15-850: Advanced Algorithms **CMU, Spring 2017**
Lecture #7: Matchings in Graphs: Linear Programs and Integrality of Polytopes
February 8, 2017 - February 4, 2015
Lecturer: Anupam Gupta Scribe: Colin White, Aditya Krishnan

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 most importantly graph matchings.

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

Definition 7.1. Let $\vec{a} \in \mathbb{R}^n$ be a vector and let $b \in \mathbb{R}$ be some 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 in $\mathbb{R}^2$ given by the set: $\{\vec{x} \mid \begin{bmatrix}1\\1\end{bmatrix} \cdot \vec{x} \geq 5\}$

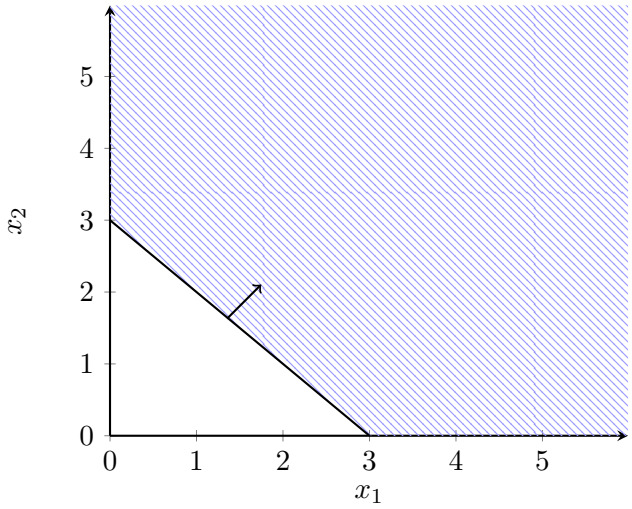

Figure 7.1: Example of a half-space in $\mathbb{R}^2$

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 satisfies some number of 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 matrix of constants, 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 polyhedron such that $\exists R > 0$ where $K \subseteq B(\mathbf{0}, R)$.

In other words, a polytope is a bounded polyhedron. The following is an example of a polytope; the bounded region of the polytope is highlighted by 

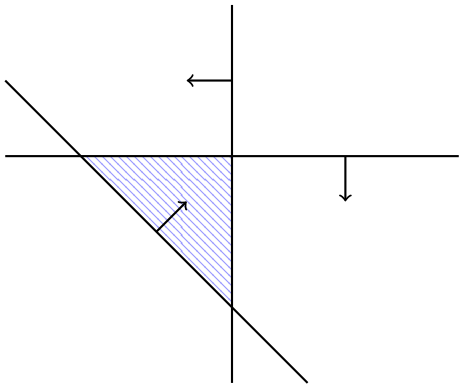

Figure 7.2: Example of a polytope in $\mathbb{R}^2$

Now we can 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

$$\text{minimize } \sum_{i=1}^{n} c_i x_i \text{ subject to } \vec{x} \in K$$

We can also have linear programs that maximize some objective function. Just flip the sign of all components of c. 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:

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

Now we will present three different definitions about types of points that may appear in a polytope.

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

In other words, an extreme point of K cannot be written as the average of two other points in K. See Figure 7.3 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 $y \neq x$, $y \in K$.

So, a vertex is the unique optimizer for some objective function. Note that there may be a linear program that does not have any vertices, as in Equation 7.1. Any assignment to x_1 and x_2 such that $x_1 + x_2 = 1$ minimizes $x_1 + x_2$, but none of these are strictly better than other points on that line. And no other objective function has a minimum in $x_1 + x_2 \geq 1$.

Now we consider one last definition about points in K.

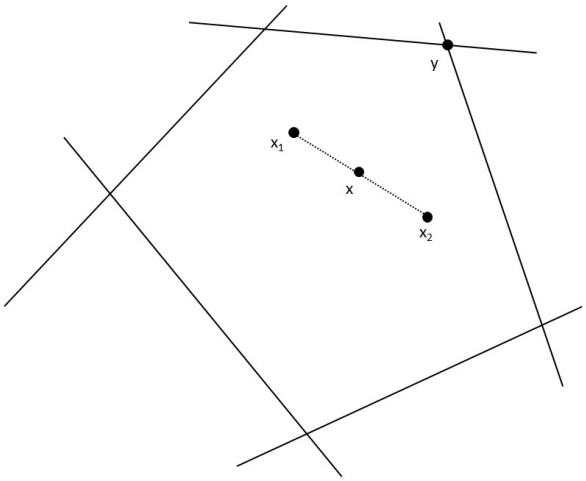

Figure 7.3: y is an extreme point, but x is not.

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 $K = \{a_i^\intercal x \leq b_i\}$ such that all constraints are linearly independent. Then x^* is a basic feasible solution if there exist n values of i such that $a_i^\intercal x^* = b_i$, and for the other values of i, $a_i^\intercal x^* \leq b_i$ (x^* must satisfy all constraints because it is in K).

As you may have guessed by now, the last three definitions are all related. In fact, the following fact shows they are all equivalent.

Fact 7.8. *Given a polyhedron K, and a point $x \in K$. Then 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. Now we will show the main fact for this section.

Fact 7.9. *For a polytope K and an 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: just 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 7.9 can be proven with weaker conditions than K being a polytope, but in this lecture, we will stick to polytopes.

Also note that when the objective function is perpendicular to a constraint, then there could be infinitely many solutions, but Fact 7.9 just states that there exists one optimal solution that is an extreme point/vertex/bfs.

We finish off this section with one more definition, which will help us 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

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

In words, the convex hull of points $x_1, \ldots, x_n$ 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 maximal 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 is also 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 $K)$.*

2 Perfect Matchings in Bipartite Graphs

Now we go back to the problem of finding a min cost perfect matching (which we have considered in previous lectures). For now, we will stick to bipartite graphs $G = (L, R, E)$.

Let us denote the matchings in G as bit-vectors in $\{0,1\}^{|E|}$ For example, see Figure 7.4.

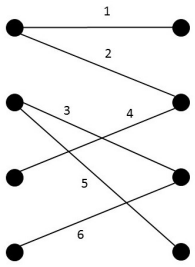

Figure 7.4: This graph has one perfect matching using edges 1, 4, 5, and 6, so we can represent it as $[1, 0, 0, 1, 1, 1]$.

This allows us to define an $|E|$ dimensional polytope which contains all perfect matchings. Let us try the obvious choice for 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 defined in vector c is

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

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

This is great news, because we do not have to deal with a fractional solution to the LP because every vertex in C_{PM} is a $\{0,1\}$ vector. But, there is one problem. It is a huge pain to write down C_{PM}. In fact, C_{PM} could potentially have exponentially many vertices because out graph could have exponentially many perfect matchings (e.g: complete bipartite graph).

Can we find a more compact way to write C_{PM} down? Let's try the following idea.

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

This polytope enforces that the weights of edges leaving every vertex is 1, so it seems plausible that it is a polytope for perfect matching. This would be much easier to use in an LP, so now we would like to show that K_{PM} is the same as C_{PM}.

Theorem 7.12. *$K_{PM} = C_{PM}$.*

We start with the easy direction, $C_{PM} \subseteq K_{PM}$. Define $\chi_{\mathcal{M}}$ as the indicator function for the edges in a matching $\mathcal{M}$.

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

Proof. Clearly, for all perfect matchings $\mathcal{M}$, $\chi_{\mathcal{M}} \in K_{PM}$ since a perfect matching satisfies the constraints that an edge weight of 1 leaves every vertex. It follows that

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

□

Now we must show that $K_{PM} \subseteq C_{PM}$. It suffices to show that all extreme points/vertices/bfs's of K_{PM} belong to C_{PM} because of Fact 7.11. We will prove this three ways, using the three definitions from the last section.

Proof. Extreme points:

Suppose x^* is an extreme point of K_{PM}. We must show that $x^* \in C_{PM}$. 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 $x_1, x_2, \ldots, x_l$. Since the graph is bipartite, l is even. All of these vertices 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 $> \epsilon$.

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

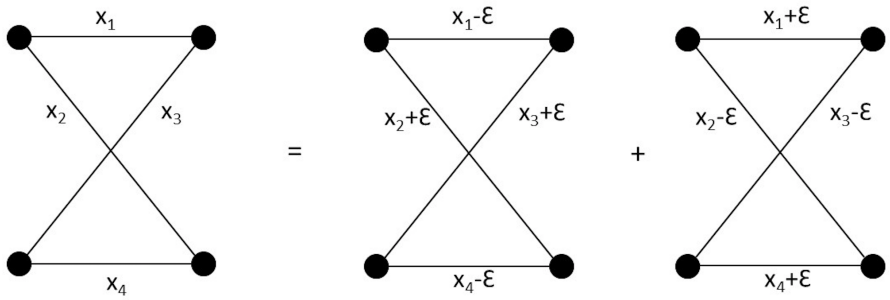

Figure 7.5: There cannot be a cycle in $supp(x^*)$, because this violates the assumption that x^* is an extreme point.

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 7.5.

Therefore, there are no cycles in the support of x^*. So there must be a leaf v in the support. Then 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 without violating its constraint. So u and v are a matched pair. Now take u and v out of the graph. In the remaining graph, there cannot be a cycle for the same reason as before, so we perform the same logic inductively to show that x^* is a perfect matching. Then $x^* \in C_{PM}$. □

Our second proof was covered in the last lecture.

Proof. Vertices:

Suppose $x^* \in K_{PM}$ is a vertex of K_{PM}. Then it is an optimizer to some objective function. Recall that in the last lecture, we showed that any unique cost function must be a perfect matching. So $x^* \in C_{PM}$. □

Now we give a proof using the definition of bfs. Recall that K_{PM} contains $2n+m$ constraints: one constraint for each of the $2n$ vertices forcing the weight of edges leaving that vertex to sum to 1, and then m constraints for nonzero edge weights.

Proof. Basic feasible solutions:

Let x^* be a basic feasible solution of K_{PM}. Then there exist m linearly independent constraints in K_{PM} which are tight.

Assume that none of these tight constraints were a nonzero edge weight constraints. Then all the tight constraints were forcing the edge weights coming out of a vertex to be 1. However, these $2n$ constraints cannot all be linearly independent.

To see this, sum up all of the constraints for vertices in L. $\sum_l \sum_r x_{lr} = n$. But notice that this is exactly the same as summing up all of the constraints for vertices in R: $\sum_r \sum_l x_{lr} = n$. Therefore, these $2n$ constraints cannot be linearly independent. So at most $2n-1$ of the linearly independent tight constraints in x^* can belong to the first $2n$ constraints.

Then $\geq m-(2n-1)$ constraints from $x_{lr} \geq 0$ are tight. So $\geq m-(2n-1)$ edges have $x_{lr} = 0$, so $|supp(x^*)| \leq 2n-1$. It follows that there must be an edge with length 1. If we pull out this edge, we can inductively perform the same argument on the smaller graph, to show that x^* is a perfect matching. This completes the proof. □

2.1 Min-cost Matchings

We looked at finding min-cost perfect matchings, but, what if our graph had no perfect matchings? Can we still find min-cost matchings? [1]

Yes. In fact, 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}.

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

We will leave it as an exercise to the reader to use the techniques from the results for perfect matchings 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.

3 Arborescences and MSTs

Now that we know 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 for designing LPs to find min-weight arborescences and MSTs.

3.1 Min-weight Arborescences

Recall the definition of a *r-arborescence*:

Definition 7.14. 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 r.
2. There exists a directed path from each vertex to r.

A min-weight r-arborescence is defined over a directed graph with non-negative edge weights and is the smallest total weight r-arborescence of the graph.

Let us define the linear program that induces the polytope K_{Arb} of all min-weight arborescences of a digraph $G = (V, A)$ with root vertex $r \in V$. As in K_{PM}, for each edge $(i, j) \in A$ we will define a variable $x_{ij} \in \mathbb{R}^{|A|}$.

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

[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.

The first and third set of inequalities should make sense. 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 so as to satisy item (2) in Definition 7.14.

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 7.15. $K_{Arb} = CH_{Arb}$

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

We will skip defining MSTs and move straight to forming the LP that induces the polytope K_{MST}. For a simple, undirected graph $G = (V, E)$, K_{MST} is the polytope in $\mathbb{R}^{|E|}$ whose vertices are exactly the MSTs of G.

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

Notice for the first constraint that $S \subset V$ not $S \subseteq V$. The second and third set of inequalities make sense, the first set says 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.

Define the convex hull of all minimum spanning trees of G to be CH_{MST}. Then, as we anticipated $CH_{MST} = K_{MST}$.

Theorem 7.16. $K_{MST} = CH_{MST}$

We made it seem in this section that finding min-weight r-arborescences and MSTs are similar to finding min-cost perfect matchings in bipartite graphs. But, notice that the polytopes K_{Arb} and K_{MST} could potentially require exponentially many constraints. Thus, it would take exponential time to simply write down the LP let alone solve it!

But, all is not lost. 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, run in polynomial time in the size of the dimension (in our case the number of edges). These separation oracles can further be used by a polytime algorithm called Ellipsoid that can solve LPs in general. Thus, there exist polytime algorithms to find the vertices of K_{MST} and K_{Arb}.

Here's a set of notes that explains the algorithms to find MSTs and min-weight r-arborescences in polytime: http://theory.epfl.ch/osven/courses/Approx13/Notes/lecture9and10.pdf

4 Perfect Matchings in General Graphs

Now we move to non-bipartite graphs. We define a polytope that is similar to the bipartite polytope. Let $x(\delta(v))$ denote the weight of all edges incident to v.

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

This is not a convex combination of all perfect matchings. For example, a triangle graph with each edge weight $\frac{1}{2}$ will satisfy the constraints of this polytope.

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

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

The above set of constraints is required because notice that a odd size vertex set cannot have a perfect matching, thus, at least one edge from the perfect matching must leave S so as to match all vertices in S. Adding these constraints to the existing K_{PM} defines the correct polytope. In fact, the proof follows from Tutte's theorem.

Thus, we have that the following LP construction, due to J.Edmonds [Edmonds, 1987], induces the polytope K_{genPM} whose vertices are the perfect matchings of a general graph $G = (V, E)$.

$$K_{\text{genPM}} = x \in \mathbb{R}^{|E|} \text{ s.t } \begin{cases} \forall v \in V, \sum_{u \in \delta(v)} x_{vu} = 1 & \text{and} \\ \forall S \text{ s.t } |S| \equiv_2 1, \sum_{e \in \delta(S)} x_e \geq 1 & \text{and} \\ \forall e \in E, \ x_e \geq 0 \end{cases}$$

Notice now that our LP contains potentially exponential number of constraints as compared to a polynomial number of constraints in the bipartite case. In fact, a theorem by [Rothvoss, 2013] 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 7.17. *$K_{PM} = CH($all perfect matchings$)$.*

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). So there are m linearly independent tight constraints. If there already exists an edge such that $x^*_e = 0$, then drop e, and argue about $G \setminus \{e\}$, i.e., G without edge e. If $x^*_e = 1$, then drop $e = (u, v)$, and induct on $G \setminus (u, v)$. Then after this process, for all v, there exist at least two edges in $\text{supp}(v)$. If all vertices have support degree 2, then there must be a cycle. This will cause a contradiction, as we saw in the proof of Theorem 7.12 for bipartite graphs. Therefore, there exists a vertex with degree ≥ 3 in the support. But then the number of edges in the support is greater than the number

of vertices. From this, we can show there is at least one $x(\delta(S)) \geq 1$ constraint that is tight. Call it S^*.

Take S^*, and shrink it down to one vertex. Then we induct on S^* itself, and on the remaining graph with the single vertex as S^*.

5 Interesting Aside

We have seen that LPs are an interesting way of formulating problems such as min-cost matchings, min-weight r-aborescences and MSTs. We reasoned about the structure of the polytopes that the LPs induce, and we were able to show that these LPs do indeed solve these combinatorial problems. Notice though that forming the LP is not sufficient, in fact, significant effort was put to show that these polytopes did indeed have integer solutions at the vertices [2]. If it were not the case that the solutions were integer valued, we would have the same problem as considering the bipartite LP for general graphs; we would get fractional solutions to these LPs that do not actually give us solutions to our problem.

There is an interesting field of study that relates to the integrality of LPs. We will briefly introduce a concept that deals with integrality of 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 n-long vector of variables and $\vec{b} \in \mathbb{R}^m$ is a m-long vector corresponding to the scalars $b_i \in \mathbb{R}$ in the constraint $A^{(i)} \cdot \vec{x} \leq b_i$.

Definition 7.18. Call a matrix $[A]_{m\times n}$ *totally unimodular* if every square submatrix B of A has the property that $\det(B) \in \{0, \pm 1\}$

We have the following neat theorem due to Kruskal-Hoffman.

Theorem 7.19. *[Hoffman and Kruskal, 2010] 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 the pains of producing combinatorial proofs as we have, instead, we can 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.

References

[Edmonds, 1987] Edmonds, J. (1987). Paths, trees, and flowers. pages 361–379. 4

[Hoffman and Kruskal, 2010] Hoffman, A. J. and Kruskal, J. B. (2010). *Integral Boundary Points of Convex Polyhedra*, pages 49–76. Springer Berlin Heidelberg, Berlin, Heidelberg. 7.19

[Rothvoss, 2013] Rothvoss, T. (2013). The matching polytope has exponential extension complexity. *ArXiv e-prints*. 4

[2] The solution vector was integer valued