M4. Lecture 3.
THE LLL ALGORITHM AND COPPERSMITH’S METHOD

Ha Tran, Dung H. Duong, Khuong A. Nguyen.

SEAMS summer school 2015
HCM University of Science
The LLL algorithm

History
Applications of LLL
Euclid’s Algorithm
The LLL algorithm: rank 2 lattices
The LLL basis
The LLL Algorithm

Analyzing the Running Time
Why LLL is polynomial?

Coppersmith’s Method
Review

We have studied:

- Lattices and basic definitions: basis, determinant, ...
- Problems on lattices: SVP, CVP, ...
- Existence of a shortest vector.
- An upper bound for the shortest vector (Minkowski’s theorem).
We have studied:

• Lattices and basic definitions: basis, determinant, ...
• Problems on lattices: SVP, CVP, ...
• Existence of a shortest vector.
• An upper bound for the shortest vector (Minkowski’s theorem).

Today: How to find the shortest vector of a given lattice?
Review

We have studied:

- Lattices and basic definitions: basis, determinant, ...
- Problems on lattices: SVP, CVP, ...
- Existence of a shortest vector.
- An upper bound for the shortest vector (Minkowski’s theorem).

Today: How to find the shortest vector of a given lattice?
LLL algorithm.
What is LLL?

László Lovász

Arjen Lenstra

Hendrik Lenstra
The LLL algorithm

A popular algorithm presented in a legendary article published in 1982


Factoring Polynomials with Rational Coefficients

A. K. Lenstra¹, H. W. Lenstra, Jr.², and L. Lovász³

1 Mathematisch Centrum, Kruislaan 413, NL-1098 SJ Amsterdam, The Netherlands
2 Mathematisch Instituut, Universiteit van Amsterdam, Roetersstraat 15, NL-1018 WB Amsterdam, The Netherlands
3 Bolyai Institute, A. József University, Aradi vėrtanúk tere 1, H-6720 Szeged, Hungary

In this paper we present a polynomial-time algorithm to solve the following problem: given a non-zero polynomial \( f \in \mathbb{Q}[X] \) in one variable with rational coefficients, find the decomposition of \( f \) into irreducible factors in \( \mathbb{Q}[X] \). It is well
How popular LLL?

A popular algorithm presented in a legendary article published in 1982

- The LLL article has been cited x1000 times.
- The LLL algorithm and/or variants are implemented in: Maple, Mathematica, GP/Pari, Magma, Number Theory Library (NTL), SAGE, etc.
- A conference was organized in 2007 to celebrate the 25th anniversary of the LLL article.
What is LLL about?

A popular algorithm presented in a legendary article published in 1982

- It is an efficient algorithm.
- It’s about finding short lattice vectors.
- It’s about finding a short basis of a lattice.
Applications of LLL: Examples

• The two-square theorem: If $p$ is a prime $\equiv 1 \pmod{4}$, then $p$ is a sum of two squares $p = x^2 + y^2$.

• This formula for $\pi$ was found in 1995 using a variant of LLL:

$$\pi = \sum_{k=0}^{\infty} \frac{1}{16^k} \left( \frac{4}{8k+1} - \frac{2}{8k+4} - \frac{1}{8k+5} - \frac{1}{8k+6} \right).$$

• Elkies used LLL in the 2000s to find:

$$5853886516781223^3 - 447884928428402042307918^2 = 1641843.$$
Applications of LLL: Examples

- Factoring polynomials over the integers or the rational number.
- Solve approximation to the SVP, as well as other lattice problems.
- Coppersmith’s Method.
- Odlyzko and te Riele used LLL in 1985 to disprove the Mertens conjecture.
- Breaking the Merkle-Hellman cryptosystem.
- Since 1982, dozens of public-key cryptosystems have been broken using LLL.
- ...
LLL is a vectorial analogue of Euclid’s algorithm to compute gcds.

- Instead of dealing with integers, it deals with vectors of integer coordinates.
- It performs similar operations, and is essentially as efficient.
Euclid’s Algorithm

- **Input**: two integers $a \geq b \geq 0$.
- **Output**: $\gcd(a, b)$.
- **While** $(b \neq 0)$
  - $a := a \mod b$
- **Swap**($a$, $b$)
- **Output**($a$)
A Vectorial Euclid’s Algorithm?

• Since $a\mathbb{Z} + b\mathbb{Z} = \gcd(a, b)\mathbb{Z}$, Euclid computes the shortest non-zero linear combination of $a$ and $b$.

• Given a finite set $B$ of vectors in $\mathbb{Z}^n$, can one compute the shortest non-zero vector in the set $L(B)$ of all linear combinations?
Reduction operations

To improve a basis, we may:

- Swap two vectors.
- Slide: subtract to a vector a linear combination of the others.

That's exactly what Euclid's algorithm does.
Lagrange’s reduction: rank 2 lattices

Let \( \mathcal{L} \) be a lattice with a basis \( B = \{ b_1, b_2 \} \). Assume that \( \| b_1 \| \leq \| b_2 \| \).

- Repeat
  - \( b_2 := b_2 - \text{round}(\mu_{2,1})b_1 \)
  - where
    \[
    \mu_{2,1} = \frac{\langle b_1, b_2 \rangle}{\| b_1 \|^2}
    \]
  - and \( \text{round}(\mu_{2,1}) \) is the closest integer to \( \mu_{2,1} \).
  - Swap\((b_1, b_2)\).
- Until \( \| b_1 \| \leq \| b_2 \| \).
- Output \( b_1, b_2 \).
Lagrange’s reduction: rank 2 lattices

Let $\mathcal{L}$ be a lattice with a basis $B = \{b_1, b_2\}$. Assume that $\|b_1\| \leq \|b_2\|$. 
Lagrange’s reduction: rank 2 lattices

**Exercise:** Show that if a basis $b_1, b_2$ of $\mathcal{L}$ is Lagrange-reduced then: $\|b_1\| = \lambda_1(\mathcal{L})$-length of the shortest vector of $\mathcal{L}$.
Lattices of rank 2: Lagrange-reduced

A basis $B = \{b_1, b_2\}$ is called Lagrange-reduced if

- $\|b_1\| \leq \|b_2\|$.
- $\left| \frac{\langle b_2, b_1 \rangle}{\|b_1\|^2} \right| \leq \frac{1}{2}$.

Such bases exist since Lagrange’s algorithm clearly outputs reduced bases.
LLL basis: general

Let $1/4 < \delta < 1$. A basis $b_1, \ldots, b_n$ is $\delta$-LLL reduced if and only if

- (Size-reduced) for $j < i$, $|\mu_{i,j}| \leq \frac{1}{2}$.
- (Lovász’s conditions)
  \[ \delta \|b_{i-1}^*\|^2 \leq \|b_i^* + \mu_{i,i-1}b_{i-1}^*\|^2. \]

where $\mu_{i,j} = \frac{\langle b_i, b_j^* \rangle}{\|b_j^*\|^2}$, $i = 2, \ldots, n$ and $j < i$. 

LLL basis: general

Let $1/4 < \delta < 1$. A basis $b_1, \ldots, b_n$ is $\delta$-LLL reduced if and only if

- (Size-reduced) for $j < i$, $|\mu_{i,j}| \leq \frac{1}{2}$.
- (Lovász’s conditions)
  \[ \delta \| b_{i-1}^* \|^2 \leq \| b_i^* + \mu_{i,i-1} b_{i-1}^* \|^2. \]

where $\mu_{i,j} = \frac{\langle b_i, b_j^* \rangle}{\| b_j^* \|^2}$, $i = 2, \ldots, n$ and $j < i$.

Note:

- \[ \| b_{i-1}^* \|^2 \leq \frac{1}{\delta - \mu_{i,i-1}^2} \| b_i^* \|^2. \]

- Usually in practice, we take $\delta = 3/4$. 
LLL basis: general

Exercise:

• Show that if a basis $b_1, b_2$ of $\mathcal{L}$ is Lagrange-reduced then it is LLL reduced for all $\frac{1}{4} < \delta < 1$.

• Let $\delta = \frac{3}{4}$ and let $\mathcal{L}$ be a lattice with an LLL reduced basis $B = \{b_1, \ldots, b_n\} \subset \mathbb{Z}^n$. Then:

$$\|b_1\| \leq 2^{(n-1)/4} \text{vol}(\mathcal{L})^{1/n} \quad \text{and} \quad \|b_1\| \leq 2^{(n-1)/2} \lambda_1(\mathcal{L}).$$
Description of the LLL Algorithm

While the basis is not LLL-reduced

1. Size-reduce the basis.
2. If Lovász’s condition does not hold for some pair \((i, i + 1)\): just swap \(b_i\) and \(b_{i+1}\).
Size-reduction

- For $i = 2$ to $d$,
  - For $j = i - 1$ down to 1
    - Size-reduce $b_i$ with respect to $b_j$: make $|\mu_{i,j}| \leq \frac{1}{2}$ by
      \[
      b_i := b_i - \text{round}(\mu_{i,j})b_j.
      \]
    - Update all $\mu_{i,j'}$ for $j' \leq j$.
  - The translation does not affect the previous $\mu_{i',j'}$ where $i' < i$ or $i' = i$ and $j' > j$. 
LLL algorithm: general

Exercise: Let $\mathcal{L} = \mathcal{L}(B)$ with $B = \{ b_1 = (10, -2, 3), b_2 = (-3, 2, 9), b_3 = (0, 2, -7) \}$.  

- Find an LLL-reduced basis of $\mathcal{L}$.  
- Find a shortest vector of $\mathcal{L}$.  
  (Hint: Use the last Corollary.)
Why LLL is polynomial?

Our analysis consists of two steps.

- Bound the number of iterations.
- Bound the running time of a single iteration.
Why LLL is polynomial?

Our analysis consists of two steps.

- Bound the number of iterations.
- Bound the running time of a single iteration.

Let $M := \max\{n, \log(\max_i \|b_i\|)\}$. Then the overall running time of the algorithm is polynomial in $M$.

**Lemma 1**

The number of iterations is polynomial in $M$.

**Lemma 2**

The running time of each iteration is polynomial in $M$. 
Why LLL is polynomial?

Lemma 1
The number of iterations is polynomial in $M$. 
Why LLL is polynomial?

Lemma 1
The number of iterations is polynomial in $M$.

proof:

• Consider the quantity

$$P = \prod_{i=1}^{n} \|b_i^*\| \cdot \cdots \cdot \|b_i^*\| = \prod_{i=1}^{n} D_i = \prod_{i=1}^{n} \|b_i^*\|^{n-i+1}$$

where $D_i$ is the volume of the lattice $\mathcal{L}(b_1, \cdots, b_i)$.

• If the $b_i$’s have integral coordinates, then $P$ is a positive integer.
  • Size-reduction does not modify $P$.
  • But each swap of LLL makes $P$ decrease by a factor $\leq \sqrt{\delta}$.

• This implies that the number of swaps is polynomially bounded.
Why LLL is polynomial?

Lemma 2
The running time of each iteration is polynomial in $M$. 
Why LLL is polynomial?

Lemma 2
The running time of each iteration is polynomial in $M$.

proof:
• Each iteration we perform only a polynomial number of arithmetic operations.
• The numbers that arise in each iteration can be represented using a polynomial number of bits. (Exercise :))
Enumerating short vectors

Proposition

Let $\mathcal{L}$ be a lattice with an LLL reduced basis $b_1, \ldots, b_n$. Let $m_1, \ldots, m_n \in \mathbb{R}$, and put $x = \sum_{i=1}^{n} m_i b_i$. Then one has

$$|m_i| \leq 2^{(n-1)/2} \left( \frac{3}{2} \right)^{n-i} \frac{\|x\|}{\|b_1\|} \text{ for all } i.$$
Corollary

Let $\mathcal{L}$ be a lattice with an LLL reduced basis $b_1, \ldots, b_n$. If $x = \sum_{i=1}^{n} m_i b_i$ is a shortest vector of $\mathcal{L}$ then

$$|m_i| \leq 2^{(n-1)/2} \left( \frac{3}{2} \right)^{n-i}$$

for all $i$. 
Coppersmith’s Method

Proposed by Don Coppersmith.
Coppersmith’s Method

Proposed by Don Coppersmith.

- One nice application of LLL.
- Is a method to find small integer roots of polynomial equations.
- This technique has been a very powerful tool in cryptanalysis.
Coppersmith’s Method

Theorem 2.1
There is an efficient algorithm that, given any monic, degree $d$ polynomial $f(x) \in \mathbb{Z}[x]$ and an integer $N$, outputs all integers $x_0$ such that $|x_0| \leq B = N^{1/d}$ and $f(x_0) = 0 \mod N$. 
Recap: LLL algorithm

Let $\mathcal{L}$ be a lattice with a basis $B$. Then LLL finds in polynomial time a basis whose first vector satisfies:

$$\|b_1\| \leq 2^{(n-1)/4} \text{vol}(L)^{1/n} \quad \text{and} \quad \|b_1\| \leq 2^{(n-1)/2} \lambda_1(L).$$
Recap: LLL algorithm

Let $\mathcal{L}$ be a lattice with a basis $B$. Then LLL finds in polynomial time a basis whose first vector satisfies:

$$\|b_1\| \leq 2^{(n-1)/4} \text{vol}(L)^{(1/n)} \text{ and } \|b_1\| \leq 2^{(n-1)/2} \lambda_1(L).$$

**Remark**

- The constant 2 can be replaced by $4/3 + \varepsilon$ and the running time becomes polynomial in $1/\varepsilon$.
- It performs “much better” than what the worst-case bounds suggest, especially in low dimension.
- LLL performs better in practice than predicted by theory, but the approximation factors remain exponential on the average and in the worst-case, except with smaller constants.
References

Eva Bayer-Fluckiger.
Lattices and number fields.

Hendrik W. Lenstra, Jr.
Lattices.

Chris Peikert.
LLL, Coppersmith.
http://www.cc.gatech.edu/~cpeikert/lic13/lec03.pdf.

Daniele Micciancio.
Basis reduction.
Lecture note of the course *CSE206A: Lattices Algorithms and Applications* (Spring 2014).

Oded Regev.
The LLL algorithm.
Lecture note of the course *Lattices in Computer Science* (Fall 2009).

Phong Nguyen.
The LLL algorithm.