In this lecture, we will start by reviewing basic concepts and definitions for linear programming. The objective of this lecture is to explore whether linear programs capture the structure of the several problems we have been studying, such as MSTs, min-weight arborescences, and graph matchings.

We shall show that there exist "small" linear programs that solve min-cost perfect matchings in bipartite graphs. This will motivate us to suggest a linear program for min-cost matchings in general graphs.

1 Linear Programming

We start with some basic definitions and results in Linear Programming. We will use these results while designing our linear program solutions for min-cost perfect matchings, min-weight arborescences and MSTs. This will be a sufficient jumping-off point for the contents of this lecture; however a introduction to the subject can be found in [**?**].

Definition 7.1. Let $\vec{a} \in \mathbb{R}^n$ be a vector and let $b \in \mathbb{R}$ a scalar. Then a *half-space* in $\mathbb{R}^n$ is a region defined by the set $\{\vec{x} \in \mathbb{R}^n \mid \vec{a} \cdot \vec{x} \geq b\}$.

The following figure is an example of a half space $S = \{\vec{x} \mid \begin{bmatrix}1\\1\end{bmatrix} \cdot \vec{x} \geq 3\}$ in $\mathbb{R}^2$:

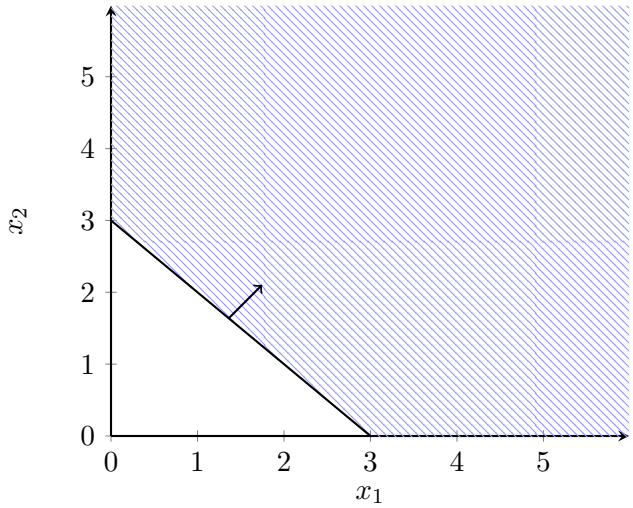

Figure 7.1: The half-space in $\mathbb{R}^2$ given by the set S

Definition 7.2. A *polyhedron* in $\mathbb{R}^n$ is the intersection of a finite number of half spaces.

A polyhedron is a convex region which is defined by finitely many linear constraints. A polyhedron in n dimensions with m constraints is often written compactly as

$$K = \{Ax \leq b\},$$

where A is an m by n constraint matrix, x is an n by 1 vector of variables, and b is an m by 1 vector of constants.

Definition 7.3. A *polytope* $K \in \mathbb{R}^n$ is a bounded polyhedron.

In other words, a polytope is polyhedron such that there exists some radius $R > 0$ such that $K \subseteq B(\mathbf{0}, R)$. The following is an example of a polytope (where the bounded region of the polytope is highlighted by ▨):

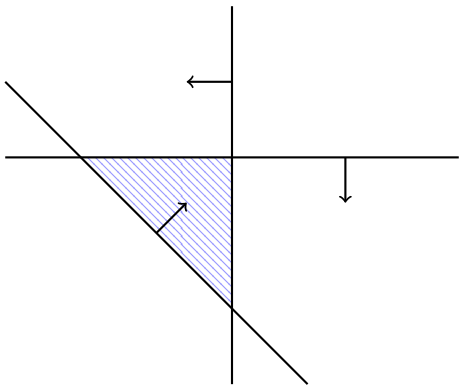

Figure 7.2: The polytope in $\mathbb{R}^2$ given by the constraints $-x_1 - x_2 \leq 1$, $x_1 \leq 0$, and $x_2 \leq 0$.

We can now define a linear program in terms of a polyhedron.

Definition 7.4. For some integer n, a polyhedron K, and an n by 1 vector c, a *linear program* in n dimensions is the linear optimization problem

$$\min_{x \in K} \{c \cdot x\}.$$

Although all linear programs can be put into this canonical form, in practice they may have many different presentations. These presentations can be shown to be equivalent to one another by adding new variables and constraints, negating the entries of A and c, etc. For example, the following are all linear programs:

$$\max_x \{c \cdot x : Ax \leq b\} \qquad \min_x \{c \cdot x : Ax = b\} \qquad \min_x \{c \cdot x : Ax \geq b\}.$$

We also note that K need not be bounded to have a solution. For example, the following linear program has a solution even though the polyhedron is unbounded:

$$\min_x \{x_1 + x_2 \mid x_1 + x_2 \geq 1\}. \tag{7.1}$$

We now introduce three different classifications of points that appear in polytopes.

Definition 7.5. Given a polytope K, a point $x \in K$ is an *extreme point of* K if there do not exist distinct $x_1, x_2 \in K$, and $\lambda \in [0, 1]$ such that $x = \lambda x_1 + (1 - \lambda)x_2$.

In other words, x is an extreme point of K if it cannot be written as the convex combination of two other points in K. See Figure **??** for an example.

Now we move to another definition about points in K.

Definition 7.6. A point $x \in K$ is a *vertex of* K if there exists an n by 1 vector $c \in \mathbb{R}^n$ such that $c^\intercal x < c^\intercal y$ for all $y \in K$ $y \neq x$.

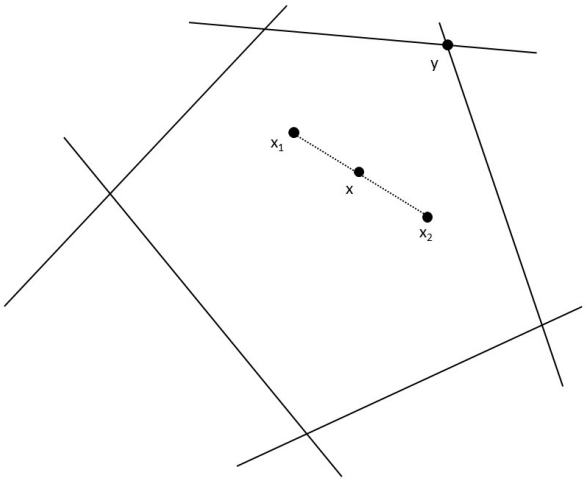

Figure 7.3: Here y is an extreme point, but x is not.

In other words, a vertex is the unique optimizer of some linear objective function. Note that there may be a linear program that does not have any vertices, as in Equation **??**. Any assignment to x_1 and x_2 such that $x_1 + x_2 = 1$ minimizes the objective $x_1 + x_2$, but none of these solutions are strictly better than the others. Additionally, no other objective function has a minimum in $x_1 + x_2 \geq 1$.

Now we consider one last definition about points in K.

Definition 7.7. Given a polytope $K \in \mathbb{R}^n$, a point $x \in K$ is a *basic feasible solution (bfs) to* K if there exist n linearly independent constraints in K which x satisfies at equality.

For example, let $K = \{x : Ax \leq b\}$ such that all constraints $a_i^\intercal x \leq b_i$ are linearly independent. Then x^* is a basic feasible solution if there exist n indices i for which $a_i^\intercal x^* = b_i$, and $a_i^\intercal x^* \leq b_i$ for all other i (x^* must satisfy all constraints because it is in K).

As you may have guessed by now, these three definitions are all related. In fact, they are all equivalent.

Fact 7.8. *Given a polyhedron K and a point $x \in K$, the following are equivalent:*

1. *x is a basic feasible solution,*
2. *x is an extreme point, and*
3. *x is a vertex.*

The proof is straightforward, and we will not present it here. Let us instead proceed directly to the main fact for this section.

Fact 7.9. *For a polytope K and a linear program $LP = \min\{c^\intercal x \mid x \in K\}$, there exists an optimal solution $x^* \in K$ such that x^* is an extreme point/vertex/bfs.*

This fact suggests an algorithm for LPs when K is a polytope: simply find all of the extreme points/vertices/bfs's and pick the one that gives the minimum solution. There are only $\binom{m}{n}$ vertices

to check in K, where m is the total number of constraints and n is the dimension (because we pick n constraints out of m to make tight).

Note that Fact **??** can be proven with weaker conditions than K a polytope, but for our purposes polytopes will suffice.

We also note that in general there can be infinitely many solutions; Fact **??** states only that there exists an optimal solution that is an extreme point/vertex/bfs.

We finish off this section with a final definition, which will help us to construct an LP for bipartite matching in the next section.

Definition 7.10. Given $x_1, x_2, \ldots, x_N \in \mathbb{R}^n$, the *convex hull* of $x_1, \ldots, x_N$ is

$$\mathrm{CH}(x_1, \ldots, x_N) = \left\{ x \in \mathbb{R}^n \;\middle|\; \exists \lambda_1, \ldots, \lambda_N \geq 0 \text{ s.t. } \sum_{i=1}^{N} \lambda_i = 1 \text{ and } x = \sum_{i=1}^{N} \lambda_i x_i \right\} \tag{7.2}$$

In words, the convex hull of $x_1, \ldots, x_N$ is the intersection of all convex sets containing $x_1, \ldots, x_N$. Another way of developing intuition for convex hulls is with the following "definition"; the convex hull defined by the points $x_1, \ldots, x_N \in \mathbb{R}^n$ is the smallest set of points that contain $x_1, \ldots, x_N$ and have the property that any path from one point in the set to another never leaves the set.

From that description, it is easy to see that every convex hull of finitely many points is a polytope. We also know the following fact:

Fact 7.11. *Given a polytope K, then $K = CH(x \in \mathbb{R}^n \mid x$ is an extreme point of $K)$.*

Proof. Let us denote $CH = \mathrm{CH}(x \in \mathbb{R}^n \mid x$ is an extreme point of $K) \subseteq K$. Clearly $K \subseteq CH$, since K is convex. For the other direction, consider some $x \in K \setminus CH$ and take $z \in CH$ to be the point witnessing $dist(x, CH) > 0$ (since CH is closed, this z is well-defined). Then x is an extreme point with respect to a linear objective c parallel to the vector $x - z$, which implies that $x \in CH$ after all. □

This captures the insight that polytopes may be represented in terms of their extreme points or their bounding half-planes. Much of the remainder of these notes will involve traversing between these two methods of representation.

2 Perfect Matchings in Bipartite Graphs

We now return to the problem of finding a minimum-cost perfect matching (which we have considered in previous lectures), beginning with bipartite graphs $G = (L, R, E)$. With an eye towards creating a linear program that finds perfect matchings, let us denote the matchings in G as bit-vectors in $\{0,1\}^{|E|}$. For example, see Figure **??**.

This allows us to define an $|E|$-dimensional polytope containing all perfect matchings on G. Let us try the obvious route for constructing such a polytope and see if it gives an efficient LP:

$$C_{PM} = CH(x \in \{0,1\}^{|E|} \mid x \text{ represents a perfect matching in } G)$$

The LP to find the min cost perfect matching of a bipartite graph with edge weights given by c is

$$\min\{c \cdot x \mid x \in C_{PM}\}.$$

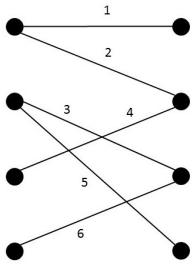

Figure 7.4: This graph has one perfect matching using edges 1, 4, 5, and 6, represented by the vector $(1, 0, 0, 1, 1, 1)$.

Then the solution will be at a vertex of C_{PM}, which by construction represents a perfect matching.

The great news is that we do not have to worry about fractional solutions to this LP, because every bfs in C_{PM} is a $\{0, 1\}$ vector, and there are are LP solvers that return bfs whenever they exist. Unfortunately there is one problem: it is a huge pain to write down C_{PM}. In fact C_{PM} can potentially have exponentially many vertices, because our graph can have exponentially many perfect matchings (e.g: a complete bipartite graph).

Can we find a more compact way to describe C_{PM}? In pursuit of a tractable polytope, let us try the following idea:

$$K_{PM} = \left\{ x \in \mathbb{R}^{|E|} \text{ s.t. } \begin{cases} \sum_{r \in N(l)} x_{lr} = 1 & \forall l \in L \\ \sum_{l \in N(r)} x_{lr} = 1 & \forall r \in R \\ x_e \geq 0 & \forall e \in E \end{cases} \right\}$$

The constraints of this polytope enforce that the weights of the edges leaving each vertex sum to 1, so it seems plausible that it is a polytope that precisely captures perfect matchings. This would be much easier to use in an LP, so let us fortify our optimism and show that $K_{PM} = C_{PM}$.

Theorem 7.12. *For any simple bipartite graph and derived K_{PM} and C_{PM}, $K_{PM} = C_{PM}$.*

To show this we start with the easy direction, $C_{PM} \subseteq K_{PM}$. Define χ_M as the indicator vector for the edges in a matching M on G.

Fact 7.13. $C_{PM} \subseteq K_{PM}$.

Proof. Clearly, for all perfect matchings M we have that $\chi_M \in K_{PM}$ since any perfect matching satisfies the constraint that one edge in M is incident to every vertex. Since K_{PM} is convex, it follows that

$$C_{PM} = \text{CH}(\chi_M \mid M \text{ is a perfect matching}) \subseteq K_{PM}$$

□

In order to prove that $K_{PM} \subseteq C_{PM}$, because of Fact **??** it suffices to show that all extreme points/vertices/bfs's of K_{PM} belong to C_{PM}.

Proof. Suppose x^* is an extreme point of K_{PM}. We must show that $x^* \in C_{PM}$, and to do so we will consider the support of x^*. Let $supp(x^*)$ denote the edges for which $x^*_e \neq 0$. First we will prove that $supp(x^*)$ is acyclic.

Suppose that $supp(x^*)$ contains a cycle of edges $x_1, x_2, \ldots, x_l$. Since the graph is bipartite, l is even. All of these edges are in the support, so each have nonzero weight. Then there exists an ϵ such that for all x_i, the weight of x_i is greater than ϵ.

Then we can create a new point $x^*_1 \in K$ by adding ϵ to the weight of each x_i where i is odd, and subtracting ϵ to the weight of each x_i, where i is even. Similarly, we define x^*_2 by adding ϵ from the even i's, and subtracting ϵ form the odd i's. But then $x^* = \frac{1}{2}x^*_1 + \frac{1}{2}x^*_2$, violating our assumption that x^* is an extreme point. See Figure **??**.

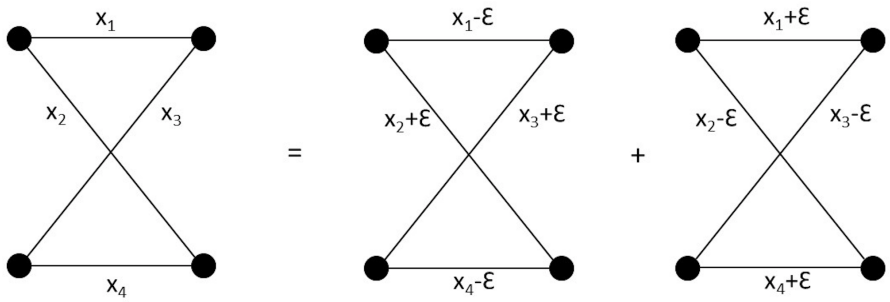

Figure 7.5: There cannot be a cycle in $supp(x^*)$, because this violates the assumption that x^* is an extreme point.

Therefore there are no cycles in $supp(x^*)$. So there must be a leaf vertex v in the support. Then, by the equality constraint at v, the single edge leaving v must have weight 1. But this edge goes to another vertex u, and because x^* is in K_{PM}, then this vertex cannot have any other edges in $supp(x^*)$ without violating its constraint. So u and v are a matched pair in x^*. Now remove u and v from consideration. We have introduced no cycles into the remainder of $supp(x^*)$, so we may perform the same step inductively to show that x^* is a perfect matching. Therefore $x^* \in C_{PM}$. □

Next we will relax the assumption that there is a perfect matching in G, and see what can be said.

2.1 Min-cost Matchings

At this point we can find min-cost perfect matchings, but what if our graph has no perfect matchings? Can we still find min-cost matchings? [1]

It turns out the answer is yes. In fact, only a slight modification of our linear constraints will give us a polytope whose vertices are matchings. For an in input graph that is bipartite, i.e $G = (L, R, E)$, let us define K_{Match}:

[1] This is not a trivial problem since the edge weights can be negative. In fact, that's how we would find max-weight matchings–by negating the weights.

$$K_{Match} = \left\{ x \in \mathbb{R}^{|E|} \text{ s.t. } \begin{cases} \sum_{j \in R} x_{ij} \leq 1 & \forall i \in L \\ \sum_{j \in L} x_{ji} \leq 1 & \forall i \in R \\ x_{i,j} \geq 0 & \forall i, j \end{cases} \right\}$$

We leave it as an exercise to apply the techniques used in the perfect matchings case to show that the vertices of K_{Match} are matchings of G, and indeed that $K_{Match} = CH_{Match}$, where CH_{Match} is the convex hull of all matchings in G.

Theorem 7.14. *For any bipartite G and derived K_{Match} and CH_{Match}, $K_{Match} = CH_{Match}$*

Proof. Left as an exercise. □

3 Arborescences and MSTs

Now that we have seen how to use LPs to find min-cost perfect matchings in bipartite graphs, let us see if we can use the ideas in the previous section as inspiration in the design of LPs for finding min-weight arborescences and MSTs.

3.1 Min-weight Arborescences

Recall the definition of a *r-arborescence*:

Definition 7.15. A *r-arborescence* of a digraph $G = (V, A)$ with root vertex $r \in V$ is a collection of arcs $B \subseteq A$ such that

1. Each vertex has one outgoing arc, except for r.
2. There exists a directed path from each vertex to r.

The weight of an arborescence is the sum of its arc weights. Given a digraph $G = (V, A)$ with root vertex $r \in V$, let us define the LP that induces the polytope K_{Arb} of all min-weight arborescences of G. As was the case for K_{PM}, for each edge $(i, j) \in A$ we will define a variable $x_{ij} \in \mathbb{R}^{|A|}$.

$$K_{Arb} = \left\{ x \in \mathbb{R}^{|A|} \text{ s.t. } \begin{cases} \sum_{u \in \delta^+(v)} x_{vu} = 1 & \forall v \neq r \\ \sum_{i \in S, j \notin S} x_{ij} \geq 1 & \forall S \subseteq V, S \not\ni r \\ x_{ij} \geq 0 & \forall (i, j) \in A \end{cases} \right\}$$

The first and third set of inequalities should be intuitive. The second set of inequalities says that for every subset of the vertices that does not contain the root, there must exist at least one edge that leaves this subsets. This is included so as to enforce item (2) in Definition **??**.

We can use the same techniques as in the previous section to show that $K_{Arb} = CH_{Arb}$, where CH_{Arb} is the convex hull formed by all r-arborscences of G.

Theorem 7.16. *For G a directed graph with non-negative edge weights and corresponding K_{Arb} and CH_{Arb} as defined above, $K_{Arb} = CH_{Arb}$*

Proof. Left as an exercise. □

3.2 Minimum Spanning Trees (a.k.a MSTs)

For a simple, undirected graph $G = (V, E)$, we may in a similar fashion construct K_{MST}, the polytope in $\mathbb{R}^{|E|}$ whose vertices are exactly the MSTs of G:

$$K_{MST} = \left\{ x \in \mathbb{R}^{|E|} \text{ s.t. } \begin{cases} \sum_{i \in S, j \notin S} x_{ij} \geq 1 & \forall S \subset V, S \neq \emptyset \\ \sum_{i,j \in S} x_{ij} \leq |S| - 1 & \forall S \subseteq V, S \neq \emptyset \\ \sum_{i,j \in V} x_{ij} = |V| - 1 & \\ x_{ij} \geq 0 & \forall i, j \in V \end{cases} \right\}$$

Notice that the first constraint excludes the case where $S = V$. The first set enforces the inequalities that for all subsets S of the vertex set that are not the empty set or the full vertex set, there should be an edge leaving S which forces the subgraph to be connected. The second and third set of inequalities should be intuitive. Define the convex hull of all minimum spanning trees of G to be CH_{MST}. Then, somewhat predictably, we will again find that $CH_{MST} = K_{MST}$.

Theorem 7.17. *For G an unweighted and undirected graph with K_{MST} and CH_{MST} as defined above, $K_{MST} = CH_{MST}$*

Proof. Omitted. □

Given the integrality of these polyhedra, it seems that min-weight r-arborescences or MSTs is similar to finding min-cost perfect matchings in bipartite graphs. But notice that the polytopes K_{Arb} and K_{MST} can have exponentially many constraints. Thus, it could take exponential time to simply write down the LP let alone solve it!

Of course, we know how to solve MSTs and min-cost arborescences without using linear programs: we saw algorithms for these very problems in the first two lectures of the course! But there is still hope, even via the linear programming approach. In fact, there are special algorithms, called separation oracles, which, given a point $x \in \mathbb{R}^n$, can reply YES if the point is within the desired polytope and NO if not. These separation oracles are of course specific to each LP, and for MSTs and r-arborescences they run in polynomial time in the size of the dimension (in our case this is the number of edges). These separation oracles can be used by a polytime algorithm called Ellipsoid that can solve LPs in general. Thus, there exist polytime algorithms for finding the vertices of K_{MST} and K_{Arb}.

4 Perfect Matchings in General Graphs

Finally, let us consider perfect matchings in non-bipartite graphs. We define a polytope that is similar to the bipartite polytope. Let

$$K_{PM} = \left\{ x \in \mathbb{R}^m \text{ s.t. } \begin{cases} \sum_{u \in \delta(v)} x_{vu} = 1 & \forall v \in V \\ x_e \geq 0 & \forall e \in E \end{cases} \right\}$$

Unfortunately, this is not a convex combination of all perfect matchings. For example, a K_3 with each edge weight $x_e = \frac{1}{2}$ will satisfy the constraints of this polytope, but there aren't even any perfect matchings in K_3.

This suggests that we need to add more constraints to K_{PM}. For a set of vertices S, let $\delta(S)$ denote the edges leaving S.

$$\left\{ \sum_{e \in \delta(S)} x_e \geq 1 \qquad \forall S \text{ such that } |S| \text{ is odd}, \qquad \right\}$$

The above set of constraints is required because an odd size vertex set cannot have a perfect matching. Thus, at least one edge from the perfect matching must leave S. Adding these constraints to the existing K_{PM} yields the correct polytope. In fact, the proof follows from Tutte's theorem.

Thus, we have that the following LP construction, due to J.Edmonds [**?**, **?**], induces the polytope K_{genPM} whose vertices are the perfect matchings of a general graph $G = (V, E)$.

$$K_{\text{genPM}} = \left\{ x \in \mathbb{R}^{|E|} \text{ s.t. } \left\{ \begin{array}{ll} \sum_{u \in \delta(v)} x_{vu} = 1 & \forall v \in V \\ \sum_{e \in \delta(S)} x_e \geq 1 & \forall S \text{ s.t. } |S| \text{ odd} \\ x_e \geq 0 & \forall e \in E \end{array} \right\} \right.$$

Notice that this LP again contains a potentially exponential number of constraints, in contrast with the polynomial number of constraints needed for the bipartite case. In fact, a theorem by Rothvoss [**?**] shows that any polytope whose vertices are the perfect matchings of the complete graph on n vertices must contain an exponential number of constraints.

Theorem 7.18. *For an undirected graph G and K_{genPM} as defined above, and CH_{genPM} the convex hull of all perfect matchings of G, $K_{genPM} = CH_{genPM}$.*

Proof. We give a sketch of the proof.

Let x^* be a basic feasible solution in K_{PM}. We would like to show that x^* is a perfect matching. Since x^* is a bfs, there are m linearly independent tight constraints. If there is some edge such that $x^*_e = 0$, then drop e, and argue about $G - e$, i.e., G without edge e. Similarly if $x^*_e = 1$, then drop $e = (u, v)$, and induct on $G \setminus \{u, v\}$. After this process, for all v, there must exist at least two edges in supp(v). If all vertices have support degree 2, then $supp(x6*)$ must contain a cycle. This will cause a contradiction with x^* an extreme point, in the same manner as in the proof of Theorem **??** for bipartite graphs. Therefore, there exists a vertex with degree at least 3 in $supp(x^*)$. But then the number of edges in $supp(x^*)$ is greater than the number of vertices. From this, we can show there is at least one $\sum_{e \in \partial(S)} x_e \geq 1$ constraint that is tight – call it S^*.

Finally, take S^* and shrink it down to one vertex. Then we induct on both S^* and on the remaining graph S^* contracted to a single vertex. □

5 An Aside

We have seen that LPs are a powerful way of formulating problems like min-cost matchings, min-weight r-aborescences, and MSTs. We reasoned about the structure of the polytopes that underly

the LPs, and we were able to show that these LPs do indeed solve their combinatorial problems. But notice that simply forming the LP is not sufficient–significant effort was expended to show that these polytopes do indeed have integer solutions at the vertices [2]. Without this guarantee, we could get fractional solutions to these LPs that do not actually give us solutions to our problem.

There is a substantial field of study concerned with proving the integrality of various LPs. We will briefly introduce a matrix property that implies the integrality of corresponding LPs. Recall that an LP can be written as

$$[A]_{m\times n} \cdot \vec{x} \leq \vec{b}$$

where A is a $m \times n$ matrix with each row corresponding to a constraint, $\vec{x}$ is a vector of n variables, and $\vec{b} \in \mathbb{R}^m$ is a vector corresponding to the m scalars $b_i \in \mathbb{R}$ in the constraint $A^{(i)} \cdot \vec{x} \leq b_i$.

Definition 7.19. A matrix $[A]_{m\times n}$ is called *totally unimodular* if every square submatrix B of A has the property that $\det(B) \in \{0, \pm 1\}$

We then have the following neat theorem, due to Hoffman and Kruskal:

Theorem 7.20. *([?]) If the constraint matrix $[A]_{m\times n}$ is totally unimodular and the vector $\vec{b}$ is integral, i.e: $\vec{b} \in \mathbb{Z}^m$, then, the vertices of the polytope induced by the LP are integer valued.*

Thus, to show that the vertices are indeed integer valued, one need not go through producing combinatorial proofs, as we have. Instead, one could just check that the constraint matrix A is totally unimodular.

Here's an interesting presentation about the relation between total unimodularity and graph matchings: http://wwwhome.math.utwente.nl/~uetzm/do/DO_Lecture6.pdf.

Acknowledgments

These lecture notes were scribed by Gregory Kehne, based on previous scribe notes of Colin White, Aditya Krishnan.

[2] i.e. that the solution vector is integer valued