today: finish Δ-graphs & M₃-net, dual SVMstruct

\[ M = L \text{ for } \triangle \text{-graphs (continuation):} \]

- If \( G \) is triangulated \( \Rightarrow \) construct a junction tree

\( G: \)

\[ \text{junction tree for } G: \]

- \( \text{every maximal clique of } G \) is a node of the JT.

\( L \): clique tree, \( G \) contains all cliques along unique path from \( C_i \) to \( C_j \) in JT.

- \( v \in C \cap C_2 \) then \( v \) belongs to all cliques along unique path from \( C_i \) to \( C_j \) in JT.

To build JT: first build weighted complete graph on cliques where weight on edges is \( |C_i \cap C_j| \):
(see Def. 10.6 in Koller & Friedman)

We need to show that
\[ f(x) \circ \mu = A \circ \mu \circ B \circ \mu \]

Let's say, want to show $M_B(y_B) = \sum y' \circ (y_B) \circ y' \\ y': y_B^*, y_B = y_B$

\[ \sum y': y_B^*, y_B = y_B \left( \sum y: M_B(y_B) \right) \]

\[ y': y_B^*, y_B = y_B \left( \sum y: M_B(y_B) \right) \]

\[ \sum y: M_B(y_B) \left( \sum y: M_B(y_B) \right) \]
Teaching Page 3

Maximize \( f(x) = \sum_{i \in I} a_i^T x + c^T y \) subject to \( x \in \mathcal{X} \) and \( y \in \mathcal{Y} \).

Let \( \text{max} \{ f(x) : x \in \mathcal{X} \} = \text{max} \{ \sum_{i \in I} a_i^T x : x \in \mathcal{X} \} \).

Given a \((I,J)\)-elimination tree, we can find \( y^* \) by solving the LP

\[
\text{max} \left\{ \sum_{i \in I} a_i^T x : x \in \mathcal{X}, \sum_{j \in J} x_j = 1 \right\}
\]

for each \( i \in I \) and \( j \in J \).

For \( f(x) = \sum_{i \in I} a_i^T x + c^T y \) with \( c \geq 0 \), we have

\[
\text{max} \{ f(x) : x \in \mathcal{X}, y \in \mathcal{Y} \} = \text{max} \{ \sum_{i \in I} a_i^T x : x \in \mathcal{X} \} + \text{max} \{ c : y \in \mathcal{Y} \}.
\]

Overall, suppose that \( l(i,y) \) decomposes as shown on page 6.

\[
\text{max} \{ l(i,y) - w^T y(i,y) : y(i,y) \in \mathcal{Y}^I \} = \max_{y \in \mathcal{Y}} \left\{ \sum_{i \in I} a_i^T x + (c + \mathbf{w})^T \mathbf{u} - \mathbf{w}^T \mathbf{y}^* \right\},
\]

where \( \mathbf{w} \) is the weight vector.

Last class:

1. Solve this LP on a developmentally structured SVM.
2. Use Mosek, CPLEX, etc. to solve to 1e-16 accuracy.
3. Saddle point formulation

\[
\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} f(x,y) = \min_{x \in \mathcal{X}} \left\{ \sum_{i \in I} a_i^T x + c^T y : y \in \mathcal{Y} \right\}
\]

The extragradient method would require projecting on \( \mathcal{X} \) (local consistency polytope).
Do check: can you use message passing for that?

and there is a version of gradient algorithm which only requires LMOS "linear minimization oracles" on L's instead of projection (saddle pt, Frank-Wolfe algorithm) \rightarrow we'll come back later

interior point method:

\[
\min f(x) \\
\text{s.t. } Ax = b, \quad x \geq 0
\]

(can transform \( A^T \tilde{z} \leq b \) \( \Rightarrow A^T \tilde{z} + \tilde{g} = \tilde{b}, \quad \tilde{z} \geq 0 \) to \( Ax = b \) and \( \tilde{z} \geq 0 \) ... Mackie transformation)

\[
\text{minimize } f(x) - t \log x \approx g_t(x) \\
\text{s.t. } Ax = b
\]

\( \text{dom}(g_t(x)) = \{ x; x > 0 \} \)

\[
\begin{aligned}
\text{feasible domain } & \\
& \text{argmin } g_t(x)
\end{aligned}
\]

utilize point method: run Newton's method with warm start on \( g_t(x) \) and decrease \( t \) with geometric schedule

Convergence: Newton's method takes \( \log(\log(\log(\log(\log(1/e)))) + C_1/(2^t) - f^\infty) \) iterations to reach \( \varepsilon \) accuracy

\# iterations

IPM: \( \log(1/e) \cdot \text{some constant overall convergence} \)

\( + \text{dimensionality is appearing; but no condition number of } f \) (vs. gradient method)

\( \text{cost on iteration } \tilde{g}_t(Y^t) \)
cost per teacher is O(d^2)