Computational Number Theory and Algebra June 13, 2012

Lecture 16

Lecturers: Markus Bläser, Chandan Saha *Scribe: Chandan Saha*

In today's class, we will complete the analysis of the LLL basis reduction algorithm, and then move on to show how short vector computation is used to factor polynomials over rationals. The topics of discussion for today's class are:

- Analysis of the LLL basis reduction algorithm (continued from previous class), and
- Factoring polynomials over rationals.

1 The LLL basis reduction algorithm (contd.)

We will continue to use the notations introduced in the previous class.

Algorithm 1 **LLL basis reduction**

1. Initialize $U = V$ and compute GSO of U. Set $i = 2$.
2. while $i \leq n$ do
3. for $j = i - 1$ to 1 do
4. Set $u_i = u_i - \lceil \mu_{ij} \rfloor u_j$ and update GSO of U.
5. if $i > 1$ and $\|u_{i-1}^*\|^2 > 2\|u_i^*\|^2$ then
6. Swap u_i and u_{i-1} and update GSO of U. Set $i = i - 1$.
7. else, set $i = i + 1$.
8. Return U.

Lemma 1 *The* LLL *algorithm halts and finds a reduced basis in polynomial time.*

Proof (Continued from the previous class) We have already shown in the previous class that the algorithm halts after polynomially many executions of the `while`-loop as the index i can decrease for at most $O(n^2 \log A)$ times in step 6. In order to complete the proof, we also need to show that the size of the numerator and denominator of any rational number involved in the computation is polynomially bounded.

We recall the following facts from the previous class. Indices i and j are fixed as in step 4. After an update in step 4, the $(k, \ell)^{th}$ entry of the projection matrix M has absolute value at most $\frac{1}{2}$ for all $1 \leq \ell < k < i$ and for all $j \leq \ell < k = i$. Further, the orthogonal basis of U (i.e. U^*) remains unaltered. After an update in step 6, $\tilde{u}_k^* = u_k^*$ for all $k \notin \{i-1, i\}$, $\|\tilde{u}_{i-1}^*\|^2 < \frac{3}{4}\|u_{i-1}^*\|^2$ and $\|\tilde{u}_i^*\| \leq \|u_{i-1}^*\|$, i.e. $\max_k\{\|\tilde{u}_k^*\|\} \leq \max_k\{\|u_k^*\|\}$. Also, recall the definition of $d_k = \det(U_k \cdot U_k^T) \in \mathbb{Z}^+$, where U_k be a $k \times m$ matrix with rows $u_1, \ldots, u_k$. It follows that $d_k = \prod_{\ell=1}^{k} \|u_\ell^*\|^2$.

We are now left with the task of showing that all the rational numbers in U, U^* and M have small numerators and denominators at any stage of the algorithm. We begin with the row vectors in U. Since, every $u_k \in \mathbb{Z}^m$, it is sufficient to bound the value of $\|u_k\|$. To start with, $\|u_k\| \leq A$ for every k. Since $\|u_k^*\| \leq \|u_k\|$ and $\max_k\{\|u_k^*\|\}$ never increases after an update in step 4 or step 6, hence $\|u_k^*\| \leq A$ for every k at all times of the algorithm.

First, we claim that $\|u_k\| \leq \sqrt{n} \cdot A$ for every k at all times of the algorithm, except in step 4 when $k = i$. Since, step 6 does only a swap between u_{i-1} and u_i, we only need to show that the claim holds for $\|u_i\|$ at the end of the `for`-loop, just before step 5. As $u_i = u_i^* + \sum_{j<i} \mu_{ij} u_j^*$, $\|u_i\|^2 \leq n m_i^2 A^2$ where

$m_i = \max_{1 \le j \le i}\{|\mu_{ij}|\}$. Notice that, at the end of the `for`-loop $|\mu_{ij}| \le \frac{1}{2}$ for every $j < i$. Hence, by taking into account that $\mu_{ii} = 1$, we have $\|u_i\| \le \sqrt{n}A$.

Let us see how $\|u_i\|$ changes within the `for`-loop. For this, we first need the following bound on the value of $\mu_{k\ell}$ for any $\ell < k$.

$$|\mu_{k\ell}| = \frac{|u_k \cdot u_\ell^*|}{\|u_\ell^*\|^2} \le \frac{\|u_k\| \cdot \|u_\ell^*\|}{\|u_\ell^*\|^2} = \frac{\|u_k\|}{\|u_\ell^*\|} \le \sqrt{d_{\ell-1}} \cdot \|u_k\|. \tag{1}$$

The last inequality holds because $\|u_\ell^*\|^2 = \frac{d_\ell}{d_{\ell-1}} \ge \frac{1}{d_{\ell-1}}$ as $d_\ell \in \mathbb{Z}^+$. Now notice that, after an update in step 4, $\mu_{i\ell}$ changes to $\mu_{i\ell} - \lceil \mu_{ij} \rfloor \mu_{j\ell}$ for every $\ell \le j$. Since $|\mu_{j\ell}| \le \frac{1}{2}$ for $\ell < j$,

$$|\mu_{i\ell} - \lceil \mu_{ij} \rfloor \mu_{j\ell}| \le |\mu_{i\ell}| + |\lceil \mu_{ij} \rfloor| \cdot |\mu_{j\ell}| \le m_i + (m_i + \frac{1}{2}) \cdot \frac{1}{2} \le 2m_i.$$

The last inequality holds because $m_i \ge 1$. Also, for $\ell = j$, $\mu_{i\ell} - \lceil \mu_{ij} \rfloor \mu_{j\ell} \le \frac{1}{2}$. Therefore, the value of m_i is at most doubled after an update in step 4. We have already shown that, at the start of the `for`-loop, $\|u_k\| \le \sqrt{n}A$ for every k. Hence, from equation (1), $m_i \le \sqrt{n}A^{n-1}$ at the start of the `for`-loop. Together with the facts that m_i can doubled for n times and $\|u_i\| \le \sqrt{n}m_i A$, we get $\|u_i\| \le n \cdot (2A)^n$. This shows that the size of any entry in u_k is at most $O(n \log A)$ bits at any time of the algorithm.

We now need to bound the numerator and denominator of $\mu_{k\ell}$ and the rational entries in u_k^*. We claim that, $d_{k-1}u_k^* \in \mathbb{Z}^m$ for every k. This can be argued as follows. It is easy to see that, every u_k^* can be expressed as $u_k^* = u_k - \sum_{\ell<k} \lambda_{k\ell} u_\ell$ for some $\lambda_{k\ell} \in \mathbb{Q}$. Since $u_j \cdot u_k^* = 0$ for any $j < k$, we get $\sum_{\ell<k} \lambda_{k\ell}(u_j \cdot u_\ell) = (u_j \cdot u_k)$ for every $1 \le j < k$. This gives a system of $k-1$ linear equations in the $\lambda_{k\ell}, 1 \le \ell < k$. The determinant of the coefficient matrix $(u_j \cdot u_\ell)_{1 \le j,\ell \le k}$ is exactly d_{k-1}. Hence, by Cramer's rule, $d_{k-1}\lambda_{k\ell} \in \mathbb{Z}$. Therefore, $d_{k-1}u_k^* = d_{k-1}u_k - \sum_{\ell<k} d_{k-1}\lambda_{k\ell}u_\ell \in \mathbb{Z}^m$. This means, the denominator of every entry in u_k^* is at most $d_{k-1} \le A^{2n}$ and the numerator is surely bounded by $d_{k-1}\|u_k^*\| \le A^{2(n-1)} \cdot A \le A^{2n}$.

Our last claim is that, $d_\ell \cdot \mu_{k\ell} \in \mathbb{Z}$. This is because, $d_\ell \mu_{k\ell} = d_\ell \cdot \frac{u_k \cdot u_\ell^*}{\|u_\ell^*\|^2} = u_k \cdot (d_{\ell-1}u_\ell^*)$. We have already shown that $d_{\ell-1}u_\ell^* \in \mathbb{Z}^m$ and so $d_\ell \mu_{k\ell} \in \mathbb{Z}$. Therefore, the denominator of $\mu_{k\ell}$ can be at most $d_\ell \le A^{2n}$ and the numerator is bounded by $d_\ell|\mu_{k\ell}|$. From equation (1),

$$d_\ell \cdot |\mu_{k\ell}| \le d_\ell \cdot \sqrt{d_{\ell-1}} \cdot \|u_k\| \le A^{2(n-1)} \cdot A^{n-2} \cdot n(2A)^n \le n \cdot (2A^4)^n.$$

Thus, the size of the numerator and denominator of any rational number involved in the computation is at most $O(n \log A)$ bits. This completes our proof. ■

The main result of this section is summarized in the following theorem.

Theorem 2 (Lenstra-Lenstra-Lovász, 1982) *Given n vectors $v_1, \ldots, v_n \in \mathbb{Z}^m$ that are linearly independent over $\mathbb{Q}$ and $\|v_i\| \le A \in \mathbb{Z}^+$ for every i, a vector $v \in \mathbb{Z}^m$ can be computed in time $\mathsf{poly}(m, \log A)$ such that $\|v\|$ is at most $2^{\frac{n-1}{2}}$ times the length of a shortest vector in the lattice $\mathcal{L} = \sum_{i=1}^n \mathbb{Z}v_i$.*

We will now see an application of this theorem in factoring polynomials over rationals.

2 Factoring polynomials over rationals

Much of this polynomial factoring algorithm, due to Lenstra, Lenstra and Lovász [LJL82], resembles the bivariate factoring algorithm discussed in lectures 12 and 13, but they differ at one crucial step. This will be clear from the following discussion.

Given a polynomial $f \in \mathbb{Q}[x]$, we can multiply f with the *lcm* of the denominators of its coefficients to get a polynomial in $\mathbb{Z}[x]$. So, without loss of generality assume that $f \in \mathbb{Z}[x]$. Before we get down to the

details of factoring f, we need to address one basic question. How large can a coefficient of a factor of f be? If a factor of f has very large coefficients, it might not be possible to even output the factor in polynomial time. Fortunately, this is not the case.

With every polynomial $f = \sum_{i=1}^{n} c_n x^i \in \mathbb{Z}[x]$ associate a coefficient vector $v_f = (c_n, \ldots, c_0) \in \mathbb{Z}^{n+1}$. Norm of f, denoted by $\|f\|$, is defined as $\|f\| = \|v_f\|$. Let $\|f\| \leq A \in \mathbb{Z}^+$ and α be a root of f in $\mathbb{C}$. Since, $f(\alpha) = 0$, $|c_n\alpha^n| = |\sum_{i=0}^{n-1} c_i\alpha^i| \leq \sum_{i=0}^{n-1} |c_i||\alpha|^i$. If $|\alpha| > 1$ then $|c_n\alpha^n| \leq nA|\alpha|^{n-1}$, implying that $|\alpha| \leq nA$. Any factor $f_1 \in \mathbb{Z}[x]$ of f is a product of at most $n-1$ linear factors over $\mathbb{C}$ and an integer with absolute value at most $|c_n| \leq A$. Therefore, absolute value of any coefficient of f_1 is bounded by $A \cdot \binom{n}{n/2} \cdot (nA)^{n-1} \leq (2nA)^n$ and hence $\|f_1\| \leq \sqrt{n}(2nA)^n$. Although, this bound is sufficient for our purpose, a sharper bound on $\|f_1\|$ is provided by Mignotte [Mig74] (also known as the Landau-Mignotte bound).

Lemma 3 (Landau-Mignotte bound) *If $f_1 \in \mathbb{Z}[x]$ is a proper factor of a polynomial $f \in \mathbb{Z}[x]$ of degree n, then $\|f_1\| \leq 2^{n-1}\|f\|$.*

Therefore, $\|f_1\| \leq 2^{n-1}A$ and size of any coefficient of f_1 is at most $O(n + \log A)$ bits. We refer to this bound on $\|f_1\|$ as B.

For simplicity, assume that f is monic and squarefree. Since the absolute value of any entry in the Sylvester matrix $S(f, \frac{df}{dx})$ is at most nA, by Hadamard's inequality $|\mathsf{Res}_x(f, \frac{df}{dx})| \leq (2n)^n(nA)^{2n} = (2n^3A^2)^n$. Searching the first $O(n \log(nA))$ primes we can find a prime p such that $f \bmod p$ is monic and squarefree. Suppose $f = gh \bmod p$, where g is monic and irreducible in $\mathbb{F}_p[x]$. We can lift this factorization using Hensel lifting for ℓ steps to obtain $f = g'h' \bmod p^{2^\ell}$, where $g' = g \bmod p$ and $h' = h \bmod p$. Arguing along the same line as in the bivariate factoring case, we can show that there is an irreducible factor $f_1 \in \mathbb{Z}[x]$ of f such that g' divides f_1 modulo p^{2^ℓ}. In bivariate factoring we could find f_1 by solving a system of linear equations over the underlying field. But this approach does not apply here straightaway.

Suppose $f_1' \in \mathbb{Z}[x]\backslash\{0\}$ with $\deg(f_1') < n$ and $\|f_1'\| = C \in \mathbb{Z}$, such that $f_1' = g'h_1'' \bmod p^{2^\ell}$ for some $h_1'' \in \mathbb{Z}[x]$. As argued in the proof of the Hadamard inequality (see Lecture 3), the absolute value of the determinant of an integer matrix with row vectors $r_1, \ldots, r_k$ is bounded by $\prod_{i=1}^{k} \|r_i\|$. Therefore, $|\mathsf{Res}_x(f, f_1')| \leq A^{n-1}C^n$. Surely, there exists s and t in $\mathbb{Z}[x]$ such that $sf + tf_1' = \mathsf{Res}_x(f, f_1')$. Since f and f_1' share a common factor g' modulo p^{2^ℓ}, if $|\mathsf{Res}_x(f, f_1')| < p^{2^\ell}$ then $\gcd(f, f_1')$ is nontrivial. Therefore, all we want is $p^{2^\ell} > A^{n-1}C^n$. This means, $\|f_1'\| = C$ has to be small in order to ensure that ℓ is small. This is the reason why just solving a f_1' with $f_1' = g'h_1'' \bmod p^{2^\ell}$ does not help. We also need to ensure that $\|f_1'\|$ is small. Putting the conditions together, we want a nonzero $f_1' \in \mathbb{Z}[x]$ of degree less than n, such that $\|f_1'\|$ is 'small' and $f_1' = g'h_1'' \bmod p^{2^\ell}$ for some $h_1'' \in \mathbb{Z}[x]$. It is this step which is solved by finding a short vector in a lattice.

Let $\deg(g') = k < n$. Consider the polynomials $x^{n-k-1}g', x^{n-k-2}g', \ldots, xg', g'$ and the polynomials $p^{2^\ell}x^{k-1}, p^{2^\ell}x^{k-2}, \ldots, p^{2^\ell}$. Let $v_1, \ldots, v_n \in \mathbb{Z}^n$ be the n coefficient vectors corresponding to these n polynomials. It is not hard to see that the coefficient vector $v_{f_1'}$ of any solution f_1', satisfying $f_1' = g'h_1'' \bmod p^{2^\ell}$ for some $h_1'' \in \mathbb{Z}[x]$, belongs to the lattice $\mathcal{L} = \sum_{i=1}^{n} \mathbb{Z}v_i$. Also, the vectors $v_1, \ldots, v_n$ are linearly independent. Since f_1 is a solution for f_1', $v_{f_1} \in \mathcal{L}$ and hence the length of the shortest vector in $\mathcal{L}$ is at most $\|v_{f_1}\| = \|f_1\| \leq B = 2^{n-1}A$. Applying Theorem 2, we can find a short vector v such that $\|v\| \leq 2^{\frac{3(n-1)}{2}} \cdot A$. Let f_1' be the polynomial corresponding to the coefficient vector v. Surely, $f_1' = g'h_1'' \bmod p^{2^\ell}$ for some $h_1'' \in \mathbb{Z}[x]$, $\deg(f_1') < n$ and $\|f_1'\| = \|v\| = C \leq 2^{\frac{3(n-1)}{2}} \cdot A$. Since, $|\mathsf{Res}_x(f, f_1')| \leq A^{n-1}C^n \leq 2^{\frac{3n(n-1)}{2}} \cdot A^{2n-1}$, we only need to choose an ℓ such that $p^{2^\ell} > 2^{\frac{3n(n-1)}{2}} \cdot A^{2n-1}$. Therefore, all the integers involved in Hensel lifting have size at most $O(n^2 + n \log A)$ bits, implying that the lifting takes only polynomial time.

The following algorithm summarizes the above discussion. It assumes that $f \in \mathbb{Z}[x]$ is a monic, squarefree polynomial of degree n, and $\|f\| \leq A$.

Algorithm 2 Polynomial factoring over rationals

1. Find a prime p such that $f \mod p$ is squarefree.
2. Let $D = 2^{\frac{3(n-1)}{2}} \cdot A$ and $E = 2^{\frac{3n(n-1)}{2}} \cdot A^{2n-1}$.
3. Factor $f = gh \mod p$ where g is monic and irreducible in $\mathbb{F}_p[x]$.
4. Let $\deg(g) = k$. If $k = n$, declare 'f is irreducible'.
5. Use Hensel lifting to compute $f = g'h' \mod p^{2^\ell}$ such that $E^2 \geq p^{2^\ell} > E$.
6. Let $v_1, \ldots, v_n$ be the coefficient vectors in $\mathbb{Z}^n$ corresponding to the polynomials $x^{n-k-1}g', x^{n-k-2}g', \ldots, xg', g'$ and $p^{2^\ell}x^{k-1}, p^{2^\ell}x^{k-2}, \ldots, p^{2^\ell}$.
7. Use LLL algorithm to find a short vector v in the lattice $\mathcal{L} = \sum_{i=1}^{n} \mathbb{Z}v_i$.
8. If $\|v\| > D$ declare 'f is irreducible'.
9. Else let f_1' be the polynomial with coefficient vector v.
10. Return $\gcd(f, f_1')$.

Correctness and time complexity - Correctness of the algorithm follows immediately from the prior discussion. In step 1, the value of prime p is $O(n \log(nA))$ and so using Algorithm 1 and 2 from Lecture 8, we can factor f over $\mathbb{F}_p$ (in step 3) in polynomial time. Since $p^{2^\ell} \leq E^2$, Hensel lifting in step 5 also takes polynomial time. Norm of g', $\|g'\| \leq \sqrt{n}E^2$. So the LLL algorithm in step 7 takes time $\mathsf{poly}(n, \log E) = \mathsf{poly}(n, \log A)$. Lastly, in step 10, gcd of two integer polynomials can be computed in polynomial time (using Chinese Remaindering). Therefore, the algorithm has an overall polynomial running time.

Exercises:

1. How can we remove the assumption, $f \in \mathbb{Z}[x]$ is monic and sqaure-free, from Algorithm 2? (Refer to Chapter 15 & 16 in von zur Gathen-Gerhard's book.)

References

[LJL82] Arjen K. Lenstra, Hendrik W. Lenstra Jr., and László Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, 1982.

[Mig74] Maurice Mignotte. An Inequality About Factors of Polynomials. *Mathematics of Computation*, 28(128):1153–1157, 1974.