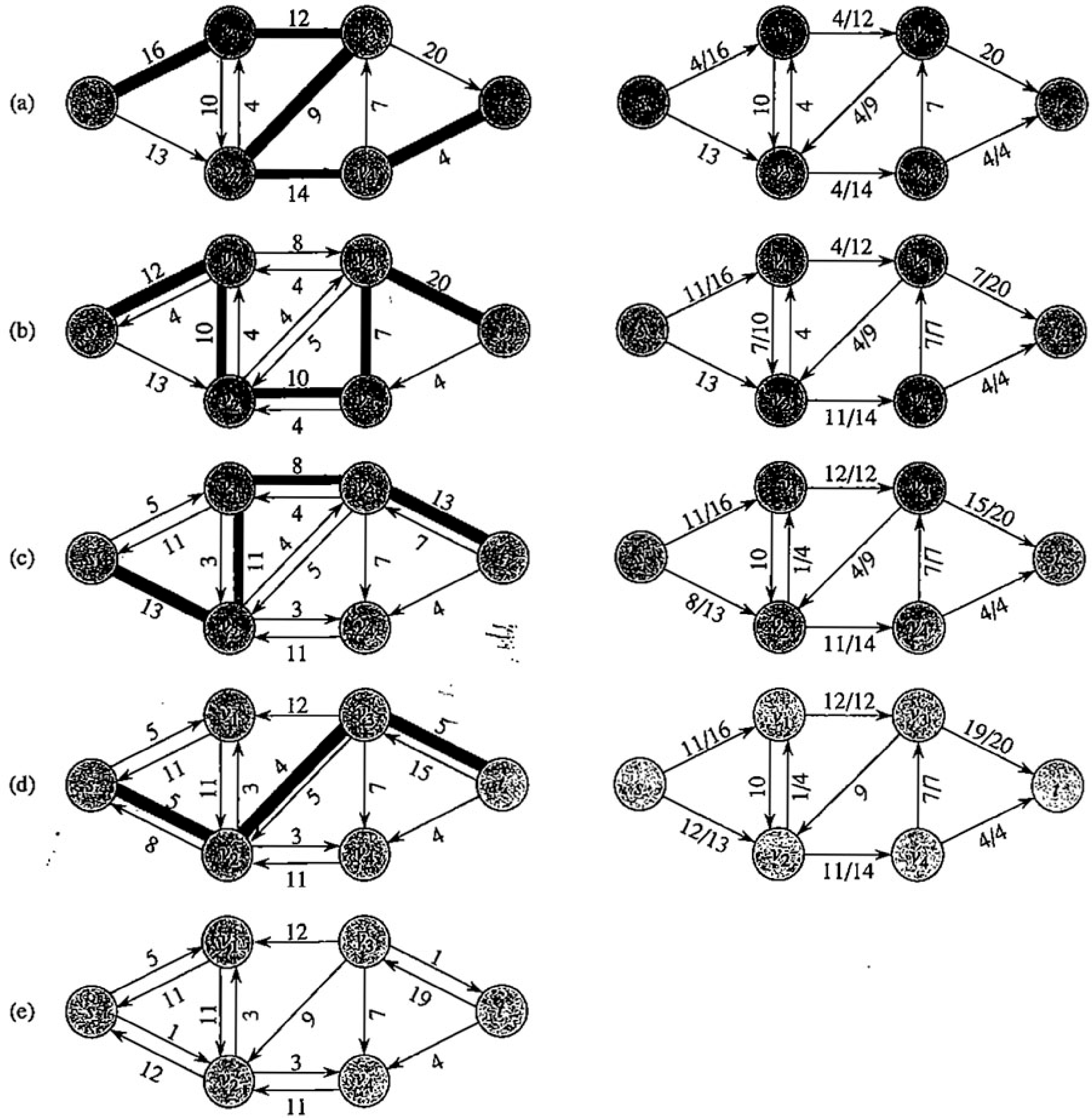
  The following conditions are equivalent:
  - f is a maximum flow in G
  - The residual network G_f contains no augmenting path
  - $|f| = c(S,T)$ for some cut (S,T) of G

Ford-Fulkerson method

- Initialize flow f=0.
- while (there exists an augmenting path p) {

        augment the flow along p

  }
- Output the final flow f


(a)
(b)
(c)
(d)
(e)

```
Ford-Fulkerson(G,s,t){
   for each edge (u,v) in E
       f(u,v)=0; f(v,u)=0;
   while ∃a path from s to t in G_f {
         c_f(p)=min{c_f(u,v) : (u,v)∈ p}
         for(each edge (u,v)∈ p) {
             f(u,v)=f(u,v)+c_f(p);
             c_f(u,v)=c(u,v)-f(u,v);
             f(v,u)=-f(u,v);
             c_f(v,u)=c(v,u)-f(v,u);
         }
   }
}
```

$O(|E||f|)$ time

Edmonds-Karp algorithm

- Ford-Fulkerson method with one change:
- When finding the augmenting path,
  - use Breadth-First search in the residual network
  - So we find the shortest augmenting path
- This method guarantees that the number of flow augmentations is $O(|E||V|)$, so the total time is $O(|E|^2 |V|)$

Observations

- If edge (u,v) is on the shortest augmenting path from s to t in G_f, then

$$d_f(s,v)=d_f(s,u)+1$$

- For each vertex u in V-{s,t}, let $d_f(s,u)$ be the distance from s to u in the residual network G_f.

  As we perform augmentations by Edmonds-Karp method, the value of $d_f(s,u)$ increases monotonically with each flow augmentation.

Application 1: Maximum (cardinality) bipartite matching problem

- Given a bipartite undirected graph G=(XUY,E), find the maximum-cardinality matching M^*.
  - A subset M of E is called Matching if no two edges in M are adjacent in G.
- Convert to Single-source-Single-target problem.
  - Generate a source s and a sink t
  - For each x in X, add edge from s to x with capacity 1.
  - For each y in Y, add edge from y to t with capacity 1.
  - For each edge xy in E, add edge from x to y with capacity 1.

Problem: Given a bipartite graph (with the partition), find a maximum matching.

Application: Matching planes to routes.

- L = set of planes.
- R = set of routes.
- $(u, v) \in E$ if plane u can fly route v.
- Want maximum # of routes to be served by planes.

Given G, define flow network $G' = (V', E')$.

- $V' = V \cup \{s, t\}$.
- $$E' = \{(s, u) : u \in L\} \cup \{(u, v) : u \in L, v \in R, (u, v) \in E\} \cup \{(v, t) : v \in R\} .$$
- $c(u, v) = 1$ for all $(u, v) \in E'$.

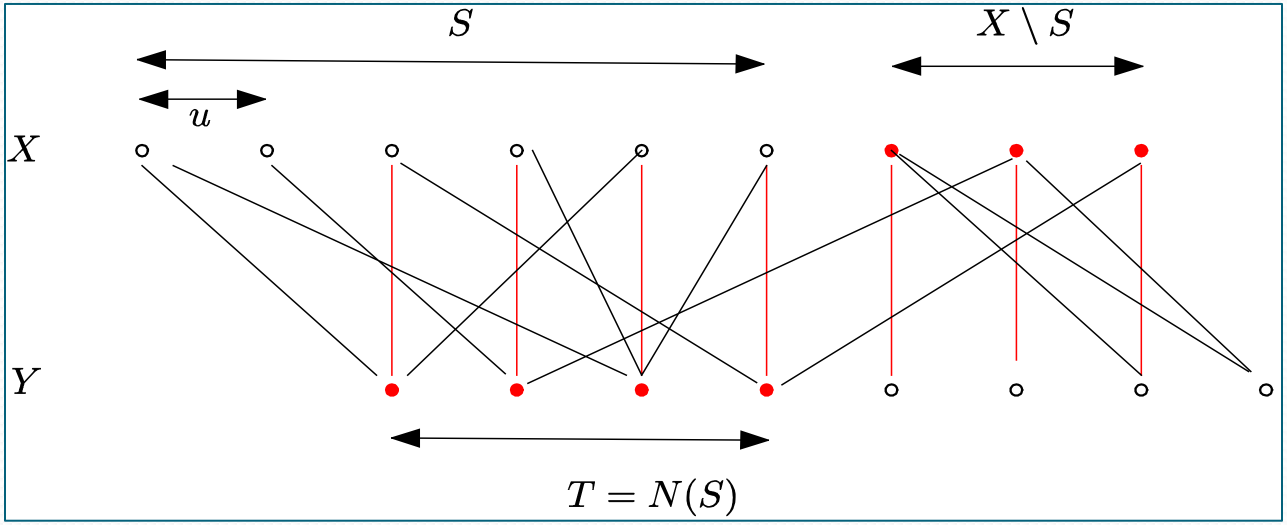
The marriage theorem

- In a bipartite undirected graph G=(XUY,E), if every vertex has degree k>0, then there exists a perfect matching M (i.e., the maximum-cardinality matching M^* satisfies that $|M^*|=|X|$).

- *If every girl in a village knows exactly k boys and every boy knows exactly k girls, then each girl can marry a boy she knows and each boy can marry a girl he knows.*

Application 2: Minimum Vertex Cover of a bipartite graph

- Given a bipartite undirected graph G=(XUY,E), find the minimum-cardinality vertex cover C^*.
  - A subset C of XUY is called a vertex cover if every edge of E has at least one end in C.
- König theorem: $|C^*|=|M^*|$



Application 3: Maximum flow with vertex capacity

- Maximum flow problem with not only edges but also vertices having maximum capacity given.
- Replace each vertex v by v’ and v”
  - every in-edges of v enter v’
  - every out-edges from v leave v”
  - Set the capacity from v’ to v” to the maximum capacity of the vertex v

Application 4: Constructing a directed graph with given degrees

- Given the in and out degrees of vertices of a directed graph, reconstruct the graph.
- A source s, a target t, and
- For each vertex v, generate
  - outvertex v' and invertex v''
  - edge from s to v' with capacity=outdegree
  - edge from v'' to t with capacity=indegree
  - edges from all v' to all v'' with capacity=1