Seminar on lattices

Complexity of the LLL algorithm

Fatima-Ezzahra EL ORCHE

June 2019

Outline

1. Recall
2. Main theorem
3. Idea of the proof of the complexity of LLL algorithm
4. Number of iterations of LLL
5. Running time of a single iteration
6. Conclusion
7. Closing remark
8. References

Recall

- Recap Gram-Schmidt Orthogonalization:
  - Definition: Given *n* linearly independent vectors $b_{1,\dots}b_n \in \mathbb{R}^n$, the Gram-Schmidt Orthogonalization of $b_{1,\dots}b_n$ is defined by $\tilde{b}_i = b_i - \sum_{j=1}^{i-1} \mu_{i,j}\tilde{b}_j$, where $\mu_{i,j} = \frac{\langle b_i | \tilde{b}_j \rangle}{\langle \tilde{b}_j | \tilde{b}_j \rangle}$.
- Definition (Reduced basis): A basis $B = \{b_1, \dots, b_n\} \in \mathbb{R}^n$ is a $\delta - LLL\ Reduced\ Basis$ if the following holds:
  1. $\forall 1 \le i \le n, j < i, |\mu_{i,j}| \le \frac{1}{2}$
  2. $\forall 1 \le i \le n, \delta \|\tilde{b}_i\|^2 \le \|\mu_{i+1,i}\tilde{b}_i + \tilde{b}_{i+1}\|^2$ **Lovazs condition**

We will take $\frac{1}{4} \le \delta \le 1$ in the rest of this presentation.

LLL algorithm

Algorithm 25 LLL algorithm with Euclidean norm (typically, choose $\delta = 3/4$)

INPUT: $\underline{b}_1, \ldots, \underline{b}_n \in \mathbb{Z}^m$.
OUTPUT: LLL reduced basis $\underline{b}_1, \ldots, \underline{b}_n$

Compute the Gram-Schmidt basis $\underline{b}_1^*, \ldots, \underline{b}_n^*$ and coefficients $\mu_{i,j}$ for $1 \leq j < i \leq n$
Compute $B_i = \langle \underline{b}_i^*, \underline{b}_i^* \rangle = \|\underline{b}_i^*\|^2$ for $1 \leq i \leq n$
$k = 2$
while $k \leq n$ **do**
    for $j = (k-1)$ downto 1 **do** ▷ Perform size reduction
        Let $q_j = \lfloor \mu_{k,j} \rceil$ and set $\underline{b}_k = \underline{b}_k - q_j \underline{b}_j$
        Update the values $\mu_{k,j}$ for $1 \leq j < k$
    end for
    if $B_k \geq (\delta - \mu_{k,k-1}^2) B_{k-1}$ **then** ▷ Check Lovász condition
        $k = k + 1$
    else
        Swap $\underline{b}_k$ with $\underline{b}_{k-1}$
        Update the values $\underline{b}_k^*$, $\underline{b}_{k-1}^*$, B_k, B_{k-1}, $\mu_{k-1,j}$ and $\mu_{k,j}$ for $1 \leq j < k$, and $\mu_{i,k}, \mu_{i,k-1}$ for $k < i \leq n$
        $k = \max\{2, k-1\}$
    end if
end while

Main theorem

The running time of LLL is polynomial in the input size.

Analyzing the running time of LLL

In order to analyze the running time of LLL, we proceed in two steps:

1. Bounding the number of iterations.
2. Bounding the running time of a single iteration.

Some notation used in this presentation:

- $X := \max_i ||b_i||$.
- $M := \max(n,\ X)$, the input size of the LLL algorithm.
- $k :=$ number of iterations of LLL.

Potential of a lattice basis

Definition: Let $B = \{b_1, \dots, b_n\}$ be a lattice basis. *The potential of* B , denoted D_B is defined by:

$$D_B = \prod_{i=1}^{n} \|\tilde{b}_i\|^{n-i+1} = \prod_{i=1}^{n} \|\tilde{b}_1\| \, \|\tilde{b}_2\| \dots \|\tilde{b}_i\| = \prod_{i=1}^{n} D_{B,i}$$

Where $D_{B,i} \coloneqq \det \Lambda_i$ and $\Lambda_i = \mathcal{L}(b_1, \dots, b_i)$ is the lattice spanned by $b_1, \dots, b_i$.

Properties:

1. Since $||\tilde{b}_i|| \leq ||b_i|| \leq X$, we have: $D_B \leq X^{n(n+1)/2}$.
2. The logarithm of this value is polynomial in M.
3. During the reduction step of LLL, D_B does not change because the GS basis does not change.
4. During the swap step of LLL, the initial value of D_B changes and it decays quickly. More precisely, we have: $\frac{D'_{B,i}}{D_{B,i}} < \sqrt{\delta}$ where $D'_{B,i}$ is the new value of $D_{B,i}$ after the swap. Which means that: $\frac{D'_B}{D_B} < \sqrt{\delta}$.
5. D_B is a nonzero integer and in particular it is at least 1.

Properties of the potential of B: Demonstration

2, 3 and 5 are obvious.

Let's prove 1 and 4:

- We have $||\tilde{b}_i|| \leq ||b_i|| \leq X$ for all $1 \leq i \leq n$, then $\prod_{j=1}^{i} ||\tilde{b}_j|| \leq X^i$ and then: $D_B = \prod_{i=1}^{n} \prod_{j=1}^{i} ||\tilde{b}_j|| \leq \prod_{i=1}^{n} X^i = X^{n(n+1)/2}$
- Consider the swap step and let's suppose that b_i is swaped with b_{i+1}. We have then:

$$\frac{D'_{B,i}}{D_{B,i}} = \frac{\det \Lambda'_i}{\det \Lambda_i} = \frac{\det \mathcal{L}(b_1,\ldots,b_{i-1},b_{i+1})}{\det \mathcal{L}(b_1,\ldots,b_i)} = \frac{\left(\prod_{j=1}^{i-1} ||\tilde{b}_j||\right) ||\mu_{i+1,i}\tilde{b}_i + \tilde{b}_{i+1}||}{\prod_{j=1}^{i} ||\tilde{b}_j||} = \frac{||\mu_{i+1,i}\tilde{b}_i + \tilde{b}_{i+1}||}{||\tilde{b}_i||} < \sqrt{\delta}$$ where the last inequality holds because the Lovazs condition is not satisfied as we are in the swap step.

This means that we have also: $\frac{D'_B}{D_B} < \sqrt{\delta}$.

Bounding the number of iterations of LLL

Theorem: The number of iterations of LLL k is polynomial on M. More precisely, we have:

$$k < \frac{1}{\log \frac{1}{\sqrt{\delta}}} . \frac{n(n+1)}{2} . \log(X)$$

Proof of the theorem

The idea of this is to prove that the *potential* of a lattice base is a stricly decreasing sequence when LLL is running. We will deduce the number of iterations of LLL when this sequence attends its limit:

- The initial value of D_B corresponds to its value before any iteration. Let's call ${D^{(j)}}_B$ the value of D_B after the j^{th} iteration in the *while* loop in the algorithm and ${D^{(0)}}_B = D_B$.
- We have from the the property 4 of D_B, that: $\frac{{D^{(j)}}_B}{{D^{(j-1)}}_B} < \sqrt{\delta}$, and this is true $\forall\, j = 1, \dots, k$ with k is the number of iterations needed for the algorithm to terminate. This means that $\frac{{D^{(k)}}_B}{{D^{(0)}}_B} < \sqrt{\delta}^k$.
- We can decuce from property 5 of D_B that ${D^{(k)}}_{B,i} \geq 1$.
- Then, $\frac{1}{{D^{(0)}}_B} < \sqrt{\delta}^k$. Which implies that: ${D^{(0)}}_B > \frac{1}{\sqrt{\delta}^k} = (\frac{1}{\sqrt{\delta}})^k$.
- From property 1 of D_B, we conclude that: $\frac{1}{\sqrt{\delta}^k} = (\frac{1}{\sqrt{\delta}})^k < X^{n(n+1)/2}$, and this terminates the proof and means that :

$$k < \frac{1}{\log(\frac{1}{\sqrt{\delta}})} . \frac{n(n+1)}{2} . \log(X)$$

Bounding the running time of a single iteration

The idea of the proof is to show that all the numbers arising during an iteration can be represented in polynomial space in M. We use two claims:

Claim 1: The Gram-Schmidt vectors $\tilde{b}_i, \ldots, \tilde{b}_n$ can be computed in polynomial time in M. Moreover, for every $1 \leq i \leq n$, we have that ${D_{B,i}}^2 \tilde{b}_i \in \mathbb{Z}^n$ and that $||\tilde{b}_i|| \leq {D_B}^2$.

Claim2: All vectors b_i appearing during an iteration can be represented using poly(M) bits.

Some explanations before demonstrating the claims

- The two properties of Claim 1 imply that the GS vectors can be represented in space polynomial in M because the bound on the norm implies that its absolute value at each coordinate of $\tilde{b}_i$ is at most ${D_B}^2$ and thus requires at most $\mathcal{O}(logD_B)$ bits to be represented, and there are n^2 of them. We obtain that the GS vectors can be represented in poly(M) bits.

- In Claim 2 we show that the basis vectors b_i do not become too large. This is necessary since these basis vectors change during the reduction step (and in fact, it is possible for vectors to "become longer" by the reduction step). We first bound the length of each b_i after the i^{th} iteration of the outer loop is done (i.e., once vector b_i is reduced). We then bound the length of b_i during the i^{th} iteration of the outer loop. For this we use the observation that to vector b_i we only add vectors b_j for $j < i$; these vectors are already reduced and hence the first bound applies.

Demonstration of the claims

- **Claim1:**
  - $\boldsymbol{D_B{}^2\tilde{b}_i \in Z^n}$:
    - The idea is to write the GS vectors as elements of $span(b_1, \dots, b_{i-1})$ since $\tilde{b}_i - b_i$ are in this set: $\tilde{b}_i = b_i + \sum_{j=1}^{i-1} a_j b_j$
    - $(a_j)_{1\le j\le i-1}$ are such that $\tilde{b}_i$ is orthogonal to each $b_l : < \tilde{b}_i, b_l > = 0 \ \forall 1 \le l \le i-1$.
    - This implies that: $< b_i, b_l > + a_1 < b_1, b_l > + \cdots + a_{i-1} < b_{i-1}, b_l > = 0 \ \forall 1 \le l \le i-1$(because $\tilde{b}_i = b_i + \sum_{j=1}^{i-1} a_j b_j$). We then obtain a system of $i-1$ linear equations in $i-1$ variables which can be solved in polynomial time using Cramer's rule.
    - $a_j = \frac{\det(\ some\ integer\ matrix)}{\begin{pmatrix} <b_1,b_1> & \cdots & <b_{i-1},b_1> \\ \vdots & \ddots & \vdots \\ <b_1,b_{i-1}> & \cdots & <b_{i-1},b_{i-1}> \end{pmatrix}} = \frac{some\ integer}{\det({B^T}_{i-1}.B_{i-1})} = \frac{some\ integer}{(\det \Lambda_{i-1})^2}$
    - Hence $\tilde{b}_i = b_i + \sum_{j=1}^{i-1} a_j b_j$ for some rational numbers a_j s.t.: $a_j.(\det \Lambda_{i-1})^2 \in$ Z. This implies that ${D^2}_{B,i}\tilde{b}_i \in$ Z and in particular ${D^2}_B\tilde{b}_i$ are integer vectors.
  - $\boldsymbol{||\tilde{b}_i|| \le D_B{}^2}$:
    - We have by definition of the potentiel D_B: $D_{B,i} = (\prod_{j=1}^{i-1} ||\tilde{b}_j||).||\tilde{b}_i||$, and so $||\tilde{b}_i|| = \frac{D_{B,i}}{\prod_{j=1}^{i-1} ||\tilde{b}_j||} \le D_{B,i} \prod_{j=1}^{i-1} {D_{B,j}}^2 \le {D_B}^2$ where the first inequality follows since $||\tilde{b}_j|| \ge \frac{1}{{D_{B,j}}^2}$.

Demonstration of the claims

- **Claim2:**
  - After the reduction step, we have for each $1 \leq i \leq n$: $||b_i||^2 = ||\tilde{b}_i||^2 + \sum_{j=1}^{i-1}(\mu_{i,j}||\tilde{b}_j||)^2 \leq {D_B}^2 + \frac{n}{4}.{D_B}^2 \leq n{D_B}^4$, where this holds using the property $|\mu_{i,j}| \leq \frac{1}{2}$ and results of claim 1. Hence, after the reduction step the length of the b_i's is not too large.
  - During the reduction step, the norm of b_i can be reprensented in poly(M) size:
    - We have: $|c_{i,j}| \leq \left|\left|\frac{<b_i,\tilde{b}_j>}{<\tilde{b}_i,\tilde{b}_j>}\right|\right| \leq \frac{\|b_i\|\|\tilde{b}_j\|}{\|\tilde{b}_j\|^2} + 1 = \frac{\|b_i\|}{\|\tilde{b}_j\|} + 1 \leq \frac{\|b_i\|}{1/{D^2}_B} + 1 \leq 2{D^2}_B\|b_i\|.$
    - Hence: $\|b_i - c_{i,j}b_j\| \leq \|b_i\| + |c_{i,j}|\|b_j\| \leq (1 + 2{D^2}_B\|b_j\|)\|b_i\| \leq (1 + 2{D^2}_B\sqrt{n}{D^2}_B)\|b_i\| \leq (4nD_B)^4\|b_i\|.$

By Claim 1 and 2 we have that is possible to represent the numbers in a polynomial number of bits. This together with the fact that in each iteration we perform a polynomial number of arithmetic operations proves that the running time of each iteration is polynomial in M.

Conclusion

- In this presentation we showed that the total number of iterations of LLL is polynomial in the input size.
- Moreover, all the numbers arising when LLL algorithm is running are not too big, they can be represented in a poly(M) space and thus they are computed in poly(M) time .
- We conclude then that the overall running time of LLL is polynomial on the input size M.

Closing remark

A more careful analysis gives the following corollary:

Corollary: Let L be a lattice in Z^m with basis: $b_1, \ldots, b_n$ and let $Y \in \mathrm{Z}_{\geq 2}$ be such that $\|b_i\|^2 \leq Y$ for $1 \leq i \leq n$. Then the LLL algorithm requires $\mathcal{O}(n^3 m.log(Y))$ arithmetic operations on integers of size $\mathcal{O}(n.log(Y))$. Using naïve arithmetic gives running time $\mathcal{O}(n^5 m.log(Y)^3)$ bits operations.

References

- https://cims.nyu.edu/~regev/teaching/lattices_fall_2004/ln/lll.pdf
- http://tomlr.free.fr/Math%E9matiques/Math%20Complete/Number%20theory/A%20course%20in%20computational%20algebraic%20number%20theory%20-%20Cohen%20H..pdf
- https://www.math.auckland.ac.nz/~sgal018/crypto-book/ch17.pdf
- http://cseweb.ucsd.edu/classes/wi10/cse206a/lec3.pdf