CMPUT 675: Topics in Combinatorics and Optimization **Fall 2016**

Lecture 13 (Oct 5): Edmonds-Gallai Decomposition, Intro to LPs

Lecturer: Zachary Friggstad *Scribe: Chris Martin*

In the previous lecture, we saw Edmonds' Blossom algorithm for finding maximum matchings in general graphs. In this lecture we finish some of the discussion around this algorithm, and then introduce Linear Programming.

13.1 Edmonds-Gallai Decomposition

There are two final results to present in relation to the Blossom algorithm. Let $\mathcal{G} = (V; E)$ be any undirected graph. For notational convenience, for any vertex set $X \subseteq V$, we use $\mathcal{G} \backslash X$ to denote the graph obtained by removing the vertices of X from $\mathcal{G}$, and all incident edges.

Recall that $\nu(\mathcal{G})$ is the size of a maximum matching in $\mathcal{G}$, and for any $X \subseteq V$, $q_{\mathcal{G}}(X)$ is the number of odd connected components in $\mathcal{G} \backslash X$. We first show a nice fact about maximum matchings in any general graph.

Corollary 1 $\nu(\mathcal{G}) = \min_{X \subseteq V} \frac{|V| - (q_{\mathcal{G}}(X) - |X|)}{2}$.

Proof. For any matching M and any $X \subseteq V$, we have

$$|M| \leq \frac{|V| - (q_{\mathcal{G}}(X) - |X|)}{2},$$

since at least $q_{\mathcal{G}}(X) - |X|$ vertices are exposed by M.

We now just need to show equality holds for any maximum matching M and some $X \subseteq V$. Let $\mathcal{G}' = (V', E')$ be the graph obtained after a series of stem and blossom contractions in $\mathcal{G}$ (so some $v \in V'$ may correspond to an entire stem/blossom in $\mathcal{G}$), removing any loops or parallel edges. Let $M' \subseteq M$ be the set of edges of M that remain in $\mathcal{G}'$ (i.e. the edges of M that did not match two vertices in the same contracted component).

Suppose we find a maximal M'-alternating forest F' in $\mathcal{G}'$; let X' be the set of odd-height vertices, Y' be the set of even-height vertices, and W' be the nodes of V' not in F'. Recall that the M'-exposed nodes of F' have height 0, their children have height 1, etc. From calculations following the previous lecture,

$$|M'| = \frac{|V'| - (q_{\mathcal{G}'}(X') - |X'|)}{2},$$

so we just need to show the equality still holds in the uncontracted graph. Let X be all the vertices in V that are either in X' or were contracted into some component in X'. Note that $q_{\mathcal{G}}(X) = q_{\mathcal{G}'}(X')$, since all contracted blossoms/stems contain an odd number of vertices, and for any contracted component C, $N(C) \subseteq X$. Counting the number of edges of M present in each contracted component, we then see that

$$|M| = \frac{|V| - (q_{\mathcal{G}}(X) - |X|)}{2},$$

completing the proof. ■

The second result we do not prove here, but it follows by careful inspection of the sets X', Y', W' found in Corollary 1.

Theorem 1 (Edmonds-Gallai decomposition) *Let $\bar{Y}$ be all non-mandatory nodes in $\mathcal{G}$, $\bar{X} = N(\bar{Y})$, and $\bar{W} = V - (\bar{X} \cup \bar{Y})$. Then:*

1. *Every component of $\bar{Y}$ is factor-critical,*
2. *Every maximum matching yields a perfect matching in each component of $\bar{W}$,*
3. *Every maximum matching matches each $v \in \bar{X}$ into distinct components of $\bar{Y}$, and*
4. *For X', Y', W' as obtained in Corollary 1, expand all contracted components and let the resulting sets be X, Y, W; then, $X = \bar{X}$, $Y = \bar{Y}$, and $W = \bar{W}$.*

The fourth is really just stating how to compute this decomposition in polynomial time.

13.2 Linear Programming

We now give a brief introduction to Linear Programming.

Definition 1 *A **linear program** (LP) in standard form consists of the following:*

1. *n variables, represented as the column vector $x \in \mathbb{R}^n_{\geq 0}$,*
2. *m linear constraints $a_i^T x \leq b_i$, where $a_i \in \mathbb{R}^n$, $b_i \in \mathbb{R}$, and $1 \leq i \leq m$, and*
3. *an objective function $c^T x$, $c \in \mathbb{R}^n$.*

The goal is to find some $\bar{x} \in \mathcal{P} = \{x \in \mathbb{R}^n_{\geq 0} : a_i^T x \leq b_i, \forall 1 \leq i \leq m\}$, maximizing $c^T \bar{x}$.

We may also write an LP in the following way, which will be more common in this course:

$$\begin{aligned} \text{maximize:} \quad & c^T x \\ \text{subject to:} \quad & Ax \leq b \\ & x \geq 0 \end{aligned} \tag{\textbf{LP}}$$

The inequalities above are element-wise; for any two k-vectors u, v, we say $u \leq v$ if $u_i \leq v_i$ for all $1 \leq i \leq k$, and similarly for $\geq$. The matrix A is the $m \times n$ *constraint matrix*, defined as

$$A = \begin{pmatrix} a_1^T \\ a_2^T \\ \dots \\ a_m^T \end{pmatrix}.$$

We similarly define the i-th entry of the vector $b \in \mathbb{R}^m$ to be b_i. Thus, $Ax \leq b$ captures all of our linear constraints.

Definition 2 *Some $\bar{x} \in \mathbb{R}^n_{\geq 0}$ is an **extreme point** in $\mathcal{P}$ if for all $y \in \mathbb{R}^n$, $y \neq 0$, either $\bar{x} - y$ or $\bar{x} + y$ is not in $\mathcal{P}$.*

A nice property of LPs, which we will discuss in more detail in future lectures, is that if an optimum solution to an LP exists, then there is always an optimum solution that is an extreme point of the LP. This property will become crucial when we consider solving LPs.

We can define linear programs more generally to allow the following. We also state how to convert each such LP into one in standard form.

- We may have constraints of the form $A_i \cdot x \geq b_i$. Rewrite as $-A_i \cdot x \leq -b_i$.
- We may allow equality constraints $A_i \cdot x = b_i$. Rewrite with two constraints $A_i x \leq b_i, -A_i x \leq -b_i$.
- The objective may be to minimize $c^T x$. Instead, maximize $-c^T x$.
- We may not have $x_i \geq 0$ for some i. Replace all occurrences of x_i with $(x_i^+ - x_i^-)$ using two new variables x_i^+, x_i^- and add the constraint $x_i^+, x_i^- \geq 0$.