Maximum Flow

Chuck Cusack

These notes are based on chapter 27 of [1] and lectures from CSCE423/823, Spring 2001.

Some real-life problems, like those involving the flow of liquids through pipes, current through wires, and delivery of goods, can be modeled using flow networks. Other problems which seem unrelated to networks can also be modeled using flow networks. We discuss the basics of flow networks, several network flow problems, and develop a few algorithms to solve the maximum network flow problem.

1 Flow Networks and Flows

Definition 1 *A* **flow network** *is a directed graph $G = (V, E)$ such that*

1. *for edge $(u, v,) \in V$, we associate a nonnegative* **capacity** *$c(u, v)$;*
2. *there are two distinguished points, the* **source** *s, and the* **sink** *t;*
3. *for every vertex $v \in V$, there is a path from s to t containing v.*

Definition 2 *Let $G = (V, E)$ be a flow network. A* **flow** *in G is a real-valued function f on pairs of vertices such that the following properties hold:*

1. **Capacity constraint***: For all $u, v \in V$, $f(u, v) \leq c(u, v)$,*
2. **Skew symmetry***: For all $u, v \in V$, $f(u, v) = -f(v, u)$,*
3. **Flow conservation***: For all $u \in V - \{s, t\}$,*

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

The three properties can be described as follows.

- **Capacity constraint** makes sure that the flow through each edge is not greater than the capacity.
- **Skew symmetry** simply means that the flow from u to v is the negative of the flow from v to u.

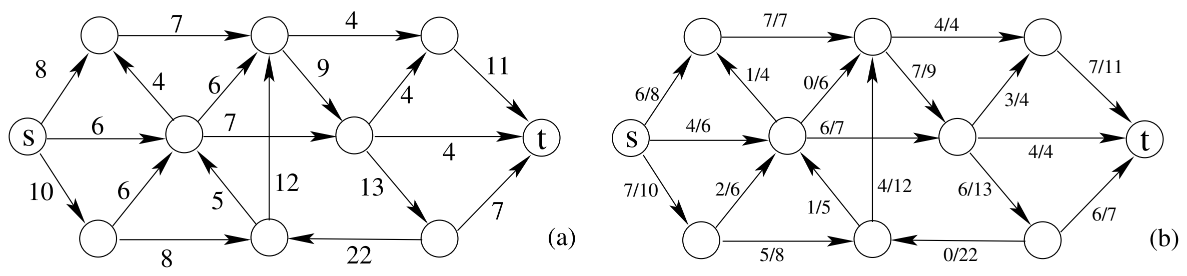

Figure 1: (a) A flow network $G = (V, E)$. (b) A flow f in G. Only the positive flows are show. Each edge is labeled with $flow/capacity$. For example, $3/4$ means the flow is 3 and the capacity is 4. The value of the flow is $|f| = 17$

- **Flow conservation** means that for every vertex $v \in V - \{s, t\}$, the net flow out of v is 0. In other words, the amount of flow into a v is the same as the amount of flow out of v.

Lemma 1 *If $(u, v) \notin E$ and $(v, u) \notin E$, then $f(u, v) = f(v, u) = 0$.*

Proof: Since

$$\begin{aligned} f(u,v) &\leq c(u,v) = 0, \\ f(v,u) &\leq c(v,u) = 0, \text{ and} \\ f(v,u) &= -f(u,v), \end{aligned}$$

the result holds. □

In other words, if there is no edge between u and v, there can be no flow between u and v.

Definition 3 *The* **value** *of the flow is the net flow from the source,*

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

Definition 4 *For each edge $(u, v) \in E$, we call $f(u, v)$ the* **net flow** *from u to v.*

Definition 5 *The* **positive net flow entering a vertex** *v is defined by*

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

The **positive net flow leaving a vertex** *is defined symmetrically.*

Definition 6 *A flow f is said to be* **integer-valued** *if $f(u, v)$ is an integer for all $(u, v) \in E$.*

Clearly the value of the flow $|f|$ is an integer in an integer-valued flow. Figure 1 shows a flow network G, and a flow f in G.

2 Implicit Sum Notation

For convenience, we will omit the set braces when it simplifies notation. For instance, we will write $V - s - t$ instead of $V - \{s, t\}$

The following notation will greatly simplify the writing of proofs involving functions like the net flow f.

Definition 7 *If X and Y are sets, and g is a 2-variable function, we define*

$$g(X, Y) = \sum_{x \in X} \sum_{y \in Y} g(x, y).$$

For example, we can express the flow conservation property as $f(u, V) = 0$ for all $u \in V - s - t$, or equivalently, $f(V, v) = 0$ for all $v \in V - s - t$. Given this, it is easy to prove the following:

Lemma 2 *If $X \subseteq V - s - t$, then $f(V, X) = f(X, V) = 0$.*

Proof: Since $X \subseteq V - s - t$,

$$f(V, X) = \sum_{x \in X} f(V, x) = \sum_{x \in X} 0 = 0.$$

The proof is similar for $f(X, V) = 0$. □

The following identities will also be useful. The proof of them is left as an exercise.

Lemma 3 *Let $G = (V, E)$ be a flow network, f a flow in G, and $X, Y, Z \subseteq V$, with $Y \cap Z = \emptyset$. Then*

1. $f(X, X) = 0$.
2. $f(X, Y) = -f(Y, X)$.
3. $f(X, Y \cup Z) = f(X, Y) + f(X, Z)$.
4. $f(Y \cup Z, X) = f(Y, X) + f(Z, X)$.

We will demonstrate the use of this lemma by proving that the value of a flow f is the total net flow into the sink.

Lemma 4 *The value of a flow is $|f| = f(V, t)$.*

Proof: We have

$$\begin{array}{rcll} |f| & = & f(s, V) & \text{(by definition)} \\ & = & f(V, V) - f(V - s, V) & \text{(by Lemma 3.3)} \\ & = & f(V, V - s) & \text{(by Lemma 3.1 and 3.2)} \\ & = & f(V, t) + f(V, V - s - t) & \text{(by Lemma 3.3)} \\ & = & f(V, t) & \text{(by Lemma 2)} \end{array}$$

□

3 Network Flow Problems

The most obvious flow network problem is the following.

Problem 1 *Given a flow network $G = (V, E)$, the* **maximum flow problem** *is to find a flow with maximum value.*

In Section 5 we will present an efficient algorithm to solve this problem. One variation of the maximum flow problem is the following.

Problem 2 *The* **multiple source and sink maximum flow problem** *is similar to the maximum flow problem, except there is a set $\{s_1, \ldots, s_m\}$ of sources and a set $\{t_1, \ldots, t_n\}$ of sinks.*

It turns out that this problem is no harder to solve than the maximum flow problem. Given a multiple source and sink flow network G, we define a new flow network G' by adding

- a **supersource** s,
- a **supersink** t,
- for each s_i, add edge (s, s_i) with capacity ∞, and
- for each t_i, add edge (t_i, t) with capacity ∞.

It is left as an exercise to prove that the maximum flows in G and G' are the same. Figure 2 shows a multiple source and sink flow network and an equivalent single source and sink flow network.

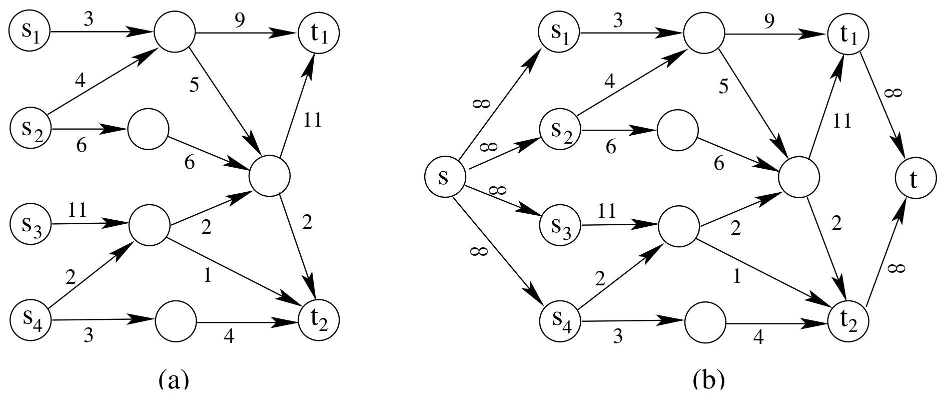

Figure 2: (a) A multiple source and sink flow network with sources $\{s_1, s_2, s_3, s_4\}$ and sinks $\{t_1, t_2\}$. (b) The equivalent flow network.

The last problem we will consider is not a network flow problem. However, we will show that it is equivalent to one. We start with a definition.

Definition 8 *A graph $G = (V, E)$ is called a* **bipartite graph** *if there exists a partition of the vertices $V = L \cup R$ such that if $u, v \in R$ or $u, v \in L$, then $(u, v) \notin E$. That is, all of the edges go from a vertex in L to a vertex in R, and there are no edges within each partition.*

Definition 9 *Let $G = (V, E)$ be a bipartite graph with partition $V = L \cup R$.*

- *A* **bipartite matching** *is a subset M of the edges of G such that for all $(u, v) \neq (u', v') \in M$, where $u, u' \in L$, and $v, v' \in R$, $u \neq u'$ and $v \neq v'$.*
- *In other words, M is a subset of the edges such that no two edges are incident with each other.*
- *Put another way, for any vertex $v \in V$, there is at most one edge containing v in the matching M.*

Problem 3 *Let $G = (V, E)$ be a bipartite graph with partition $V = L \cup R$. We wish to find a* **maximal bipartite matching** *of G. That is, a matching with the maximal number of edges.*

Given a bipartite graph $G = (V, E)$ with partition $V = L \cup R$, we can construct a flow network $G' = (V', E')$ as follows:

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

Figure 3 shows a bipartite graph G and the corresponding flow network G'.

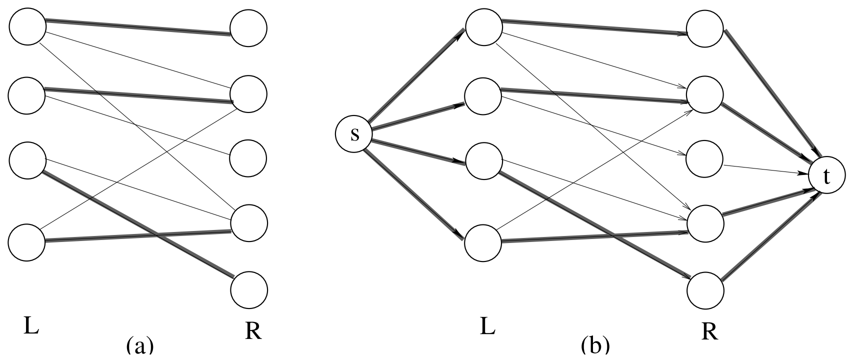

Figure 3: (a) A bipartite graph $G = (V, E)$, with partition $V = L \cup R$. A maximal matching M is shown with the grey edges. (b) A flow network G' corresponding to G. Each edge has capacity 1. The grey edges have flow 1, and the others have flow 0.

The following theorem shows the relationship between flows and matchings.

Theorem 5 *Let $G = (V, E)$ be a bipartite graph with partition $V = L \cup R$, and G' be the corresponding flow network. Then each matching M in G corresponds to an integer-valued flow f in G' with $|f| = |M|$.*

Proof: Let M be a matching in G. Define an integer-valued flow f as follows:

- $f(u, v) = f(s, u) = f(v, t) = 1$, for each edge $(u, v) \in M$,

- $f(v,u) = f(u,s) = f(t,v) = -1$, for each edge $(u,v) \in M$,
- $f(u,v) = 0$ for each edge $(u,v) \notin M$.

To see that f is indeed a flow, we need to prove that the three properties are satisfied.

1. Since $f(u,v) \leq 1 = c(u,v)$ for all $(u,v) \in E'$, the capacity constraint is satisfied.
2. It is not hard to see that $f(u,v) = -f(v,u)$ for all u and v, so skew symmetry is satisfied.
3. Let $u \in V - \{s,t\}$.
   - If u is not in an edge of the matching M, then $f(u,V) = 0$, since no flow goes through u.
   - Otherwise, u is in exactly one edge of M.
   - If $u \in L$ and $(u,v) \in M$, then $f(u,s) = -1$, $f(u,v) = 1$, and $f(u,w) = 0$ if $w \in V - \{s,v\}$, so $f(u,V) = 0$.
   - A similar argument shows that if $u \in R$, then $f(u,V) = 0$.
   - Therefore flow conservation is satisfied.

Since there are $|M|$ edges in the matching, there are $|M|$ edges (s,u) such that $f(s,u) = 1$, so

$$|f| = \sum_{u \in V} f(s,u) = |M|.$$

Conversely, let f be an integer-valued flow in G'. Define

$$M = \{(u,v) : u \in L, v \in R, \text{ and } f(u,v) > 0\}.$$

We will show that M is a matching.

- The only edge entering each vertex $u \in L$ is (s,u), which has capacity 1. Therefore, the net flow into u is at most 1.
- In light of this, the net flow out of vertex u is at most 1.
- Since f is integer valued, the net flow into and out of vertex u is 1 if and only if there is a vertex $v \in R$ such that $f(u,v) = 1$.
- Thus, at most one edge leaving u carries weight.
- A similar argument proves the same for edges entering $v \in R$.
- Thus, M is a matching.

To see that $|f| = |M|$, notice that:

$$\begin{array}{rcll} |M| & = & f(L,R) & \text{(by definition)} \\ & = & f(L,V') - f(L,L) - f(L,s) - f(L,t) & \text{(by splitting up the sum)} \\ & = & 0 - 0 + f(s,L) - 0 & \text{(by conservation)} \\ & = & f(s,L) + f(s,R+t+s) & \text{(no flow from } s \text{ to } R \cup \{t\} \cup \{s\}) \\ & = & f(s,V') & \text{(by Lemma 3)} \\ & = & |f| & \text{(by definition)} \end{array}$$

□

In particular, this means that every maximum bipartite matching in G corresponds to a maximum integer-valued flow in G'. Thus, a solution to the maximum flow network problem for G' should give us a solution to the maximum bipartite matching problem for G. The problem with this is that the algorithm we use to compute a maximum flow of G' might not produce an integer-valued flow. It turns out that it is not a problem if we use the right algorithm.

Theorem 6 *There exists an algorithm (the* FORD-FULKERSON METHOD*) to solve the maximum flow problem such that if the capacities c are integers, then the resulting maximum flow f will be integer-valued.*

Proof: We will explore the FORD-FULKERSON METHOD in Section 5. The proof is then left as an exercise. □

Corollary 7 *Let $G = (V,E)$ be a bipartite graph and G' be the corresponding flow network. Then the cardinality of a maximum matching of G is the value of a maximum flow in G'.*

Proof: Left as an exercise. □

In light of these results, we can solve the maximum bipartite matching problem by solving the equivalent maximum network flow problem, as long as we use an algorithm such as the FORD-FULKERSON METHOD to guarantee the flow to be integer-valued.

4 Residual Networks, Augmenting Paths, and Cuts

Definition 10 *Let $G = (V,E)$ be a graph with flow f. For each pair of vertices $u, v \in V$, we define the* **residual capacity** *of (u,v) by*

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

That is, it is the additional net flow that can be pushed from u to v.

Definition 11 *Given a flow network $G = (V,E)$ with flow f, the* **residual network** *of G induced by f is $G_f = (V, E_f)$, where*

$$E_f = \{(u,v) \in V \times V : c_f(u,v) \geq 0\}.$$

An edge in E_f is called a **residual edge**.

Think of the residual network as the network of edges through which more flow can be pushed. Notice that an edge does not need to be present in G to be present in G_f, although if edge $(u,v) \in E_f$, then at least one of (u,v) or (v,u) is in E.

Since the residual network G_f is a flow network, we can find a flow f_r in G_f. As Lemma 8 shows, we can add the flows f and f_r to obtain another flow in G.

Lemma 8 *Let $G = (V,E)$ be a flow network with flow f, G_f the residual network induced by f, and f_r a flow in G_f. Let $f' = f + f_r$. That is, for each pair $u,v \in V$, $f'(u,v) = f(u,v) + f_r(u,v)$. Then f' is a flow in G with value $|f'| = |f| + |f_r|$.*

Proof: We need to verify that the 3 properties of a flow are satisfied.

1. By definition,

$$f_r(u,v) \le c_f(u,v) = c(u,v) - f(u,v) \text{ for all } u,v \in V.$$

Given this, we can demonstrate capacity constraint as follows.

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

2. To see that f' has skew symmetry, notice that for $u,v \in V$,

$$\begin{aligned} f'(u,v) = f(u,v) + f_r(u,v) &= -f(v,u) - f_r(v,u) \\ &= -(f(v,u) + f_r(v,u)) \\ &= -f'(v,u) \end{aligned}$$

3. f' has flow conservation, since for all $u \in V - s - t$,

$$\begin{aligned} \sum_{v \in V} f'(u,v) &= \sum_{v \in V} (f(u,v) + f_r(u,v)) \\ &= \sum_{v \in V} f(u,v) + \sum_{v \in V} f_r(u,v) \\ &= 0 + 0 = 0 \end{aligned}$$

Finally, we can easily see that

$$\begin{aligned} |f'| &= \sum_{v \in V} f'(s,v) \\ &= \sum_{v \in V} f(s,v) + f_r(s,v) \\ &= \sum_{v \in V} f(s,v) + \sum_{v \in V} f_r(s,v) \\ &= |f| + |f_r|. \end{aligned}$$

□

Definition 12 *Let $G = (V, E)$ be a flow network with flow f. An* **augmenting path** *p is a simple path from s to t in the residual network G_f.*

Definition 13 *Let $G = (V, E)$ be a flow network with flow f. The* **residual capacity** *of an augmenting path p is*

$$c_f(p) = \min\{c_f(u, v) : (u, v) \in p\}$$

The residual capacity is the maximal amount of flow that can be pushed through the augmenting path p. Notice that if there is an augmenting path, then each edge on it has positive capacity. We will use this fact to compute a maximum flow in a flow network.
Figure 4 demonstrates these concepts.

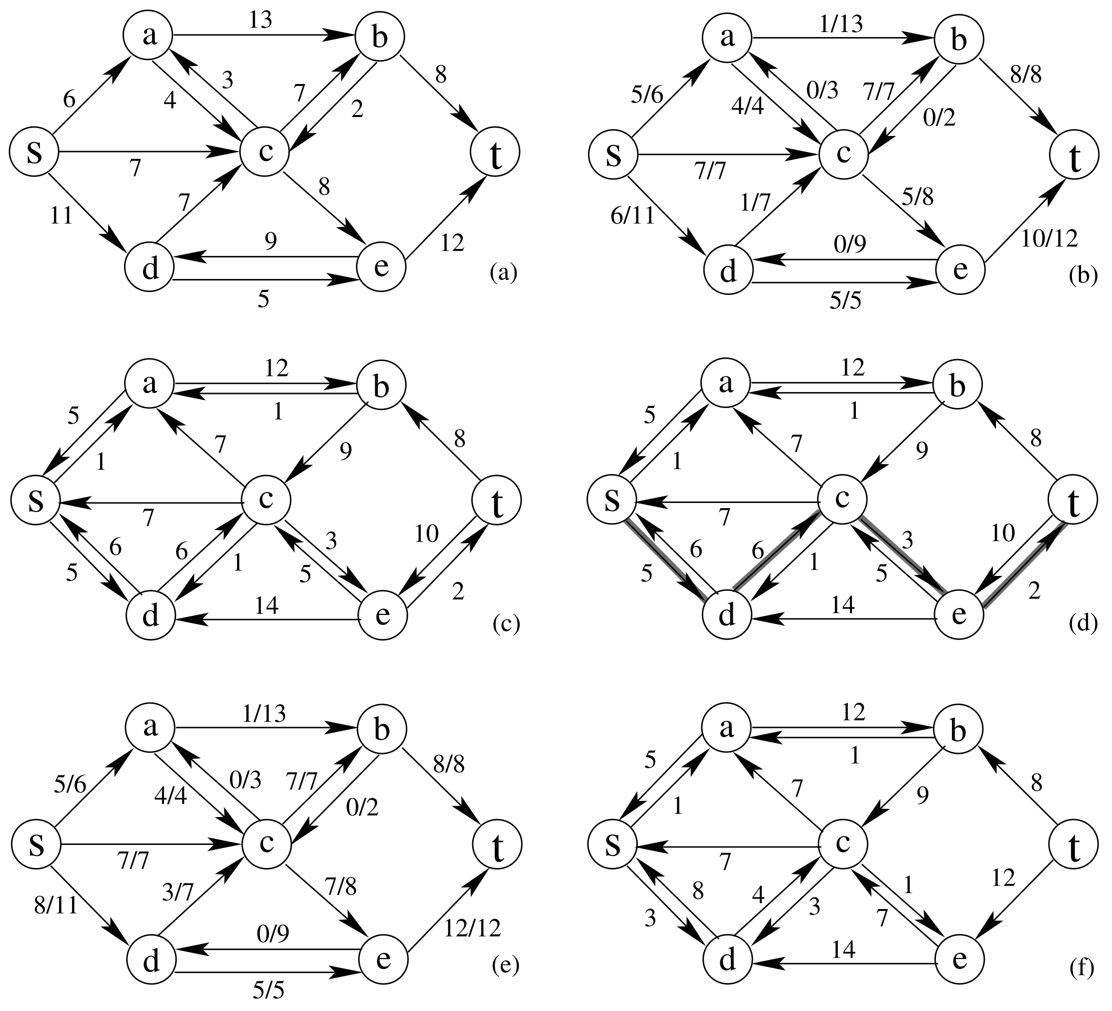

Figure 4: (a) A flow network $G = (V, E)$. (b) A flow f in G. (c) The residual network G_f. (d) The grey edges form an augmenting path with capacity 2. (e) A new flow $f' = f + f_p$. (f) The residual network $G_{f'}$.

The following is not hard to prove.

Lemma 9 *Let $G = (V, E)$ be a flow network, f a flow in G, and p an augmenting path in G_f. Define f_p by*

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

Then f_p is a flow in G_f with value $|f_p| = c_f(p) > 0$.

Proof: Left as an exercise. □

Lemmas 8 and 9 lead immediately to the following.

Corollary 10 *Let $G = (V, E)$ be a flow network, f a flow in G, and p an augmenting path in G_f. Then $f' = f + f_p$ is a flow in G with value $|f'| = |f| + |f_p| > |f|$.* □

Definition 14 *Let $G = (V, E)$ be a flow network with flow f.*

- *A* **cut** *(S, T) is a partition of V into S and $T = V - S$ such that $s \in S$ and $t \in T$.*
- *The* **net flow** *across the cut (S, T) is $f(S, T)$.*
- *The* **capacity** *of the cut is $c(S, T)$.*

For example, consider the flow network $G = (V, E)$ with flow f from Figure 4(b). Let $(\{s, a, b, c, d\}, \{e, t\})$ be a cut in G. Then the flow across the cut is

$$f(b,t) + f(c,e) + f(d,e) + f(e,d) = 8 + 5 + 5 + 0 = 18,$$

and the capacity is

$$f(b,t) + f(c,e) + f(d,e) = 8 + 8 + 5 = 21.$$

Lemma 11 *Let $G = (V, E)$ be a flow network, and f a flow in G. Let (S, T) be a cut of G. Then $f(S, T) = |f|$.*

Proof: It is straightforward to see that

$$\begin{aligned} f(S,T) &= f(S,V) - f(S,S) \\ &= f(S,V) \\ &= f(S-s,V) + f(s,V) \\ &= f(s,V) \\ &= |f| \end{aligned}$$

□

Corollary 12 *Let $G = (V, E)$ be a flow network. Then the value of any flow in G is bounded above by the capacity of any cut.*

Proof: Let f be any flow of G, and (S,T) be any cut. Using Lemma 11, we can show that

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

□

Finally, we will prove the **max-flow min-cut theorem**.

Theorem 13 *Let $G=(V,E)$ be a flow network with flow f. The the following statements are equivalent:*

1. *f is a maximum flow in G.*
2. *G_f has no augmenting paths.*
3. *$|f| = c(S,T)$ for some cut (S,T) of G.*

Proof:

$1 \Rightarrow 2$ Assume that f is a maximum flow, but G_f has an augmenting path p. By Corollary 10, $f' = f + f_p$ is a flow in G with $|f'| > |f|$, contradicting that f was a maximum flow.

$2 \Rightarrow 3$
- Assume that G_f has no augmenting path.
- Define

$$\begin{aligned}
S &= \{v\in V : \text{there exists a path from } s \text{ to } v \text{ in } G_f\}, \text{ and} \\
T &= V - S.
\end{aligned}$$

- Clearly $s \in S$.
- Since there is no augmenting path, there is no path from s to t.
- Thus $t \in T$.
- Therefore (S,T) is a cut.
- By construction, when $u\in S$ and $v \in T$, $(u,v)\notin E_f$, so $f(u,v) = c(u,v)$.
- Thus, $f(S,T) = c(S,T)$.
- By lemma 11, $|f| = f(S,T) = c(S,T)$.

$3 \Rightarrow 1$ By Corollary 12, $|f| \le c(S,T)$ for any cut (S,T). Thus, is $|f| = c(S,T)$, f must be a maximum cut. □

5 The Ford-Fulkerson method

The FORD-FULKERSON METHOD computes a maximum flow in a flow network. It works by repeatedly finding augmenting paths and adding them to the flow until there are no more augmenting paths. More specifically, the algorithm is as follows.

```
FORD-FULKERSON($G$)
1   For each edge $(u, v) \in E$
2       $f(u, v) = 0$
3       $f(v, u) = 0$
4   While there exists an augmenting path in $G_f$
5       Let $C = \min\{c_f(u, v) : (u, v) \in p\}$
6       For each edge $(u, v) \in p$
7           $f(u, v) = f(u, v) + C$
8           $f(v, u) = -f(v, u)$
```

Notice that the FORD-FULKERSON METHOD does not specify *how* the augmenting paths are chosen. Depending on how this is handled, the algorithm can perform very differently. Since most network-flow problems have integer capacities, or can be converted to integer capacities, our discussion and analysis will assume this.

Let $|E| = e$ and $|V| = v$. Notice that lines 1-3 require $O(e)$ time, as do lines 5-8, assuming we store the graph, including capacities and flow values, using an appropriate data structure (an adjacency matrix where each entry stores the capacity and flow would suffice). If lines 5-8 are executed N times, then the complexity of FORD-FULKERSON is $O(e) + N\,O(e) = O(N\,e)$.

In the worst case, the value of the flow of the augmenting path is 1 for each iteration of the while loop. Thus, if the value of a maximal flow f is $|f| = M$, then the worst-case complexity of FORD-FULKERSON is $O(M\,e)$. This worst-case behavior can be avoided by choosing the augmenting paths in an intelligent way.

The EDMONDS-KARP algorithm is an implementation of FORD-FULKERSON that chooses as an augmenting path a *shortest path* in the residual flow network, using breadth-first search. It can be shown that this implementation has a complexity of $O(v\,e^2)$. Two other implementations which achieve a better running times are the PREFLOW-PUSH algorithm, with complexity $O(v^2\,e)$, and the LIFT-TO-FRONT algorithm, with complexity $O(v^3)$. The details of all three of these implementations, including proofs of the complexities, can be found in [1].

In Figure 5 we show how the EDMONDS-KARP algorithm works on a small graph.

References

[1] Cormen, Leiserson, and Rivest, *Introduction to Algorithms*, McGraw Hill, 1990.

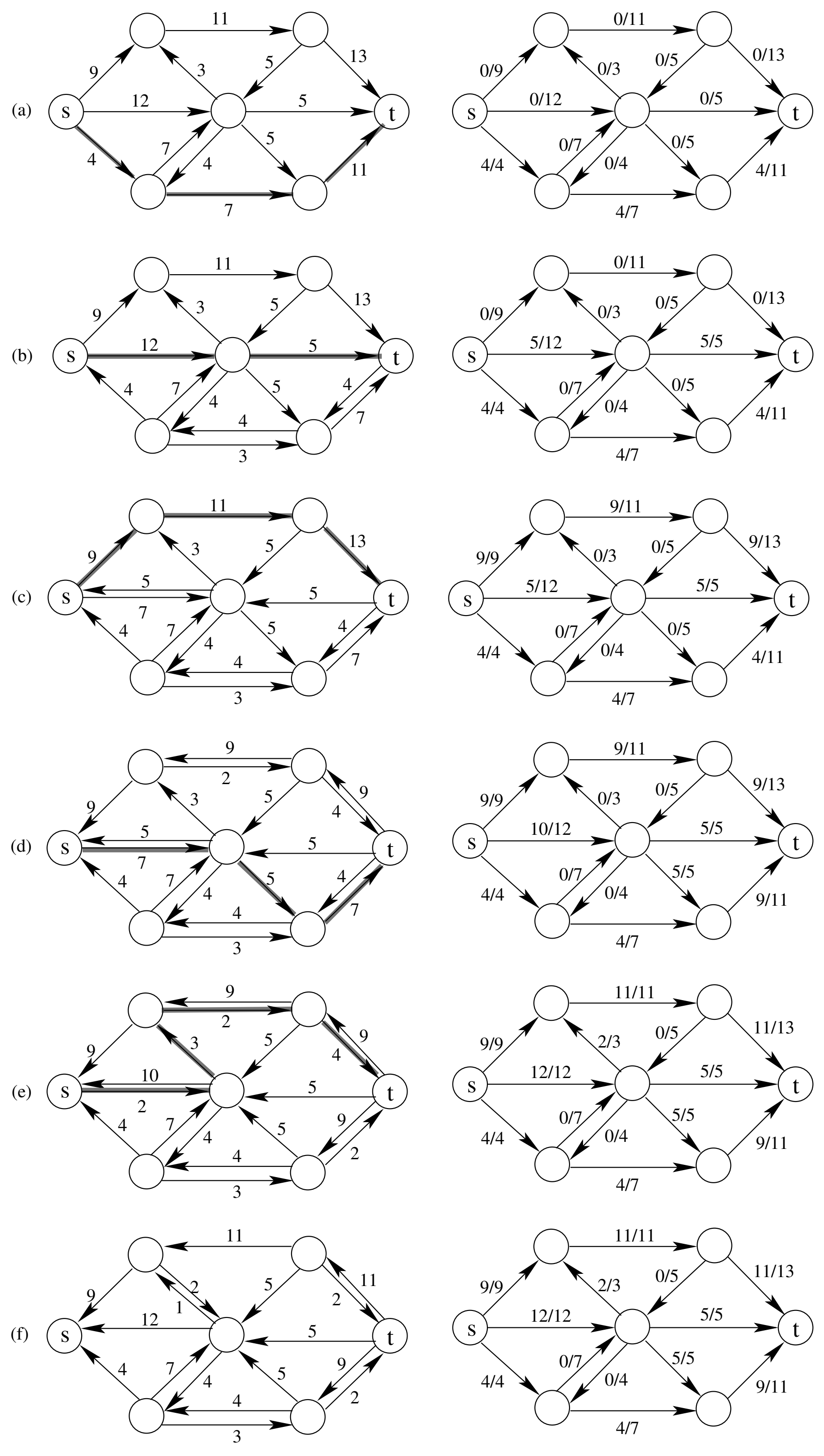

Figure 5: **The Ford-Fulkerson Algorithm.** In steps (a)-(e) we find an augmenting path, and add it into the flow. The residual graph is on the left, and the resulting flow is on the right. (f) The final residual network has no augmenting path, and the final flow is shown on the right. The flow has value 25.