Review: Ford-Fulkerson Algorithm

⊿ Augment flow as long as it is possible
while there exists an s-t path $P$ in residual graph $G_f$ do
\[ f = \text{Augment}(f, P) \]
update $G_f$
end while
return $f$

Pearson Demo
Correctness: relate maximum flow to minimum cut

Step 3: F-F returns a maximum flow

We will prove this by establishing a deep connection between flows and cuts in graphs: the max-flow min-cut theorem.

- An s-t cut $(A, B)$ is a partition of the nodes into sets $A$ and $B$ where $s \in A$, $t \in B$
- Capacity of cut $(A, B)$ equals
  \[ c(A, B) = \sum_{e \text{ from } A \text{ to } B} c(e) \]
- Flow across a cut $(A, B)$ equals
  \[ f(A, B) = \sum_{e \text{ out of } A} f(e) - \sum_{e \text{ into } A} f(e) \]

Clicker Question
What is the capacity of the cut and the flow across the cut?

<table>
<thead>
<tr>
<th>Capacity</th>
<th>Flow</th>
</tr>
</thead>
<tbody>
<tr>
<td>A. 16+4+9+14   11+1-3+11</td>
<td></td>
</tr>
<tr>
<td>B. 16+4-9+14   11+1-4+11</td>
<td></td>
</tr>
<tr>
<td>C. 16+4+14     11+1-4+11</td>
<td></td>
</tr>
<tr>
<td>D. 16+4+14     11+1+11</td>
<td></td>
</tr>
</tbody>
</table>

Flow Value Lemma

First relationship between cuts and flows

Lemma: let $f$ be any flow and $(A, B)$ be any s-t cut. Then
\[ \nu(f) = \sum_{e \text{ out of } A} f(e) - \sum_{e \text{ into } A} f(e) \]

Proof: see book. Basic idea is to use conservation of flow: all the flow out of $s$ must leave $A$ eventually.
Corollary: Cuts and Flows

Really important corollary of flow-value lemma

**Corollary:** Let \( f \) be any \( s-t \) flow and let \((A, B)\) be any \( s-t \) cut. Then \( v(f) \leq c(A, B) \).

**Proof:**

\[
v(f) = \sum_{e \text{ out of } A} f(e) - \sum_{e \text{ into } A} f(e)
\]

\[
\leq \sum_{e \text{ out of } A} f(e)
\]

\[
\leq \sum_{e \text{ out of } A} c(e)
\]

\[
= c(A, B)
\]

Duality

**Illustration on board**

Claim If there is a flow \( f^* \) and cut \((A^*, B^*)\) such that \( v(f^*) = c(A^*, B^*) \), then

- \( f^* \) is a maximum flow
- \((A^*, B^*)\) is a minimum cut

Clicker

Suppose \( f \) is a flow, and there is a path from \( s \) to \( u \) in \( G_f \), but no path from \( s \) to \( v \) in \( G_f \). Then

A. There is no edge from \( u \) to \( v \) in \( G \).

B. If there is an edge from \( u \) to \( v \) in \( G \) then \( f \) does not send any flow on this edge.

C. If there is an edge from \( u \) to \( v \) in \( G \) then \( f \) fully saturates it with flow.

D. None of the above.

F-F finds a minimum cut

**Theorem:** The cut \((A, B)\) where \( A \) is the set of all nodes reachable from \( s \) in the residual graph is a minimum-cut.

F-F returns a maximum flow

**Theorem:** The \( s-t \) flow \( f \) returned by F-F is a maximum flow.

- Since \( f \) is the final flow there are no residual paths in \( G_f \).
- Let \((A, B)\) be the \( s-t \) cut where \( A \) consists of all nodes reachable from \( s \) in the residual graph.
  - Any edge out of \( A \) must have \( f(e) = c(e) \) otherwise there would be more nodes than just \( A \) that reachable from \( s \).
  - Any edge into \( A \) must have \( f(e) = 0 \) otherwise there would be more nodes than just \( A \) that reachable from \( s \).
- Therefore

\[
v(f) = \sum_{e \text{ out of } A} f(e) - \sum_{e \text{ into } A} f(e)
\]

\[
= \sum_{e \text{ out of } A} c(e) = c(A, B)
\]
Ford-Fulkerson Running Time

- Flow increases at least one unit per iteration
- F-F terminates in at most \( C \) iterations, where \( C \) is the sum of capacities leaving source.
- \( C \leq n C_{\text{max}} \), where \( C_{\text{max}} \) = maximum edge capacity
- Running time: \( O(m n C_{\text{max}}) \)

Is this polynomial? pseudo-polynomial (exponential in \( \log C_{\text{max}} \))

Improving Running Time

Good path choice will find:
- \( s \rightarrow u \rightarrow t \), flow \( C \)
- \( s \rightarrow v \rightarrow t \), flow \( C \)

Worst-case: keep incrementing by 1:
- \( s \rightarrow u \rightarrow v \rightarrow t \), flow 1
- \( s \rightarrow v \rightarrow u \rightarrow t \), flow 1
- \( s \rightarrow u \rightarrow v \rightarrow t \), flow 1
- ...

Solution: choose good augmenting paths, with
- Large enough bottleneck capacity: capacity-scaling algorithm
- Fewest edges: Edmonds-Karp, Dinitz

Capacity-scaling algorithm

Start with large \( \Delta \), divide by two in each phase

let \( f(e) = 0 \) for all \( e \in E \)
let \( \Delta = \) largest power of 2 \( \leq C_{\text{max}} \)

while \( \Delta \geq 1 \) do

prune residual graph \( G_f \) to \( G_f(\Delta) \)

while there is augmenting \( s \rightarrow t \) path \( P \) in \( G_f(\Delta) \) do

\( f = \) Augment \((f, P)\)

update \( G_f(\Delta) \) \( \triangleright \) only \( e \geq \Delta \)

end while

\( \Delta = \Delta / 2 \) \( \triangleright \) refine

end while
Capacity-Scaling: Running Time

- How many scaling phases? \( \Theta(\log C_{\text{max}}) \)
- How much does the flow increase at every augmentation? \( \geq \Delta \)
- How many augmentations per phase? \( \leq 2m \)
  - Can show: at end of \( \Delta \) phase, flow value within \( m\Delta \) of max.
  - \( \implies \) at most \( 2m \) iterations \( \Delta/2 \) phase
  - (Sketch) Construct cut \( (A, B) \) as in max-flow / min-cut theorem.
  - Edges from \( A \) to \( B \) are within \( \Delta \) of being saturated.
  - Edges from \( B \) to \( A \) carry less than \( \Delta \) flow.
  - \( \implies \) Cut capacity at most \( m\Delta \) more than flow value.
- Recall: time to find augmenting path? \( O(m) \)
- Overall: \( O(m^2 \log C_{\text{max}}) \), polynomial

Running Times

- Basic F-F: \( O(mnC_{\text{max}}) \) pseudo-polynomial
  - polynomial in magnitude
- Capacity-scaling: \( O(m^2 \log C_{\text{max}}) \) polynomial
  - polynomial in number of bits
- Edmonds-Karp: \( O(m^2n) \) strongly-polynomial
  - does not depend on values, only \( m, n \)
- Dinitz: \( O(mn^2) \) even better
- Edmonds-Karp and Dinitz: choose short augmenting paths