6.966 Algebra and Computation October, 21 1998

Lecture 11

Lecturer: Madhu Sudan *Scribe: Adam Klivans*

1 Introduction

In the last lecture, we saw how finding a short vector in a lattice plays an important part in a polynomial time algorithm for factoring polynomials with rational coefficients. Here, we present the well known basis reduction algorithm of Lenstra, Lenstra, and Lovasz. We can use this algorithm to find a relatively short vector in a lattice. Many of the details in this set of notes are based on L. Lovasz's book "An Algorithmic Theory of Numbers, Graphs, and Convexity" and R. Kannan's "Algorithmic Geometry of Numbers." Most details can be also be read from the original source, i.e., the paper "Factoring polynomials with rational coefficients" by A. K. Lenstra, H. W. Lenstra Jr. and L. Lovasz.

1.1 Preliminaries

We would like to point out that without loss of generality, all of our proofs will concern lattices that are specified by a set of basis vectors with integral entries (A lattice specified by basis vectors with rational entries can be transformed into one with integral entries via multiplication by the least common multiple).

2 Orthogonalization

The LLL algorithm makes use of some important relationships between orthogonal bases and short vectors. To begin, we review the Gram-Schmidt orthogonalization procedure. This algorithm takes in as input a set of basis vectors for R^n and produces an orthogonal basis. The algorithm works by iterating over all basis vectors and, at step i, takes basis vector i and subtracts from basis vector i its projection onto the already constructed orthogonal basis vectors.

GS-Orthogonalization Algorithm

Input: A set of basis vectors $(a_1, \ldots, a_n)$ for R^n.
Output: A set of orthogonal basis vectors $(a_1^*, \ldots, a_n^*)$ for R^n.

1 Let $a_1^* = a_1$

2 Compute $a_i^* = a_i - \sum_{j=1}^{i-1} \frac{(a_i, a_j^*)}{\|a_j^*\|^2} a_j^*$.

(Note we repeat step (2) for all $1 \leq i \leq n$.)
Furthermore, by expanding the resulting expressions, we can write $a_i = \sum_{j=1}^{i} \mu_{ij} a_j^*$.

Note that in the above algorithm, we started with a basis for R^n and retrieved a new basis. For the rest of this lecture, we will take a basis for a lattice and retrieve its orthogonalization. This orthogonalization is not necessarily a basis for the lattice. Given a basis we can, however, still refer to its orthogonalization as the output of the Gram-Schmidt process. Now that we have seen how to compute an orthogonalization from a set of basis vectors, we show what relationship the orthogonal vectors have to the shortest vector in the lattice. The following important lemma (that we will use later) shows that the size of the shortest vector in an orthogonalization computed via the Gram-Schmidt procedure is always a lower bound for the shortest vector in the lattice.

Lemma 1 *Let $(b_1, \ldots, b_n)$ be a basis for a lattice L and let $(b_1^*, \ldots, b_n^*)$ be the vectors obtained from its orthogonalization. Then for any $b \in L$, $||b|| \geq \min_i\{||b_i^*||\}$.*

Proof: Let $b \in L$. Then $b = \sum_{i=1}^k \lambda_i b_i$ where $1 \leq k \leq n$ and $\lambda_i \neq 0$. Then we can substitute in for each b_i its orthogonal representation. I.e., $b_i = \sum_{j=1}^i \mu_{ij} b_j^*$. So, we can rewrite $b = \sum_{i=1}^k \lambda_i' b_i^*$ and since $u_{ii} = 1$ for all i, $\lambda_k' = \lambda_k$, a non-zero integer. So

$$||b||^2 = \sum_{i=1}^k ||\lambda_i'|^2 ||b_i^*||^2 \geq |\lambda_k|^2 ||b_i^*||^2 \geq ||b_k^*||^2$$

□

2.1 Gauss' Reduction Algorithm

The LLL algorithm can be viewed as a generalization of a simple basis reduction algorithm in two dimensions due to Gauss. The idea is, given u and v, try to subtract off as much of u's projection onto v from u as possible. It is a discrete form of an orthogonalization. We formalize this as follows:

Lemma 2 *Given two lattice "basis" vectors u and v we can in polynomial time output a new lattice basis u' and v' such that the acute angle between u' and v' is at least 60 degrees.*

Proof: The algorithm is simple: suppose without loss of generality that $||u|| \leq ||v||$, and replace v with v' equal to the shortest vector of the form $v - mu$ for some $m \in \mathbb{Z}$. Notice that choosing the nearest integer to $\frac{v \cdot u}{u \cdot u}$ suffices. Also notice the length of the component of v' in the direction of u, i.e., $\frac{v \cdot u}{u \cdot u} u$, is less than $||\frac{1}{2}u||$. Now if $||v'|| \geq ||u||$ stop. Otherwise swap u and v' and repeat.
□

Note that by modifying Gauss' algorithm to stop only if $||v'|| \geq (1-\epsilon)||u||$ then the length of the shortest basis vector decreases by a factor of at least $(1-\epsilon)$ after each iteration.

3 The LLL Algorithm

Before giving the LLL algorithm, we state formally what it means for a basis to be reduced. The definition may seem somewhat arbitrary, but it will be useful for finding a short vector in a lattice.

Definition 3 *We say that a basis $(b_1, \ldots, b_n)$ is reduced if each μ_{ij} corresponding to its orthogonalization is less than $\frac{1}{2}$ and, for each i, $||b_i^*||^2 \leq \frac{4}{3}||b_{i+1}(i)||^2$ where $b_{i+1}(i)$ is the component of b_{i+1} orthogonal to $b_1, \ldots, b_{i-1}$.*

Here is an outline of the LLL basis reduction algorithm: The algorithm keeps a list of the vectors corresponding to our candidate short basis. Initially the list is simply our input vectors. We will sort the vectors in increasing length. Then, we will subtract small vectors from large vectors (a la Lemma 2) until one vector's magnitude decreases significantly and should be moved around in the list in order to maintain our length increasing invariant. We then repeat these two steps. For clarity, we separate the two steps and describe them in more detail.

3.1 Step 1– Applying Gauss' Reduction

Lemma 4 *Given a basis $(b_1, \ldots, b_n)$ with orthogonalization $(b_1^*, \ldots, b_n^*)$ we can, in polynomial time, find a new basis $(b_1', \ldots, b_n')$ where $b_i' = \sum_{j=1}^i \mu_{ij}' b_j^*$ with $|\mu_{ij}'| \leq \frac{1}{2}$.*

Proof and Algorithm for Step 1: Assume that $(b_1, \ldots, b_n)$ does not have the desired property. Then there must be indices ij such that $|\mu_{ij}| \geq \frac{1}{2}$. Choose the indices ij where j is maximized. Let m be the closest integer to $|\mu_{ij}|$. Replace b_i with $b_i' = b_i - mb_j$. It should be clear that in our updated basis $(b_1, \ldots, b_{i-1}, b_i', \ldots, b_n)$, μ_{ij} (now μ_{ij}') is less than $\frac{1}{2}$ as we have performed one iteration of Gauss' algorithm above. Repeating this at most $O(n^2)$ times (one time for each pair i, j) we will achieve the desired basis.
□

3.2 Step 2– Swapping Vectors

Find an index i such that such that $||b_i^*||^2 > \frac{4}{3}||b_{i+1}(i)||^2$ and swap b_i with b_{i+1}.

3.3 Proof of Termination

Certainly if we cannot carry out step 1 or step 2 we have found a reduced basis. We now prove that we only need to repeat steps 1 and 2 a polynomial number of times.
Proof of termination: Consider the quantity $D(b_1, \ldots, b_n) = \prod_{i=1}^{n} ||b_i^*||^{n-i}$. We will show that Step 1 does not change the quantity $D(b_1, \ldots, b_n)$ and that Step 2 only decreases the quantity $D(b_1, \ldots, b_n)$. Since initially $D(b_1, \ldots, b_n)$ is at most exponentially large in the size of the largest vector,and $D(b_1, \ldots, b_n)$ is always larger than 1, we can only decrease $D(b_1, \ldots, b_n)$ by a constant factor a polynomial number of times. This will prove that our algorithm terminates in polynomial time.

- Clearly Step 1 leaves the quantity $D(b_1, \ldots, b_n)$ unchanged.
- In Step 2, we are putting a bigger lattice vector farther down our list. Thus the quantity $D(b_1, \ldots, b_n)$ must be decreasing by at least a factor of $\frac{2}{\sqrt{3}}$.

As a crude upper bound we see that $D(b_1, \ldots, b_n) \leq (\max_i ||b_i||)^{n^2}$.

Now we need to show $D(b_1, \ldots, b_n)$ is always greater than 1. Let C_k be the $k \times k$ matrix whose ith column is b_i and R_k matrix be the matrix whose ith row is b_i. Notice that for any k, $det(R_k C_k) = det(R_k) det(C_k) \geq 1$ (recall our basis vectors have integer entries, and $\det(A) = \det(B)$ for any bases A and B). But, $det(R_k C_k) = ||b_1^*||^2 \cdots ||b_k^*||^2$ since the determinant of the matrix consisting of orthogonal basis vectors is the product of the norms of the vectors. Thus, $D(b_1, \ldots, b_n) \geq 1$.

Since $D(b_1, \ldots, b_n)$ is only a single exponential in the size of the input and is always greater than 1, we can only repeat steps 1 and 2 a polynomial number of times.
□

3.4 Retrieving a Short Vector

Here we show how finding a short basis immediately yields a relatively "short" vector.

Theorem 5 *Let $(b_1, \ldots, b_n)$ be a reduced basis of the lattice L. Then $||b_1|| \leq 2^{\frac{n-1}{2}} \lambda(L)$ where $\lambda(L)$ is the length of the shortest vector in the lattice.*

Proof: (taken directly from Lovasz's An Algorithmic Theory of Numbers, Graphs, and Convexity)

$$\begin{aligned}
||b_i^*||^2 \quad = ||b_i(i)||^2 \quad &\leq \frac{4}{3}||b_{i+1}(i)||^2 \\
&= \frac{4}{3}||b_{i+1}^* + \mu_{i+1,i}||b_i^*||^2 \\
&= \frac{4}{3}||b_{i+1}^*||^2 \frac{4}{3}\mu_{i+1,i}^2||b_i^*||^2 \\
&\leq \frac{4}{3}||b_{i+1}^*||^2 + \frac{1}{3}||b_i^*||^2
\end{aligned}$$

Thus, $||b_{i+1}^*||^2 \geq \frac{1}{2}||b_i^*||^2$ and inductively we see that $||b_i^*||^2 \geq 2^{1-i}||b_1^*||^2 = 2^{1-i}||b_1||^2$. By Lemma 1, $||b_1||^2 \leq \min_i\{2^{i-1}||b_i^*||^2\} \leq 2^{n-1}\min_i ||b_i^*||^2 \leq 2^{n-1}\lambda(L)^2$ which proves the theorem.

□

4 Root Finding and Factorization

In this course we've covered a number of different factorization algorithms. One category of factorization algorithms we don't cover is factorization over the reals. Here we briefly mention one important concept that arises in this study: namely, the method of Sturm sequences. We illustrate this concept to describe its role in (approximately) finding real roots of a real polynomial. ' Observe that in many cases, root-finding seems much easier than factorization. For example, if we have a bivariate polynomial $f(x, y)$ and we know that $y - p(x)$ divides $f(x, y)$ for some polynomial $p(x)$, then we know that for any polynomial $r(x)$, $f(x, r(x))$ will have $r(x) - p(x)$ as an irreducible factor.

4.1 Sturm sequences

Sturm sequences are used to determine the number of real roots that a given polynomial over the reals has in some interval (a, b). Given f, we compute f', its derivative, and we define a sequence of polynomials h_i inductively as follows:

$$h_{i-2} = q_{i-1}h_{i-1} - h_i$$

Now let $Var_a(h_0(a), \ldots, h_n(a))$ = the number of sign changes of this sequence. The interesting fact about these polynomials is captured in the following theorem:

Theorem 6 *For a given polynomial f, the number of distinct zero locations in an interval (a, b) is equal to $Var_a - Var_b$.*

One can find a proof for this in B. Mishra's book "Algorithmic Algebra" published by Springer-Verlag.