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 $F$.

Ex: In testing network reliability
Thm. [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.

i) We showed that max number edge disjoint s-t paths = max flow.

ii) Max-flow Min-cut theorem => min s-t cut = max-flow

iii) For a s-t cut (A,B), cap(A,B) is equal to the number of edges out of A. In other words, every s-t cut (A,B) corresponds to cap(A,B) edges whose removal disconnects s from t.

So, max number of edge disjoint s-t paths = min number of edges to disconnect s from t.
Label each pixel as foreground/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.
- $p_{i,j} \geq 0$ 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:

$$\sum_{i \in A} a_i + \sum_{j \in B} b_j - \sum_{(i,j) \in E} p_{i,j}$$
Difficulties:
• Maximization (as opposed to minimization)
• No source or sink
• Undirected graph

Step 1: Turn into Minimization

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

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

Equivalent to minimizing
\[ + \sum_{j \in B} a_j + \sum_{i \in A} b_i + \sum_{(i,j) \in E} p_{i,j} \]
Min cut Formulation (cont’d)

G' = (V', E').  
Add s to correspond to foreground;  
Add t to correspond to background  
Use two anti-parallel edges instead of undirected edge.
Consider min cut \((A, B)\) in \(G'\). \((A = \text{foreground.})\)

\[
cap(A, B) = \sum_{j \in B} a_j + \sum_{i \in A} b_i + \sum_{(i,j) \in E, i \in A, j \in B} p_{i,j}
\]

Precisely the quantity we want to minimize.
Linear Programming
System of Linear Equations

Find a solution to

$$x_3 - x_1 = 4$$
$$x_3 - 2x_2 = 3$$
$$x_1 + 2x_2 + x_3 = 7$$

Can be solved by Gaussian elimination method in $O(n^3)$ when we have n variables/n constraints
Let $a$ be a column vector in $\mathbb{R}^d$ and $x$ a column vector of $d$ variables.

$$\langle a, x \rangle = a^T x = a_1 x_1 + a_2 x_2 + \cdots + a_d x_d$$

**Hyperplane:** A hyperplane is the set of points $x$ such that $\langle a, x \rangle = b$ for some $b \in \mathbb{R}$

**Halfspace:** A halfspace is the set of points on one side of a hyperplane.

$$\{x : \langle a, x \rangle \leq b\} \text{ or } \{x : \langle a, x \rangle \geq b\}$$
\[ 3x_2 + x_1 = 0 \]

\[ a = \left( \frac{1}{3} \right) \]
$3x_2 + x_1 \leq 0$

$a = \begin{pmatrix} 1 \\ 3 \end{pmatrix}$
\[3x_2 + x_1 \leq -3\]

\[a = \left(\frac{1}{3}\right)\]
Find the smallest point in a polytope
\begin{align*}
\min & \quad x_2 \\
\text{s. t.,} & \quad \langle a_1, x \rangle \leq b_1 \\
& \quad \langle a_2, x \rangle \geq b_2 \\
& \quad \langle a_3, x \rangle \geq b_3 \\
& \quad \ldots
\end{align*}
Optimize a linear function subject to linear inequalities

\[
\begin{align*}
\text{max} & \quad 3x_1 - 4x_3 \\
\text{s.t.,} & \quad x_1 + x_2 \leq 5 \\
& \quad x_3 - x_1 = 4 \\
& \quad x_3 - x_2 \geq -5 \\
& \quad x_1, x_2, x_3 \geq 0
\end{align*}
\]

• We can have equalities and inequalities,
• We can have a linear objective functions
Let \( a \) be a column vector in \( \mathbb{R}^d \) and \( x \) a column vector of \( d \) variables.

\[
\langle a, x \rangle = a^T x = a_1 x_1 + a_2 x_2 + \cdots + a_d x_d
\]

\[
A = \begin{bmatrix}
  a_1^T \\
  a_2^T \\
  \vdots \\
  a_m^T
\end{bmatrix}
\]

\[
Ax = \begin{pmatrix}
  \langle a_1, x \rangle \\
  \langle a_2, x \rangle \\
  \vdots \\
  \langle a_m, x \rangle
\end{pmatrix}
\]

\[
Ax \leq b
\]

\[
\langle a_1, x \rangle \leq b_1 \\
\langle a_2, x \rangle \leq b_2 \\
\vdots \\
\langle a_m, x \rangle \leq b_m
\]
Linear Programming Standard Form

\[ \begin{align*}
\text{max} & \quad \langle c, x \rangle \\
\text{s.t.,} & \quad Ax \leq b \\
x & \geq 0
\end{align*} \]

Any linear program can be translated into the standard form.

\[ \begin{align*}
\text{min} & \quad y_1 - 2y_2 \\
\text{s.t.,} & \quad y_1 + 2y_2 = 3 \\
& \quad y_1 - y_2 \geq 1 \\
& \quad y_1 \geq 0
\end{align*} \]

Replace \( y_2 \) with \( z_2 - z'_2 \)

\[ \begin{align*}
\text{max} & \quad -y_1 + 2(z_2 - z'_2) \\
\text{s.t.,} & \quad y_1 + 2(z_2 - z'_2) \leq 3 \\
& \quad -(y_1 + 2(z_2 - z'_2)) \leq -3 \\
& \quad -(y_1 - (z_2 - z'_2)) \leq -1 \\
& \quad y_1, z_2, z'_2 \geq 0
\end{align*} \]
Applications of Linear Programming

Generalizes: $Ax=b$, 2-person zero-sum games, shortest path, max-flow, matching, multicommodity flow, MST, min weighted arborescence, ...

Why significant?
• We can solve linear programming in polynomial time.
• Useful for approximation algorithms
• We can model many practical problems with a linear model and solve it with linear programming

Linear Programming in Practice:
• There are very fast implementations: IBM CPLEX, Gorubi in Python, CVX in Matlab, ....
• CPLEX can solve LPs with millions of variables/constraints in minutes
Example 1: Diet Problem

Suppose you want to schedule a diet for yourself. There are four categories of food: veggies, meat, fruits, and dairy. Each category has its own (p)rice, (c)alory and (h)appiness per pound:

<table>
<thead>
<tr>
<th></th>
<th>veggies</th>
<th>meat</th>
<th>fruits</th>
<th>dairy</th>
</tr>
</thead>
<tbody>
<tr>
<td>price</td>
<td>(p_v)</td>
<td>(p_m)</td>
<td>(p_f)</td>
<td>(p_d)</td>
</tr>
<tr>
<td>calorie</td>
<td>(c_v)</td>
<td>(c_m)</td>
<td>(c_f)</td>
<td>(c_d)</td>
</tr>
<tr>
<td>happiness</td>
<td>(h_v)</td>
<td>(h_m)</td>
<td>(h_f)</td>
<td>(h_d)</td>
</tr>
</tbody>
</table>

**Linear Modeling:** Consider a linear model: If we eat 0.5lb of meat, 0.2lb of fruits we will be \(0.5 \cdot h_m + 0.2 \cdot h_f\) happy

- You should eat 1500 calories to be healthy
- You can spend 20 dollars a day on food.

**Goal:** Maximize happiness?
Diet Problem by LP

- You should eat 1500 calories to be healthy
- You can spend 20 dollars a day on food.

**Goal:** Maximize happiness?

<table>
<thead>
<tr>
<th></th>
<th>veggies</th>
<th>meat</th>
<th>fruits</th>
<th>dairy</th>
</tr>
</thead>
<tbody>
<tr>
<td>price</td>
<td>$p_v$</td>
<td>$p_m$</td>
<td>$p_f$</td>
<td>$p_d$</td>
</tr>
<tr>
<td>calorie</td>
<td>$c_v$</td>
<td>$c_m$</td>
<td>$c_f$</td>
<td>$c_d$</td>
</tr>
<tr>
<td>happiness</td>
<td>$h_v$</td>
<td>$h_m$</td>
<td>$h_f$</td>
<td>$h_d$</td>
</tr>
</tbody>
</table>

\[
\begin{align*}
\text{max} & \quad x_v h_v + x_m h_m + x_f h_f + x_d h_d \\
\text{s.t.} & \quad x_v p_v + x_m p_m + x_f p_f + x_d p_d \leq 20 \\
& \quad x_v c_v + x_m c_m + x_f c_f + x_d c_d \leq 1500 \\
& \quad x_v, x_m, x_f, x_d \geq 0
\end{align*}
\]

#pounds of veggies, meat, fruits, dairy to eat per day
Components of a Linear Program

- Set of variables
- Bounding constraints on variables,
  - Are they nonnegative?
- Objective function
- Is it a minimization or a maximization problem
- LP Constraints, make sure they are linear
  - Is it an equality or an inequality?
Example 2: Max Flow

Define the set of variables
• For every edge $e$ let $x_e$ be the flow on the edge $e$

Put bounding constraints on your variables
• $x_e \geq 0$ for all edge $e$ (The flow is nonnegative)

Write down the constraints
• $x_e \leq c(e)$ for every edge $e$, (Capacity constraints)
• $\sum_{e \text{ out of } v} x_e = \sum_{e \text{ in to } v} x_e \quad \forall v \neq s, t$ (Conservation constraints)

Write down the objective function
• $\sum_{e \text{ out of } s} x_e$

Decide if it is a minimize/maximization problem
• $\text{max}$
Example 2: Max Flow

\[
\begin{align*}
\text{max} & \quad \sum_{e \text{ out of } s} x_e \\
\text{s.t.} & \quad \sum_{e \text{ out of } v} x_e = \sum_{e \text{ in to } v} x_e \quad \forall v \neq s, t \\
& \quad x_e \leq c(e) \quad \forall e \\
& \quad x_e \geq 0 \quad \forall e
\end{align*}
\]

Q: Do we get exactly the same properties as Ford Fulkerson? 
A: Not necessarily, the max-flow may not be integral
Example 3: Min Cost Max Flow

Suppose we can route 100 gallons of water from $s$ to $t$. But for every pipe edge $e$ we have to pay $p(e)$ for each gallon of water that we send through $e$.

**Goal:** Send 100 gallons of water from $s$ to $t$ with minimum possible cost

\[
\begin{align*}
\min & \quad \sum_{e \in E} p(e) \cdot x_e \\
\text{s.t.} \quad & \sum_{e \text{ out of } v} x_e = \sum_{e \text{ into } v} x_e \quad \forall v \neq s, t \\
& \sum_{e \text{ out of } s} x_e = 100 \\
x_e & \leq c(e) \quad \forall e \\
x_e & \geq 0 \quad \forall e
\end{align*}
\]
Linear Programming and Approximation Algorithms
Integer Program for Vertex Cover

Given a graph \(G=(V,E)\) with costs \(c_v\) on the vertices. Find a vertex cover of \(G\) with minimum cost, i.e., \(\min \sum_{v \in S} c_v\)

Write LP with Integrality Constraint:

- Variables: One variable \(x_v\) for each vertex \(v\)
- Bound: \(x_v \in \{0,1\}\)
- Edge cover Constraints: \(x_u + x_v \geq 1\) for every edge \((u,v) \in E\)
- Obj: \(\min \sum_v c_v x_v\)
IP for Vertex Cover

\[
\begin{align*}
\min & \quad \sum_v c_v x_v \\
\text{s.t.,} & \quad x_v + x_u \geq 1 \quad \forall (u, v) \in E \\
& \quad x_v \in \{0, 1\} \quad \forall v \in V
\end{align*}
\]

**Fact:** The optimum solution of the above program is min vertex cover.

**Pf:**

- First, any vertex cover \( S \), \( x_v = \begin{cases} 1 & \text{if } v \in S \\ 0 & \text{o.w.} \end{cases} \) is feasible.
- For any feasible solution \( x \), the \( S = \{v: x_v = 1\} \) is a vertex cover.

IP is NP-complete general! But there are fast algorithms in practice that often work.
LP Relaxation Vertex Cover

\[
\begin{align*}
\text{min} & \quad \sum_v c_v x_v \\
\text{s.t.,} & \quad x_v + x_u \geq 1 \quad \forall (u, v) \in E \\
& \quad 0 \leq x_v \leq 1 \quad \forall v \in V
\end{align*}
\]

Fact: \( \text{OPT-LP} \leq \text{Min Vertex Cover} \)

Pf: Min vertex cover is a feasible solution of the LP

Q: Can we hope to get an integer solution?
Bad Optimum solutions

\[
\begin{align*}
\min & \quad \sum_{v} c_v x_v \\
\text{s.t.,} & \quad x_v + x_u \geq 1 \quad \forall (u, v) \in E \\
& \quad 0 \leq x_v \leq 1 \quad \forall v \in V
\end{align*}
\]

A feasible solution:
Set \( x_v = 0.5 \) for all \( v \) in the complete graph

If \( c_v = 1 \) for all \( v \), then
Min vertex cover=\( n - 1 \)
But OPT LP=\( n/2 \).
Approximation Alg for Vertex Cover

Given a graph $G=(V,E)$ with costs $c_v$ on the edges. Find a vertex cover of $G$ with minimum cost, i.e., $\min \sum_{v \in S} c_v$

**Thm:** There is a 2-approximation Alg for weighted vertex cover.

**ALG:** Solve LP. Let $S = \{v: x_v \geq 0.5\}$. Output $S$.

**Pf:** First, for every edge $(u, v)$, $x_u + x_v \geq 1$ So at least one is in $S$. So, $S$ is a vertex cover.
Second,

$$\sum_{v \in S} c_v \leq \sum_{v \in S} c_v (2x_v) \leq 2 \text{OPTLP} \leq \text{Min Vertex Cov}$$