**Flow Networks**

**Definition 1**

A flow network consists of

- A directed graph \( G = (V, E) \).
- A capacity function \( c : V \times V \to \mathbb{R} \) such that \( c(u, v) \geq 0 \) if \( (u, v) \in E \) and \( c(u, v) = 0 \) for all \( (u, v) \notin E \).
- Two distinguished vertices \( s, t \in V \) called the source and the sink, respectively.

We read \( (u, v) \) to mean \( u \to v \).

**Assumption**

Each vertex \( v \in V \) is on some directed path from \( s \) to \( t \). This implies that \( G \) is connected (but not necessarily strongly connected), and that \( |E| \geq |V| - 1 \).

**Network Flows**

**Definition 2**

Let \( N = (G = (V, E), c, s, t) \) be a flow network.

A flow in \( N \) is a function

\[
 f : V \times V \to \mathbb{R}
\]

satisfying the following conditions:

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

\[
 \sum_{v \in V} f(u, v) = 0.
\]
Network Flows (cont’d)

\[ N = (G = (V, E), c, s, t) \text{ flow network, } f : V \times V \to \mathbb{R} \text{ flow in } N. \]

- For \( u, v \in V \) we call \( f(u, v) \) the net flow at \( (u, v) \).
- The value of the flow \( f \) is the number
  \[ |f| = \sum_{v \in V} f(s, v). \]

Notice that our particular definition of flow (the “skew-symmetry” constraint) ensures that \( f(u, v) \) is truly the “net flow” in the usual sense of the word (e.g. if \((r, y)\) on slide 2 was to carry flow 3, and \((y, r)\) to carry flow 4, we will have \( f(r, y) = -1 \)).

The Maximum-Flow Problem

Input: Network \( N \)
Output: Flow of maximum value in \( N \)

The problem is to find the flow \( f \) such that \( |f| = \sum_{v \in V} f(s, v) \) is the largest possible (over all “legal” flows).

The Ford-Fulkerson Algorithm

Published in 1956 by Delbert Fulkerson and Lester Randolph Ford Jr.

Algorithm Ford-Fulkerson(\( N \))

1. \( f \leftarrow \) flow of value 0
2. while there exists an \( s \to t \) path \( P \) in the “residual network” do
3. \( f \leftarrow f + f_P; \)
4. Update the “residual network”.
5. return \( f \)

The “residual network” is \( N \) with the “used-up” capacity removed.

To make this precise, we need notation, and proofs - this lecture.
Some Technical Observations

\( N = (G = (V, E), c, s, t) \) flow network, \( f : V \times V \to \mathbb{R} \) flow in \( N, u, v \in V \).

1. \( f(u, u) = 0 \) for all \( u \in V \).
   "Proof": \( f(u, u) = -f(u, u) \) by skew symmetry.

2. For any \( v \in V \setminus \{s, t\} \),
   \[ \sum_{u \in V} f(u, v) = 0. \]
   Proof: \( \sum_{u \in V} f(u, v) = -\sum_{v \in V} f(v, u) = 0 \) by skew symmetry and flow conservation.

3. If \( (u, v) \notin E \) and \( (v, u) \notin E \) then \( f(u, v) = f(v, u) = 0 \).
   Proof: Either \( f(u, v) \) or \( f(v, u) \geq 0 \) by skew symmetry. Say, \( f(u, v) \geq 0 \).
   Then \( 0 \leq f(u, v) \leq c(u, v) = 0 \) by the capacity constraint. So \( f(u, v) = 0 \).
   By skew symmetry, this shows \( f(v, u) = 0 \).

One More Technical Observation

4. The positive net flow entering \( v \) is:
   \[ \sum_{u \in V} f(u, v) \text{ for } f(u, v) > 0. \]
   The positive net flow leaving \( v \) is defined symmetrically.
   Flow conservation now says:
   "positive net flow in = positive net flow out".

All these observations are just to make it easy for us to talk about flows.

Working with Flows

Implicit summation notation: For \( X, Y \subseteq V \) put
\[ f(X, Y) = \sum_{u \in X} \sum_{v \in Y} f(u, v) = \sum_{(u, v) \in X \times Y} f(u, v). \]

Abbreviations:

\[ f(u, Y) \] stands for \( f([u], Y) \) and
\[ f(X, v) \] stands for \( f(X, [v]) \).

Conservation of flow is now:

\[ f(u, V) = 0 \quad \text{for all } u \in V \setminus \{s, t\}. \]

Working with Flows (cont’d)

Lemma 3
\( N = (G = (V, E), c, s, t) \) flow network, \( f \) flow in \( N \).
Then for all \( X, Y, Z \subseteq V \),
1. \( f(X, X) = 0 \).
2. \( f(X, Y) = -f(Y, X) \).
3. If \( X \cap Y = \emptyset \) then
   \[ f(X \cup Y, Z) = f(X, Z) + f(Y, Z), \]
   \[ f(Z, X \cup Y) = f(Z, X) + f(Z, Y). \]

Lemma “lifts” Network flow properties to sets-of-vertices.
Proof of Lemma 3

1. \( f(X, X) = \sum_{(u, v) \in X \times X} f(u, v) \) by defn. of \( f(X, X) \)
\( = \sum_{\{u, v\} \subseteq X} f(u, v) \) by skew-symm
\( = f(Y, X) \) by defn of \( f(Y, X) \)

\( \text{ADS: lects 10 & 11 – slide 13 – 24th & 28th Oct, 2014} \)

Working with Flows (cont’d)

Corollary 4
\( N = (G = (V, E), c, s, t) \) flow network, \( f \) flow in \( N \). Then
\[ |f| = f(V, t). \]

Proof:
\[ |f| = f(s, V) \] (by definition)
\[ = f(V, V) - f(V \setminus \{s\}, V) \] (by Lemma 3 (3.))
\[ = -f(V \setminus \{s\}, V) \] (by Lemma 3 (1.))
\[ = f(V, V \setminus \{s\}) \] (by Lemma 3 (2.))
\[ = f(V, t) + f(V, V \setminus \{s, t\}) \] (by Lemma 3 (3.))
\[ = f(V, t) + \sum_{v \in V \setminus \{s, t\}} f(V, v) \] (by Definition)
\[ = f(V, t) \] (by flow conservation)

Residual Networks

Idea is to capture possible extra flow given current flow.

Definition 5
\( N = (G = (V, E), c, s, t) \) flow network, \( f \) flow in \( N \).

1. For all \( u, v \in V \times V \), the residual capacity of \( (u, v) \) is
\[ c_r(u, v) = c(u, v) - f(u, v). \]

2. The residual network of \( N \) induced by \( f \) is
\[ N_f((V, E_f), c_r, s, t), \]
where
\[ E_f = \{(u, v) \in V \times V | c_r(u, v) > 0\} \]

Notice that \( E_f \) may contain edges not originally in \( E \) ("back-edges").

\( \text{ADS: lects 10 & 11 – slide 15 – 24th & 28th Oct, 2014} \)

Proof of Lemma 3 (cont’d)

2. \( f(X, Y) = \sum_{(u, v) \in X \times Y} f(u, v) \) by defn of \( f(X, Y) \)
\( = \sum_{(u, v) \in X \times Y} -f(v, u) \) by skew-symmetry
\( = -\sum_{(v, u) \in Y \times X} f(v, u) \) take − outside the summation
\( = -f(Y, X). \) by defn of \( f(Y, X) \)

\( \text{ADS: lects 10 & 11 – slide 14 – 24th & 28th Oct, 2014} \)

3. \( f(X \cup Y, Z) = \sum_{u \in X \cup Y} \sum_{v \in Z} f(u, v) \)
\( = \sum_{u \in X \cap Y} \sum_{v \in Z} f(u, v) \) (expand sum into \( X \) and \( Y \), subtract duplicates in \( X \cap Y \))
\( = \sum_{u \in X} \sum_{v \in Z} f(u, v) \) (but \( X \cap Y = \emptyset \), so third term disappears)
\( = f(X, Z) + f(Y, Z). \)

Moreover,
\( f(Z, X \cup Y) = -f(X \cup Y, Z) = -(f(X, Z) + f(Y, Z)) = f(Z, X) + f(Z, Y). \)
Adding Flows

Lemma 6
Let $N = (G = (V, E), c, s, t)$ be a flow network.
Let $f$ be a flow in $N$.
Let $g : V \times V \to \mathbb{R}$ be a flow in the residual network $N_f$.
Then the function $f + g : V \times V \to \mathbb{R}$ defined by
$$(f + g)(u, v) = f(u, v) + g(u, v)$$
is a flow of value $|f| + |g|$ in $N$.

Proof of Lemma 6
First we have to check that $f + g$ is actually a flow in $N$.

Capacity constraints:
$$(f + g)(u, v) = f(u, v) + g(u, v) \leq f(u, v) + c(u, v) = f(u, v) + c(u, v) - f(u, v) = c(u, v).$$

Skew symmetry:
$$(f + g)(u, v) = f(u, v) + g(u, v) = -f(v, u) - g(v, u) = -(f + g)(v, u).$$

Flow Conservation: For every $u \in V \setminus \{s, t\}$:
$$\sum_{v \in V} (f + g)(u, v) = \sum_{v \in V} f(u, v) + \sum_{v \in V} g(u, v) = 0 + 0 = 0.$$

Proof of Lemma 6 (cont’d)
Next we have to check that $f + g$ does have the value that we claimed for it.

Value:
$$|f + g| = \sum_{v \in V} (f + g)(s, v)$$
$$= \sum_{v \in V} f(s, v) + \sum_{v \in V} g(s, v)$$
$$= |f| + |g|. $$
Augmenting Paths

Definition 7
\( N = \langle G = (V, E), c, s, t \rangle \) flow network, \( f \) flow in \( N \).

Then an augmenting path for \( f \) is a path \( P \) from \( s \) to \( t \) in the residual network \( N_f \).

The residual capacity of \( P \) is
\[
    c_f(P) = \min\{c_f(u, v) \mid (u, v) \text{ edge on } P\}.
\]

Note that \( c_f(P) > 0 \), by definition of \( E_f \) (recall that we only keep edges in \( E_f \) if their residual capacity is strictly positive).

Example

An augmenting path of residual capacity 10

Pushing Flow through an Augmenting Path

Lemma 8
\( N = \langle G = (V, E), c, s, t \rangle \) flow network, \( f \) flow in \( N \).

Let \( P \) be an augmenting path. Then \( f + f_P \) is a flow in \( N \) of value
\[
    |f| + c_f(P) > |f|.
\]

Proof: Follows from Lemma 6 and Lemma 8.

Corollary 9
\( N = \langle G = (V, E), c, s, t \rangle \) flow network, \( f \) flow in \( N \). Let \( P \) be an augmenting path. Then \( f + f_P \) is a flow in \( N \) of value
\[
    |f| + c_f(P) > |f|.
\]

Proof: Follows from Lemma 6 and Lemma 8.
The Ford-Fulkerson Algorithm

Algorithm Ford-Fulkerson(N)
1. \( f \leftarrow \text{flow of value 0} \)
2. while there exists an augmenting path \( P \) in \( N_f \) do
3. \( f \leftarrow f + f_P \)
4. return \( f \)

To prove that Ford-Fulkerson correctly solves the Maximum Flow problem, we have to prove that:
1. The algorithm terminates.
2. After termination, \( f \) is a maximum flow.

Definition 10
\( N = (G = (V, E), c, s, t) \) flow network.
A cut of \( N \) is a pair \((S, T)\) such that:
1. \( s \in S \) and \( t \in T \),
2. \( V = S \cup T \) and \( S \cap T = \emptyset \).
The capacity of the cut \((S, T)\) is
\[
c(S, T) = \sum_{u \in S, v \in T} c(u, v).\]

Example
A cut of capacity 45.

Example
A cut of capacity 25.
Cuts and Flows

Lemma 11
\(\mathcal{N} = (\mathcal{G} = (V, E), c, s, t)\) flow network, \(f\) flow in \(\mathcal{N}\), \((S, T)\) cut of \(\mathcal{N}\).

Then
\[|f| = f(S, T).\]

Proof: We apply Lemma 3:
\[
|f| = f(s, V) = f(s, V) + f(S - \{s\}, V) \quad [t \not\in S \Rightarrow f(S - \{s\}, V) = 0]
\]
\[= f(S, V)
\]
\[= f(S, T) + f(S, S)
\]
\[= f(S, T).
\]

Cuts and Flows (cont’d)

Corollary 12
The value of any flow in a network is bounded from above by the capacity of any cut.

Proof: Let \(f\) be a flow and \((S, T)\) a cut. Then
\[|f| = f(S, T) \leq c(S, T).
\]

The Max-Flow Min-Cut Theorem

Theorem 13
Let \(\mathcal{N} = (\mathcal{G} = (V, E), c, s, t)\) be a flow network.
Then the maximum value of a flow in \(\mathcal{N}\) is equal to the minimum capacity of a cut in \(\mathcal{N}\).

Proof of the Max-Flow Min-Cut Theorem
Let \(f\) be a flow of maximum value and \((S, T)\) a cut of minimum capacity in \(\mathcal{N}\). We shall prove that
\[|f| = c(S, T).
\]

1. \(|f| \leq c(S, T)\) follows from Corollary 12.
   So all we have to prove is that there is a cut \((S, T)\) such that
   \[c(S, T) \leq |f|.
   \]
2. First remember that \(|f|\) has no augmenting path.
   \[Proof:\] If \(P\) was an augmenting path, then \(f + f_P\) would be a flow of larger value
   (because by definition of \(\mathcal{N}_r\), all edges in \(\mathcal{N}_r\) have strictly positive weights).
3. Thus there is no path from \(s\) to \(t\) in \(\mathcal{N}_r\). Let
   \[S = \{v | there is a path from s to v in \mathcal{N}_r\}\]
   and \(T = V \setminus S\). Then \((S, T)\) is a cut.
4. By definition of $S$, and because reachability in graphs is a transitive relation, there cannot be any edge from $S$ to $T$ in $N_f$. Thus for all $u \in S$, $v \in T$ we have $c(u, v) - f(u, v) = 0$.

5. Thus

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

(by Lemma 11).

---

Corollary 14

A flow is maximum if, and only if, it has no augmenting path.

Proof: This follows from the proof of the Max-Flow Min-Cut theorem.

Corollary 15

If the Ford-Fulkerson algorithm terminates, then it returns a maximum flow.

Proof: The flow returned by Ford-Fulkerson has no augmenting path.

---

Termination

Let $f^*$ be a maximum flow in a network $N$.

- If all capacities are integers, then Ford-Fulkerson stops after at most $|f^*|$ iterations of the main loop.
- If all capacities are rationals, then Ford-Fulkerson stops after at most $q \cdot |f^*|$ iterations of the main loop, where $q$ is the least common multiple of the denominators of all the capacities.
- For arbitrary real capacities, it may happen that Ford-Fulkerson does not stop.
The Edmonds-Karp Heuristic

Idea
Always choose a shortest augmenting path.

\( n \) number of vertices, \( m \) number of edges. Recall that \( n \leq m + 1 \)

A shortest augmenting path can be found by Breadth-First-Search (reading assignment) in time \( O(n + m) = O(m) \).

Theorem 16
The Ford-Fulkerson algorithm with the Edmonds-Karp heuristic stops after at most \( O(nm) \) iterations of the main loop.
Thus the running time is \( O(nm^2) \).

Interesting Example

We will run Ford-Fulkerson (with the Edmonds-Karp heuristic) on this network. This is interesting because we will see the "back-edges" being used to "undo" part of an previous augmenting path.

1st augmenting path: \( s \rightarrow r \rightarrow w \rightarrow t \).
Length is 3 (so we satisfy Edmonds-Karp rule to take a shortest possible path). Min capacity is 10, so we push flow of 10 along the path. Starting flow becomes 10.

Residual network after adding first flow of value 10 along \( s \rightarrow r \rightarrow w \rightarrow t \).
The newly-created "back-edges" are shown in red.
There is no longer any augmenting path of length \( \leq 3 \), and the only one of length 4 is \( s \to x \to y \to z \to t \), which has a minimum capacity \( \min\{10, 10, 15, 15\} \), ie 10.

We push this extra flow of value 10 along \( s \to x \to y \to z \to t \), bringing overall flow to 20.

Residual network after adding flow from second augmenting path \( s \to u \to v \to w \to r \to y \to z \to t \), overall flow now 20.

Now there is only one simple augmenting path - \( s \to u \to v \to w \to r \to y \to z \to t \), with minimum residual capacity 5.

Notice we use the “back-edge” \( w \to r \) in our path. This is essentially “re-shipping” 5 units from the first flow-path away from \( r \to w \to t \) and along \( r \to y \to z \to t \) instead.

Residual network after adding 3rd flow, of value 5 \( \Rightarrow \) total flow 25.

There is no longer any augmenting path in our residual network (set of vertices “reachable” from \( s \) is \( \{s, u, v, x, w, r\} \)).
Problems

   
   Not in [CLRS] (ed 3). Question is: consider Figure 26.1(b) and find a pair of subsets $X, Y \subseteq V$ such that $f(X, Y) = -f(V \setminus X, Y)$. After that, find a pair of subsets $X', Y' \subseteq V$ for which $f(X', Y') \neq -f(V \setminus X', Y')$.


4. Problem 26-4 of [CLRS].