Reductions to Network Flow Problem

- Bipartite Matching ≈ Network Flow
- The Gee-ball Problem ≈ Network Flow

Matching

- Let $G=(V,E)$ be a graph.
- $E' \subseteq E$ is a matching if every vertex of $V$ is incident to at most one edge of $E'$.

Matching Example

Bipartite Graph

- Let $G=(V,E)$ be a graph.
- $G$ is bipartite if $V$ can be partitioned into $V_1$ and $V_2$ such that no pair of vertices in $V_i$ (i=1,2) have an edge.

Bipartite Example

Bipartite Matching

- Input: Bipartite graph $G$
- Output: A largest matching of $G$
Bipartite Matching $\propto$ Network Flow

- Transform Input – Step 1

Bipartite Matching $\propto$ Network Flow

- Transform Input – Step 2

Reductions to Network Flow Problem

- Bipartite Matching $\propto$ Network Flow
- The Gee-ball Problem $\propto$ Network Flow

Matching

- Let $G=(V,E)$ be a graph.
- $E' \subseteq E$ is a matching if every vertex of $V$ is incident to at most one edge of $E'$. 
Matching Example

Let $G=(V,E)$ be a graph.

- $G$ is bipartite if $V$ can be partitioned into $V_1$ and $V_2$ such that no pair of vertices in $V_i$ ($i=1,2$) have an edge.

Bipartite Graph

Bipartite Example

Bipartite Matching

- Input: Bipartite graph $G$
- Output: A largest matching of $G$

Bipartite Matching $\Leftrightarrow$ Network Flow

- Transform Input – Step 1

Bipartite Matching $\Leftrightarrow$ Network Flow

Algorithm for Problem Bipartite Matching

Algorithm for Problem Network Flow

Diagram:

Input $\rightarrow$ Transform $\rightarrow$ Output

Diagram:

Input $\rightarrow$ Transform $\rightarrow$ Output

Diagram:

Input $\rightarrow$ Transform $\rightarrow$ Output

Diagram:

Input $\rightarrow$ Transform $\rightarrow$ Output
Bipartite Matching ∝ Network Flow

- Transform Input – Step 2

\[
\begin{array}{c}
\text{s} & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 \\
\end{array}
\]

Bipartite Matching ∝ Network Flow

- Transform output

\[
\begin{array}{c}
\text{s} & 1/1 & 1/1 & 1/1 & 1/1 \\
1/1 & 1/1 & 1/1 & 1/1 & 1/1 \\
\end{array}
\]

Reduction

- Is it correct?
- Is it efficient?

Integrality theorem

- If the capacities in a network are integral, then the max flow can be achieved with integral flows on each edge.
- Further the Ford-Fulkerson method yields an integral solution.

Proof of correctness

There is a 1-1 correspondence between 0/1 flows in the network and matchings in the input graph.
Reduction

- Is it correct?
- Is it efficient?
  \[ T_{BM}(n) = cn + T_{BF}(n+2) \]

Reductions to Network Flow Problem

- Bipartite Matching \( \preceq \) Network Flow
- The Gee-ball Problem \( \preceq \) Network Flow

The Gee-ball Problem

- The southwestern conference of the gee-ball league consists of \( n+1 \) teams. Team \( n+1 \) is from HMC.
- We want to know whether it is possible for HMC to win more games this season than any other team in the conference.
- No ties allowed.

Example

- The teams are Pitzer, CMC, Pomona, and HMC
- Games won so far:
  - Pitzer 4, CMC 3, Pomona 2, HMC 2
- Games to play:
  - 1 game: Pitzer vs. HMC
  - 2 games: Pomona vs. HMC

The Gee-ball Problem

- Teams \( t_1, t_2, \ldots, t_n, t_{n+1} \)
- So far this year team \( i \) has won \( w_i \) games.
- Teams \( i \) and \( j \) will play each other \( g_{ij} \) more times this season (\( g_{ij} = g_{ji} \)).

Gee-ball \( \preceq \) Network Flow

Gee-ball \( \preceq \) Network Flow

\[ \begin{align*}
\text{Input} & \quad \text{Algorithm for Gee-ball} \\
& \quad \text{Algorithm for Network Flow} \\
& \quad \text{Output}
\end{align*} \]
Transform Input

1. Create a source $s$.
2. Create vertex $v_{i,j}$ for $1 \leq i < j \leq n$
3. Create edge from $s$ to $v_{i,j}$ with capacity $g_{i,j}$

Let $w = w_{n+1} + \sum_{i=1}^{n} g_{n+1,i}$

4. Create vertices $u_i$, $1 \leq i \leq n$
5. Create edge from $v_{i,j}$ to $u_i$ and $u_j$ with infinite capacity.