Maximum Flow

Jie Wang

University of Massachusetts Lowell
Department of Computer Science

Flow Networks

- A *flow network* is a weighted digraph $G = (V, E)$, where the weight is called *capacity*, denoted by c, such that
  - For all $(u, v) \in E$: $c(u, v) \geq 0$.
    - For $(u, v) \notin E$: $c(u, v) = 0$.
  - If $(u, v) \in E$, then $(v, u) \notin E$.
    - Flow networks do not allow *antiparallel edges*.
    - That is, (u, v) and (v, u) cannot be both $\in E$.
  - No self loops.
  - G has one source node $s \in V$.
  - G has one sink node $t \in V$.
  - Every node $u \in V - \{s, t\}$ is on a path from s to t.
  - Note that for each node $v \in V - \{s, t\}$, it's possible to have $(s, v) \in E$ or $(v, s) \in E$, but not both.

Flow Network Constraints

- A flow is a function $f : V \times V \to R$ with the following constraints:
  - **Capacity constraint**: For all $u, v \in V : 0 \leq f(u, v) \leq c(u, v)$.
  - **Flow conservation**: For all $u \in V \setminus \{s, t\}$,

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

    If $(u, v) \notin E$, then $f(u, v) = 0$.
- The *value of a flow*, denoted by $|f|$, is defined as follows:

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

Note that the second summation is 0; but we keep it here for later use when dealing with residual networks.

Flow Network Example

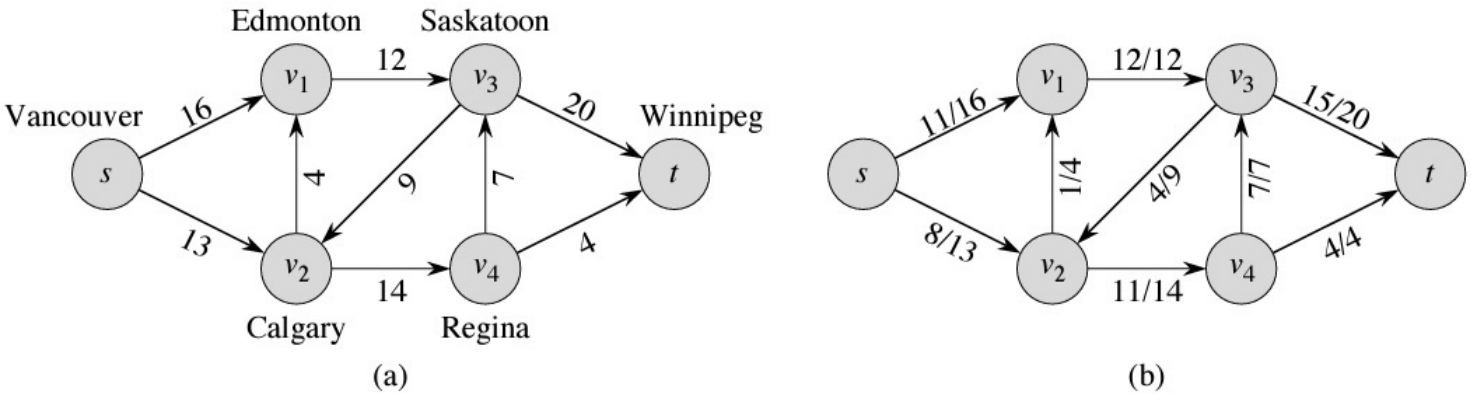


Figure 26.1 **(a)** A flow network $G = (V, E)$ for the Lucky Puck Company's trucking problem. The Vancouver factory is the source s, and the Winnipeg warehouse is the sink t. The company ships pucks through intermediate cities, but only $c(u, v)$ crates per day can go from city u to city v. Each edge is labeled with its capacity. **(b)** A flow f in G with value $|f| = 19$. Each edge (u, v) is labeled by $f(u, v)/c(u, v)$. The slash notation merely separates the flow and capacity; it does not indicate division.

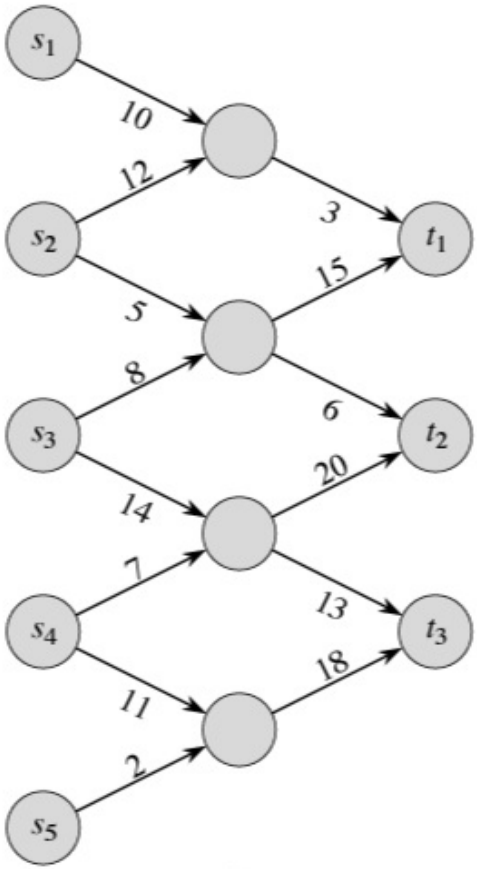
Maximum Flow

Input: A flow network $G = (V, E)$, a source $s \in V$, a sink $t \in V$, and a flow function $f : V \times V \to R$.
Output: $\max_f \{|f| \mid f \text{ is a flow of } G \text{ from } s \text{ to } t\}$ (namely, the value of the maximum flow in G).

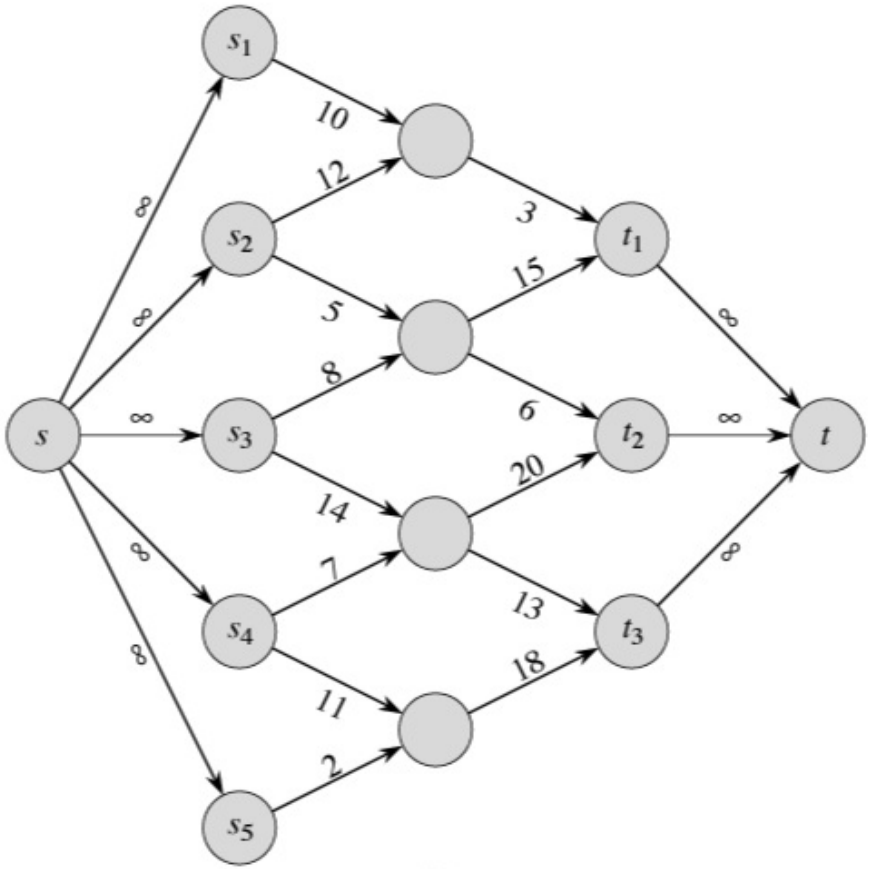
Network Flow Applications include

- Liquid flowing through pipes
- Parts through assembly lines
- Current through power grids
- Data through communication networks
- Maximum bipartite matching
- Minimum cuts

Handling Multiple Sources and Sinks



(a)



(b)

Ford-Fulkerson Method

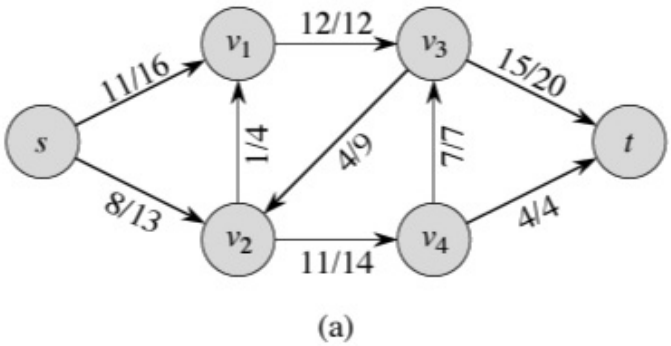
- **Residual networks**. A residual network of (G, f), denoted by $G_f = (V, E_f)$, is a flow network with positive **residual capacity** c_f, where for each $u, v \in V$:

$$c_f(u,v) = \begin{cases} c(u,v) - f(u,v), & \text{if } (u,v) \in E, \\ f(v,u), & \text{if } (v,u) \in E, \\ 0, \text{otherwise.} \end{cases}$$

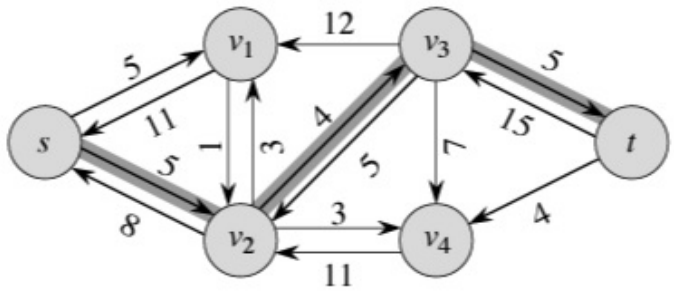
  Thus, $E_f = \{(u,v) \in V \times V \mid c_f(u,v) > 0\}$.
- **Augmenting paths**. An augmenting path is a path from s to t in a residual network G_f.
  - The flow allowed on an augmenting path is the minimum residual capacity on the path, denoted by f'.
  - The augmentation of flow f by f', denoted by $f \uparrow f'$, is defined as follows:

$$f \uparrow f'(u,v) = \begin{cases} f(u,v) + f'(u,v) - f'(v,u), & \text{if } (u,v) \in E, \\ 0, & \text{otherwise.} \end{cases}$$

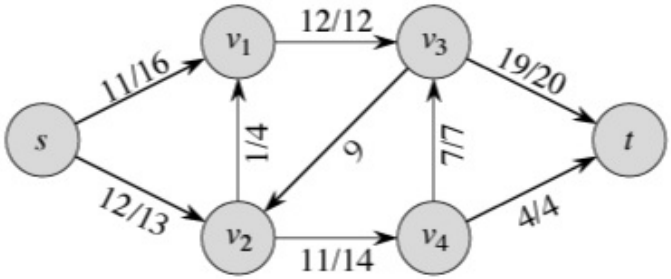
Residual Network and Augmenting Path Example



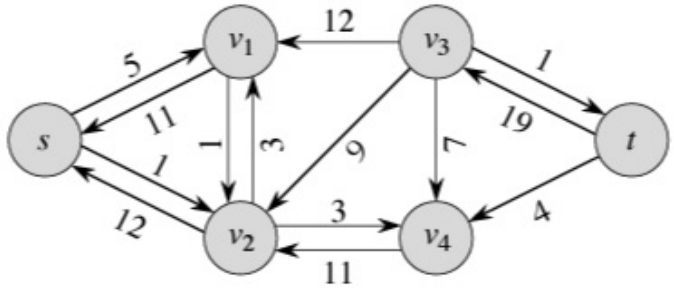
(a)



(b)



(c)



(d)

Augmentation Satisfies Capacity Constraint

- Want to show for $(u, v) \in E$:

$$0 \leq f \uparrow f'(u, v) \leq c(u, v).$$

Since $(v, u) \notin E$ and $(u, v) \in E$, $c_f(v, u) = f(u, v)$. Thus, $f'(v, u) \leq c_f(v, u) = f(u, v)$. Hence, for each $(u, v) \in E$:

$$\begin{aligned} f \uparrow f'(u, v) &= f(u, v) + f'(u, v) - f'(v, u) \\ &\geq f(u, v) + f'(u, v) - f(u, v) \\ &= f'(u, v) \\ &\geq 0. \end{aligned}$$

$$\begin{aligned} f \uparrow f'(u, v) &= f(u, v) + f'(u, v) - f'(v, u) \\ &\leq f'(u, v) + f'(u, v) \\ &\leq f'(u, v) + c_f(u, v) \\ &= f(u, v) + c(u, v) - f(u, v) \\ &= c(u, v). \end{aligned}$$

Augmentation Satisfies Flow Conservation

For each $u \in V - \{s, t\}$:

$$\begin{aligned}
\sum_{v \in V} f \uparrow f'(u, v) &= \sum_{v \in V} (f(u, v) + f'(u, v) - f'(v, u)) \\
&= \sum_{v \in V} f(u, v) + \sum_{v \in V} f'(u, v) - \sum_{v \in V} f'(v, u) \\
&= \sum_{v \in V} f(v, u) + \sum_{v \in V} f'(v, u) - \sum_{v \in V} f'(u, v) \\
&= \sum_{v \in V} f \uparrow f'(v, u).
\end{aligned}$$

Augmentation Increases Flow

We want to compute the value of $|f \uparrow f'|$ for the original network G. Recall that it's possible to have $(s, v) \in E$ or $(v, s) \in E$, but not both. Let

$$V_1 = \{v \mid (s, v) \in E\}, \ V_2 = \{v \mid (v, s) \in E\}.$$

Then $V_1 \cap V_2 = \emptyset$ and $V_1 \cup V_2 \subseteq V$.

$$\begin{aligned}
|f \uparrow f'| &= \sum_{v \in V} f \uparrow f'(s, v) - \sum_{v \in V} f \uparrow f'(v, s) \\
&= \sum_{v \in V_1} f \uparrow f'(s, v) - \sum_{v \in V_2} f \uparrow f'(v, s) \\
&= \sum_{v \in V_1} (f(s, v) + f'(s, v) - f'(v, s)) - \sum_{v \in V_2} (f(v, s) + f'(v, s) - f'(s, v)) \\
&= \sum_{v \in V_1} f(s, v) - \sum_{v \in V_2} f(v, s) + \sum_{v \in V_1 \cup V_2} (f'(s, v) - f'(v, s)) \\
&= \sum_{v \in V} (f(s, v) - f(v, s)) + \sum_{v \in V} (f'(s, v) - f'(v, s)) \\
&= |f| + |f'|.
\end{aligned}$$

Ford-Fulkerson Method Continued

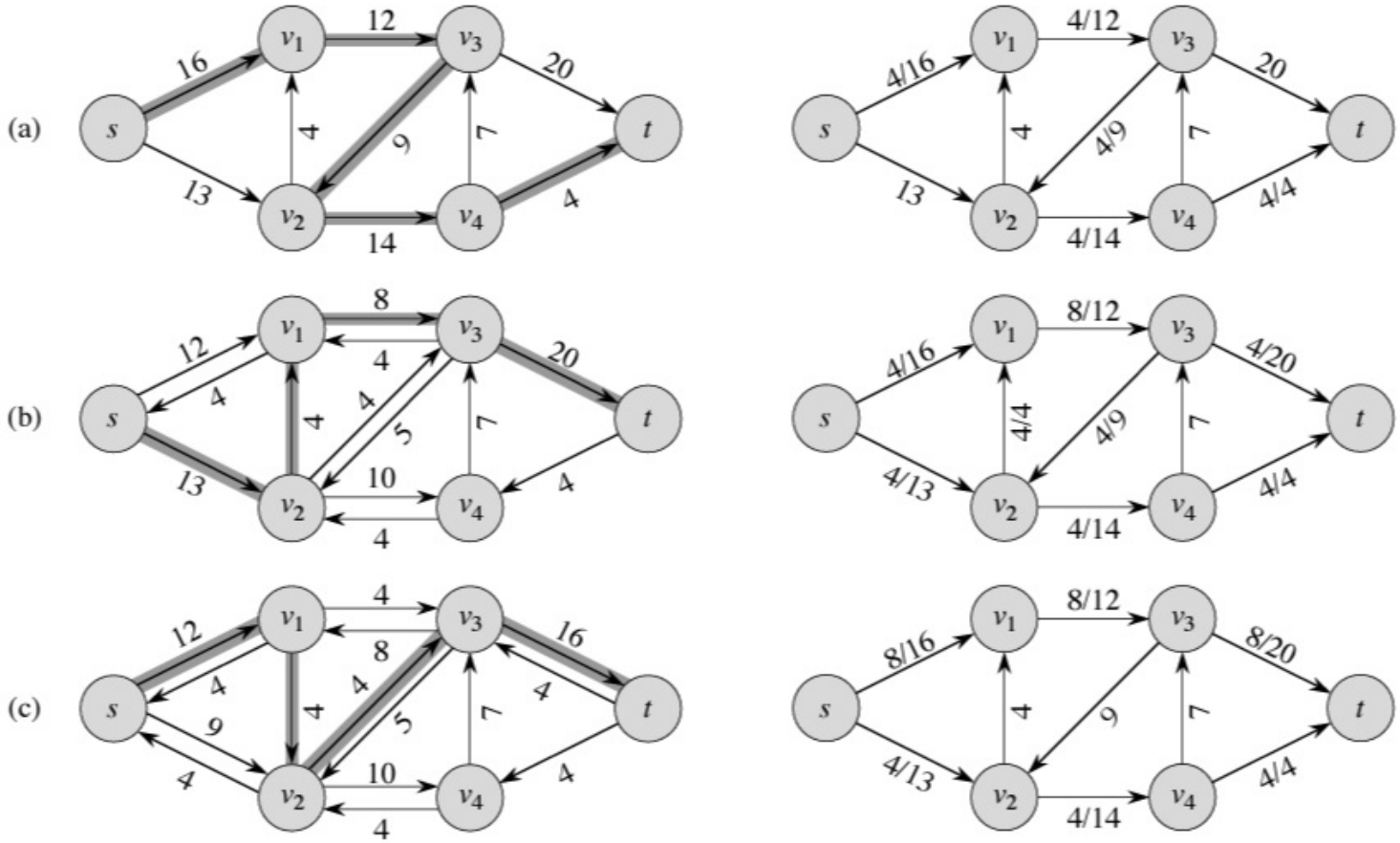
1. Initialize flow f to 0.
2. While there still exists an augmenting path p in the residual network G_f, augment flow f along p.

FORD-FULKERSON(G, s, t)

```
1  for each edge (u, v) ∈ E
2      f(u, v) = 0
3  while there is an augmenting path p from s to t in G_f
4      x = min{c_f(u, v) | (u, v) ∈ p}
5      for each edge (u, v) ∈ p
6          y = f(u, v) − f(v, u) + x
7          f(u, v) = max{0, y}
8          f(v, u) = max{0, −y}
```

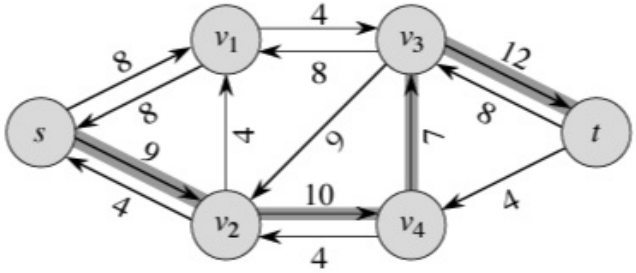
Every path from s to t is now at full capacity and thus a maximum flow is found. Will prove this result later.

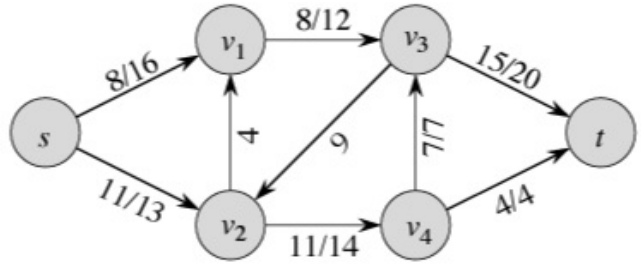
Ford-Fulkerson Method Example



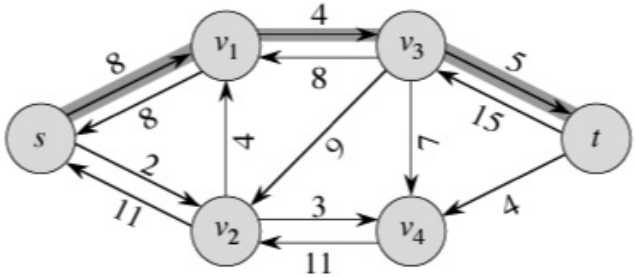
Ford-Fulkerson Method Example Continued

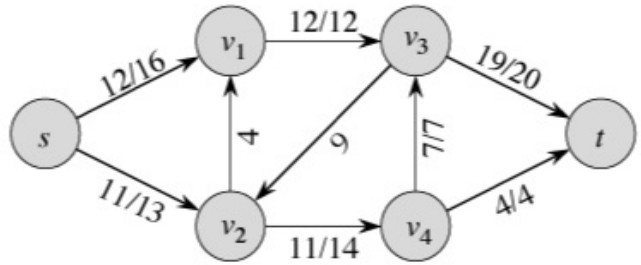
(d)



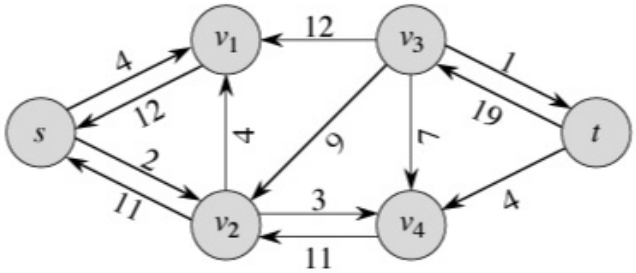


(e)





(f)



Time Complexity of Ford-Fulkerson

- First, we need to use a data structure to store the network, which is a special graph $G' = (V, E')$ with

$$E' = \{(u, v) \mid (u, v) \in E \text{ or } (v, u) \in E\}.$$

  Since all edges in G are also in G', the flows and capacities can be stored as attributes on the edges. We can use adjacency list representation for this purpose.
- The first for loop runs in $\Theta(|E|)$ time.
- The while loop starts by finding an augmenting path.
  - Let's assume that the capacity function is $c : V \times V \to Z$.
  - If the maximum flow is f^* then the while loop runs at most $|f^*|$ times, since the flow increases by at least one each time.
  - The residual network for any flow f is part of G', for it is given by all the edges $(u, v) \in E'$ such that $c_f(u, v) > 0$.
  - We can either use BFS or DFS to find an augmenting path. Either one runs in time $O(|V| + |E'|)$ if G' is implemented using an adjacency list representation.

Time Complexity of Ford-Fulkerson Continued

- The runtime using BFS or DFS is $O(|f^*|(|V| + |E'|)$ if the capacities are integrals. This is a pseudo-polynomial-time algorithm.
- In practice, capacities may be given as rational numbers. We can scale all the capacities to integrals.
- We can improve the runtime of Ford-Fulkerson to polynomial time using BFS to find the shortest augmented path from s to t. This implementation is called the Edmonds-Karp algorithm.
- The runtime of Edmonds-Karp algorithm is $O(|V||E|^2)$, independent of the value of the maximum flow.

Proof of Edmonds-Karp Runtime

Lemma 26.7. The shortest-path distance $\delta_f(s, v)$ in G_f increases monotonically with each flow augmentation, where $v \in V - \{s, t\}$.

Proof. By contradiction.

- Let f be the flow just before the first augmentation that decreases some shortest-path distance.
- Let f' be the flow just afterward.
- Let v be the closest node to s on a shortest path p with $\delta_f(s, v) > \delta_{f'}(s, v)$.
- Let $p = s \rightsquigarrow u \to v$ be a shortest path from s to v in $G_{f'}$ so that

$$(u, v) \in E_{f'},$$
$$\delta_{f'}(s, u) = \delta_{f'}(s, v) - 1.$$

- Because of how v is chosen, we have $\delta_{f'}(s, u) \geq \delta_f(s, u)$.

Proof Continued

- Claim: $(u, v) \notin E_f$. If $(u, v) \in E_f$, then

$$\begin{aligned}\delta_f(s, v) &\leq \delta_f(s, u) + 1 \text{ (by triangle inequality)} \\ &\leq \delta_{f'}(s, u) + 1 \\ &= \delta_{f'}(s, v),\end{aligned}$$

  contradicting to the assumption that $\delta_f(s, v) > \delta_{f'}(s, v)$.
- To have $(u, v) \in E_{f'}$ but $(u, v) \notin E_f$, the augmentation must increase the flow from v to u.
- Edmonds-Karp always augments flow along shortest paths. Thus, the shortest path from $s \rightsquigarrow u \in G_f$ must have $s \rightsquigarrow v \to u$. Hence,

$$\begin{aligned}\delta_f(s, v) &= \delta_f(s, u) - 1 \\ &\leq \delta_{f'}(s, u) - 1 \\ &= \delta_{f'}(s, v) - 2,\end{aligned}$$

  Contradicting to the assumption that $\delta_f(s, v) > \delta_{f'}(s, v)$.
- This completes the proof.

Proof Continued

Theorem 26.8. Edmonds-Karp performs $O(|V||E|)$ flow augmentations.
Proof. An edge $(u, v) \in G_f$ is **critical** on an augmenting path p if $c_f(u, v) = \min\{c_f(e) \mid e \text{ is an edge of } p\}$.

- Each edge $e = (u, v) \in E$ can become critical at most $|V|/2$ times. A proof is presented below.
  - When e is critical for the first time, we have
  $$\delta_f(s, v) = \delta_f(s, u) + 1.$$
  - Once the flow is augmented, e disappears from E_f.
  - e cannot reappear until after (v, u) appears on an augmenting path.
  - Let f' be the flow when this happens. Then
  $$\delta_{f'}(s, u) = \delta_{f'}(s, v) + 1.$$
  - Since $\delta_{f'}(s, v) \geq \delta_f(s, v)$ (Lemma 26.7), we have
  $$\begin{aligned}\delta_{f'}(s, u) &\geq \delta_f(s, v) + 1 \\ &= \delta_f(s, u) + 2.\end{aligned}$$

Proof Continued

- In a cycle that (u, v) becomes critical, the distance from s to u increases by at least 2.
- The distance from s to u is initially at least 0.
- The intermediate nodes on a shortest path from s to u cannot contain s, u, or t.
- The distance for any shortest path from s to u is at most $|V| - 2$.
- Thus, after (u, v) becomes critical, it can become critical at most $(|V| - 2)/2 = |V|/2 - 1$ times more. (Reason: Let k be the number of times that it can become critical, then we must have $2k = |V| - 2$.)
- There are $O(|E|)$ edges in a residual network.
- The total number of critical edges is $O(|V||E|)$.
- Each augmenting path has at least one critical edge. This completes the proof.

Cuts

- The node V in a flow network can be partitioned into a cut (S, T), where $s \in S$, $t \notin S$, and $T = V - S$.
- The capacity of a cut (S, T) is defined by

$$c(S, T) = \sum_{u \in S, v \in T} c(u, v).$$

- Let (S, T) be any cut of G and f a flow. Then the flow from S to T, denoted by $f(S, T)$, is always equal to $|f|$.

$$\begin{aligned}
|f| &= \sum_{v \in V} f(s, v) - \sum_{v \in V} f(v, s) + \sum_{u \in V - \{s\}} \left(\sum_{v \in V} f(u, v) - \sum_{v \in V} f(v, u) \right) \\
&= \sum_{v \in V} \left(f(s, v) + \sum_{u \in S - \{s\}} f(u, v) \right) - \sum_{v \in V} \left(f(v, s) + \sum_{u \in S - \{s\}} f(v, u) \right) \\
&= \sum_{v \in V} \sum_{u \in S} f(u, v) - \sum_{v \in V} \sum_{u \in S} f(v, u).
\end{aligned}$$

Proof Continued

- Since $V = S \cup T$ and $S \cap T = \emptyset$, split each summation over V into summations over S and T to get

$$\begin{aligned} |f| &= \sum_{u \in S} \sum_{v \in T} f(u, v) - \sum_{u \in S} \sum_{v \in T} f(v, u) \\ &= f(S, T). \end{aligned}$$

- $f(S, T) \leq c(S, T)$ for the following reason:

$$\begin{aligned} f(S, T) &= \sum_{u \in S} \sum_{v \in T} f(u, v) - \sum_{u \in S} \sum_{v \in T} f(v, u) \\ &\leq \sum_{u \in S} \sum_{v \in T} f(u, v) \\ &\leq \sum_{u \in S} \sum_{v \in T} c(u, v) \\ &= c(S, T). \end{aligned}$$

Max-Flow Min-Cut Theorem

Theorem 26.6. (1) f is a maximum flow in G iff (2) G_f contains no augmenting paths iff (3) $|f| = c(S, T)$ for some cut (S, T) of G (this implies that this (S, T) is a minimum cut).
Proof. That (1) implies (2) is straightforward.
To show that (2) implies (3), let

$$S = \{v \in V \mid \text{there exists a path from } s \text{ to } t \text{ in } G_f\}.$$

Let $T = V - S$. Then (S, T) is a cut because $t \notin S$.

- Let $u \in S$ and $v \in T$. If $(u, v) \in E$, then $f(u, v) = c(u, v)$. This is because if $f(u, v) < c(u, v)$, then $(u, v) \in E_f$, and so $v \in S$, a contradiction.
- If $(u, v) \notin E$, then $f(u, v) = 0$. This is because if $f(u, v) > 0$, then $c_f(u, v) = f(v, u) > 0$, and so $(u, v) \in E_f$, contradicting to $v \notin S$.

Proof Continued

- $f(u, v) = 0$ for other cases. Thus,

$$\begin{aligned} |f| &= f(S, T) \\ &= \sum_{u \in S} \sum_{v \in T} f(u, v) - \sum_{v \in T} \sum_{u \in S} f(v, u) \\ &= \sum_{u \in S} \sum_{v \in T} c(u, v) - \sum_{v \in T} \sum_{u \in S} 0 \\ &= C(S, T). \end{aligned}$$

To show that (3) implies (1), it suffices to note that $|f| = f(S, T) \leq c(S, T)$.

Maximum Bipartite Matching

- Sometimes a problem that appears to have **nothing** to do with maximum flow can be recast into a maximum flow problem.
- Given an undirected graph $G = (V, E)$, a matching is a subset of edges such that no two edges share a common node.
- A matching with the largest number of edges is a maximum matching.
- Let $G = (L \cup R, E)$ be an undirected bipartite graph. We want to find a maximum matching.
  - Construct a flow network by connecting a new node s to all nodes in L and connecting all nodes in R to another new node t.
  - Assign each edge in the new network a capacity of 1.
  - Then the maximum flow of the network corresponds to the maximum matching.

Maximum Bipartite Matching Example

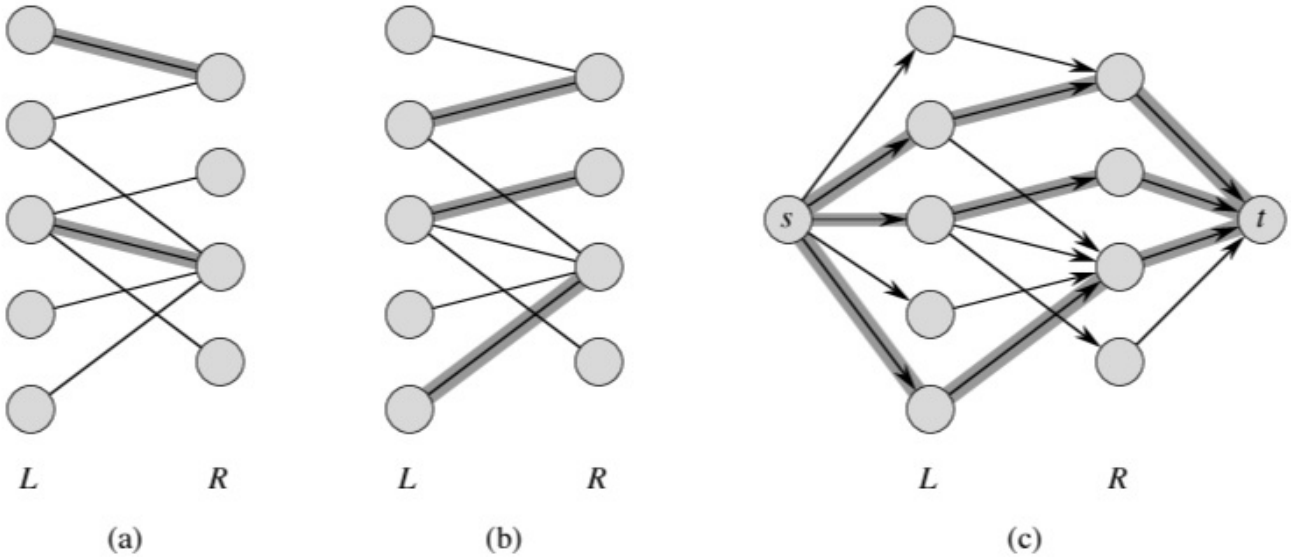


Figure 26.8 A bipartite graph $G = (V, E)$ with vertex partition $V = L \cup R$. **(a)** A matching with cardinality 2, indicated by shaded edges. **(b)** A maximum matching with cardinality 3. **(c)** The corresponding flow network G' with a maximum flow shown. Each edge has unit capacity. Shaded edges have a flow of 1, and all other edges carry no flow. The shaded edges from L to R correspond to those in the maximum matching from (b).

An Application of Maximum Bipartite Matching

- In a local Apple store there are m sales reps working on a typical working day. At a peak hour there are n customers acquiring different types of services. A sales rep may be able to provide the service needed by certain customers. For example, sales rep Eric may be able to serve customers Alice, Bob, and Charlie. Moreover, each sales rep can only serve one customer at a time.
- Find an assignment so that maximum number of customers are served at a time.
- This is a maximum bipartite matching problem.