Approximation Algorithms and Hardness of Approximation March 19, 2013

Lecture 9 and 10: Iterative rounding II

Lecturer: Ola Svensson *Scribes: Olivier Blanvillain*

In the last lecture we saw a framework for building approximation algorithms using iterative rounding:

1. Formulate the problem as a linear program (LP)
2. Characterise extreme point structure
3. Iterative algorithm
4. Analysis

We used this framework to solve two problems: *Matchings in Bipartite Graphs* and the *Generalised Assignment Problem*. A negative point about this approach is that it requires solving a linear program polynomial many time.
In this lecture we use iterative rounding to derive new results. We will talk about the minimum spanning tree problem. Ultimately we aim to design an algorithm to find a minimum spanning tree that (almost) respects given degree bounds.

1 Minimum spanning tree

Minimum spanning tree problem:
Given a graph $G = (V, E)$ and a cost function on the edges $c : E \to \mathbb{R}$ we want to find a spanning tree of minimum total edge cost.

Numerous exact polynomial time algorithm are know for this classical problem. As we will see in this lecture our approach using a linear programming formulation will introduce some interesting techniques that can be used to solve other problems.

1.1 Linear Program Relaxation

In this section we will discuss two formulations of the minimum spanning tree problem as a linear program. Our algorithm will have to pick a subset of the edges of the graph that form a spanning tree. Naturally we define one variable for each edge in the graph to indicate whether or not this edge in part of the tree:

$$\begin{aligned} x_e &= 1 \qquad \text{if edge } e \text{ is in tree} \\ &= 0 \qquad \text{otherwise.} \end{aligned}$$

The objective to minimize the total edge cost can be written as:

$$\text{minimise} \quad \sum_{e \in E} c_e x_e$$

For the first formulation we will consider a spanning tree as a minimal 1-edge-connected subgraph. As a result we have that each cut in the graph must have at least one edge:

$$\sum_{e \in \delta(S)} x_e \geq 1 \quad \forall S \subset V, S \neq \emptyset$$

Recall that $\delta(S)$ is the set of edges with exactly one endpoint in S. Unfortunately, the LP relaxation of this formulation is not exact for the minimum spanning tree problem. Consider cycle on 4 vertices, where each edge has cost 1. By each variable to $x_e = 1/2$ we obtain a feasible solution to this LP formulation with total cost 2 whereas any minimum spanning tree has cost 3.

We will now consider an alternative formulation that will be used for the remaining of this lecture.
LP-MST:

$$\begin{aligned}
\text{minimise} \quad & \sum_{e \in E} c_e x_e \\
\text{subject to} \quad & \sum_{e \in E(V)} x_e = |V| - 1 \\
& \sum_{e \in E(S)} x_e \leq |S| - 1 \quad && \forall S \subset V, S \neq \emptyset \\
& x_e \geq 0 && \forall e \in E
\end{aligned}$$

For $S \subseteq V$, we let $E(S)$ be the set of edges with both endpoints in S. A tree on n vertices can be seen as a graph with $n-1$ edges and no simple cycles. The first constraint captures the relation between the number of edges and the number of vertices: $|E| = |V| - 1$. As simple cycles on n vertices have n edges we can eliminate the cycles by ensuring all subset S of the vertices have less than $|S|$ edges with both endpoints in S. This constraint is called a subtour elimination constraint.

We have a problem with this linear program: the number of constraints is exponential in the number of vertices of the graph. To deal with this problem we will use what is called a *separation oracle*. Informally, a separation oracle is a procedure for a set of linear constraints that, given a candidate solution, will either certify that this solution is feasible, or exhibit a violated constraint.

Theorem 1 *Given a linear program P with constraints C and a polynomial-time separation oracle for C, we can optimally solve P in polynomial time via the Ellipsoid algorithm.*

Proof (intuition) Feasibility verification is an equivalent formulation. Usually we can lower and upper bound the objective function, then add the objective function as a constraint and finally do a binary search in the feasibility region.

The idea behind the Ellipsoid algorithm is to start with a big ellipsoid that contains all the feasible region, then query "is the center of the ellipsoid feasible?". For this the algorithm will use a separation oracle. If the point is inside the polytope formed by the constraints, this point can be returned as a feasible solution. If it's outside, the separation oracle will output a constraint that separates this point from the polytope. Then the Ellipsoid algorithm will use this constraint to shrink the ellipsoid to get closer and closer to the feasible region. ■

1.2 Separation oracle for LP-MST

Theorem 2 *There is a polynomial time separation oracle for LP-MST.*

Proof We can construct an algorithm that, given a fractional solution x, will find a set $S \subseteq V$ such that $x(E(S)) > |S| - 1$ if this set exists. For a set F of edges, $x(F)$ denotes $\sum_{e \in F} x_e$. We consider the graph as a flow network. We add a source vertex s, a sink vertex t, and edges (s, v), (v, t) $\forall v \in V$ with:

$$\begin{aligned}
c(s, v) &= \frac{x(\delta(v))}{2} \\
c(v, t) &= 1 \\
c(e) &= \frac{x_e}{2}
\end{aligned}$$

Let $S \cup s$ be a s-t cut in the graph. It's value is:

$$|S| + \sum_{v \in V \setminus S} \frac{x(\delta(v))}{2} + \sum_{e \in \delta(S)} \frac{x_e}{2} = |S| + x(E(V)) - x(E(S))$$

We can see that this is very related to our constraints: a s-t cut $S \cup s$ has value smaller than $|V|$ if and only if the constraint corresponding to this set S is violated. Hence it is sufficient to check that the minimum cut has value greater or equal to $|V|$.

We can find the min-cut in polynomial time, but there is another problem. What do we do if S is empty? How do we do to ensure that S has size at least two? For any pair $v, w \in V$, consider instance with $c(s, v) = \infty$. If in each case the minimum cut has value $|V|$ we can safely return that the solution is feasible. Furthermore, if in one case the minimum cut M has value smaller than $|V|$ we can output the constraint corresponding to M. So we have run n^2 instances of the minimum cut algorithm. ■

1.3 Uncrossing Technique

By using this two theorems we see that we can solve our LP-MST in polynomial time. Is there any extreme point solution structure? We have so far considered linear programs which had very few tight constraints. Here we may have exponential many tight constraints: given a spanning tree, any subtree corresponds to a tight constraint.

Consider two tight constraints S_1 and S_2 such that S_1 covers e_4 and e_3, S_2 covers e_3, e_2 and e_1.

$$\begin{aligned} (S_1):&\ x_{e_4} + x_{e_3} = 2 \\ (S_2):&\ x_{e_1} + x_{e_2} + x_{e_3} = 3 \end{aligned}$$

We also have two constraints $S3 = S1 \cup S2$ and $S4 = S1 \cap S2$:

$$\begin{aligned} (S_3):&\ x_{e_1} + x_{e_2} + x_{e_3} + x_{e_4} = 4 \\ (S_4):&\ x_{e_3} = 1 \end{aligned}$$

As $(S_1) + (S_2) = (S_3) + (S_4)$, these four constraints are linearly dependent. Therefore they cannot all define an extreme point. We will prove that we can always uniquely characterise an extreme point by a set of "non crossing" linearly independent constraints. Such a set is called a $laminar family$.

We define the characteristic vector of a set of edges F to be:

$$\begin{aligned} \chi(F)_e &= 1 \quad \text{if } e \in F, \\ &= 0 \quad \text{otherwise.} \end{aligned}$$

Proposition 3 *For $S, T \subseteq V$, $\chi(E(S)) + \chi(E(T)) \leq \chi(E(S \cup T)) + \chi(E(S \cap T))$, and equality if and only if the set of edges $E(S \backslash T, T \backslash S) = \emptyset$.*

Proof We can see that $\chi(E(S)) + \chi(E(T)) = \chi(E(S \cup T)) + \chi(E(S \cap T)) - \chi(E(S \backslash T, T \backslash S))$. ■

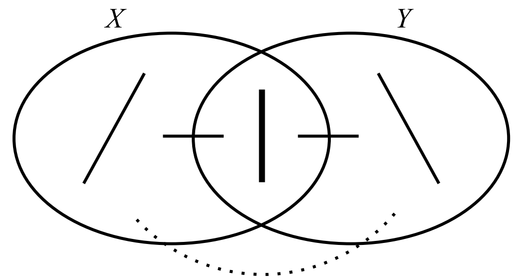

Figure 1: In this diagram the regular edges are counted once in both the right hand side $\chi(E(S)) + \chi(E(T))$ and left hand side $\chi(E(S \cup T)) + \chi(E(S \cap T))$, and the bold edge is counted twice by the two sides. However the dashed edge is counted in the right hand side but not in the left hand side.

Let $\mathcal{F} = \{S | x(E(S)) = |S| - 1\}$ correspond to tight constraints for an extreme point x. The following lemma shows that $\mathcal{F}$ is closed under union and intersection.

Lemma 4 *If $S, T \in \mathcal{F}$ and $S \cap T \neq \emptyset$, then both $S \cap T$ and $S \cup T$ are in $\mathcal{F}$. Moreover, $\chi(E(S)) + \chi(E(T)) = \chi(E(S \cup T)) + \chi(E(S \cap T)$.*

Proof As both S and T are tight, we have:

$$\begin{aligned} |S| - 1 + |T| - 1 &= x(E(S)) + x(E(T) \\ &\leq x(E(S \cup T)) + x(E(S \cap T)) \quad \text{by proposition 3} \\ &\leq |S \cup T| - 1 + |S \cap T| - 1 \quad \text{because } \chi \text{ is feasible} \\ &= |S| - 1 + |T| - 1 \end{aligned}$$

As both size of the inequality are equal we must have equality everywhere. The first equality implies that $S \cap T$ and $S \cup T$ are linearly independent. The second inequality shows that $S \cap T$ and $S \cup T$ have to be tight. ■

1.4 Laminar Family

Two sets S, T are intersecting if $S \cap T$, $S \backslash T$ and $T \backslash S$ are non-empty. A family $\mathcal{L}$ of sets is called laminar if all sets are non intersecting. We can see that a laminar family defines a nice tree structure, "like an onion". Let $span(\mathcal{F})$ be the vector space generated by the set of vectors $\{\chi(E(S)) : S \in \mathcal{F}\}$ (recall that $\mathcal{F}$ is the set consisting of tight constraints).

Lemma 5 *If $\mathcal{L}$ is a maximal laminar family then $span(\mathcal{L}) = span(\mathcal{F})$.*

Proof Suppose toward contradiction that $span(\mathcal{L})$ is contained in $span(\mathcal{F})$. Then there exist $\chi(E(S)) \notin span(\mathcal{L})$ and $S \in \mathcal{F}$. Pick the set S that intersects fewest sets in $\mathcal{L}$. Let $T \in \mathcal{L}$ that intersects S. But then by Lemma 4, $S \cap T$ and $S \cup T$ are also in $\mathcal{F}$. And $S \cap T$ and $S \cup T$ can be seen to intersect fewer sets in $\mathcal{L}$. This implies that (by the selection of S):

$$\chi(E(S \cap T)), \chi(E(S \cup T)) \in span(\mathcal{L}).$$

Again, by lemma 4 we have that:

$$\chi(E(S)) = \chi(E(S \cup T)) + \chi(E(S \cap T)) - \chi(E(T))$$

but as $\chi(E(T))$, $\chi(E(S \cup T))$ and $\chi(E(S \cap T)$ are in $span(\mathcal{L})$, $\chi(E(S))$ is also in $span(\mathcal{L})$. This is a contradiction. ■

Proposition 6 *A laminar family $\mathcal{L}$ over V without singletons has at most $|V|-1$ distinct member.*

Proof By induction on the size of V. If $|V|=2$ we clearly have 1 non singleton set and the proposition holds. Let $n=|V|$ and suppose that the claim is true for all laminar families of size smaller than n. Let $S \neq V$, $S \subset \mathcal{L}$ be a maximal set under those conditions. Then for each set $S' \neq V$, $S' \subset \mathcal{L}$, either $S' \subset S$ or S' do not intersect S. By induction hypothesis the number of sets in $\mathcal{L}$ contained in S is at most $|S|-1$. The set in $\mathcal{L}$ not intersecting S form a laminar family over $V \backslash S$ so by induction hypothesis there are at most $|V|-|S|-1$ such sets. By adding the set V we obtain a total of at most $(|S|-1)+(|V|-|S|-1)+1=|V|-1$ sets. ■

Corollary 7 *LP-MST is integral.*

Proof A Laminar family $\mathcal{L}$ without singleton defines an extreme point. Also $|V|-1 \geq |\mathcal{L}|=|E|$. We have $\sum_e x_e = |V|-1$ and $0 \leq x_e \leq 1$ $\forall e \in E$. This implies that $x_e=1$ for each edges e in our graph. ■

This corollary has an alternative proof that can be generalised to the minimum Bounded degree Spanning Tree problem:
Proof By Lemma 5 and the Rank-Lemma we know that there is a laminar set $\mathcal{L}$ of linearly independent tight constraints that define the extreme point solution. In addtion, $|E|=|\mathcal{L}|$. We will do a proof by contradiction using a fractional token counting argument. The idea is start with a set of $|E|$ tokens and to distribute them fractionally among the sets in $\mathcal{L}$ such that each $S \in \mathcal{L}$ gets at least one token. We will then see that if LP-MST is not integral there will be some left over tokens, contradicting $|E|=|L|$.

The redistribution of tokens we use will for each edge $e \in E$ redistribute x_e tokens to the smallest set in $\mathcal{L}$ containing both endpoints of e. We now show that each set $S \in \mathcal{L}$ will get at least one unit of token. Let $R_1, R_2, ..., R_k$ be the children of S. The constraints corresponding to these sets are all tight:

$$\begin{aligned} X(E(S)) &= |S|-1 \\ X(E(R_i)) &= |R_i|-1 \quad \forall i = 1...k \end{aligned}$$

Let A be the set of edges that have both endpoint in S but not in R_i $\forall i=1...k$. By construction we have:

$$x(A) = X(E(S)) - \sum_{i=1}^{k} X(E(R_i)) = |S|-1-\sum_{i=1}^{k}|R|-1 = |S|-\sum_{i=1}^{k}|R|+k-1$$

By our redistribution policy, S obtains x_e fractional token for each edge of A. If $A=\emptyset$, $\chi(E(S)) = \sum_{i=1}^{k}\chi(E(R_i))$ which is a contradiction with the linear independence of the constrains in $\mathcal{L}$. In the last equation we have that $x(A)$ equals a sum of integers, so $x(A)$ has to be integer. As $A \neq \emptyset$ we have $x(A)>0$ implying $x(A) \geq 1$. Hence, at least one token is assigned to S. If there exist e with $x_e<1$, that edge has $1-x_e$ unused token which is a contradiction. ■

2 Minimum Bounded degree Spanning Tree

Given a graph $G(V,E)$ with a cost function on edges $c: E \rightarrow \mathbb{R}$ and degree Bounds B_v for each $v \in W$, $W \subseteq V$, find a minimum cost spanning tree respecting degree bounds.

One application of this problem would be the construction of trees for broadcasting in computer networks. When looking for a spanning tree one would topically want to avoid a "star" configuration where all the communication has to pass through a single node.

Even if we restrict this problem to respecting the degree bounds (without cost minimisation), this problem is NP-complete. Indeed by setting the degree bounds of two vertices to 1 and all the remaining degree bounds to 2, the problem is equivalent to determining the existence of a Hamiltonian path in the graph, which is a classical NP-complete problem. Hence, we cannot hope to satisfy the degree bounds completely.

Theorem 8 *There exists a polynomial time algorithm that find spanning tree of cost C where each degree bound is violated at most by an additive 1. C denotes the optimal cost of a tree respecting the degree bounds.*

2.1 Linear Program Relaxation

LP-MBDST:

$$
\begin{aligned}
\text{minimise} \quad & \sum_{e \in E} c_e x_e \\
\text{subject to} \quad & \sum_{e \in E(V)} x_e = |V| - 1 \\
& \sum_{e \in E(S)} x_e \leq |S| - 1 && \forall S \subset V, S \neq \emptyset \\
& X(\delta(V)) \leq B_v && \forall v \in W \\
& x_e \geq 0 && \forall e \in E
\end{aligned}
$$

For a solution x of this LP we will remove all edges with $x_e = 0$. Using the Rank-Lemma and the results from the Uncrossing Technique (Lemma 5) we can derive the following extreme point characterization:

Lemma 9 *Let x be an extreme point solution to LP-MBDST such that $x_e > 0$ $\forall e \in E$. Then there exists a Laminal family $\mathcal{L}$ and a set $T \subseteq W$ such that:*

- $\forall s \in \mathcal{L} : x(E(S)) = |S| - 1$ *and* $\forall v \in T : x(\delta(v)) = B_v$
- *the vectors* $\{\chi(E(S)) : s \in \mathcal{L}\} \cup \{\chi(\delta(v)) : v \in T\}$ *are linearly independent*
- $|E| = |\mathcal{L}| + |T|$
- $span(\mathcal{L}) = span(\mathcal{F})$ *where* $\mathcal{F} = \{S | x(E(S)) = |S| - 1\}$.

2.2 Iterative Algorithm

If $T = \emptyset$, we know that the previous LP-MST analysis that the solution is integral. In the case of $T \neq \emptyset$ we have the following lemma:

Lemma 10 *If $T \neq \emptyset$ then exist $v \in W$ such that $d_E(V) \leq B_v + 1$.*

Before proving this Lemma we will see how it can be used to construct an iterative algorithm for the minimum bounded degree spanning tree.

Iterative Algorithm for MBDST:

Initially set $W = V$

. Loop:

1. Solve LP-MBDST to find an extreme point solution x and remove all edges with $x_e = 0$.

2. If the solution is integral then return x.

3. Else

   (a) By the Key Lemma there exists $v \in W$ such that $d_E(v) \leq B_v + 1$.

   (b) Remove v from W.

We now present a proof of Lemma 10 that uses a fractional token counting argument similar to one used in the second proof of Corollary 7:

Proof Suppose toward contradiction that $T \neq \emptyset$ and $d_E(v) > B_v + 1\ \forall v \in W$. Consider a set of $|E|$ tokens. For each edge $e \in E$ assign x_e to the smallest set in $\mathcal{L}$ containing both endpoints of e, and assign $(1 - x_e)/2$ tokens to each endpoint. By the previous token counting proof we know that each $S \subset \mathcal{L}$ gets a least one token. We can see that each $v \in W$ gets at least one token:

$$\sum_{e \in \delta(v)} \frac{1 - x_e}{2} \geq \frac{d_E(v) - B_v}{2} \geq 1$$

where the the last inequality holds because we assumed $d_E(v) > B_v + 1\ \forall v \in W$.

In the remaining of the proof we will show that there are some left over tokens, which will contradict $|E| = |\mathcal{L}| + |T|$ (from Lemma 9).
- If $V \notin \mathcal{L}$ then there exist an edge e not contained in any set of $\mathcal{L}$. Thus the corresponding x_e token are never assigned and we have a contradiction.
- If there exists a vertex $v \in W \backslash T$ then v gets one unit of token it doesn't need and we have a contradiction.
- If there exists a vertex $v \in V \backslash T$ then each edge e in $\delta(v)$ must have $x_e = 1$, otherwise $(1-x_e)/2$ tokens are left over. Note that for each edge e with $x_e = 1$, e is a tight set of size two, implying $\chi(e) \in span(\mathcal{L})$. Hence:

$$\begin{aligned} 2\chi(E(V)) &= \sum_{v \in V} \chi(\delta(v)) \\ &= \sum_{v \in T} \chi(\delta(v)) \ + \sum_{v \in V \backslash T} \chi(\delta(v)) \\ &= \sum_{v \in T} \chi(\delta(v)) \ + \sum_{v \in V \backslash T} \sum_{e \in \delta(v)} \chi(e) \\ &\Rightarrow\ 2\chi(E(V)) \ - \sum_{v \in V \backslash T} \sum_{e \in \delta(v) \chi(e)} \ = \ \sum_{v \in T} \chi(\delta(v)) \end{aligned}$$

By previous argument the left hand side of this equation is in $span(\mathcal{L})$. Since $T \neq \emptyset$, this contradicts the linear independence of the tight constraints in T and in $\mathcal{L}$. ■