Advanced Heuristic and Approximation Algorithms Grenoble-INP (ORCO)

Lecture 4

Lecturer: Alantha Newman *October 23, 2017*

1 Efficient Algorithms for Linear Programming

Our goal is to present an efficient algorithm for the linear programming problem:

$$\begin{aligned} \min \quad & \mathbf{c}^\mathsf{T}\mathbf{x} \\ \text{subject to:} \quad & \mathbf{A}\mathbf{x} \ \geq \ \mathbf{b} \\ & \mathbf{x} \geq \ 0. \end{aligned}$$

This means that we want to find a feasible point $\mathbf{x} \in \mathbb{R}^n$ such that the value of $\mathbf{c}^\mathsf{T}\mathbf{x}$ is minimized. By *efficient*, we mean *polynomial in the size of the input.* Specifically, suppose that the input consists of $poly(n)$ integers, each with value at most U. (Note that as long as the input is rational, we can scale the input values so that all entries are integers.) Then this input instance can be described using at most $\log U \cdot poly(n)$ bits, since the integer U can be written in binary using $\log U$ bits. In this case, we say that the *size of the input* is upper bounded by $\log U \cdot poly(n)$. Thus, we say an algorithm is efficient if it runs in $O(\log U \cdot poly(n))$ steps.

In order to state the running times of our algorithms in terms on the input size, we will use some lemmas, which can be found in Chapter 8 of [BT97].

Lemma 1 (Lemma 8.4 [BT97]). *Let $P = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{A}\mathbf{x} \geq \mathbf{b}\}$ be a non-empty, full-dimensional and bounded polyhedron, where the entries of $\mathbf{A}$ and $\mathbf{b}$ are integer and have absolute value bounded by the integer U. Then,*

$$Vol(P) \ \geq \ \gamma \ = \ n^{-n}(nU)^{-n^2(n+1)}.$$

If P is not bounded or full-dimensional, this fact can be stated in modified form, but the details are somewhat tedious.

Lemma 2 (Lemma 8.2 [BT97]). *Let $P = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{A}\mathbf{x} \geq \mathbf{b}\}$ be a bounded polyhedron, where the entries of $\mathbf{A}$ and $\mathbf{b}$ are integer and have absolute value bounded by the integer U. Then every point $\mathbf{x} \in P$ satisfies*

$$-(nU)^n \leq x_j \leq (nU)^n, \quad j \in \{1, \ldots, n\}.$$

Again, if P is not bounded, we can bound the value of the *extreme points* of P, but for the sake of simplicity, let us assume that P is bounded. We will use two useful corollaries of Lemma 2.

Corollary 3. *Let $P = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{A}\mathbf{x} \geq \mathbf{b}\}$ be a bounded polyhedron, where the entries of $\mathbf{A}$ and $\mathbf{b}$ are integer and have absolute value bounded by the integer U. Then P is contained in a cube of width $(nU)^n$. So P is contained in a cube of volume Γ, where*

$$\Gamma \ = \ (nU)^{n^2}.$$

Corollary 4. *Let $P = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{A}\mathbf{x} \geq \mathbf{b}\}$ be a bounded polyhedron and let $\mathbf{c} \in \mathbb{R}^n$ be a linear objective function, where the entries of $\mathbf{A}$, $\mathbf{b}$ and $\mathbf{c}$ are integer and have absolute value bounded by the integer U. Then for any $\mathbf{x} \in P$, we have*

$$-(nU)^{n+1} \leq \mathbf{c}^\mathsf{T}\mathbf{x} \leq (nU)^{n+1}.$$

Note that neither the lower bound on the volume of P provided by Lemma 1 nor the upper bound provided by Corollary 3 depend on the number of constraints used to define the polyhedron P.

2 Reducing Optimization to Feasibility

Let K denote a convex set in $\mathbb{R}^n$. For example, K could be the polyhedron: $P = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{A}\mathbf{x} \geq \mathbf{b}\}$. For a linear objective function $\mathbf{c}$, we can define the following *optimization* problem.

$$\begin{aligned} \min \quad & \mathbf{c}^{\intercal}\mathbf{x} \\ \text{subject to:} \quad & \mathbf{x} \in K. \end{aligned}$$

We can also define the following—possibly easier—*feasibility* problem.

$$\text{Find any point } \mathbf{x} \in K \text{ or return "}K\text{ is empty".}$$

We will show that the optimization problem can be reduced to the feasibility problem. In other words, given a black box to solve the feasibility problem, we can efficiently solve the optimization problem. To do this, we can simply add another constraint so that the feasibility problem becomes:

$$\text{Find } \mathbf{x} \text{ such that } \mathbf{x} \in K \text{ and } \mathbf{c}^{\intercal}\mathbf{x} \geq t \text{ or return "}\{\mathbf{x} \mid \mathbf{x} \in K \text{ and } \mathbf{c}^{\intercal}\mathbf{x} \geq t\} \text{ is empty}''. \tag{1}$$

In other words, consider the convex set K', which consists of all the points in K that obey the constraint $\mathbf{c}^{\intercal}\mathbf{x} \geq t$. Then, then (1) becomes:

$$\text{Find any point } \mathbf{x} \in K' \text{ or return "}K'\text{ is empty".}$$

By conducting a binary search on t, we can find $\mathbf{x} \in K$ such that $\mathbf{c}^{\intercal}\mathbf{x}$ is maximized. Suppose the maximum value is $\tau = \mathbf{c}^{\intercal}\mathbf{z}$ attained for some $\mathbf{z} \in K$. Then finding a point with value τ requires at most $\log|\tau|$ iterations of an algorithm for the feasibility problem. Although we may not know the value of τ, by Corollary 4, we do know that $|\tau| \leq (nU)^{n+1}$. Thus, in at most $(n+1)\log(nU)$ steps, we can find a point $\mathbf{z} \in K$ with value τ. So to find an optimal solution for a linear program, it is sufficient to return a feasible point in a polyhedron.

3 Separation Oracles

Our next goal is to present an effcient algorithm for the feasibility problem. Suppose we have a polyhedron that has the following implicit description: Given a point $\mathbf{x} \in \mathbb{R}^n$, we have access to an efficient algorithm that returns one of two alternatives:

1. $\mathbf{x} \in P$, or
2. $\mathbf{x} \notin P$ and returns a violated constraint.

Such an algorithm is called a *separation oracle*. We claim that if we have a separation oracle for a polyhedron P (or more generally, a convex body), then we can solve the feasibility problem for P. This allows us to solve the feasibility problem for a polyhedron P with an exponential number of constraints, provided it has a separation oracle. We now give a specific example of a linear program with an exponential number of constraints that has a separation oracle.

3.1 Separation Oracle for TSP

Given a graph $G = (V, E)$, the *traveling salesman problem* is defined as follows. Find a connected, spanning Eulerian multigraph of G containing the minimum number of edges. This problem is NP-hard. Consider the following linear programming relaxation (P_{TSP}). For a nonempty subset $S \subset V$, let

$\delta(S) \subset E$ denote all edges with exactly one endpoint in S.

$$\begin{aligned} \min \sum_{e\in E} x_e & \\ x(\delta(i)) = 2, &\quad \text{for all } i \in V, \\ x(\delta(S)) \geq 2, &\quad \text{for all } S \subset V \text{ such that } S \neq \emptyset, \\ x_e \geq 0, &\quad \text{for each } e \in E. \end{aligned} \tag{P_{TSP}}$$

The linear program (P_{TSP}) has an exponential number of constraints, because there are an exponential number of subsets $S \subset V$. Given an $\mathbf{x} \in \mathbb{R}^{|E|}$, we wish to determine if $\mathbf{x} \in (P_{TSP})$ and if not, we want to find a violated constraint. We can easily check that the degree constraints and the nonnegativity constraints are respected. How can we check the cut constraints? We can find a minimum cut in G using the values x_e as edge weights. If the minimum cut has value at least 2, we can conclude that $\mathbf{x}$ belongs to (P_{TSP}). Otherwise, a cut with value less than 2 corresponds to a violated constraint.

3.2 Separating Hyperplanes

The concept of a separation oracle is closely related to that of a *separating hyperplane.*

Theorem 5. [Separating Hyperplane Theorem] *Let K be a non-empty closed convex subset of $\mathbb{R}^n$ and let $\mathbf{x}^* \in \mathbb{R}^n$ be a point such that $\mathbf{x}^* \notin K$. Then there exists some hyperplane $\mathbf{a}^\intercal \mathbf{y} = h$ such that $\mathbf{a}^\intercal \mathbf{x}^* \leq h$ and $\mathbf{a}^\intercal \mathbf{x} > h$ for all $\mathbf{x} \in K$.*

If K is a polyhedron (i.e. described by a set of linear inequalities), and we have a separation oracle for K, then this immediately yields a separating hyperplane for a point $\mathbf{x}^* \notin K$. Specifically, the violated constraint found by the separation oracle can be converted into a separating hyperplane. For example, let $K = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{A}\mathbf{x} \geq \mathbf{b}\}$ and let $\mathbf{a}_j$ denote the j^{th} constraint (i.e. the j^{th} row in the matrix $\mathbf{A}$). For some $\mathbf{x}^* \in \mathbb{R}^n$, suppose there is a constraint such that $\mathbf{a}_j^\intercal \mathbf{x}^* < b_j$ but $\mathbf{a}_j^\intercal \mathbf{x} \geq b_j$ for all $\mathbf{x} \in K$. Then setting $h = \mathbf{a}_j^\intercal \mathbf{x}^*$, we observe that $\mathbf{a}_j^\intercal \mathbf{y} = h$ is a separating hyperplane. (Notice that there are other ways to obtain a separating hyperplane from a violated constraint. We use this convention because the resulting hyperplane has the additional property that it goes "through" the point $\mathbf{x}^*$, which—as we will see in the next section—is a useful property.)

4 An Efficient Algorithm for the Feasibility Problem

Given a convex set $K \in \mathbb{R}^n$, our goal is now to find a point $\mathbf{x} \in K$ or to determine that K is empty. We can solve this problem efficiently if we are given an (efficient) separation oracle for K. If K is nonempty, then we have the following:

(i) A convex body C_0 with volume $Vol(C_0) = \Gamma$ that contains K. (Lemma 1.)

(ii) A lower bound on the volume of K, i.e. $Vol(K) \geq \gamma$. (Corollary 3.)

Moreover, we will show that we can construct a series of t convex bodies such that $C_{i+1} \subset C_i$ and $K \subset C_i$ for all $i \in \{0, \dots, t\}$. If K is nonempty and the volume of each successive convex body C_i decreases by a constant factor, then we can use the following lemma to lower bound the value of t in terms of Γ and γ, i.e. in terms of the size of the input.

Lemma 6. *Let K be a nonempty convex body. Let $C_0, C_1, \dots, C_t$ be a series of convex bodies such that:*

(a) $K \subset C_i$ for all $i \in \{0, \dots, t\}$.

(b)

$$\frac{Vol(C_{i+1})}{Vol(C_i)} < 1 - \frac{1}{e} \text{ for all } i \in \{0, \ldots, t\},$$

then $t < 3 \cdot \log \frac{\Gamma}{\gamma}$.

Proof. Proof by contradiction. Let $t^* = 3 \cdot \log \frac{\Gamma}{\gamma}$. Then we have:

$$\frac{Vol(C_{t^*})}{Vol(C_0)} = \prod_{i=0}^{t^*} \frac{Vol(C_{i+1})}{Vol(C_i)} < \left(1 - \frac{1}{e}\right)^{t^*} < \left(\frac{1}{e}\right)^{\log \frac{\Gamma}{\gamma}} < \frac{\gamma}{\Gamma}.$$

This implies:

$$Vol(C_{t^*}) < \gamma.$$

By assumption we have $K \subset C_{t^*}$. Therefore, if K is nonempty, then $Vol(C_{t^*}) \geq Vol(K) \geq \gamma$. We can conclude that $t < t^*$. □

Now we will show that we can construct this series of convex bodies $\{C_0, \ldots, C_t\}$ with the required properties and use them to solve the feasibility problem.

4.1 Constructing a Shrinking Set of Convex Bodies

Recall that we are given a separation oracle for K. That is, for some $\mathbf{x}^* \in \mathbb{R}^n$, the separation oracle returns "yes" if $\mathbf{x}^* \in K$ and "no" if $\mathbf{x}^* \notin K$. Additionally, if $\mathbf{x}^* \notin K$, the separation oracle provides a separating hyperplane $\mathbf{a}^\intercal \mathbf{y} = h$ such that $\mathbf{a}^\intercal \mathbf{x}^* \leq h$ and $\mathbf{a}^\intercal \mathbf{x} > h$ for all $\mathbf{x} \in K$. We will use this to show that we can construct a series of convex bodies $\{C_0, \ldots, C_t\}$ such that:

(i) $K \subset C_i$ for all $i \in \{0, \ldots, t\}$, and

(ii) $\frac{Vol(C_{i+1})}{Vol(C_i)} < 1 - \frac{1}{e}$.

The centroid of a convex body K is defined to be the average value of all the points in K. The following theorem is due to Grünbaum.

Theorem 7. [Grü60] *Any halfspace that contains the centroid of K contains at least a $\frac{1}{e}$-fraction of the volume of K.*

We test the centroid of C_0 (which is the origin) and ask if it belongs to K. If so, we are done. If not, we find a separating hyperplane and construct a new, smaller convex body C_1 (possibly no longer a cube) that still contains K. Specifically, this new convex body is the intersection of the current convex body (i.e. C_0, initially) with the separating hyperplane. By Theorem 7, this separating hyperplane cuts off a $\frac{1}{e}$-fraction of the volume, so we satisfy condition (ii). We can also verify that condition (i) is satisfied by the new convex body, since K belongs to C_0 and K is fully contained on the positive side of the separating hyperplane. We then repeat this procedure on the new convex body. We summarize this procedure here. Let $t^* = 3 \cdot \log \frac{\Gamma}{\gamma}$.

CUTTING-CENTROID(K)

1. Let C_0 be an axis-aligned cube that contains K with center $\mathbf{z}_0 = 0$.
2. For $i = 0$ to t^*:
   (a) If $\mathbf{z}_i \in K$, terminate and output $\mathbf{z}_i$.
   (b) Otherwise, find a constraint $\mathbf{a}_j^\intercal \mathbf{x} \geq b_j$ in K violated by $\mathbf{z}_i$.
      i. Set $h = \mathbf{a}_j^\intercal \mathbf{z}_i$.
      ii. Let $C_{i+1} = C_i \cap \{\mathbf{x} \mid \mathbf{a}_j^\intercal \mathbf{x} \geq h\}$.
      iii. Let $\mathbf{z}_{i+1}$ be the point at the center of C_{i+1}.
3. Terminate and declare $K = \emptyset$.

Note that after t^* iterations of Step 2, if we have not yet found a point in K, then by Lemma 6, we can conclude that K is empty. Given an efficient subroutine for computing the centroid of a convex body, the algorithm CUTTING-CENTROID(K) is efficient. However, the problem of finding the centroid of a convex body is computationally intractable. But we can modify the algorithm to use an efficient procedure for computing an *approximate* centroid. The efficiency of the modified algorithm relies on the following theorem.

Theorem 8. **[BV04]** *Let K be a convex body in $\mathbb{R}^n$ and let $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$ denote $N = O(n)$ points in K, each chosen uniformly at random. Define $\mathbf{z} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i$. Then with high probability, any halfspace that contains $\mathbf{z}$ contains at least a $\frac{1}{3}$-fraction of the volume of K.*

Theorem 8 shows that an approximate centroid can be generated with high probability, using subroutines for generating random points in polytopes, which is a well-studied problem and can be done efficiently. Using Theorem 8, we obtain the following algorithm.

CUTTING-RANDOM-CENTROID(K)

1. Let C_0 be axis-aligned cube that contains K with center $\mathbf{z}_0 = 0$.
2. For $i = 0$ to t^*:
   (a) If $\mathbf{z}_i \in K$, terminate and output $\mathbf{z}_i$.
   (b) Otherwise, find a constraint $\mathbf{a}_j^\intercal \mathbf{x} \geq b_j$ in K violated by $\mathbf{z}_i$.
      i. Set $h = \mathbf{a}_j^\intercal \mathbf{z}_i$.
      ii. Let $C_{i+1} = C_i \cap \{\mathbf{x} \mid \mathbf{a}_j^\intercal \mathbf{x} \geq h\}$.
      iii. Generate $N = O(n)$ random points $\mathbf{x}_1, \mathbf{x}_, \ldots, \mathbf{x}_N$ in C_{i+1}.
      iv. Let $\mathbf{z}_{i+1} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i$.
3. Terminate and declare $K = \emptyset$.

Note that Lemma 6 still applies even when we cut off only a $\frac{1}{3}$-fraction of the convex body at each step rather than a $\frac{1}{e}$-fraction. The algorithm CUTTING-RANDOM-CENTROID(K) can be viewed as a reduction from the problem of finding a point in a convex body to the problem of finding an approximate centroid of a convex body. For more on the complexity of the latter problem, see [Rad07].

References

[BT97] Dimitris Bertsimas and John N. Tsitsiklis. *Introduction to linear optimization.* Athena Scientific Belmont, MA, 1997.

[BV04] Dimitris Bertsimas and Santosh Vempala. Solving convex programs by random walks. *Journal of the ACM*, 51(4):540–556, 2004.

[Grü60] Branko Grünbaum. Partitions of mass-distributions and of convex bodies by hyperplanes. *Pacific Journal of Mathematics*, 10(4):1257–1261, 1960.

[Kha80] Leonid G. Khachiyan. Polynomial algorithms in linear programming. *USSR Computational Mathematics and Mathematical Physics*, 20(1):53–72, 1980.

[Lev65] A. Yu. Levin. On an algorithm for the minimization of convex functions. *Soviet Mathematics Doklady*, 6:286–290, 1965.

[Rad07] Luis A. Rademacher. Approximating the centroid is hard. In *Proceedings of the twenty-third Annual Symposium on Computational Geometry*, pages 302–305. ACM, 2007.

[Sho72] Naum Z. Shor. Utilization of the operation of space dilatation in the minimization of convex functions. *Cybernetics and Systems Analysis*, 6(1):7–15, 1972.

[YN76] David B. Yudin and Arkadii S. Nemirovski. Evaluation of the information complexity of mathematical programming problems. *Ekonomika i Matematicheskie Metody*, 12:128–142, 1976.

These lectures notes are based in part on Chapter 8 of [BT97]. The algorithms in Section 4 are due to Levin [Lev65] and to Bertsimas and Vempala [BV04]. We note that at a high level, these centroid-cutting algorithms resemble the classic *ellipsoid algorithm*, but in several key ways, they are much simpler. The ellipsoid algorithm was developed in the works of Shor [Sho72], Yudin and Nemirovskii [YN76] and proved to be implementable in polynomial time in the famous paper of Khachiyan [Kha80]. In the ellipsoid algorithm, each iteration requires a non-trivial computation of a new ellipsoid that is guaranteed to contain K. But determining the center of an ellipsoid is easy. In contrast, in the algorithm of Bertsimas and Vempala, it is easy to compute the next convex body, but computing the (approximate) center is non-trivial.