6.854J / 18.415J Advanced Algorithms
Fall 2008

18.415/6.854 Advanced Algorithms October 15, 2008

Lecture 11

Lecturer: Michel X. Goemans

In this lecture, we will start continuing from where we left in the last lecture on linear programming. We then argue that $LP \in NP \cap co-NP$. In the end of this lecture, we introduce the first polynomial algorithm to solve LP, known as the *Ellipsoid Algorithm.*

1 LP continuation

Last time, we had proved that, given a polyhedral set $P = \{x : Ax = b,\ x \geq 0\}$, a point x is a vertex of P if and only if $A_{\{j:\ x_j>0\}}$ has linearly independent columns. Now assume that $rank(A) = m$, where m is the number of rows. We had then defined the notion of a basic feasible solution (bfs) corresponding to a basis B, see last lecture for details.

Theorem 1 *Consider the polyhedral set $P = \{x : Ax = b,\ x \geq 0\}$ where $rank(A) = m$. A point x is a vertex of P if and only if it is a basic feasible solution.*

Proof: If x is a vertex of P, then we know that $A_{\{j\ :x_j>0\}}$ has linearly independent columns. Let $J == \{j : x_j > 0\}$. Thus $rank(A_J) = |J|$. Since $rank(A) = m$, we can add columns to J to get a set B with $|B| = m$ and $rank(A_B) = m$, i.e. A_B is invertible. We must have that:

$$x_B = A_B^{-1} b$$
$$x_N = 0.$$

Therefore, x is a basic feasible solution.

Conversely, assume x is a basic feasible solution, that is,

$$x_B = A_B^{-1} b$$
$$x_N = 0.$$

By definition, $J = \{j : x_j > 0\} \subseteq B$ and the fact that $rank(A_B) = |B|$ implies that A_J has linearly independent columns. Thus, x is a vertex of P. □

Theorem 2 *Let $P = \{x : Ax = b,\ x \geq 0\}$. Assume $min\{c^T x : x \in P\}$ is finite. Then, for any $x \in P$, there exists a vertex $x' \in P$ such that $c^T x' \leq c^T x$*

Proof: If x is a vertex, we are done. Otherwise, there exists $y \neq 0$ such that $x \pm y$ is in P. Note that, as $Ay = 0$ (because $A(x + y) = b = Ax$), for any $\alpha \in \mathbb{R}$, $A(x + \alpha y) = b$. Observe that,

$$(x + \alpha y)_j \geq 0 \begin{cases} \text{for } \alpha \leq \frac{x_j}{-y_j}, \text{ if } y_j < 0 \\ \text{always, if } y_j \geq 0 \end{cases}$$

We may assume that $c^T y \leq 0$ (otherwise choose $-y$). Moreover, if $c^T y = 0$, we can assume that there exists j such that $y_j < 0$.

Assume, by contradiction, that for all j, $y_j \geq 0$. Then, $c^T y < 0$. But this implies that

$$c^T(x + \alpha y) \to -\infty \text{ as } \alpha \to \infty$$

Then $\min\{c^T x : x \in P\}$ is not finite. Contradiction!

Therefore, there exists j such that $y_j < 0$. Choose

$$\alpha = \min_{j:\ y_j<0} \frac{x_j}{-y_j}. \tag{1}$$

This implies that $x + \alpha y$ is in P, and $c^T(x + \alpha y) \leq c^T x$. Moreover, one more component of x is 0. We can apply the same procedure to $x' = x + \alpha y$, and eventually we are going to get to a vertex. (Formally, we could apply induction on the number of nonzero entries of x).

□

2 Size of LP

In order to be able to discuss the complexity for solving a linear program, we need first to discuss the size of the input. We assume that every integer data is given in binary encoding, thus for $n \in \mathbb{Z}$, we need

$$size(n) = 1 + \lceil \log_2(|n| + 1) \rceil$$

bits, for $v \in \mathbb{Z}^p$, we need

$$size(v) = \sum_{i=1}^{p} (v_i)$$

bits, and for $A \in \mathbb{Z}^{nxm}$, we need

$$size(A) = \sum_{i=1}^{n} \sum_{j=1}^{m} (a_{i,j}).$$

bits. As a result, to represent all the data of a linear program, we need a size equal to

$$size(LP) = size(b) + size(c) + size(A).$$

The above size is not very convenient when proving the complexity of a linear programming algorithm. Instead, we will be considering another size, defined by

$$L = m + n + log_2(det_{max}) + \log_2(b_{max}) + \log_2(c_{max}),$$

where $\det_{max} = \max |\det(A')|$ over all submatrices A' of A, $b_{max} = \max_i |b_i|$ and $c_{max} = \max_j |c_j|$.

In the following two lemmas, we show that L is polynomially comparable with $size(LP)$, which implies that an algorithm has a running time polynomially bounded in terms of L if, and only if, it is polynomial in $size(LP)$.

Lemma 3 *If* $A' \in \mathbb{Z}^{n\times n}$ *then* $|det(A')| \leq 2^{size(A')-n^2} - 1$.

Proof: Recall that for $A' = [a_1, a_2, ..., a_k]$, $|\det(A')|$ can be visualized as the volume of the parallelipiped spanned by the column vectors. Hence,

$$1 + |det(A')| \leq 1 + \prod_{i=1}^{n} \|a_i\| \leq \prod_{i=1}^{n}(1 + \|a_i\|) \leq \prod_{i=1}^{n} 2^{size(a_i)-n} = 2^{size(A')-n^2}.$$

□

Lemma 4 $L \leq \text{size(LP)} \leq mnL$.

Proof: Using the fact that $size(n) \leq 2 + \log_2(n)$ for $n \geq 1$, we have that the second inequality holds because:

$$size(A) \leq mn \max_{i,j}(size(a_{ij})) \leq mn(2 + \log_2(\det_{\max})),$$

$$size(b) \leq m(2 + \log_2(b_{\max})),$$

and

$$size(c) \leq n(2 + \log_2(c_{\max})).$$

Adding these together gives the desired inequality for $m \geq 2$, $n \geq 2$. The first $\leq$ holds because, by the previous lemma, the determinant of any minor of A is bounded by the size of A. Hence,

$$det_{\max} \leq 2^{\text{size(A)}}.$$

Also,

$$m + \log b_{\max} \leq \text{size(b)},$$

and

$$n + \log c_{\max} \leq \text{size(c)}.$$

Finally,

$$2^L = 2^m 2^n \det_{\max} c_{\max} b_{\max} \leq 2^{\text{size(LP)}}$$

. □

From the definition of L, the following remark follows; this is what we will need mostly when analyzing running times or sizes.

Remark 1 $det_{max} * b_{max} * c_{max} * 2^{m+n} = 2^L$.

3 Complexity of LP

Here is the decision problem corresponding to linear programming.

Given $A \in \mathbb{Z}^{m\times n}$, $b \in \mathbb{Z}^m$, $c \in \mathbb{Z}^n$, and λ, determine whether

$$\min\{c^T x : Ax = b, x \geq 0\} \leq \lambda. \tag{2}$$

To show that LP is in NP, we need to be able to provide a concise (i.e. polynomially bounded in the size of the input) certificate for yes instances. A feasible point of cost less or equal to λ will clearly be a certificate, but will it be concise?

Claim 5 *$LP \in NP$*

We now show that if we take not just any feasible solution, but a basic feasible solution, then its size will be polynomially bounded in the size of the input.

Theorem 6 *Let x be a vertex (or basic feasible solution) of $Ax = b, x \geq 0$. Then $x_i = \frac{p_i}{q}$. for i=1,...,n where $p_i, q \in \mathbb{N}$ and $p_i < 2^L$ and $q < 2^L$.*

Proof: Since x is a vertex, then x is a basic feasible solution with basis B such that $x_B = A_B^{-1}b$ and $x_N = 0$ (notice that A_B is square). By Cramer's rule:

$$x_B = A_B^{-1}b = \frac{1}{\det(A_B)} cof(A_B)b,$$

where $cof(A)$ is a matrix whose entries are all determinants of submatrices of A. Letting $q = \det(A_B)$, we get that $q \leq det_{\max} < 2^L$ and $p_i \leq m\ det_{\max}\ b_{\max} < 2^L$. □

Now, to prove Claim 5, for yes instances, the certificate will be a vertex of $\{x : Ax = b, x \geq 0\}$ such that $c^T x \leq \lambda$.

However, to be precise, we also have to deal with the case in which the LP is unbounded, since in that case, there might not be any such vertex. But in that case, we can give a certificate of unboundedness by (i) exhibiting a vertex of $\{x : Ax = b, x \geq 0\}$ (showing it is not empty, and it is concise by the above theorem) and (ii) showing that the dual feasible region $\{y : A^T y \leq c\}$ is empty by using Farkas' lemma and exhibiting a vertex of $Ax = b, x \geq 0, c^T x = -1$ which is also concise by the above theorem.

Alternatively, one can show a concise feasible solution to

$$\min\{c^T x : Ax = b, x \geq 0, c^T x \leq \lambda - 1\}. \tag{3}$$

Claim 7 *$LP \in co - NP$.*

Indeed, for the complement instances of LP, we can use strong duality and exhibit a basic feasible solution of $A^T y \leq c$ s.t. $b^T y > \lambda$ (or show that $\{x \geq 0 : Ax = b\}$ is empty using Farkas' lemma). In the case when $\{x : Ax = b, x \geq 0\}$ is feasible, the correctness follows from strong duality saying that

$$\min\{c^T x : Ax = b, x \geq 0\} = \max\{b^T y : A^T y \leq c\}.$$

Thus, $LP \in NP \cap co - NP$ which makes it likely to be in P. And indeed, LP was shown to be polynomially solvable through the ellipsoid algorithm.

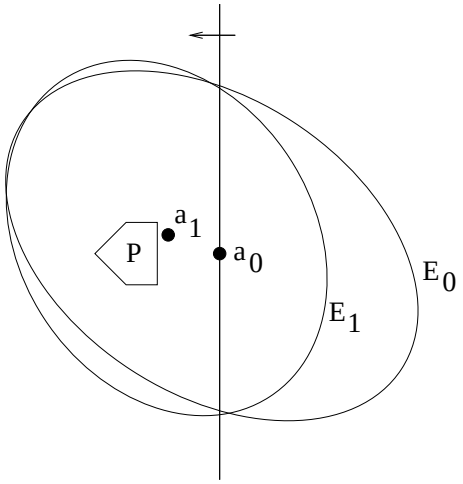

Figure 1: One iteration of the ellipsoid algorithm.

4 The Ellipsoid Algorithm

The Ellipsoid algorithm was proposed by the Russian mathematician Shor in 1977 for general convex optimization problems, and applied to linear programming by Khachyan in 1979. The problem being considered by the ellipsoid algorithm is:

> Given a bounded, convex, non-empty and full-dimensional set $P \in \mathbb{R}^n$ find $x \in P$.

We will see that we can reduce linear programming to an instance of this problem.

The ellipsoid algorithm works as follows. We start with a big ellipsoid E that is guaranteed to contain P. We then check if the center of the ellipsoid is in P. If it is, we are done, we found a point in P. Otherwise, we find an hyperplane passing through the center of the ellipsoid, so that P is contained in one of the half spaces defined by it. One iteration of the ellipsoid algorithm is illustrated in Figure 1. The ellipsoid algorithm is the following.

- Let E_0 be an ellipsoid containing P
- while center a_k of E_k is not in P do:
  - Let $c_k^T x \leq c_k^T a_k$ be such that $\{x : c_k^T x \leq c_k^T a_k\} \supseteq P$
  - Let E_{k+1} be the minimum volume ellipsoid containing $E_k \cap \{x : c_k^T x \leq c_k^T a_k\}$
  - $k \leftarrow k + 1$

The ellipsoid algorithm has the important property that the ellipsoids constructed shrink by, at least, a constant (depending on the dimension) factor in volume as the algorithm proceeds; this is stated precisely in the next lemma. As P is full dimensional, we will eventually find a point in P.

Lemma 8 $\frac{Vol(E_{k+1})}{Vol(E_k)} < e^{-\frac{1}{2n+2}}$.

Note that the ratio is independent of k.

Before we can state the algorithm more precisely, we need to define ellipsoids.

Definition 1 *Given a center a, and a positive definite matrix A, the ellipsoid* $E(a, A)$ *is defined as* $\{x \in \mathbb{R}^n : (x-a)^T A^{-1}(x-a) \leq 1\}$.

One important fact about a positive definite matrix A is that there exists B such that $A = B^T B$, and hence $A^{-1} = B^{-1}(B^{-1})^T$. Ellipsoids are in fact just affine transformations of unit balls. To see this, consider the (bijective) affine transformation $T : x \to y = (B^{-1})^T(x-a)$. It maps $E(a, A) \to \{y : y^T y \leq 1\} = E(0, I)$, the unit ball.

This gives a motivation for the fact that the ratio $\frac{Vol(E_{k+1})}{Vol(E_k)}$ is independent of k. Indeed, as linear transformations preserve ratio of volumes, we can reduce to the case when E_k is the unit ball. In this case, by symmetry of the ball, the volume ratio will be independent of k.