Chapter 7
Network Flow

7.5 Bipartite Matching

Matching

- Input: undirected graph $G = (V, E)$
- $M \subseteq E$ is a matching if each node appears in at most one edge in $M$.
- Max matching: find a maximum cardinality matching.

Bipartite Matching

- Input: undirected bipartite graph $G = (L \cup R, E)$
- $M \subseteq E$ is a matching if each node appears in at most one edge in $M$.
- Max matching: find a maximum cardinality matching.

Diagram:

- Matching example:
  - Nodes $1, 2, 3, 4, 5$
  - Edges connecting nodes

- Bipartite matching example:
  - Nodes $L$ and $R$
  - Matching edges $1-2', 3-1', 4-5'$
Bipartite Matching

Bipartite matching.
- Input: undirected, bipartite graph $G = (L \cup R, E)$.
- $M \subseteq E$ is a matching if each node appears in at most one edge in $M$.
- Max matching: find a max cardinality matching.

```
$G$

$G'$
```

**Theorem.** Max cardinality matching in $G = \text{value of max flow in } G'$.

**Pf.** $\leq$
- Given max matching $M$ of cardinality $k$.
- Consider flow $f$ that sends 1 unit along each of $k$ paths.
- $f$ is a flow, and has cardinality $k$.

```
$G$

$G'$
```

**Bipartite Matching: Proof of Correctness**

```
$G$

$G'$
```

**Theorem.** Max cardinality matching in $G = \text{value of max flow in } G'$.

**Pf.** $\geq$
- Let $f$ be a max flow in $G'$ of value $k$.
- Integrality theorem $\Rightarrow k$ is integral and can assume $f$ is 0-1.
- Consider $M = \text{set of edges from } L \text{ to } R \text{ with } f(e) = 1$.
- Each node in $L$ and $R$ participates in at most one edge in $M$.
- $|M| = k$: consider cut $(L \cup s, R \cup t)$.

```
$G$

$G'$
```
**Perfect Matching**

**Def.** A matching $M \subseteq E$ is **perfect** if each node appears in exactly one edge in $M$.

**Q.** When does a bipartite graph have a perfect matching?

**Structure of bipartite graphs with perfect matchings.**
- Clearly we must have $|L| = |R|$.
- What other conditions are necessary?
- What conditions are sufficient?

**Marriage Theorem.** [Frobenius 1917, Hall 1935] Let $G = (L \cup R, E)$ be a bipartite graph with $|L| = |R|$. Then, $G$ has a perfect matching if $|N(S)| \geq |S|$ for all subsets $S \subseteq L$.

**Pf.**
- Suppose $G$ does not have a perfect matching.
- Formulate as a max flow problem and let $(A, B)$ be min cut in $G'$.
- By max-flow min-cut, $\text{cap}(A, B) \leq |L|$.
- Define $L_A = L \cap A$, $L_B = L \cap B$, $R_A = R \cap A$.
- $\text{cap}(A, B) = |L_A| + |R_A|$.
- Since min cut can’t use $\infty$ edges: $N(L_B) \subseteq R_A$.
- $|N(L_B)| \leq |R_A| \leq \text{cap}(A, B) - |L_B| \leq |L| - |L_B| = |L_A|$.
- Choose $S = L_A$. □

**Proof of Marriage Theorem**

Let $S = \{2, 4, 5\}$, $N(S) = \{2', 5'\}$.

$G'$

$L_A = \{2, 4, 5\}$
$L_B = \{1, 3\}$
$R_A = \{2', 5'\}$
$N(L_B) = \{2', 5'\}$

No perfect matching:
$S = \{2, 4, 5\}$
$N(S) = \{2', 5'\}$. 

No perfect matching:
$S = \{2, 4, 5\}$
$N(S) = \{2', 5'\}$. 

Notation. Let $S$ be a subset of nodes, and let $N(S)$ be the set of nodes adjacent to nodes in $S$.

**Observation.** If a bipartite graph $G = (L \cup R, E)$ has a perfect matching, then $|N(S)| > |S|$ for all subsets $S \subseteq L$.

**Pf.** Each node in $S$ has to be matched to a different node in $N(S)$.

<table>
<thead>
<tr>
<th>L</th>
<th>R</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
</tr>
</tbody>
</table>
Which max flow algorithm to use for bipartite matching?
- Generic augmenting path: $O(m \text{val}(f^*)) = O(mn)$.
- Capacity scaling: $O(m^2 \log C) = O(m^2)$.
- Shortest augmenting path: $O(m n^{1/2})$.

Non-bipartite matching.
- Structure of non-bipartite graphs is more complicated, but well-understood. [Tutte-Berge, Edmonds-Galai]
- Blossom algorithm: $O(n^5)$. [Edmonds 1965]
- Best known: $O(m n^{1/2})$. [Micali-Vazirani 1980]

---

**Disjoint Paths**

**Disjoint path problem.** Given a digraph $G = (V, E)$ and two nodes $s$ and $t$, find the max number of edge-disjoint $s$-$t$ paths.

**Def.** Two paths are edge-disjoint if they have no edge in common.

**Ex:** communication networks.

---

**Edge Disjoint Paths**

**Disjoint path problem.** Given a digraph $G = (V, E)$ and two nodes $s$ and $t$, find the max number of edge-disjoint $s$-$t$ paths.

**Def.** Two paths are edge-disjoint if they have no edge in common.

**Ex:** communication networks.
**Edge Disjoint Paths**

Theorem. Max number edge-disjoint s-t paths equals max flow value.

\[ \text{Pf. } \leq \]
- Suppose there are k edge-disjoint paths \( P_1, \ldots, P_k \).
- Set \( f(e) = 1 \) if \( e \) participates in some path \( P_i \); else set \( f(e) = 0 \).
- Since paths are edge-disjoint, \( f \) is a flow of value \( k \).  

\[ \text{Pf. } \geq \]
- Suppose max flow value is \( k \).
- Integrality theorem \( \Rightarrow \) there exists 0-1 flow \( f \) of value \( k \).
- Consider edge \( (s, u) \) with \( f(s, u) = 1 \).
  - by conservation, there exists an edge \( (u, v) \) with \( f(u, v) = 1 \)
  - continue until reach \( t \), always choosing a new edge
- Produces \( k \) (not necessarily simple) edge-disjoint paths.  

**Network Connectivity**

Network connectivity. Given a digraph \( G = (V, E) \) and two nodes \( s \) and \( t \), find min number of edges whose removal disconnects \( t \) from \( s \).

Def. A set of edges \( F \subseteq E \) disconnects \( t \) from \( s \) if all s-t paths uses at least one edge in \( F \).

**Edge Disjoint Paths and Network Connectivity**

Theorem. [Menger 1927] The max number of edge-disjoint s-t paths is equal to the min number of edges whose removal disconnects \( t \) from \( s \).

\[ \text{Pf. } \leq \]
- Suppose the removal of \( F \subseteq E \) disconnects \( t \) from \( s \), and \( |F| = k \).
- All s-t paths use at least one edge of \( F \). Hence, the number of edge-disjoint paths is at most \( k \).
Disjoint Paths and Network Connectivity

**Theorem.** [Menger 1927] The max number of edge-disjoint s-t paths is equal to the min number of edges whose removal disconnects t from s.

**Pf.**
- Suppose max number of edge-disjoint paths is k.
- Then max flow value is k.
- Max-flow min-cut ⇒ cut (A, B) of capacity k.
- Let F be set of edges going from A to B.
- |F| = k and disconnects t from s. □

Circulation with Demands

**Circulation with demands.**
- Directed graph $G = (V, E)$.
- Edge capacities $c(e), e \in E$.
- Node supply and demands $d(v), v \in V$.

**Def.** A circulation is a function that satisfies:
- For each $e \in E$: $0 \leq f(e) \leq c(e)$ (capacity)
- For each $v \in V$: $\sum_{e \in \text{in}} f(e) - \sum_{e \in \text{out}} f(e) = d(v)$ (conservation)

**Circulation problem:** given $(V, E, c, d)$, does there exist a circulation?

7.7 Extensions to Max Flow

**Necessary condition:** sum of supplies = sum of demands.

$$\sum_{v \in \text{in}} d(v) = \sum_{v \in \text{out}} d(v) = D$$

**Pf.** Sum conservation constraints for every demand node $v$. 

\[
\begin{align*}
\sum f(e) &= d(v) & \text{demand if } d(v) > 0; \\
\sum f(e) &= -d(v) & \text{supply if } d(v) < 0; \\
\sum f(e) &= 0 & \text{transshipment if } d(v) = 0
\end{align*}
\]
Circulation with Demands

Max flow formulation.

\[ G: \]

\begin{align*}
\text{supply} & : 3 & 10 & 6 \\
\text{demand} & : 4 & 9 & 7 & 4
\end{align*}

Add new source \( s \) and sink \( t \).

- For each \( v \) with \( d(v) < 0 \), add edge \((s, v)\) with capacity \(-d(v)\).
- For each \( v \) with \( d(v) > 0 \), add edge \((v, t)\) with capacity \( d(v) \).

Claim: \( G \) has circulation iff \( G' \) has max flow of value \( D \).

Claim: \( G \) has circulation iff \( G' \) has max flow of value \( D \).

\[ G': \]

\begin{align*}
\text{supply} & : 3 & 10 & 6 \\
\text{demand} & : 4 & 9 & 7 & 4
\end{align*}

Integrality theorem. If all capacities and demands are integers, and there exists a circulation, then there exists one that is integer-valued.

\[ \text{Pf.} \] Follows from max flow formulation and integrality theorem for max flow.

Characterization. Given \((V, E, c, d)\), there does not exists a circulation iff there exists a node partition \((A, B)\) such that \( \sum_{e \in \delta_{B}} d_{e} \geq \text{cap}(A, B) \).

\[ \text{Pf idea.} \] Look at min cut in \( G' \).

Feasible circulation.

- Directed graph \( G = (V, E) \).
- Edge capacities \( c(e) \) and lower bounds \( l(e) \), \( e \in E \).
- Node supply and demands \( d(v) \), \( v \in V \).

\[ \text{Def.} \] A circulation is a function that satisfies:

\begin{align*}
&\text{For each } e \in E: \quad l(e) \leq f(e) \leq c(e) \quad \text{(capacity)} \\
&\text{For each } v \in V: \quad \sum_{e \in \delta_{v}^+} f(e) - \sum_{e \in \delta_{v}^-} f(e) = d(v) \quad \text{(conservation)}
\end{align*}

Circulation problem with lower bounds. Given \((V, E, l, c, d)\), does there exists a a circulation?
Circulation with Demands and Lower Bounds

Idea. Model lower bounds with demands.
- Send \( i(e) \) units of flow along edge \( e \).
- Update demands of both endpoints.

\[
\begin{array}{c|c|c}
\text{lower bound} & \text{upper bound} \\
\hline
2 & 3 \\
\hline
\end{array}
\]

\[
\begin{array}{c|c|c}
\text{capacity} & \\
\hline
2 & \\
\hline
\end{array}
\]

Theorem. There exists a circulation in \( G \) iff there exists a circulation in \( G' \). If all demands, capacities, and lower bounds in \( G \) are integers, then there is a circulation in \( G \) that is integer-valued.

Pf sketch. \( f(e) \) is a circulation in \( G \) iff \( f'(e) = f(e) - i(e) \) is a circulation in \( G' \).

Survey Design

Survey design.
- Design survey asking \( n_1 \) consumers about \( n_2 \) products.
- Can only survey consumer \( i \) about product \( j \) if they own it.
- Ask consumer \( i \) between \( c_1 \) and \( c_1' \) questions.
- Ask between \( p_j \) and \( p_j' \) consumers about product \( j \).

Goal. Design a survey that meets these specs, if possible.

Bipartite perfect matching. Special case when \( c_1 = c_1' = p_1 = p_1' = 1 \).

7.8 Survey Design

Algorithm. Formulate as a circulation problem with lower bounds.
- Include an edge \((i, j)\) if customer own product \( i \).
- Integer circulation \( \iff \) feasible survey design.
7.10 Image Segmentation

Image Segmentation

- Central problem in image processing.
- Divide image into coherent regions.

Ex: Three people standing in front of complex background scene. Identify each person as a coherent object.

Foreground / background segmentation.
- Label each pixel in picture as belonging to foreground or background.
- $V$ = set of pixels, $E$ = pairs of neighboring pixels.
- $a_i \geq 0$ is likelihood pixel $i$ in foreground.
- $b_i \geq 0$ is likelihood pixel $i$ in background.
- $p_{ij} \geq 0$ is separation penalty for labeling one of $i$ and $j$ as foreground, and the other as background.

Goals.
- Accuracy: if $a_i > b_i$ in isolation, prefer to label $i$ in foreground.
- Smoothness: if many neighbors of $i$ are labeled foreground, we should be inclined to label $i$ as foreground.

Find partition $(A, B)$ that maximizes:

\[
\sum_{i \in A} a_i + \sum_{j \in B} b_j - \sum_{(i,j) \in E} p_{ij}
\]

or alternatively

\[
\sum_{i \in A} a_i + \sum_{j \in B} b_j + \sum_{(i,j) \in E} p_{ij}
\]

Formulate as min cut problem.
- Maximization.
- No source or sink.
- Undirected graph.

Turn into minimization problem.
- Maximizing

\[
\sum_{i \in A} a_i + \sum_{j \in B} b_j - \sum_{(i,j) \in E} p_{ij}
\]

is equivalent to minimizing

\[
\frac{(\sum_{i \in A} a_i + \sum_{j \in B} b_j)}{\text{constant}} - \sum_{i \in A} - \sum_{j \in B} + \sum_{(i,j) \in E}
\]

or alternatively

\[
\sum_{i \in A} a_i + \sum_{j \in B} b_j + \sum_{(i,j) \in E}
\]
Image Segmentation

Formulate as min cut problem.
- $G' = (V', E')$.
- Add source to correspond to foreground; add sink to correspond to background.
- Use two anti-parallel edges instead of undirected edge.

Consider min cut $(A, B)$ in $G'$.
- $A = \text{foreground}$.
- $\text{cap}(A, B) = \sum_{i \in A} \sum_{j \in B} p_{ij} + \sum_{i \in A, j \in B} p_{ij}$ if $i$ and $j$ on different sides, $p_{ij}$ counted exactly once.
- Precisely the quantity we want to minimize.

Project Selection

Projects with prerequisites.
- Set $P$ of possible projects. Project $v$ has associated revenue $p_v$.
  - Some projects generate money: create interactive e-commerce interface, redesign web page
  - Others cost money: upgrade computers, get site license
- Set of prerequisites $E$. If $(v, w) \in E$, can’t do project $v$ and unless also do project $w$.
- A subset of projects $A \subseteq P$ is \textbf{feasible} if the prerequisite of every project in $A$ also belongs to $A$.

Project selection. Choose a feasible subset of projects to maximize revenue.
Project Selection: Prerequisite Graph

Prerequisite graph:
- Include an edge from v to w if can’t do v without also doing w.
- \(\{v, w, x\}\) is feasible subset of projects.
- \(\{v, x\}\) is infeasible subset of projects.

![Graph showing prerequisite relationships]

Project Selection: Min Cut Formulation

Min cut formulation:
- Assign capacity \(\infty\) to all prerequisite edge.
- Add edge \((s, v)\) with capacity \(p_v\) if \(p_v > 0\).
- Add edge \((v, t)\) with capacity \(-p_v\) if \(p_v < 0\).
- For notational convenience, define \(p_s = p_t = 0\).

![Graph showing min cut formulation]

Claim. \((A, B)\) is min cut iff \(A - \{s\}\) is optimal set of projects.
- Infinite capacity edges ensure \(A - \{s\}\) is feasible.
- Max revenue because:
  \[\text{cap}(A, B) = \sum_{v \in A} p_v + \sum_{v \in B} (-p_v)\]
  \[= \sum_{p_v \geq 0} p_v - \sum_{p_v < 0} p_v\]

Open Pit Mining

Open-pit mining (studied since early 1960s)
- Blocks of earth are extracted from surface to retrieve ore.
- Each block v has net value \(p_v\) = value of ore – processing cost.
- Can’t remove block v before w or x.
7.12 Baseball Elimination

"See that thing in the paper last week about Einstein? . . . Some reporter asked him to figure out the mathematics of the pennant race. You know, one team wins so many of their remaining games, the other teams win this number or that number. What are the myriad possibilities? Who's got the edge?"

"The hell does he know?"

"Apparently not much. He picked the Dodgers to eliminate the Giants last Friday."

- Don DeLillo, Underworld

<table>
<thead>
<tr>
<th>Team</th>
<th></th>
<th>Wins</th>
<th>Losses</th>
<th>To play</th>
<th>Against = ( r_i )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Atlanta</td>
<td>83</td>
<td>71</td>
<td>8</td>
<td>Atl 1 Phi 6 1</td>
<td></td>
</tr>
<tr>
<td>Philly</td>
<td>80</td>
<td>79</td>
<td>3</td>
<td>1</td>
<td>0 2</td>
</tr>
<tr>
<td>New York</td>
<td>78</td>
<td>78</td>
<td>6</td>
<td>6 0</td>
<td>0</td>
</tr>
<tr>
<td>Montreal</td>
<td>77</td>
<td>82</td>
<td>3</td>
<td>1 2 0</td>
<td>-</td>
</tr>
</tbody>
</table>

Which teams have a chance of finishing the season with most wins?

- Philly can win 83, but still eliminated . . .
- If Atlanta loses a game, then some other team wins one.

**Remark.** Answer depends not just on how many games already won and left to play, but also on whom they're against.
Baseball Elimination

Baseball elimination problem.
- Set of teams $S$.
- Distinguished team $s \in S$.
- Team $x$ has won $w_x$ games already.
- Teams $x$ and $y$ play each other $r_{xy}$ additional times.
- Is there any outcome of the remaining games in which team $s$ finishes with the most (or tied for the most) wins?

Baseball Elimination: Max Flow Formulation

Theorem. Team 3 is not eliminated iff max flow saturates all edges leaving source.
- Integality theorem $\Rightarrow$ each remaining game between $x$ and $y$ added to number of wins for team $x$ or team $y$.
- Capacity on $(x, t)$ edges ensure no team wins too many games.

Baseball Elimination: Explanation for Sports Writers

Which teams have a chance of finishing the season with most wins?
- Detroit could finish season with $49 + 27 = 76$ wins.
Which teams have a chance of finishing the season with most wins?
- Detroit could finish season with 49 + 27 = 76 wins.

Certificate of elimination. \( R = \{ \text{NY}, \text{Bal}, \text{Bos}, \text{Tor} \} \)
- Have already won \( w(R) = 278 \) games.
- Must win at least \( r(R) = 27 \) more.
- Average team in \( R \) wins at least \( 305/4 > 76 \) games.

<table>
<thead>
<tr>
<th>Team</th>
<th>Wins ( w_i )</th>
<th>Losses ( l_i )</th>
<th>To play ( r_i )</th>
<th>Against + ( r_{ij} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>NY</td>
<td>75</td>
<td>59</td>
<td>28</td>
<td>-</td>
</tr>
<tr>
<td>Bal</td>
<td>71</td>
<td>63</td>
<td>28</td>
<td>3</td>
</tr>
<tr>
<td>Bos</td>
<td>69</td>
<td>66</td>
<td>27</td>
<td>8</td>
</tr>
<tr>
<td>Tor</td>
<td>63</td>
<td>72</td>
<td>27</td>
<td>7</td>
</tr>
<tr>
<td>Detroit</td>
<td>49</td>
<td>86</td>
<td>27</td>
<td>3</td>
</tr>
</tbody>
</table>

**Certificate of elimination.**
- \( T \subseteq S \), \( w(T) = \sum_{x \in T} w_x \), \( g(T) = \sum_{(x,y) \in T^*} g_{x,y} \).
- If \( \frac{w(T) + g(T)}{|T|} > w_z + g_z \), then \( z \) is eliminated (by subset \( T \)).

**Theorem.** [Hoffman-Rivlin 1967] Team \( z \) is eliminated iff there exists a subset \( T^* \) that eliminates \( z \).

**Proof idea.** Let \( T^* \) = team nodes on source side of min cut.

**Proof of theorem.**
- Use max flow formulation, and consider min cut \((A, B)\).
- Define \( T^* = \) team nodes on source side of min cut.
- Observe \( x - y \in A \) iff both \( x \in T^* \) and \( y \in T^* \).
- Infinite capacity edges ensure if \( x - y \in A \) then \( x \in A \) and \( y \in A \).
- If \( x \in A \) and \( y \in A \) but \( x - y \notin T \), then adding \( x - y \) to \( A \) decreases capacity of cut.

\[
\begin{align*}
g(S - \{ z \}) & > \text{cap}(A, B) \\
& = \frac{g(S - \{ z \}) - g(T^*)}{\sum_{(x,y) \in T^*} g_{x,y}} + \sum_{x \in T^*} (w_x + g_x - w_x) \\
& = g(S - \{ z \}) - g(T^*) - w(T^*) + |T^*| (w_z + g_z)
\end{align*}
\]

- Rearranging terms: \( w_z + g_z < \frac{w(T^*) + g(T^*)}{|T^*|} \)