CSEP 521
Applied Algorithms

Richard Anderson
Lecture 9
Network Flow Applications
Announcements

• Reading for this week
  – 7.5-7.12. Network flow applications
  – Next week: Chapter 8. NP-Completeness

• Final exam, March 18, 6:30 pm. At UW.
  – 2 hours
  – In class (CSE 303 / CSE 305)
  – Comprehensive
    • 67% post midterm, 33% pre midterm
Network Flow
Review

- Network flow definitions
- Flow examples
- Augmenting Paths
- Residual Graph
- Ford Fulkerson Algorithm
- Cuts
- Maxflow-MinCut Theorem
Network Flow Definitions

- Flowgraph: Directed graph with distinguished vertices s (source) and t (sink)
- Capacities on the edges, \( c(e) \geq 0 \)
- Problem: assign flows \( f(e) \) to the edges such that:
  - \( 0 \leq f(e) \leq c(e) \)
  - Flow is conserved at vertices other than s and t
    - Flow conservation: flow going into a vertex equals the flow going out
  - The flow leaving the source is as large as possible
Find a maximum flow
Residual Graph

• Flow graph showing the remaining capacity
• Flow graph $G$,  Residual Graph $G_R$
  – $G$: edge $e$ from $u$ to $v$ with capacity $c$ and flow $f$
  – $G_R$: edge $e'$ from $u$ to $v$ with capacity $c - f$
  – $G_R$: edge $e''$ from $v$ to $u$ with capacity $f$
Augmenting Path Lemma

- Let $P = v_1, v_2, \ldots, v_k$ be a path from $s$ to $t$ with minimum capacity $b$ in the residual graph.
- $b$ units of flow can be added along the path $P$ in the flow graph.
Ford-Fulkerson Algorithm (1956)

while not done

Construct residual graph $G_R$
Find an $s$-$t$ path $P$ in $G_R$ with capacity $b > 0$
Add $b$ units along in $G$

If the sum of the capacities of edges leaving $S$
is at most $C$, then the algorithm takes at most $C$ iterations
Cuts in a graph

- **Cut**: Partition of $V$ into disjoint sets $S$, $T$ with $s$ in $S$ and $t$ in $T$.
- **Cap($S,T$)**: sum of the capacities of edges from $S$ to $T$.
- **Flow($S,T$)**: net flow out of $S$
  - Sum of flows out of $S$ minus sum of flows into $S$.
- **Flow($S,T$) $\leq$ Cap($S,T$)***
Ford Fulkerson MaxFlow – MinCut Theorem

- There exists a flow which has the same value of the minimum cut
  - Shows that a cut is the dual of the flow
  - Proves that the augmenting paths algorithm finds a maximum flow
  - Gives an algorithms for finding the minimum cut
Better methods of for constructing a network flow

• Improved methods for finding augmenting paths or blocking flows
• Goldberg’s Preflow-Push algorithm
  – Text, section 7.4

\[ O(nm) \]

Efficient Network Flow Algorithms
Applications of Network Flow
Problem Reduction

- Reduce Problem A to Problem B
  - Convert an instance of Problem A to an instance of Problem B
  - Use a solution of Problem B to get a solution to Problem A
- Practical
  - Use a program for Problem B to solve Problem A
- Theoretical
  - Show that Problem B is at least as hard as Problem A
Problem Reduction Examples

Reduce the problem of finding the longest path in a directed graph to the problem of finding a shortest path in a directed graph.

1. Negate weights of graph. Call SoPoA.
2. Reverse signs. Return $P_{AH}$.

Construct an equivalent minimization problem.
Undirected Network Flow

- Undirected graph with edge capacities
- Flow may go either direction along the edges (subject to the capacity constraints)

Construct an equivalent flow problem
Multi-source network flow

- Multi-source network flow
  - Sources $s_1, s_2, \ldots, s_k$
  - Sinks $t_1, t_2, \ldots, t_j$

- Solve with Single source network flow
Bipartite Matching

A graph $G = (V, E)$ is bipartite if the vertices can be partitioned into disjoint sets $X, Y$.

A matching $M$ is a subset of the edges that does not share any vertices.

Find a matching as large as possible.
Application

- A collection of teachers
- A collection of courses
- And a graph showing which teachers can teach which courses
Converting Matching to Network Flow
Finding edge disjoint paths

Construct a maximum cardinality set of edge disjoint paths
Theorem

• The maximum number of edge disjoint paths equals the minimum number of edges whose removal separates s from t
Finding vertex disjoint paths

Construct a maximum cardinality set of vertex disjoint paths
Network flow with vertex capacities
Balanced allocation
Problem 9, Page 419

• To make a long story short:
  – N injured people
  – K hospitals
  – Assign each person to a hospital with 30 minutes drive
  – Assign N/K patients to each hospital
Baseball elimination

- Can the Dinosaurs win the league?
- Remaining games:
  - AB, AC, AD, AD, AD, BC, BC, BC, BD, CD

<table>
<thead>
<tr>
<th></th>
<th>W</th>
<th>L</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ants</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>Bees</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Cockroaches</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>Dinosaurs</td>
<td>6</td>
<td>5</td>
</tr>
</tbody>
</table>

A team wins the league if it has strictly more wins than any other team at the end of the season. A team ties for first place if no team has more wins, and there is some other team with the same number of wins.
Baseball elimination

- Can the Fruit Flies win or tie the league?
- Remaining games:

<table>
<thead>
<tr>
<th></th>
<th>W</th>
<th>L</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ants</td>
<td>2</td>
<td>17 12</td>
</tr>
<tr>
<td>Bees</td>
<td>3</td>
<td>16  7</td>
</tr>
<tr>
<td>Cockroaches</td>
<td>3</td>
<td>16  7</td>
</tr>
<tr>
<td>Dinosaurs</td>
<td>5</td>
<td>14  13</td>
</tr>
<tr>
<td>Earthworms</td>
<td>5</td>
<td>14  10</td>
</tr>
<tr>
<td>Fruit Flies</td>
<td>19</td>
<td></td>
</tr>
</tbody>
</table>
Assume Fruit Flies win remaining games

- Fruit Flies are tied for first place if no team wins more than 19 games
- Allowable wins
  - Ants (2)
  - Bees (3)
  - Cockroaches (3)
  - Dinosaurs (5)
  - Earthworms (5)
- 18 games to play

<table>
<thead>
<tr>
<th></th>
<th>W</th>
<th>L</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ants</td>
<td>17</td>
<td>13</td>
</tr>
<tr>
<td>Bees</td>
<td>16</td>
<td>8</td>
</tr>
<tr>
<td>Cockroaches</td>
<td>16</td>
<td>9</td>
</tr>
<tr>
<td>Dinosaurs</td>
<td>14</td>
<td>14</td>
</tr>
<tr>
<td>Earthworms</td>
<td>14</td>
<td>12</td>
</tr>
<tr>
<td>Fruit Flies</td>
<td>19</td>
<td>15</td>
</tr>
</tbody>
</table>
Remaining games

Solving problems with a minimum cut

- Image Segmentation
- Open Pit Mining / Task Selection Problem

$S, T$ is a cut if $S, T$ is a partition of the vertices with $s$ in $S$ and $t$ in $T$

The capacity of an $S, T$ cut is the sum of the capacities of all edges going from $S$ to $T$
Image Segmentation

- Separate foreground from background
- Reduction to min-cut problem

S, T is a cut if S, T is a partition of the vertices with s in S and t in T

The capacity of an S, T cut is the sum of the capacities of all edges going from S to T
Image analysis

- \(a_i\): value of assigning pixel \(i\) to the foreground
- \(b_i\): value of assigning pixel \(i\) to the background
- \(p_{ij}\): penalty for assigning \(i\) to the foreground, \(j\) to the background or vice versa
- \(A\): foreground, \(B\): background
- \(Q(A, B) = \sum_{i \in A} a_i + \sum_{j \in B} b_j - \sum_{\{(i, j)\} \in E, i \in A, j \in B} p_{ij}\)

\[
A^* = \sum_{i \in A} a_i \\
B^* = \sum_{j \in B} b_j \\
Q(A, B) = A^* + B^* - \text{cap}(A, B)
\]
Pixel graph to flow graph
Mincut Construction
Open Pit Mining
Application of Min-cut

- Open Pit Mining Problem
- Task Selection Problem
- Reduction to Min Cut problem

S, T is a cut if S, T is a partition of the vertices with s in S and t in T
The capacity of an S, T cut is the sum of the capacities of all edges going from S to T
Open Pit Mining

- Each unit of earth has a profit (possibly negative)
- Getting to the ore below the surface requires removing the dirt above
- Test drilling gives reasonable estimates of costs
- Plan an optimal mining operation
Mine Graph

-4 -3 -2 +3 = -6

+15 -12

Profit +3
Determine an optimal mine
Generalization

- Precedence graph $G=(V,E)$
- Each $v$ in $V$ has a profit $p(v)$
- A set $F$ is feasible if when $w$ in $F$, and $(v,w)$ in $E$, then $v$ in $F$.
- Find a feasible set to maximize the profit
Min cut algorithm for profit maximization

- Construct a flow graph where the minimum cut identifies a feasible set that maximizes profit
Precedence graph construction

- Precedence graph $G=(V,E)$
- Each edge in $E$ has infinite capacity
- Add vertices $s, t$
- Each vertex in $V$ is attached to $s$ and $t$ with finite capacity edges
Show a finite value cut with at least two vertices on each side of the cut.
The sink side of a finite cut is a feasible set

- No edges permitted from S to T
- If a vertex is in T, all of its ancestors are in T
Setting the costs

- If $p(v) > 0$,
  - $\text{cap}(v,t) = p(v)$
  - $\text{cap}(s,v) = 0$
- If $p(v) < 0$
  - $\text{cap}(s,v) = -p(v)$
  - $\text{cap}(v,t) = 0$
- If $p(v) = 0$
  - $\text{cap}(s,v) = 0$
  - $\text{cap}(v,t) = 0$
Enumerate all finite s,t cuts and show their capacities

\[ B = \text{Ben}(S) + \text{Ben}(T) \]

\[ \text{Cap}(S, T) = \text{Cost}(T) + \text{Ben}(S) + \text{Ben}(T) - \text{Ben}(T) \]

\[ S = \text{Cost}(T) + \text{Ben}(S) + \text{Ben}(T) - \text{Ben}(T) \]

\[ = \text{Cost}(T) - \text{Ben}(T) + B \]

\[ \text{Profit}(T) = B - \text{Cap}(s, T) \]
Summary

• Construct flow graph
  – Infinite capacity for precedence edges
  – Capacities to source/sink based on cost/benefit

• Finite cut gives a feasible set of tasks

• Minimizing the cut corresponds to maximizing the profit

• Find minimum cut with a network flow algorithm