CSC2411 - Linear Programming and Combinatorial Optimization[*] Lecture 9: Ellipsoid Algorithm (Contd.) and Interior Point Methods

Notes taken by Vincent Cheung

March 16, 2005

Summary: The ellipsoid algorithm is shown to be capable of solving certain LP problems where the number of constraints is exponential in polynomial time. This lecture then introduces interior point methods, which is one of the most commonly used algorithms for solving linear programming problems. In particular, Ye's algorithm, that takes a primal-dual approach, is discussed.

1 Ellipsoid algorithm for LP with "too many" constraints

Recall that for the ellipsoid algorithm to work, we need these oracles:

1. Membership oracle - is $x \in P$?
2. Separation oracle - what is a plane separating x from P?

The ellipsoid algorithm tells us that given these oracles to a problem, guarantees of not too large of an initial search space, and not too small of a possible volume for $P \neq 0$, we get a polynomial solution. With this observation, we may hope to achieve polynomial algorithms to certain LP with many more constraints than the natural parameters of the problem. Specifically, if m, the number of constraints, is exponential in the number of variables n, and the entries of A, b, and c are not big in terms of n, we may still get a polynomial algorithm in n as long as we have a separation oracle.

Example 1.1. Given an undirected graph G with n edges and cost, c, on the edges and k terminal pairs of sources and sinks, (s_i, t_i). The goal is to find the minimal weight set of edges, F, that leaves s_i connected to t_i. This problem can be formulated as a linear program.

[*] Lecture Notes for a course given by Avner Magen, Dept. of Computer Sciecne, University of Toronto.

Let $\mathcal{S}_i$ be the set of partitions S of the nodes which separate s_i and t_i, i.e. $|S \cap \{s_i, t_i\}| = 1$. The problem is characterized by the following integral program.

$$\begin{aligned}
&\min \sum c_e x_e \\
&\forall\, i = 1, \dots, k,\ \forall\, S \ \in\ \mathcal{S}_i \quad \sum_{e \text{ in the cut of } S} x_e \geq 1 \\
&x_e \in \{0, 1\}
\end{aligned}$$

Relaxing the binary constraints on x and permitting values in the interval $[0, 1]$ gives the following formulation.

$$\begin{aligned}
&\min \sum c_e x_e \\
&\forall\, i = 1, \dots, k,\ \forall\, S \ \in\ \mathcal{S}_i \quad \sum_{e \text{ in the cut of } S} x_e \geq 1 \\
&0 \leq x_e \leq 1
\end{aligned}$$

Recall that for the ellipsoid algorithm, we had bounds that relates to n and to the size of n x n matrices since all coefficients of A, b, and c are $0, 1$ this means that really, if we can only show separation oracle, we actually know the ellipsoid algorithm is polynomial. For $x \in \mathbb{R}^n$, we need to know if

1. x is feasible
2. a separating hyperplane, a, such that $\langle a, y \rangle < \langle a, x \rangle$ $\forall$ feasible y

Algorithm 1.2.

Input: $x \geq 0$
Output: x feasible or separating hyperplane, a

for $i = 1, \dots, k$

1. look at network flow problem with s_i source, t_i sink, and capacities are the x_e
2. run max flow-min cut algorithm on this network
3. if flow < 1 and S is the min-cut, then $\sum_{e \text{ in the cut of } S} x_e < 1$ is a separating hyperplane

if all the k "max flows" we get are ≥ 1, then x_e is feasible.

For a single source, s_i, and sink, t_i, if the max flow-min cut algorithm yields a maximum flow less than 1, then it cannot be the case that the edges have binary weights in forming a path between s_i and t_i. Thus, step 3 forms a separating hyperplane for the original formulation of the LP, as required.

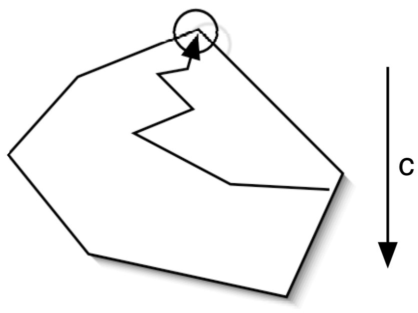

Figure 1: Moving through the polytope.

2 The Interior Point Method

The simplex algorithm performs well in practice, but has a poor theoretical worst case performance and works only with linear programming problems. The ellipsoid algorithm does not provide much in terms of practical performance improvement over the simplex algorithm, but has a polynomial time theoretical complexity and is able to solve more than just linear programming problems. Karmakar's paper in 1984 introduced a new method that not only performs well in practice and in theory, but, as we will see, is even more extendable beyond linear programs than the ellipsoid algorithm. This method is called "interior point".

Like the simplex algorithm, the interior point method holds a feasible solution in each iteration, but unlike simplex, all this is happening in the interior of the polytope, int(P), rather than the vertices of the polytope. The interior point method proceeds by moving through the polytope towards the optimal value. Figure 1 illustrates this movement through the polytope. How can we hope to find the optimum, opt(P), if we are always in the interior? We know that the optimal solution lies on a vertex of the simplex, so intuitively, we can stop when the solution is within some small distance to the optimum and declare the nearest vertex to be the optimum.

Definition 2.1. A feasible solution is "almost optimal" if $\langle x, c\rangle < \text{opt}(P) + 2^{-2L}$, where L is the size of the LP.

Claim 2.2. *If x is almost optimal and if v is a vertex with $\langle v, c\rangle \leq \langle x, c\rangle$, then v is optimal.*

Proof. If u_1 and u_2 are vertices such that $\langle u_1, c\rangle < \langle u_2, c\rangle$, then $\langle u_1, c\rangle \leq \langle u_2, c\rangle - 2^{-2L}$ because of the bounds on the size of the LP. So, if v is not optimal, then the optimal solution, w, is such that $\langle w, c\rangle \leq \langle v, c\rangle - 2^{-2L} \leq \langle x, c\rangle - 2^{-2L}$, which is in contradiction to x being almost optimal. □

The simplex can be quite complicated and it might seem obvious to move through the polytope with the interior point method by moving in the direction that best decreases the objective function. However, this naïve improvement is inefficient when

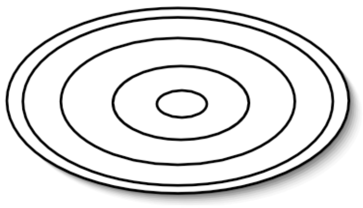

Figure 2: The barrier function.

the solution is near the boundary, as seen Fig. 1. The first movement in the polytope moves the solution away from the boundary, while doing something reasonable with respect to the objective function so that the following steps can make better gains. This desire to stay away from the boundary of the polytope is formulated in the form of a barrier function. A barrier function, as shown in Fig. 2, is like a deep well, where it has a low value in the most interior of the polytope and has infinitely steep walls at the boundary. Staying within the interior of the polytope is essential for the interior point algorithm and it is the minimization of both the barrier and objective functions that gives an efficient and optimal solution.

Definition 2.3. A barrier function is a convex function on the affine hull, int(P) that tends to infinity on the boundary of P.

Consider a linear programming problem in standard form,

$$\begin{aligned} &\min\langle x, c\rangle \\ &Ax = b \\ &x \geq 0 \end{aligned}$$

A barrier function that is commonly used is

$$\Phi(x) = -\sum \ln(x_i)$$

This barrier function clearly goes to infinity as x approaches the boundary of the polytope, i.e. as x_i approaches 0. This barrier function is convex because it is simply a negative summation of concave logarithmic functions.

Equipped with a barrier function for P, we can define a potential function that takes into account both the barrier and objective function. The interior point method proceeds by taking a step in the direction that minimizes this potential function. Interior point is not a specific algorithm, but rather, a method, and so, a family of algorithms exist, which differ by:

- their setting as primal, dual, and primal-dual
- their barrier function and their potential function
- how a step is performed

2.1 Ye's Interior Point Algorithm

Ye took a primal-dual approach for interior point in 1990 when he introduced a barrier function called the primal-dual potential function.

2.1.1 A primal-dual approach

Consider the following linear program:

$$\begin{aligned} &\min\langle x, c\rangle \\ &Ax = b \\ &x \geq 0 \end{aligned}$$

It's dual is given by:

$$\begin{aligned} &\max\langle y, b\rangle \\ &yA \leq c \\ &y \gtreqless 0 \end{aligned}$$

By adding a slack variable, the dual becomes:

$$\begin{aligned} &\max\langle y, b\rangle \\ &yA + s = c \\ &s \geq 0 \end{aligned}$$

The interior point algorithm holds x and s as solutions. Note that s is just part of the dual solution. The algorithm will hold pairs $(x^{(0)}, s^{(0)}) \to (x^{(1)}, s^{(1)}) \to \cdots$

From complementary slackness,

$$\langle x, c\rangle - \langle x, s\rangle = \langle x, c - s\rangle = y^T Ax = \langle y, b\rangle$$

thus, $\langle x, s\rangle$ is an upper bound to the distance of $\langle x, c\rangle$ ($\langle y, b\rangle$) from the optimal in the primal (dual), thus, we require both the primal and dual to be close to the optimal. An abstract visualization of this concept is shown in Fig. 3. The primal polytope is shown below the dual polytope. Note that in this particular instance, the dual quickly gets close to the optimal, but the primal requires several more steps. Another thing to notice is that the interior point method may actually increase the objective function in order to decrease the barrier function in a given iteration, as seen in the second step in the primal polytope.

Definition 2.4. The primal-dual potential function associated with the primal-dual linear program is

$$G(x, s) = (n + \sqrt{n}) \ln(\langle x, s\rangle) - \sum_{i=1}^{n} \ln(x_i, s_i)$$

Claim 2.5. *If $G(x, s) \leq 2\sqrt{n}L$, then x is "almost optimal"*

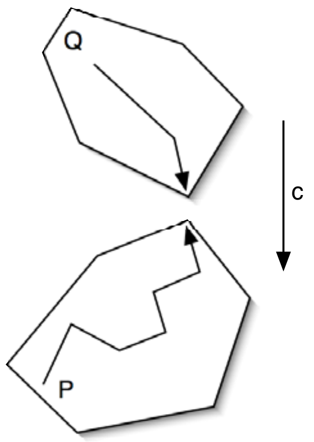

Figure 3: Primal-dual approach to interior point.

Proof. It is enough to show that if $G(x,s) \leq 2\sqrt{n}L$, then $\langle x,s\rangle < 2^{-2L}$ to prove this claim since $\langle x,s\rangle$ is an upper bound to the distance of $\langle x,c\rangle$ from the optimal and x is almost optimal.

$$G(x,s) = \sqrt{n}\ln(\langle x,s\rangle) + [n\ln(\langle x,s\rangle) - \sum \ln(x_i s_i)]$$

The term in brackets can be shown to be ≥ 0 by using Jensen's inequality because of the concavity of the logarithm function.

$$\begin{aligned} \ln(\langle x,s\rangle) = \ln(\sum x_i s_i) = \ln(n\frac{1}{n}\sum x_i s_i) \geq& \ln(n) + \frac{1}{n}\sum \ln(x_i s_i) \\ n\ln(\langle x,s\rangle) \geq& n\ln(n) + \sum \ln(x_i s_i) \\ n\ln(\langle x,s\rangle) - \sum \ln(x_i s_i) \geq& n\ln(n) \geq 0 \end{aligned}$$

$2\sqrt{n}L \geq G(x,s) \geq \sqrt{n}\ln(\langle x,s\rangle)$ and

$e^{-2L} \geq 2^{-2L} \geq \langle x,s\rangle$ as required.

□

2.1.2 The algorithm

One way to decrease the potential function, G, is to fix s and move x in a manner that follows the opposite direction of g, the gradient of G with respect to x evaluated at $(x^{(i)}, s^{(i)})$. At iteration i, the solution held is $(x^{(i)}, s^{(i)})$. For now, assume that $x^{(i)}$ is the all ones vector, i.e $x^{(i)} = \mathbf{1}$ (it is later shown how a linear transform can transform

x into this form).

$$\begin{aligned} g = \nabla_x G(x,s)\Big|_{(x^{(i)},s^{(i)})} &= \frac{n+\sqrt{n}}{\langle x,s\rangle} s - \begin{pmatrix} 1/x_1 \\ \vdots \\ 1/x_n \end{pmatrix}\Bigg|_{(x^{(i)},s^{(i)})} \\ &= \frac{n+\sqrt{x}}{\sum s_i} s - \mathbf{1} \end{aligned}$$

We can not simply move $x^{(i)}$ in the direction of $-g$ because $A(x^{(i)} - \epsilon g) = b - \epsilon A g \not\equiv b$. Consequently, the step must be restricted to the nullspace of A to satisfy the constraints.

$$\begin{aligned} Ax^{(i)} &= b \\ A(x^{(i)} - \epsilon d) &= b \\ Ax^{(i)} - \epsilon A d &= b \Rightarrow Ad = 0 \end{aligned}$$

Let d be the projection of g onto the kernel of A

$$d = (I - A^T(AA^T)^{-1}A)g$$

At this point, we know the preferred direction and we only need to determine the size of the step. One obvious limiting factor is that $x^{(i+1)} \geq 0$. Suppose a step size of $\frac{1}{4}$ in the direction of $-d$,

$$\begin{aligned} & x^{(i+1)} = x^{(i)} - \frac{1}{4}\frac{d}{\|d\|_2}, \text{ clearly } x_j^{(i+1)} = 1 - \frac{1}{4}\frac{d_j}{\|d\|_2} \geq \frac{3}{4} \\ & s^{(i+1)} = s^{(i)} \end{aligned}$$

Claim 2.6. *If* $\|d\|_2 \geq 0.4$*, then* $G(x^{(i)}, s^{(i)}) - G(x^{(i+1)}, s^{(i+1)}) \geq \frac{7}{120}$

Proof.

$$\begin{aligned} G(x^{(i)}, s^{(i)}) - G(x^{(i+1)}, s^{(i+1)}) &= G(\mathbf{1}, s^{(i)}) - G(\mathbf{1} - \frac{1}{4}\frac{d}{\|d\|_2}, s^{(i)}) \\ &= (n+\sqrt{n})\ln(\mathbf{1}^T s^{(i)}) - \sum_{j=1}^n \ln 1 - \sum_{j=1}^n \ln s_j^{(i)} \\ &\quad - (n+\sqrt{n})\ln(\mathbf{1}^T - \frac{d^T s^{(i)}}{4\|d\|_2}) \\ &\quad + \sum_{j=1}^n \ln(1 - \frac{d_j}{4\|d\|_2}) + \sum_{j=1}^n \ln s_j^{(i)} \\ &= \sum_{j=1}^n \ln(1 - \frac{d_j}{4\|d\|_2}) \\ &\quad - (n+\sqrt{n})\ln(1 - \frac{d^T s^{(i)}}{4\|d\|_2}\mathbf{1}^T s^{(i+1)}) \end{aligned}$$

Using the following inequality obtained from the Taylor series expansion of $\ln(1-x)$:
$-x - \frac{x^2}{2(1-a)} \geq \ln(1-x) \leq -x$, for $|x| \leq a < 1$

for $a = \frac{1}{4}$,

$$G(x^{(i)}, s^{(i)}) - G(x^{(i+1)}, s^{(i+1)}) \geq \frac{\|d\|_2}{4} - \frac{1}{24} \geq \frac{7}{120}, \text{ if } \|d\|_2 \geq 0.4$$

□

If $\|d\|_2$ is small, i.e. $\|d\|_2 < 0.4$, which can occur when g is almost perpendicular to the kernel of A, then a move in x does not gain much. In such a situation, we keep $x^{(i)}$ unchanged and move the dual variable in the opposite direction of the gradient, $g = \nabla_s G(x,s)\big|_{(x^{(i)},s^{(i)})}$, in an analogous manner to x above. It can be shown that also in the event, G decreases by a constant.

Up to this point, we had assumed that $x^{(i)} = \mathbf{1}$. Of course, this will not hold in general, but the problem can be transformed by a linear transformation, $\hat{X}$, so that $x' = \hat{X}^{-1}x = \mathbf{1}$,
$\hat{X} = \begin{pmatrix} \hat{x}_1 & & \\ & \ddots & \\ & & \hat{x}_n \end{pmatrix}$

Let $\hat{A} = A\hat{X}$ and $\hat{c} = c\hat{X}$. The transformed problem then becomes

$$\begin{aligned} &\min\langle \hat{c}, x' \rangle \\ &\hat{A}x' = b \\ &x' \geq 0 \end{aligned}$$

In the dual, the transformation is $S \rightarrow \hat{X}s$, so $x'_j s'_j$ remains unchanged, i.e. $x'_j s'_j = x_j s_j$ and G is related to x and s through $x_j s_j$.

Algorithm 2.7.
Ye's primal-dual interior point algorithm, adopted from [Mei97].

Input: $m, n \in \mathbb{N}, \epsilon > 0$, and$A \in \mathbb{R}^{mxn}$ of full row rank, $c \in \mathbb{R}^n, b \in \mathbb{R}^m$ and initial $x^{(0)}, y^{(0)}, s^{(0)}$ such that
$Ax^{(0)} = b, x^{(0)} > 0, y^{(0)T}A + s^{(0)T} = c^T, s^{(0)} > 0.$

Output: A feasible pair $(x^{(k)}, s^{(k)})$ such that $s^{(k)T}x^{(k)} < \epsilon$.

```
while($s^{(k)T}x^{(k)} < \epsilon$)

  1. Transform the feasible solution with respect to $x^{(k)}$:
     $x'^{(k)} = \hat{X}^{-1}x^{(k)} = \mathbf{1}, \quad s'^{(k)} = \hat{X}s^{(k)}, \quad A' = A\hat{X}$

  2. $g \leftarrow \frac{n+\sqrt{x^{(k)}}}{\sum s_i^{(k)}} s^{(k)} - \mathbf{1}$
```

3. $d \leftarrow (I - A'^T(A'A'^T)^{-1}A')g$
4. If $\|d\|_2 \geq 0.4$, then do a primal step: $x'^{(k+1)} \leftarrow \mathbf{1} - \frac{1}{4\|d\|_2}d$,
   else, do a dual step: $s'^{(k+1)} \leftarrow \frac{s'^T\mathbf{1}}{n+\sqrt{n}}(d+1)$
5. Transform the solution back to the original domain:
   $x^{(k+1)} \leftarrow \hat{X}x'^{(k+1)}, \quad s^{(k+1)} \leftarrow \hat{X}^{-1}s'^{(k+1)}$
6. $k \leftarrow k+1$

2.1.3 Analysis

Ye showed that that an initial pair $(x^{(0)}, s^{(0)})$ can always be constructed cheaply such that $G(x^{(0)}, s^{(0)}) = O(\sqrt{n})$. In each iteration, the potential function, G is decreased by at least some constant amount. From claim 2.5, the algorithm then requires $O(\sqrt{n}L)$ iterations for the duality gap, s^Tx to be guaranteed to be less than ϵ, i.e. for x to be almost optimal. Further, the operations in each iteration are rather minimal, with the most expensive being the computation of the projection, d, of the gradient, which requires $O(n^3)$ operations.

2.1.4 Implementation issues

- Robust to rounding because $\|d\|_2 \geq 0.4$ is arbitrary
- There are a few with respect to precision we may have, that unlike the ellipsoid algorithm, are not critical - they don't have bad performance consequences

References

[Mei97] G. Meinsma. Interior point methods. Mini course, Spring 1997.

[Tun98] L. Tunel. Convex optimization: Barrier functions and interior-point methods. Technical Report B-336, Department of Mathematical and Computing Sciences, Tokyo Institute of Technology, Tokyo, Japan, March 1998.