Lecture 20

Plan: 1) Finish LCIS algo.
2) Matroid intersection polytope
3) Start min-cost arborescence

After today, ~4 more lectures! ~3 ellipsoids

Matroid intersection polytope

Let $M_1 = (E, I_1)$ & $M_2 = (E, I_2)$ be two matroids, rank functions $r_1, r_2$.

Analogously to the matroid polytope, let

$$X = \{ 1 \leq \sum_{E \in R \subseteq E} r(S) : S \in I_1 \cap I_2 \}$$
ie. $X \in \mathbb{R}^E$ is set of indicators of common independent sets.

- Define the matroid intersection polytope $P_{M_1, M_2} \coloneqq \text{conv}(X)$ (can use to optimize linear functions over $X$).

- Main result: $P_{M_1, M_2}$ is the intersection $P_{M_1} \cap P_{M_2}$ of the matroid polytopes $P_{M_1}, P_{M_2}$ of $M_1, M_2$.

$$\Rightarrow \text{vertices}(P_{M_1}) \cap \text{vertices}(P_{M_2}) = \text{vertices}(P_{M_1 \cap M_2})$$

- This is surprising! In general, for polytopes $P_1, P_2$

$$\text{verts}(P_1) \cap \text{verts}(P_2) \neq \text{verts}(P_1 \cap P_2)$$

$$\Rightarrow \text{verts}(P_{M_1}) = \text{indicators of } I_1 \Rightarrow \text{intersection is common independent sets.}$$

$$= \text{vertices of } P_{M_1 \cap M_2} \text{ if } P_{M_1, M_2} = P_{M_1 \cap M_2}, \text{ same set of vertices.}$$
e.g. \( P_1, P_2 \) share no vertices but
\( P_1 \cap P_2 \neq \emptyset \)
(\& hence has vertices).

- Interm of inequalities?
- Recall matroid polytope:
  for \( r \) rank function of \( M \),
  \[
P_m = \{ x \in \mathbb{R}^E : x(S) \leq r(S) \ \forall S \subseteq E \}
  \]
  \[
i.e. \quad x \geq 0 \quad \forall e \in E
  \]
  \( P_m \cap P_{m_2} \) has both sets of constraints, so
1981: can efficiently decide membership in \( P_m, m_2 \).
Theorem: Let $P = P_m \cap P_{m_2}$, i.e.

$$P = \{ x \in \mathbb{R}^E : \begin{align*}
    x(s) &\leq c_1(s) \quad \forall s \\
    x(s) &\leq c_2(s) \quad \forall s \\
    x &\geq 0 \quad \forall e \in E \end{align*} \}$$

Then

$$P_{m,m_2} = P$$

i.e. $P_{m,m_2} = P_m \cap P_{m_2}$.

Proof: Plan: Similar to Lecture 17, vertex proof for matroid polytope.

- Like second proof for matroid polytope, use vertex integrality.
- Integrality suffices by the usual logic:
  - Clearly $\text{conv}(X) \subseteq P$, i.e. $X \subseteq P_m \cap P_{m_2}$. 
On the other hand, if $P$ integral then $P \subseteq \text{conv}(X)$ because integral points in $P_{m_1}, P_{m_2}$ are indicators of independent sets in $M_1, M_2$. ⇒ integral points in $P_{m_1} \cap P_{m_2} \subseteq P$ are common independent.

- Again, if $P = \{x : Ax \leq b, x \geq 0\}$, matrix $A$ is not totally unimodular, there are matrices.

- But submatrices describing vertices will be T.U. (enough).

Let $x^* \in \mathbb{R} \quad x^* = (1,1,\sqrt{2})$?

- We know $x^*$ characterized by which inequalities are tight for it.
For $i \in \{1, 2, 3\}$, let 
$$T_i = \left\{ S \subseteq E : x^*(S) = f_i(S) \right\}$$

i.e. $T_i$ sets of tight rank constraints in $M_i$.

Let $J = \left\{ e : x^*e = 0 \right\}$.

Then $x^*$ is unique solution to

$$T_1 \begin{cases} 
  x(S) = f_1(S) \quad \forall S \in T_1 \\
  x(S) = f_2(S) \quad \forall S \in T_2 \\
  xe = 0 \quad \forall e \in J.
\end{cases}$$
That is, $\{x^x\}$ is the intersection of two faces $F_1, F_2$ in $Pm_1, Pm_2$.

$$F_i = \{x \in Pm_i : x(s) = r_i(s) \land s \in \mathbb{S}, x_e = 0 \forall e \in J\}.$$ 

Recall from lem 17: $T_i$ can be replaced by a chain $C_i$ without changing $F_i$.

$$F_i = \{x \in Pm_i : x(s) = r_i(s) \land s \in \mathbb{S}, x_e = 0 \forall e \in J\}.$$
Thus, assume $x^*$ is solution to:

$x(s) = r_1(s)$ \forall s \in C_1,

$x(s) = r_2(s)$ \forall s \in C_2

$x_e = 0 \quad \forall e \in \mathcal{J}$. 
This is $Ax=b$ for $b \in \mathbb{R}^n$.

**Claim:** A T.U. $\Rightarrow x^*$ integral.

Why? Rows of $A$ are 1s of $S$ in chain $C_1$ or $C_2$.

**Example:**

$$
A = \begin{bmatrix}
1 & 1 & 1 \\
0 & 1 & 0 \\
0 & 0 & 1 \\
0 & 0 & 0
\end{bmatrix}
\begin{aligned}
\begin{cases}
3 C_1, \\
3 C_2
\end{cases}
\end{aligned}
$$

- We use discrepancy to prove Theorem 3.14 in polyhedral notes.
- Recall: A T.U. $\iff$ $A$ submatrices $A'$ of $A$, $\exists$ partition $R_1, R_2$ of rows of $A'$.
\[ E_{a_i} - E_{a_i} \text{ has } \{ -1, 0, +1 \}. \]

\[ \text{if } i \in R_1, \ i \in R_2 \]

\[ A' \]

\[ R_1 \]
\[ R_2 \]
\[ = \begin{array}{ccccccc}
1 & 0 & 0 & -1 & 1 & \ldots & 1
\end{array} \]

- Consider submatrix \( A' \) of \( A \) corresponds to subchains \( C_1', C_2' \) (same form as \( A \))

- Assign \( R_1, R_2 \) as follows:
  - Assign largest element of \( C_1' \) to \( R_1 \), then alternately assign remaining \\
  - elements of \( C_1' \) to \( R_2, R_1 \).
e.g., \[ C'_1 \begin{bmatrix} r_1 \\ r_2 \end{bmatrix} \pm r_1. \]

sum has entries in \( \mathbb{Z}_0, 13 \). 

\[ \begin{array}{c}
\text{\textbf{\textcircled{1}} For } C'_2, \text{ assign oppositely.} \\
\text{e.g., } C'_2 \begin{bmatrix} r_1 \\ r_2 \end{bmatrix} \pm r_1. \\
\text{sum has entries in } \mathbb{Z}_0, -13.
\end{array} \]

- Overall, sum has entries in \( \mathbb{Z}_0, 13 \).
- Complete the proof. \( \square \).

\underline{Matroid intersection optimization}
• Given a cost function $c : E \rightarrow \mathbb{R}$
  can we efficiently compute
  \[
  \max_{S \subseteq I, \lambda I} c(S) = \sum_{e \in S} c(e) = c \cdot 1_S .
  \]
equiv: optimize $c^T x$ over $x \in P_m, M_2$.

• For just one matroid: greedy alg works.

• For $C = 1$: just L.C.I.S.
  \((1, \ldots, 1) \text{ iter times} \)

• For perfect matching: Hungarian algo.
  e.g. min-cost p.m.

• Can also compute min cost L.C.I.S.
  Exercise: equiv. to max cost indep
  for $c' = K - c$ for $K$ large.
In general, YES, can efficiently compute

d-ellipsoid

d-complicated primal-dual algs.

\[ \rightarrow \text{strongly poly. time} \]

4 steps indep of \( c \)

\( \text{if arithmetic is unit cost.} \)

\text{Today!}

\text{simple primal-dual alg. for}

**Min-cost arborescence**

\text{Recall: given directed graph } D \&

\text{vertex } r, \text{ arborescence } A \text{ is a spanning tree in } D \text{ directed away from } r.

\text{e.g.}

\[ \text{Diagram of arborescence} \]
- **min-cost arborescence:**
  \[
  \min_{\mathcal{A}} \sum_{e \in \mathcal{A}} c(e) = c(A)
  \]
  for an arborescence \( e \in \mathcal{A} \)

**Example:**

- **Example:**
  - edges = roads to be fixed
  - \( r \) = distribution center
  - cost = expense of fixing road.

- **First, I.P. formulation:**
  - assume \( c \) nonnegative.
  \[
  \text{OPT} = \min_{\mathcal{A} \subseteq E} \sum_{e \in \mathcal{A}} c(e) x_e \quad x_e \in \mathbb{R}_+ \quad \forall e \in E
  \]

  Shrink \( x = 1_{\mathcal{A}} \)
  for A arborescence
  \( \sum_{e \in \mathcal{A}} c(e) x_e = c(A) \)
Subject to
\[ \sum_{e \in \delta^{-}(s)} x_e = 1 \quad \forall s \in V - r \]
\[ x_e \geq 1 \quad \forall e \in \delta^{-}(s) \]
\[ \sum_{e \in \delta^{+}(s)} x_e = 1 \quad \forall s \in V - r \]
\[ x_e \in \{0, 1\} \]

Indegrees = (except r).
x is an indicator.

Check: only solutions are \( s \)A where A is an arborescence.
in particular, all arbors satisfy constraints.

Miraculously, we'll show even w/out integrality constraint & indegree constraint, there's still an optimal solution that's an arborescence.
I.e., the following LP has optimizer \( \mathbf{A} \) s.t. \( \mathbf{A} \) is an arborescence.

\[
\text{LP} = \min \sum c(e) x_e \\
\text{subject to } \sum_{e \in \delta^+(s)} x_e \geq 1 \quad \forall s \in \mathcal{V} - \{r\} \\
\text{(primal)} \quad x_e \geq 0 \quad \forall e \in \mathcal{E} 
\]

(note LP \( \leq \) OPT w/c LP has fewer constraints.
(hypo in pre-lecture!)
"Symmetric version"

Dual LP is
\[ LP = \max \sum_{s \in S^V} y_s \]

subject to
\[ \sum_{s \in S^V} y_s \leq c(e) \quad \forall e \in E \]

\[(\text{dual}) \quad \sum_{s \in S^V} y_s \geq 0 \quad \forall s \in S.\]

- **Algorithm sketch:** construct
  - arb. \( A \)
  - dual. feas. \( y \)
  - satisfying complementary slackness

Then \( c(A) = LP \), but \( LP \leq \text{OPT} \)

\( c(A) \leq \text{OPT} \Rightarrow c(C(A)) = \text{OPT} \)
• Complementary slackness for $x \in \mathcal{L}A$, $y$ says:

a.) $y_A > 0 \Rightarrow |A \cap \delta^-(S)| = 1$

b.) $e \in A \Rightarrow \exists y s = c(e), S : e \in \delta^-(S)$

• Two phases of algorithm:

1) Construct

A dual feas $y$

A set $F$ of edges s.t.

every vertex of $V \in$ reachable from $r$ in $F$

$F$ might not be an arborescence.

& $y_A, A \subseteq F$ satisfy (b).
2) Remove unnecessary edges from $F$, get arborescence which satisfies both (a) & (b).

**Phase 1**

Initialize $F = \emptyset$, $j = 0$

counter $k = 1$

D While not everything reachable from $r$ in $F$

D select $S \subseteq V - r$

i) $F$ strongly connected in $S$

(every vertex can reach every other using only edges contained entirely in $S$)

ii) $F \cup \delta^-(S) = \emptyset$

$S$ is a “source” in decomp. of $F$ into s.c.c.'s.

(digraph has decomp. where if s.c.c.'s are contracted, left with DAG).
S is a subset of vertices, F subset of edges, does $S \subseteq$ vertices "touched by" F? not necessarily, initially.
\[ \text{increase } y \text{ s until new inequality } \sum y \leq c(e_k) \]
\[ \text{S: } e_k \in \delta(s) \]
becomes an equality. (\(y\) remain dual feas, b/c it was before)

\[ F \leftarrow F + e_k, \quad k \leftarrow k+1 \]
new \(F, y\) don't violate \((b)\) because \(e_k\) is tight.

\[ \text{Return } F, y \text{ satisfying } (b), \]
& eventl. reachable from \(r\) in \(F\).
Phase 2: eliminate as many edges as we can in reverse order they were added.

\[ \text{For } i = K \ldots 1:\]
\[ \text{If } F - e_i \text{ contains a directed path from } r \text{ to every vertex, } F \leftarrow F - e_i. \]

\[ \text{Return } A = F \]

Claim 1: \( A \) is an arborescence

Proof: We'll show \( |A| = N1 - 1 \) and \( d^{-}(v) = 1 \) for \( v \in V - r \)

- If indegree \( < 1 \) for \( v \neq r \), contradicts reachability in \( A \)
- If \( |A| > |N1 - 1 \) then collision \( e_i \)
Suppose $i < j \Rightarrow$ in reverse delete, would have removed $e_j$. \( b/c \) any vertex reached thru $e_j$ is reachable thru $e_i$. \qed

Finally:

**Claim 2:** Condition (a) of complementary slackness holds.

a.) $\gamma_s > 0 \Rightarrow |A \cap \delta^{-}(s)| = 2$. 

**PF:** Assume not $\exists S$ s.t. $\gamma_S > 0$ & $|A \cap \delta^{-}(s)| > 1$. 

$S$ was chosen at some step $e$ of phase 1 when we added $ee$ to $F$. 

• F had no other edges in \( \delta(s) \) when \( e_e \) was added. (by construction)

\[ \Rightarrow \text{all edges of } A \cap \delta(s) \text{ are } e_j \text{ for } j > i. \]

• When \( S \) chosen, \( F \) strongly connected within \( S \)

\[ \Rightarrow S \text{ strongly connected using only } e_i \text{ if } i < l. \]
Subclaim: All $e_j, j > k$ should have been removed in Phase 2.

**Why?** Suppose $e_j$ necessary to visit some vertex $v$

- Let $P : r \rightarrow v$ path using $e_j$
- Let $w$ last vertex in $S$ on $P$

**Note:** $P$ first enters $S$ along $e_j$, else could shortcut $e_j$ because $S$ stays connected.
• Because $e_2$ is necessary at step 1 of Phase 2, there must be another path $Q$ through $e_k$.

Similarly: $Q$ must enter $S$ first through $e_k$.

• Can use $Q$ to shortcut $e_j$; thus $e_j$ not necessary.