6.4 Ford-Fulkerson Demo

click to begin demo
Ford-Fulkerson algorithm

computing a min cut
Ford-Fulkerson algorithm

**Initialization.** Start with 0 flow.
Augmenting path. Find an undirected path from $s$ to $t$ such that:

- Can increase flow on forward edges (not full).
- Can decrease flow on backward edge (not empty).

1st augmenting path
Idea: increase flow along augmenting paths

**Augmenting path.** Find an undirected path from $s$ to $t$ such that:
- Can increase flow on forward edges (not full).
- Can decrease flow on backward edge (not empty).

1st augmenting path
Augmenting path. Find an undirected path from $s$ to $t$ such that:

- Can increase flow on forward edges (not full).
- Can decrease flow on backward edge (not empty).

Idea: increase flow along augmenting paths

**1st augmenting path**

[Diagram of a network flow graph with labeled edges and nodes, illustrating the first augmenting path and the bottleneck capacity of 10.]
Idea: increase flow along augmenting paths

**Augmenting path.** Find an undirected path from \( s \) to \( t \) such that:
- Can increase flow on forward edges (not full).
- Can decrease flow on backward edge (not empty).

**1st augmenting path**

![Graph](image)
Idea: increase flow along augmenting paths

Augmenting path. Find an undirected path from $s$ to $t$ such that:
- Can increase flow on forward edges (not full).
- Can decrease flow on backward edge (not empty).

2nd augmenting path

\[ \begin{array}{c}
\text{s} \\
0 / 5 \\
10 / 10 \\
0 / 15 \\
0 / 15 \\
0 / 16 \\
\end{array} \quad \begin{array}{c}
0 / 4 \\
0 / 4 \\
0 / 8 \\
0 / 6 \\
0 / 16 \\
\end{array} \quad \begin{array}{c}
0 / 9 \\
10 / 15 \\
0 / 15 \\
0 / 10 \\
0 / 10 \\
\end{array} \quad \begin{array}{c}
10 / 10 \\
0 / 10 \\
0 / 15 \\
0 / 15 \\
0 / 10 \\
\end{array} \quad \begin{array}{c}
\text{t} \\
10 \\
\end{array} \]
Idea: increase flow along augmenting paths

**Augmenting path.** Find an undirected path from $s$ to $t$ such that:
- Can increase flow on forward edges (not full).
- Can decrease flow on backward edge (not empty).

**2nd augmenting path**
Idea: increase flow along augmenting paths

**Augmenting path.** Find an undirected path from \( s \) to \( t \) such that:

- Can increase flow on forward edges (not full).
- Can decrease flow on backward edge (not empty).

2nd augmenting path

\[
\begin{align*}
&10 + 10 = 20
\end{align*}
\]
Idea: increase flow along augmenting paths

**Augmenting path.** Find an undirected path from $s$ to $t$ such that:
- Can increase flow on forward edges (not full).
- Can decrease flow on backward edge (not empty).

*3rd augmenting path*
Augmenting path. Find an undirected path from $s$ to $t$ such that:

- Can increase flow on forward edges (not full).
- Can decrease flow on backward edge (not empty).

Idea: increase flow along augmenting paths
Idea: increase flow along augmenting paths

**Augmenting path.** Find an undirected path from $s$ to $t$ such that:

- Can increase flow on forward edges (not full).
- Can decrease flow on backward edge (not empty).

**3rd augmenting path**

The network diagram illustrates a third path from $s$ to $t$ with capacities and flows as indicated.
Idea: increase flow along augmenting paths

**Augmenting path.** Find an undirected path from $s$ to $t$ such that:
- Can increase flow on forward edges (not full).
- Can decrease flow on backward edge (not empty).

4th augmenting path
Augmenting path. Find an undirected path from $s$ to $t$ such that:

- Can increase flow on forward edges (not full).
- Can decrease flow on backward edge (not empty).

Idea: increase flow along augmenting paths
Augmenting path. Find an undirected path from \( s \) to \( t \) such that:

- Can increase flow on forward edges (not full).
- Can decrease flow on backward edge (not empty).

Idea: increase flow along augmenting paths

4th augmenting path

\[
\begin{align*}
&\text{backward edge} \\
&\text{(not empty)}
\end{align*}
\]
Idea: increase flow along augmenting paths

**Termination.** All paths from $s$ to $t$ are blocked by either a
- Full forward edge.
- Empty backward edge.

no more augmenting paths
Ford-Fulkerson algorithm
computing a min cut
To compute mincut \((A, B)\) from maxflow \(f\):

- By augmenting path theorem, no augmenting paths with respect to \(f\).
- Compute \(A = \text{set of vertices connected to } s \text{ by an undirected path with no full forward or empty backward edges.}\)
To compute mincut \((A, B)\) from maxflow \(f\):

- By augmenting path theorem, no augmenting paths with respect to \(f\).
- Compute \(A = \text{set of vertices connected to } s \text{ by an undirected path with no full forward or empty backward edges.}\)

---

**Diagram:**

[Diagram of a network flow with vertices labeled and capacities indicated on the edges.]
Computing a mincut from a maxflow

To compute mincut \((A, B)\) from maxflow \(f\):

- By augmenting path theorem, no augmenting paths with respect to \(f\).
- Compute \(A = \) set of vertices connected to \(s\) by an undirected path with no full forward or empty backward edges.
To compute mincut \((A, B)\) from maxflow \(f\):

- By augmenting path theorem, no augmenting paths with respect to \(f\).
- Compute \(A = \text{set of vertices connected to } s \text{ by an undirected path with no full forward or empty backward edges.}\)
Computing a mincut from a maxflow

To compute mincut \((A, B)\) from maxflow \(f\):
• By augmenting path theorem, no augmenting paths with respect to \(f\).
• Compute \(A = \) set of vertices connected to \(s\) by an undirected path with no full forward or empty backward edges.
Computing a mincut from a maxflow

To compute mincut $(A, B)$ from maxflow $f$:

- By augmenting path theorem, no augmenting paths with respect to $f$.
- Compute $A =$ set of vertices connected to $s$ by an undirected path with no full forward or empty backward edges.
To compute mincut \((A, B)\) from maxflow \(f\):

- By augmenting path theorem, no augmenting paths with respect to \(f\).
- Compute \(A = \) set of vertices connected to \(s\) by an undirected path with no full forward or empty backward edges.
Computing a mincut from a maxflow

To compute mincut \((A, B)\) from maxflow \(f\):

- By augmenting path theorem, no augmenting paths with respect to \(f\).
- Compute \(A = \text{set of vertices connected to } s \text{ by an undirected path with no full forward or empty backward edges.}\)
To compute mincut \((A, B)\) from maxflow \(f\):

- By augmenting path theorem, no augmenting paths with respect to \(f\).
- Compute \(A = \) set of vertices connected to \(s\) by an undirected path with no full forward or empty backward edges.