Course Recap: (Or, What Could be On the First Midterm?)

- Gale-Shapley, Stable Matching Problem
- Asymptotic Analysis (e.g., Big O notation)
- Recurrence Relationships
- Greedy Algorithms
- Graph Algorithms
- Divide-And-Conquer + Recurrence Relationships
- Dynamic Programming
- Basic Questions about Network Flow (today)

Midterm 1

- Practice Midterm and Solutions Posted on Blackboard
  - Solutions posted yesterday (Monday)
  - No electronics (laptop, calculator, smart phone etc.)
  - May prepare one 3x5 inch index card with any notes you want
    - No additional notes
  - Exam is 2 hours (8PM to 10PM)
    - Practice exam is longer than the real midterm
    - Topics are reasonably representative of real midterm

Chapter 7

Network Flow

Max Flow and Min Cut

- Max flow and min cut.
- Two very rich algorithmic problems.
- Cornerstone problems in combinatorial optimization.
- Beautiful mathematical duality.

- Practical applications / reductions:
  - Data mining.
  - Open-pit mining.
  - Project selection.
  - Airline scheduling.
  - Bipartite matching.
  - Baseball elimination.
  - Image segmentation.
  - Multi-camera scene reconstruction.
  - Network connectivity.
  - Many many more...
Flow network.
- Abstraction for material flowing through the edges.
- $G = (V, E)$: directed graph, no parallel edges.
- Two distinguished nodes: $s =$ source, $t =$ sink.
- $c(e)$: capacity of edge $e$.

**Minimum Cut Problem**

$\begin{array}{cccccccc}
2 & 3 & 4 & 5 & 6 & 7 & t \\
15 & 5 & 30 & 15 & 10 & 8 & 15 & 9 & 6 & 10 & 10 \\
4 & 4 & & & & & & & & & \\
\end{array}$

**Cuts**

- An s-t cut is a partition $(A, B)$ of $V$ with $s \in A$ and $t \in B$.
- The capacity of a cut $(A, B)$ is: $\text{cap}(A, B) = \sum_{e \in A \to B} c(e)$.

**Def.** An s-t flow is a function that satisfies:
- For each $e \in E$: $0 \leq f(e) \leq c(e)$ (capacity)
- For each $v \in V - \{s, t\}$: $\sum_{e \in A \to v} f(e) = \sum_{e \in v \to B} f(e)$ (conservation)

**Def.** The value of a flow $f$ is: $\nu(f) = \sum_{e \in A \to t} f(e)$.
Max flow problem. Find s-t flow of maximum value.

**Flow value lemma.** Let f be any flow, and let (A, B) be any s-t cut. Then, the net flow sent across the cut is equal to the amount leaving s.

\[ \sum_{e \in \delta^+ (A)} f(e) - \sum_{e \in \delta^- (A)} f(e) = v(f) \]

Weak duality. Let f be any flow, and let (A, B) be any s-t cut. Then the value of the flow is at most the capacity of the cut.

\[ \text{Cut capacity} \leq \text{Flow value} \leq \text{Capacity of Cut} \]
Flows and Cuts

Weak duality. Let \( f \) be any flow. Then, for any s-t cut \((A, B)\) we have \( v(f) \leq \text{cap}(A, B) \).

**Pf.**

\[
 v(f) = \sum_{e \in A \rightarrow B} f(e) - \sum_{e \in B \rightarrow A} f(e) 
\leq \sum_{e \in A \rightarrow B} \text{cap}(e) 
= \text{cap}(A, B) 
\]

Certificate of Optimality

Corollary. Let \( f \) be any flow, and let \((A, B)\) be any cut. If \( v(f) = \text{cap}(A, B) \), then \( f \) is a max flow and \((A, B)\) is a min cut.

Towards a Max Flow Algorithm

**Greedy algorithm.**
- Start with \( f(e) = 0 \) for all edge \( e \in E \).
- Find an s-t path \( P \) where each edge has \( f(e) < c(e) \).
- Augment flow along path \( P \).
- Repeat until you get stuck.

Residual Graph

Original edge: \( e = (u, v) \in E \).
- Flow \( f(e) \), capacity \( c(e) \).

Residual edge:
- "Undo" flow sent.
- \( e = (u, v) \) and \( e^R = (v, u) \).
- Residual capacity:
  - \( c_r(e) = |c(e) - f(e)| \) if \( e \in E \).
  - \( f(e) \) if \( e^R \in E \).

Residual graph: \( G_f = (V, E_f) \)
- Residual edges with positive capacity.
- \( E_f = \{ e : f(e) < c(e) \} \cup \{ e^R : f(e) > 0 \} \).
Ford-Fulkerson Algorithm

Max-Flow Min-Cut Theorem

Augmenting Path Algorithm

Proof of Max-Flow Min-Cut Theorem

7.3 Choosing Good Augmenting Paths
Ford-Fulkerson: Exponential Number of Augmentations

Q. Is generic Ford-Fulkerson algorithm polynomial in input size?
A. No. If max capacity is $C$, then algorithm can take $C$ iterations.

Choosing Good Augmenting Paths

Use care when selecting augmenting paths.
- Some choices lead to exponential algorithms.
- Clever choices lead to polynomial algorithms.
- If capacities are irrational, algorithm not guaranteed to terminate!

Goal: choose augmenting paths so that:
- Can find augmenting paths efficiently.
- Few iterations.

Choose augmenting paths with: [Edmonds-Karp 1972, Dinitz 1970]
- Max bottleneck capacity.
- Sufficiently large bottleneck capacity.
- Fewest number of edges.

Capacity Scaling

Intuition. Choosing path with highest bottleneck capacity increases flow by max possible amount.
- Don’t worry about finding exact highest bottleneck path.
- Maintain scaling parameter $\Delta$.
- Let $G_\Delta$ be the subgraph of the residual graph consisting of only arcs with capacity at least $\Delta$.

Capacity Scaling: Correctness

Assumption. All edge capacities are integers between 1 and $C$.

Integrality invariant. All flow and residual capacity values are integral.

Correctness. If the algorithm terminates, then $f$ is a max flow.

Proof:
- By integrality invariant, when $\Delta = 1 \Rightarrow G_\Delta = G_f$.
- Upon termination of a $\Delta$ phase, there are no augmenting paths.

Capacity Scaling: Running Time

Lemma 1. The outer while loop repeats $1 + \log_2 C$ times.
Proof. Initially $\Delta \leq 2C$. $\Delta$ decreases by a factor of 2 each iteration.

Lemma 2. Let $f$ be the flow at the end of a $\Delta$-scaling phase. Then the value of the maximum flow is at most $v(f) + m \cdot \Delta$.

Lemma 3. There are at most $2m$ augmentations per scaling phase.
Proof. Let $f$ be the flow at the end of the previous scaling phase.
- $LZ \Rightarrow v(f^*) \leq v(f) \cdot m(2\Delta)$.
- Each augmentation in a $\Delta$-phase increases $v(f)$ by at least $\Delta$.

Theorem. The scaling max-flow algorithm finds a max flow in $O(m \log C)$ augmentations. It can be implemented to run in $O(m^2 \log C)$ time.
Lemma 2. Let $f$ be the flow at the end of a $\Delta$-scaling phase. Then value of the maximum flow is at most $v(f) + m\Delta$.

**Proof.** (almost identical to proof of max-flow min-cut theorem)

- We show that at the end of a $\Delta$-phase, there exists a cut $(A, B)$ such that $\text{cap}(A, B) \leq v(f) + m\Delta$.
- Choose $A$ to be the set of nodes reachable from $s$ in $G_f(\Delta)$.
- By definition of $A$, $s \in A$.
- By definition of $f$, $t \notin A$.

\[
\begin{align*}
v(f) &= \sum_{e \in A} f(e) - \sum_{e \in \overline{A}} f(e) \\
&\geq \sum_{e \in A} (c(e) - \Delta) - \sum_{e \in \overline{A}} \Delta \\
&= \sum_{e \in A} c(e) - \Delta - \sum_{e \in \overline{A}} \Delta \\
&\geq \text{cap}(A, B) - m\Delta
\end{align*}
\]