Optimization II	**Winter 2009/10**
Lecture 12 — January 25	
Lecturer: Julián Mestre	

In this final lecture we will cover how to solve the linear program for the MDST problem. In fact, we will see two different ways of accomplishing this!

$$\text{minimize} \sum_{e \in E} w_e x_e \tag{LP1}$$

subject to

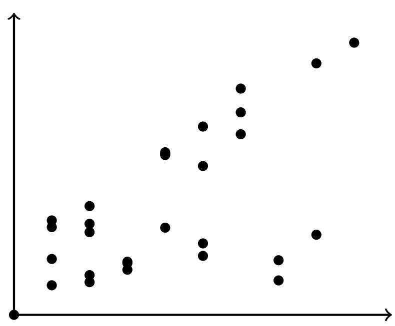
$$x(E(V)) = |V| - 1 \tag{12.1}$$
$$x(E(S)) \leq |S| - 1 \qquad \forall\, S \subseteq V \tag{12.2}$$
$$x(\delta(u)) \leq d(u) \qquad \forall\, u \in V \tag{12.3}$$
$$x_e \geq 0 \qquad \forall\, e \in E$$

12.1 Via separation oracle

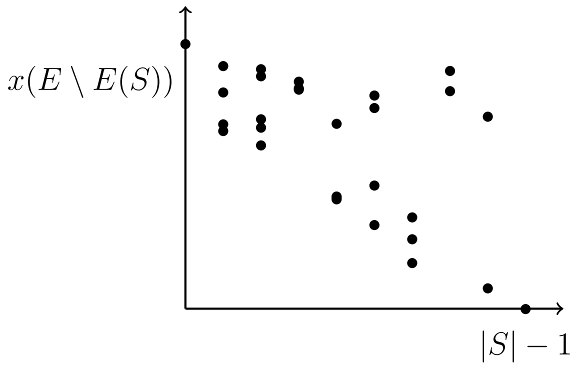
Our first approach to solving (LP1) will be to design a separation oracle. That is, given a solution x we need to either establish the feasibility of x or produce a violated constraint. If we can do that in polynomial time, then the Ellipsoid algorithm can be used to solve the program. We note that testing the feasible of Constraints (12.1) and (12.3) is trivial. The issue is how to test for Constraints (12.2), which are exponentially many.

The abstract problem we are to solve is as follows. We are given a solution x and we want to find a set $S \subseteq$ with maximum *density* $\frac{x(E(S))}{|S|-1}$. Let S^* be a maximizer of this function (among the maximizes we break ties by choosing one with minimum size) and λ^* be the value attained by this set. If $\lambda^* > 1$, then S^* defines our violating constraint, otherwise we know x satisfies all Constraints (12.2).

Imagine plotting for each set $S \subseteq V$ with $|S| \geq 2$, the point $(|S| - 1, x(E(S)))$. It is convenient to include a dummy point at the origin $(0, 0)$ in our point set. Notice that the density of a given set S is the slope of the line going trough the corresponding point and the dummy point. Thus, finding the point with maximum density is equivalent to finding the line with the smallest slope going through the dummy point origin that leaves all points under the line.



It is convenient to view the problem from a slightly different perspective. Consider the same plot for the points $(|S| - 1, x(E \setminus E(S)))$. This transformation only flips the previous graph. The dummy point is mapped accordingly to $(0, x(E))$. Now we are interested in finding the line going through the dummy having smallest negative slope (closest to 0) that leaves all points above the line.



Let P be the set of point in our scatter plot. Suppose we had an oracle for finding the point $(a, b) \in P$ minimizing $\alpha \cdot a + b$ for some fix but arbitrary $\alpha > 0$. If $\alpha > \lambda^*$ then the oracle would return the dummy point. If $\alpha > \lambda^*$ the oracle will return some $S \neq \emptyset$. At $\alpha = \lambda^*$ our oracle can return the dummy point or some set with maximum density. Let us assume that if there is a choice the oracle never returns the dummy point. Suppose we run our oracle with $\alpha = 1$. If the oracle returns the dummy point, then we know there is some violating constraint and we can find it using binary search on α. Otherwise, we know that all constraints are satisfied.

The oracle can be implemented using the following reduction to minimum weight vertex cover in bipartite graphs. First we guess some vertex v in the set S. Our bipartite graph has two parts corresponding to $V - v$ and E in the original graph. For each vertex u we assign its node a weight of α. For each edge e, we assign its corresponding node a weight of x_e. We connect the node corresponding to an edge with the nodes corresponding to its endpoints; nodes incident on v are only connected to the other endpoint. Any minimal vertex cover in the graph picks a subset $S \subseteq V - v$. If $S = \emptyset$ the edges chosen are $E(V)$; otherwise, when $S \neq \emptyset$ the edges chosen are $E \setminus E(S + v)$. Finding a minimum weight vertex cover is precisely the kind of oracle we need.

12.2 Via alternative formulation

Another avenue for solving (LP1) is to come up with an alternative formulation that has polynomial size. To simplify the treatment, we focus on the spanning tree polytope.

First consider the follow formulation for an out-branching in a directed graph (V, D). Let r be the be the root of our out-branching.

$$\text{minimize } \sum_{e \in D} w_e x_e \tag{LP2}$$

subject to

$$x(\delta^{out}(S)) \geq 1 \qquad \forall\, S \subset V : r \in S \tag{12.4}$$

$$x_e \geq 0 \qquad \forall\, e \in D$$

Theorem 12.1. *The extreme points of* (LP2) *correspond to* 0-1 *incident vectors of out-branchings.*

The basic idea is to bi-direct every edge in our undirected graph. That is, we replace each undirected edge $\{u, v\} \in E$ with two edges (u, v) and (v, u) and we choose arbitrarily a root. Asking for an out-branching in this graph is equivalent to asking for spanning tree in the original graph.

To deal with exponential number of cut constraints we turn to a flow-based formulation. Let $\mathcal{F}_u$ be the polytope corresponding to flow vectors that ship at least unit of flow from u to r. The flow, of course, is fractional. Also, in this flow we have for each edge

$(u, v) \in E$ two variables $f^u_{uv}, f^u_{vu} \geq 0$ that encode how much flow goes along the edge in each direction. The basic idea is to let each vertex send one unit of flow to the root and then pick the edge capacities that will allow us to ship each of these flows separately.

$$\text{minimize} \sum_{e \in E} w_e x_e \tag{LP3}$$

subject to

$$\begin{aligned} f^u &\in \mathcal{F}_u && \forall\, u \in V - r && (12.5)\\ f^u_{ab} + f^v_{ba} &\leq x_{\{a,b\}} && \forall\, \{a, b\} \in E \text{ and } u, v \in V - r \end{aligned}$$

Theorem 12.2. *Let* (x, f) *be an extreme point of* (LP3) *then the vector* x *corresponds to the incidence vector of a spanning tree in* G. *Also, for every incidence vector* x *a spanning tree there exists a flow vector* f *such that* (x, f) *is feasible for* (LP3)