18.409 An Algorithmist's Toolkit 10/29/09

Lecture 14

Lecturer: Jonathan Kelner

1 Approximating the volume of a convex body

Exactly computing the volume a convex body is known to be #P-hard, so the fact that we can approximate its volume in P is surprising–the kind of result you would bet against until you saw it was true.

Before discussing any algorithms, though, we need to say what it means to be given a convex body K. To keep our implementation as general as possible, we'll assume that K is given by some oracle:

1. Membership oracle
   Given a point p, returns 'yes' if $p \in K$ and 'no' if $p \notin K$.

2. Separation oracle
   Given a point p, returns 'yes' if $p \in K$ and a separating hyperplane H if $p \notin K$.

Given a reasonable description of a convex body, it is easy to build a separation oracle. For example,

1. A ball of radius r
   Given p, compute its norm $|p|$. If less than r, return 'yes'; if greater than r, return the hyperplane tangent to the boundary sphere at $rp/|p|$.

2. A cube of side length s
   Given p, compute its l_∞ norm. If less than $s/2$, return 'yes'; if greater, return the face of the cube in the violated direction.

3. A polytope
   Given p, check each inequality. If it satisfies them all, return 'yes'; if not, return the failed inequality.

In what follows, we'll assume that our convex body contains a ball of radius 1 centered at the origin, and is contained within a ball of radius $2^{\mathrm{poly(n)}}$. These conditions are reasonable–after suitable translation and dilation, they hold for any K specified by inequalities of polynomial bit length.

Given a membership oracle, how could we approximate volume? The naive Monte Carlo algorithm–pick points from some designated region (say a ball) and check if they're in K–in general fails. If K is an ellipse with major axis of exponential length l and minor axis l^{-1}, then the probability of a successful trial is exponentially small. No chance of a polynomial time algorithm. But if the body is well-rounded, say $B_2^n \subseteq K \subseteq nB_2^n$, then the following algorithm has a chance:

1. Pick points $p_1, \cdots, p_m$

2. Check if $p_i \in K$

3. Set $K' := \mathrm{conv}\{p_i | p_i \in K\}$

4. Return the volume of K'

If $n = 2$, this algorithm works:

Theorem 1 *For any $\epsilon > 0$, there exists a set $P = \{p_1, \cdots, p_m\}$ s.t. m is polynomial in $1/\epsilon$ and for any well-rounded 2-dimensional convex body K,* $\mathrm{Vol}(\mathrm{conv}(P \cap K)) \geq \mathrm{Vol}(K)/(1+\epsilon)$.

For example, a grid with spacing $\epsilon/8$ works. In higher dimensions, though, a grid has exponentially many points. It's also difficult to compute the convex hull of a bunch of points in high dimensions. We could try to construct our set of points more carefully, perhaps tailoring them based on our knowledge of the body so far. It turns out that such an algorithm cannot succeed.

Theorem 2 *There is no deterministic poly time algorithm that, given a membership oracle for K, computes* $\mathrm{Vol}(K)$ *within a polynomial factor.*

Proof Since the algorithm is deterministic, an adversary can construct a worst-case K depending on the queries. Her evil plan is to answer 'yes' to each query p if $p \in B_2^n$, so at the end of the algorithm, the only data known about the convex body is that it contains a polynomial number of points $P = \{p_1, \ldots, p_m\} \subset B_2^n$, and not certain points outside of the ball. Hence the algorithm cannot distinguish between $K_1 = B_2^n$ and $K_2 = \mathrm{conv}(p_1, \ldots, p_m)$. We will show that for *any* such polynomial collection of points, the ratio $\mathrm{Vol}(K_1)/\mathrm{Vol}(K_2)$ is exponentially large, dooming our algorithm to defeat.

For each p_i, denote B_i by the ball centered at $p_i/2$ of radius $|p_i|/2$. Now we claim that $\mathrm{conv}(P) \subseteq \bigcup B_i$. We can rewrite B_i as $\{x | \angle p_i x O \geq \pi/2\}$. Let $v \in \overline{p_i p_j}$. We'll show that $B_v \subset B_i \cup B_j$, where B_v is the ball centered at $v/2$ of radius $|v|/2$. For any point $x \in B_v$, we have $\angle vxO \geq \pi/2$. We consider three cases:

1. $x \in \triangle O p_i p_j$. Then $\angle Oxp_i + \angle Oxp_j + \angle p_j x p_i = 2\pi$ gives $\angle Oxp_i + \angle Oxp_j \geq \pi$, and one angle must be at least $\pi/2$.
2. x is outside the triangle in the p_i direction, so $\angle Oxp_i \geq \angle Oxv \geq \pi/2$
3. x is outside the triangle in the p_j direction, so $\angle Oxp_j \geq \angle Oxv \geq \pi/2$

Hence $B_v \subset \bigcup B_i$ for any v in the boundary of the convex hull of the p_i. Since any $x \in$ conv(P) is a linear combination of two points v, w on the boundary, $x \in B_x \subset B_v \cup B_w \subset \bigcup B_i$. Hence the volume of the convex hull is at most

$$\mathrm{conv}(\mathrm{Vol}(P)) \leq \sum_{i=1}^{m} \mathrm{Vol}(B_i) \leq \frac{m}{2^n} \mathrm{Vol}(B_2^n).$$

■

One can show that separation oracles are also insufficient for creating a deterministic polynomial time algorithm. It is worth noting that, together with our randomized algorithm for approximating the volume of a convex body, we have proved that the separation oracle A separates P from BPP, i.e., $P^A \neq BPP^A$. But it is widely believed that $P = BPP$. What's going on? There exist bodies without polynomial time separation oracles.

2 The Algorithm

We will give a randomized, polynomial time algorithm for approximating the volume of a convex body, given a separation oracle. The presentation roughly follows the original Dyer, Frieze, Kannan paper, and gives a very bad polynomial (degree ≈ 30). There are now algorithms running in $O(n^4)$. The strategy is similar to the one we used to approximate the permanent, finding a nested sequence of sets where random sampling hits with polynomial probability, and then multiplying the ratios.

Given a method for sampling from a convex body, we can implement the following (sketched) algorithm:

1. Change coordinates so that K is well-rounded , $B \subseteq K \subseteq nB$
2. Let $\rho = 1 + 1/n$, and let $K_i = K \cap \rho^i B$. Compute

$$\gamma_i = \frac{\mathrm{Vol}(K_{i-1})}{\mathrm{Vol}(K_i)}$$

3. Return $\text{Vol}(B)\prod \gamma_i$

The first step can be done with the separating oracle and the ellipsoid algorithm, or the method on the problem set. The last step works since $K_0 = K \cap B = B$ and $K_N = K \cap nB = K$. For the second step, we need to sample. It's easy to sample from highly symmetric objects: the cube is given by n uniforms, $U[0,1]$, the sphere by n gaussians, appropriately rescaled, the ball by picking the direction, then the radius. For nonsymmetric bodies, the best bet is a random walk. There are a few ways walk:

1. **Grid Walk**
   Intersect a grid with the body; walk on the resulting graph.
2. **Ball Walk**
   At a point p, pick a random neighbor in a small ball centered at p, and walk there.
3. **Hit and Run**
   At a point p, draw a random line l through p and walk to a random point $l \cap K$.

We'll use the grid walk. Drop a width δ grid on $\mathbb{R}$, the graph H with vertices $\delta\mathbb{Z}^n$ with edges $p \to p \pm \delta e_i$, and set $G = H \cap K$. We can walk on G using a membership oracle; walk on H, and if you would go to a neighbor not in G, choose again. Note that H has degree $2n$, but exponentially many vertices, so we need to show that the walk mixes very quickly. This is plausible though, and is easily seen when G is just a cube with side length $\leq n/\delta$. Since the path $P_{n/\delta}$ mixes in time polynomial in n/δ, and the cube is just the product P^n, its mixing time is n times that of the path, so still polynomial. There are still many problems with using a the walk on G to approximate K.

1. We're only sampling lattice points. After walk mixes, we could take a random vector v from the cube of width δ centered at $p \in G$. But if $p + v \notin K$, we're in trouble. We could throw it out and re-sample, but this would overweight points near the boundary. Alternatively, we could start the whole walk over, which is acceptable as long as the probability of landing outside is small.
2. The graph might be (close to) bipartite. Just use the lazy walk.
3. The graph has nonconstant degree. Throw in self-loops for vertices near boundary. Equivalently, our walk is: pick a random vector $v \in \pm e_i$; if $p + v \in K$, go there, otherwise, stay put.
4. The graph G might not be connected! If K has a sharp angle, then the vertex of G closest to the corner will not be adjacent to any other vertices of G. Finer grids don't help, as this is a problem with the angle itself.

We'll fix the last problem next lecture, by walking on the graph G associated to $K' = (1 + \epsilon)K$.

18.409 Topics in Theoretical Computer Science: An Algorithmist's Toolkit
Fall 2009