Решетки, алгоритмы и современные проблемы криптографии. LLL-алгоритм

Шокуров А.В.

23 февраля 2024 г.

Задача аппроксимации с обязательством

При исследовании вычислительной сложности задач аппроксимации на решетках удобно их формулировать как *задачи с обязательствами*. Это понятие обобщает понятие задач распознавания и весьма удобно для исследования сложности соответствующих задач аппроксимации.

Задача с обязательством представляет пару непересекающихся языков (Π_{YES}, Π_{NO}), т.е. $\Pi_{YES}, \Pi_{NO} \subseteq \Sigma^*$ и $\Pi_{YES} \cap \Pi_{NO} = \emptyset$.

Определение

Алгорим решает задачу с обязательством (Π_{YES}, Π_{NO}), если для входа $I \in \Pi_{YES} \cup \Pi_{NO}$ он правильно определяет выполняется ли $I \in \Pi_{YES}$ или $I \in \Pi_{NO}$. Поведение алгоритма для $I \notin \Pi_{YES} \cup \Pi_{NO}$ (в том случае если не выполняется обязательство) не специфицируется, т.е. в этом случае ответ может быть каким угодным.

Задачи распознавания — частный случай задач с обязательством: для таких задач выполняется соотношение $\Pi_{NO} = \Sigma^* \setminus \Pi_{YES}$. Определим задачи с обязательствами ассоциированные с задачами SVP и CVP.

Задача аппроксимации с обязательством для SVP

Определение

Задача с обязательством $G_{AP}SVP_{\gamma}$, где γ — функция ранга решетки, определяется так

- *YES соответствует парам $(\boldsymbol{B}, r)$, где $\boldsymbol{B} \in \mathbb{Z}^{m \times n}$ задает базис решетки, а $r \in \mathbb{Q}$ такое число, что для некоторого $\boldsymbol{z} \in \mathbb{Z}^n \setminus \{\boldsymbol{0}\}$ выполняется неравенство $\|\boldsymbol{Bz}\| \leq r$.*
- *NO соответствует парам $(\boldsymbol{B}, r)$, где $\boldsymbol{B} \in \mathbb{Z}^{m \times n}$, а $r \in \mathbb{Q}$ такое число, что для некоторого $\boldsymbol{z} \in \mathbb{Z}^n \setminus \{\boldsymbol{0}\}$ выполняется неравенство $\|\boldsymbol{Bz}\| > \gamma r$.*

Задача аппроксимации с обязательством для CVP

Определение

Задача с обязательством $G_{AP}CVP_{\gamma}$, где γ — функция ранга решетки, определяется так

- *YES соответствует тройкам $(\boldsymbol{B}, \boldsymbol{t}, r)$, где $\boldsymbol{B} \in \mathbb{Z}^{m \times n}$ задает базис решетки, $\boldsymbol{t} \in \mathbb{Z}^n$, а $r \in \mathbb{Q}$ такое число, что для некоторого $\boldsymbol{z} \in \mathbb{Z}^n$ выполняется неравенство $\|\boldsymbol{Bz} - \boldsymbol{t}\| \leq r$.*
- *NO соответствует тройкам $(\boldsymbol{B}, \boldsymbol{t}, r)$, где $\boldsymbol{B} \in \mathbb{Z}^{m \times n}$, $\boldsymbol{t} \in \mathbb{Z}^n$, а $r \in \mathbb{Q}$ такое число, что для некоторого $\boldsymbol{z} \in \mathbb{Z}^n$ выполняется неравенство $\|\boldsymbol{Bz} - \boldsymbol{t}\| > \gamma r$.*

Задача аппроксимации с обязательством для CVP

Задача с обязательством $\mathrm{G_{AP}SVP}_\gamma$ и задача нахождения приближенного решения с множителем γ эквивалентны в следующем смысле. Пусть алгоритм $\mathcal{A}$ находит приближение для задачи SVP с множителем γ. Тогда можно решить задачу $\mathrm{G_{AP}SVP}_\gamma$ так. На входе алгоритма $(\mathbf{B}, r)$ алгоритм $\mathcal{A}$ находит аппроксимацию $r' = \|x\| \in [\lambda_1, \gamma\lambda_1]$ кратчайшего вектора. Если $r' > \gamma r$, тогда $\lambda_1 > r$, т.е. пара $(\mathbf{B}, r)$ в задаче $\mathrm{G_{AP}SVP}_\gamma$ не соответствует ответу YES, а поскольку $(\mathbf{B}, r) \in \Pi_{YES} \cup \Pi_{NO}$ на этой паре принимается значение NO. Если же выполняется неравенство $r' \leq \gamma r$, из условия выполнения обязательства $(\mathbf{B}, r) \in \Pi_{YES} \cup \Pi_{NO}$ заключаем, что на паре $(\mathbf{B}, r)$ принимается значение YES.

Задача аппроксимации с обязательством для CVP

Пусть теперь имеется оракул $\mathcal{A}$, решающий задачу $\mathrm{G_{AP}SVP}_\gamma$. Заметим, что при невыполнении обязательства, оракул может выдать любой ответ. Пусть $u \in \mathbb{Z}$ верхняя граница $\lambda(\mathbf{B})^2$, например, u квадрат длины базисного вектора. В этом случае всегда $\mathcal{A}(\mathbf{B}, \sqrt{u}) = \mathit{YES}$, а $\mathcal{A}(\mathbf{B}, 0) = \mathit{NO}$. Далее, используя бинарный поиск найдем такое целое $r \in \{0, \ \ldots, u\}$, что $\mathcal{A}(\mathbf{B}, \sqrt{r}) = \mathit{YES}$, а $\mathcal{A}(\mathbf{B}, \sqrt{r-1}) = \mathit{NO}$. Тогда $\lambda_1(\mathbf{B})$ лежит в полуинтервале $[\sqrt{r}, \gamma\sqrt{r})$. Аналогичное рассуждение применимо к задаче нахождения ближайшего вектора.

Задача аппроксимации с обязательством для CVP

Класс *NP* можно расширить на задачи с обязательствами.

Определение

Задача (Π_{YES}, Π_{NO}) *лежит в NP, если существует отношение* $R \subset \Sigma^* \times \Sigma^*$*, такое что* $(x, y) \in R$ *подтверждается за полиномиальное время от* $|x|$ *и для каждого* $x \in \Pi_{YES}$ *существует такой y, что* $(x, y) \in R$*, а для* $x \in \Pi_{NO}$ *нет такого y, что* $(x, y) \in R$.

Дополнением задачи распознавания (Π_{YES}, Π_{NO}) является задача $(\Pi_{NO}, \Pi_{YES},)$. Класс задач распознования языков, дополнение которых лежит в *NP* называется классом *coNP*. Соответственно этот класс расширяется на класс дополнений к задачам с обязательствами из класса *NP*.

Задача аппроксимации с обязательством для CVP

Определение

Сведением по Карпу задачи (Π_{YES}, Π_{NO}) *к* (Π'_{YES}, Π'_{NO}) *называется функция* $f : \Sigma^* \to \Sigma^*$*, преобразующая ответы YES в YES, а NO в NO.*

Если алгоритм A решает задачу (Π'_{YES}, Π'_{NO}), то этот же алгоритм может решить задачу (Π_{YES}, Π_{NO}). А именно, пусть $I \in (\Pi_{YES}, \Pi_{NO})$. Примениая к $f(I) \in (\Pi'_{YES}, \Pi'_{NO})$ алгоритм A, получим ответ для I. Заметим, что $f(I)$ — задача с обязательством, т.е. $f(I) \in (\Pi'_{YES}, \Pi'_{NO})$ и если для $f(I)$ ответ *YES* тогда и только тогда, когда для I ответ *YES*.
Задача с обязательством A называется *NP*-трудной, если любая задача с обязательством B из класса *NP* сводится эффективно к задаче A.

Ортогонализация Грамма-Шмидта

Пусть $\mathbf{b}_1, \ldots, \mathbf{b}_n$ — базис, тогда ортогональные векторы $\mathbf{b}_i^*$ определяются формулами

$$\begin{aligned} \mathbf{b}_i^* &= \mathbf{b}_i - \sum_{j=1}^{i-1} \mu_{i,j} \mathbf{b}_j^* \\ \mu_{i,j} &= \frac{(\mathbf{b}_i, \mathbf{b}_j^*)}{(\mathbf{b}_j^*, \mathbf{b}_j^*)}. \end{aligned}$$

Как выполняется алгоритм Гаусса? Какие действия выполняются?

1. Приведение базиса: $b_2 := b_2 - cb_1$, где $c = \lceil \frac{(\mathbf{b}_1, \mathbf{b}_2)}{(\mathbf{b}_1, \mathbf{b}_1)} \rceil$.
2. Перестановка: **if** $\|\mathbf{b}_1\| > \|\mathbf{b}_2\|$ **then swap** $(\mathbf{b}_1, \mathbf{b}_2) := (\mathbf{b}_2, \mathbf{b}_1)$.
3. **if** $(\mathbf{b}_1, \mathbf{b}_2)$ не приведенный базис **repeat**.

Приведенный базис

Определение

Базис $B = (b_1, \ldots, b_n) \in \mathbb{R}^{m \times n}$ *называется* δ*-LLL-приведенным относительно параметра* $\frac{1}{4} < \delta < 1$*, если*

1. $|\mu_{ij}| \leq 1/2$ *при* $i > j$*, где* μ_{ij} *— коэффициенты Грамма-Шмидта,*
2. *для любой последовательной пары векторов* b_i, b_{i+1} *выполняется неравенство*

$$\delta \|\pi_i(b_i)\|^2 \leq \|\pi_i(b_{i+1})\|^2,$$

где π_i *— проекция на оболочку* $\mathbf{span}(b_i^*, b_{i+1}^*, \ldots, b_n^*)$*. Иначе это условие задается соотношением*

$$\delta \|b_i^*\|^2 \leq \|b_{i+1}^* + \mu_{i+1,i} b_i^*\|^2 = \|b_{i+1}^*\|^2 + \mu_{i+1,i}^2 \|b_i^*\|^2.$$

Свойства приведенного базиса

Теорема

Пусть $b_1, \ldots, b_n$ *—* δ*-LLL-приведенный базис решетки* L*. Тогда*

1. $\det L \le \prod_{i=1}^{n} \|b_i\| \le \left(\frac{4}{4\delta-1}\right)^{n(n-1)/4} \det L$,
2. $\|b_j\| \le \left(\frac{4}{4\delta-1}\right)^{(i-1)/2} \|b_i^*\|$ *при* $1 \le j \le i \le n$,
3. $\|b_1\| \le \left(\frac{4}{4\delta-1}\right)^{(n-1)/4} (\det L)^{1/n}$,
4. *если* $x \ne 0$ *— элемент решетки, то* $\|b_1\| \le \left(\frac{4}{4\delta-1}\right)^{(n-1)/2} \|x\|$,
5. *если векторы решетки* $x_1, \ldots, x_t$ *— линейно независимы, то* $\|b_j\| \le \left(\frac{4}{4\delta-1}\right)^{(n-1)/2} \max\{\|x_1\|, \ldots, \|x_t\|\}$ *при* $1 \le j \le t$.

Доказательство. Согласно определению приведенного базиса выполняется неравенство

$$\|\mathbf{b}_{i+1}^*\|^2 \geq \left(\delta - \mu_{i+1,i}^2\right) \|\mathbf{b}_i^*\|^2 \geq \left(\delta - \frac{1}{4}\right) \|\mathbf{b}_i^*\|^2$$

для всех $1 \leq i < n$. Следовательно, для всех $1 \leq j \leq i < n$ выполняются неравенства $\|\mathbf{b}_j^*\|^2 \leq \left(\frac{4}{4\delta-1}\right)^{i-j} \|\mathbf{b}_i^*\|^2$. Введем обозначение $\tau = \frac{4}{4\delta-1}$. Тогда ввиду ограничения на параметр δ, получаем, что $\tau > \frac{4}{3}$.

Поэтому из условия приведенности базиса

$$\begin{aligned}
\|\mathbf{b}_{i+1}\|^2 &= \|\mathbf{b}^*_{i+1}\|^2 + \sum_{j=1}^{i} \mu_{i+1,j}\|\mathbf{b}^*_j\|^2 \\
&\le \|\mathbf{b}^*_{i+1}\|^2 + \sum_{j=1}^{i} \frac{1}{4} \cdot \left(\frac{4}{4\delta-1}\right)^{i+1-j} \|\mathbf{b}^*_{i+1}\|^2 \\
&= \|\mathbf{b}^*_{i+1}\|^2 + \sum_{j=1}^{i} \frac{1}{4} \cdot \tau^{i+1-j}\|\mathbf{b}^*_{i+1}\|^2 \\
&= \left(1 + \frac{1}{4}\tau\frac{\tau^i - 1}{\tau - 1}\right) \cdot \|\mathbf{b}^*_{i+1}\|^2 \\
&\le \tau^i \cdot \|\mathbf{b}^*_{i+1}\|^2,
\end{aligned}$$

т.к. $\left(1 + \frac{1}{4}\tau\frac{\tau^i - 1}{\tau - 1}\right) \le \tau^i$ при $\tau > \frac{4}{3}$.
Следовательно,

$$\|\mathbf{b}_j\|^2 \le \tau^{j-1}\|\mathbf{b}^*_j\|^2 \le \tau^{i-1}\|\mathbf{b}^*_i\|^2$$

при $1 \le j \le i \le n$. Формула 2 теоремы доказана.

Из полученного неравенства выводим

$$\prod_{i=1}^{n} \|\mathbf{b}_i\|^2 \leq \prod_{i=1}^{n} \tau^{i-1} \prod_{i=1}^{n} \|\mathbf{b}_i^*\| = \tau^{n(n-1)/2} \cdot (\det L)^2.$$

Поскольку первая половина неравенства 1 представляет собой, доказанное ранее неравенство Адамара, соотношение 1 доказано.

Для доказательства соотношения 3 воспользуемся соотношениями 2. Перемножая соотношения

$$\|\mathbf{b}_1\| \leq \tau^{\frac{i-1}{2}} \|\mathbf{b}_i^*\|,$$

получим

$$\|\mathbf{b}_1\|^n \leq \prod_{i=1}^{n} \left(\tau^{\frac{i-1}{2}} \|\mathbf{b}_i^*\| \right) = \tau^{\frac{n(n-1)}{4}} \cdot \det L.$$

Соотношение 3 доказано.

Докажем соотношение 4. Поскольку $\mathbf{x}$ — вектор решетки, имеют место разложения

$$\mathbf{x} = \sum_{i=1}^{n} r_i \mathbf{b}_i = \sum_{i=1}^{n} r'_i \mathbf{b}^*_i,$$

причем все $r_k \in \mathbb{Z}$. Пусть k — максимальное целое, для которого $r_k \neq 0$. Тогда из определения процесса ортогонализации следует, что выполняется равенство $r_k = r'_k$, и в частности $\|r_k\| = \|r'_k\| \geq 1$. Воспользовавшись соотношением 2, получаем

$$\tau^{n-1}\|\mathbf{x}\|^2 \geq \tau^{n-1}\|r'_k\|^2\|\mathbf{b}^*_k\|^2 \geq \tau^{k-1}\|\mathbf{b}^*_k\|^2 \geq \|\mathbf{b}_1\|^2.$$

Соотношение 4 доказано.

Докажем соотношение 5. Выразим векторы $\mathbf{x}_j$ через элементы базиса $\mathbf{b}_i$

$$\mathbf{x}_j = \sum_{i=1}^{n} r_{i,j}\mathbf{b}_i, \;\; r_{i,j} \in \mathbb{Z}.$$

Для каждого j обозначим через $k(j)$ наибольшее целое число k, для которого $r_{k,j} \neq 0$. Перенумеруем векторы $\mathbf{x}_j$ так, чтобы $k(1) \leq k(2) \leq \;\; \ldots k(t)$. Тогда для всех $1 \leq j \leq t$ выполняются неравенства $\|\mathbf{x}_j\|^2 \geq \|\mathbf{b}^*_{k(j)}\|^2$. Выполняется неравенство $j \leq k(j)$ для всех $1 \leq j \leq t$, поскольку векторы $\mathbf{x}_1, \;\; \ldots, \mathbf{x}_t$ линейно независимы. Воспользовавшись этим неравенством и уже доказанным соотношением 2, получаем

$$\|\mathbf{b}_j\|^2 \leq \tau^{k(j)-1} \cdot \|\mathbf{b}^*_{k(j)}\|^2 \leq \tau^{n-1} \cdot \|\mathbf{b}^*_{k(j)}\|^2 \leq \tau^{n-1} \cdot \|\mathbf{x}_j\|^2.$$

Свойства приведенного базиса

Следствие

Если $B = (b_1, \ldots, b_n) \in \mathbb{R}^{m \times n}$ *—* $\delta-$*LLL-приведенный базис с* $\delta \in (1/4, 1)$*, то* $\|b_1\| \leq \left(2/\sqrt{4\delta - 1}\right)^{n-1} \lambda_1$*. В частности, если* $\delta = 1/4 + (3/4)^{n/(n-1)}$*, то* $\|b_1\| \leq \left(2/\sqrt{3}\right)^n \lambda_1$.

LLL(Lenstra, Lenstra, Lovasz)-алгоритм

Вход: Базис решетки $\mathbf{B} = (b_1, \ldots, b_n) \in \mathbb{Z}^{m \times n}$

Выход: **B** — LLL-приведенный базис решетки.

(loop):

for $i = 1, \ldots, n$

for $j = i - 1, \ldots, 1$

$b_i := b_i - c_{i,j} b_j$ где $c_{i,j} = \left\lfloor (b_i, b_j^*)/(b_j^*, b_j^*) \right\rceil$

if $\delta \|\pi_i(b_i)\|^2 > \|\pi_i(b_{i+1})\|^2$ для некоторого i

then swap(b_i, b_{i+1}) **go to** (loop)

else B — выход

Stop

Корректность алгоритма

Отметим, что выполняемые в алгоритме преобразования переводят базис решетки в базис той же решетки, причем в процессе редукции (до перестановки векторов) соответствующий ортогональный базис $\mathbf{B}^*$ не изменяется. При попытке ортогонализации все промежуточные значения $|\mu_{i,j}| \leq 1/2$ при $i > j$ сщхраняются. В этом случае, если алгоритм заканчивает работу, то в результате получается приведенный базис решетки.

Поэтому, для доказательства полиномиальности алгоритма необходимо сначала доказать, что алгоритм заканчивает выполнение за конечное число шагов и оценить их число, а также сложность выполнения каждого шага.

Доказательство корректности алгоритма

Из определения процедуры ортогонализации Грамма-Шмидта следует

Лемма

Пусть $j < i$. При преобразовании базиса решетки

$$\varphi_{i,j} : B = (\boldsymbol{b}_1, \ \ldots, \boldsymbol{b}_n) \to B' = (\boldsymbol{b}'_1, \ \ldots, \boldsymbol{b}'_n),$$

по формуле

$$\varphi_{i,j}(\boldsymbol{b}_k) = \begin{cases} \boldsymbol{b}_k & \text{при} \quad k \neq i \\ \boldsymbol{b}_k - c_{k,j}\boldsymbol{b}_j & \text{при} \quad k = i \end{cases},$$

где $c_{k,j} = \left\lfloor (\boldsymbol{b}_k, \boldsymbol{b}^*_j)/(\boldsymbol{b}^*_j, \boldsymbol{b}^*_j) \right\rceil$, ортогональный базис Грамма-Шмидта нового базиса не изменяется.

Доказательство корректности алгоритма

Из определения коэффициентов $c_{k,j} = \lfloor (\mathbf{b}_k, \mathbf{b}_j^*)/(\mathbf{b}_j^*, \mathbf{b}_j^*) \rceil$

Лемма

Пусть $j < i$. При преобразовании базиса решетки

$$\varphi_{i,j} : B = (\boldsymbol{b}_1, \ \ldots, \boldsymbol{b}_n) \to B' = (\boldsymbol{b}'_1, \ \ldots, \boldsymbol{b}'_n),$$

заданном формулой

$$\varphi_{i,j}(\boldsymbol{b}_k) = \begin{cases} \boldsymbol{b}_k & \text{при} \quad k \neq i \\ \boldsymbol{b}_k - c_{k,j}\boldsymbol{b}_j & \text{при} \quad k = i \end{cases},$$

где $c_{k,j} = \lfloor (\boldsymbol{b}_k, \boldsymbol{b}_j^*)/(\boldsymbol{b}_j^*, \boldsymbol{b}_j^*) \rceil$, для всех $k > m$ выполняются соотношения

$$\mu'_{k,m} = \begin{cases} \mu_{k,m} & \text{при} \quad k \neq i \\ \mu_{k,m} & \text{при} \quad k = i > m > j \end{cases}$$

и $|\mu'_{i,j}| \leq 1/2$.

Доказательство корректности алгоритма

Отметим, что выполняемые в алгоритме преобразования переводят базис решетки в базис той же решетки, причем в силу доказанных выше двух лемм, если алгоритм заканчивает работу, то в результате получается δ-LLL приведенный базис решетки.
Поэтому, для доказательства корректности алгоритма достаточно убедиться, что алгоритм заканчивает выполнение за конечное число шагов. Мы докажем, что алгоритм останавливается не только за конечное время, даже его полиномиальность относительно длины входа. Для этого потребуется оценить число обращений к циклу (loop), а также сложность выполнения каждого шага.

Целозначные функции d_i

Зададим целозначные положительные функции d_i на базисах решетки формулами

$$d_i(b_1, \ \ldots, b_i) = \begin{vmatrix} (b_1, b_1) & \cdots & (b_1, b_i) \\ \cdots & \cdots & \cdots \\ (b_i, b_1) & \cdots & (b_i, b_i) \end{vmatrix}.$$

Согласно доказанному ранее выполняется равенство

$$d_i(b_1, \ \ldots, b_i) = \prod_{j=1}^{n} \|b_j^*\|^2.$$

Целозначная функция D

Свяжем теперь с базисом $b_1, \ldots, b_n$ натуральное число

$$D = D(b_1, \ldots, b_n) = \prod_{i=1}^{n} d_i(b_1, \ldots, b_i).$$

Заметим, что если в процессе выполнения алгоритма

не выполняется перестановка векторов, то величины d_i, являющиеся квадратами детерминантов базисов соответствующих решеток не изменяются. Следовательно, и величина D в этом случае остается неизменной.

Шаг алгоритма, переставляющий векторы

Рассмотрим теперь шаг алгоритма, на котором выполняется перестановка двух соседних элементов базиса. А именно, пусть векторы $b_1, \ldots, b_i$ определяют приведенный базис в решетке **span**$(b_1, \ldots, b_i)$, порожденной этими векторами. Пусть также векторы $b_1, \ldots, b_{i+1}$ представляют базис, для которого выполняется условие 1, но не выполняется условие 2 определения δLLL-приведенности. Тогда на этом шаге

алгоритма получаем базис

$$(\tilde{b}_1, \ldots, \tilde{b}_i, \tilde{b}_{i+1}, \ldots, \tilde{b}_n) = (b_1, \ldots, b_{i-1}, b_{i+1}, b_i, b_{i+1}, \ldots, b_n).$$

Изменение величины D

Посмотрим, как изменится при этом значение величины D. Отметим, что значения d_k при $k \neq i$ остаются неизменными, поскольку соответствующие решетки не меняются.

$$\frac{D(\tilde{b}_1, \ldots, \tilde{b}_n)}{D(b_1, \ldots, b_n)} = \frac{d_i(\tilde{b}_1, \ldots, \tilde{b}_i)}{d_i(b_1, \ldots, b_i)}$$

$$= \frac{\left(\prod_{j=1}^{i-1} \|b_j^*\|^2\right) \|\pi_i(b_{i+1})\|^2}{\prod_{j=1}^{i} \|b_j^*\|^2} = \frac{\|\pi_i(b_{i+1})\|^2}{\|b_i^*\|^2}.$$

Измение D

Поскольку выполняется перестановка, второе условие δ-LLL приводимости не выполняется, т.е. $\frac{\|\pi_i(b_{i+1})\|^2}{\|b_i^*\|^2} = \frac{\|\pi_i(b_{i+1})\|^2}{\|\pi_i(b_i)\|^2} \leq \delta$. Поэтому выполняется неравенство

$$D(\tilde{b}_1, \ldots, \tilde{b}_n) \leq \delta D(b_1, \ldots, b_n). \qquad (1)$$

Пусть $D_0 = D(b_1, \ldots, d_n)$ — значение целозначной функции D на исходном базисе решетки на входе LLL -алгортима, а D_k — соответствующее значение после k -й итерации (loop). Тогда из формулы 1 следует соотношение $D_k \leq \delta^k D_0$.

Оценка числа итераций в цикле (loop)

Поскольку D — целозначная положительная функция и $\delta < 1$, выполняется неравенство

$$k \leq \frac{\log D_0}{\log(1/\delta)}.$$

Поскольку D_0 вычислима за полиномиальное время от длины входа, значение $\log D_0$ полиномиально от длины входа. Следовательно, если $\delta < 1$ — константа, то число итераций полиномиально от длины входа.

Полиномиальность остается для последовательности δ_n

Следующая лемма показывает, что для некоторой возрастающей функции δ_n, для которой $\lim_{n\to\infty} \delta_n = 1$, число итераций остается полиномиальным.

Лемма

Если $\delta_n = (1/4) + (3/4)^{n/(n-1)}$, *то для всех* $c > 1$ *существует такое* N, *что для всех* $n > N$ *выполняется неравенство* $(\log(1/\delta_n))^{-1} \leq n^c$.

Доказательство. Неравенство $(\log(1/\delta_n))^{-1} \leq n^c$ эквивалентно неравенству $1 - 2^{-\left(\frac{1}{n}\right)^c} \leq 1 - \delta_n$. Согласно определению δ_n выполняется равенство

$$1 - \delta_n = (3/4) - (3/4)^{n/(n-1)} = (3/4)\left(1 - (3/4)^{1/(n-1)}\right)$$

Следовательно, доказываемое неравенство эквивалентно соотношению

$$\frac{1 - 2^{-\left(\frac{1}{n}\right)^c}}{(3/4)(1 - (3/4)^{1/(n-1)})} \leq 1. \tag{2}$$

Чтобы доказать это соотношение, достаточно проверить, что выполняется равенство

$$\lim_{n\to\infty} \frac{1 - 2^{-\left(\frac{1}{n}\right)^c}}{(3/4)(1 - (3/4)^{1/(n-1)})} = 0. \tag{3}$$

Сделаем замену переменных $x = 1/(n-1)$. Тогда подставляя $n = 1 + 1/x$ в соотношение 3, получим эквивалентное равенство

$$\lim_{x\to 0} \frac{1 - 2^{-\left(\frac{x}{x+1}\right)^c}}{(3/4)(1 - (3/4)^x)} = 0.$$

Для вычисления предела воспользуемся правилом Лопиталя. Имеем

$$\lim_{x\to 0}\frac{1-2^{-\left(\frac{x}{x+1}\right)^c}}{(3/4)(1-(3/4)^x)}=\lim_{x\to 0}\frac{2^{-\left(\frac{x}{x+1}\right)^c}\frac{c\ln 2}{(1+x)^2}\left(\frac{x}{1+x}\right)^{c-1}}{\frac{3}{4}\left(\frac{3}{4}\right)^x\ln(4/3)}=0.$$

Из доказанной леммы получаем, что δ_n-LLL алгоритм находит приближение кратчайшего вектора решетки с точностью до множителя $(2\sqrt{3})^n$ за полиномиальное относительно длины входа число итераций.

Полиномиальность

Число арифметических операций, выполняемых в каждом цикле полиномиально. Поэтому, чтобы получить полиномиальную оценку времени выполнения алгоритма, достаточно проверить, что размеры всех чисел, получаемых в процессе выполнения алгоритма, полиномиальны относительно длины входа. Поскольку при выполнении LLL -алгоритма операции производятся над рациональными числами, достаточно проверить полиномиальность размеров их числителей и знаменателей.

Ортогонализация Грамма-Шмидта

Из формул ортогонализации Грамма-Шмидта следует, что $b_i - b_i^* \in \text{span}(b_1, \ldots, b_{i-1})$, т.е.

$$b_i - b_i^* = \sum_{j=1}^{i-1} v_{i,j} b_j \tag{4}$$

для некоторых вещественных чисел $v_{i,j}$. Для любого $t < i$ выполняется равенство

$$(b_i, b_t) = \sum_{j=1}^{i-1} v_{i,j} (b_j, b_t). \tag{5}$$

Система линейных уравнений для коэффициентов $v_{i,j}$

Составим матрицу $B_t = (b_1, \ \ldots, b_t)$ из столбцов и вектор-столбец $v_i = (v_{i,1}, \ \ldots, v_{i,i-1})'$. Объединяя уравнения (5) при $t = 1, \ \ldots, i-1$, получим систему линейных уравнений

$$b_i' B_{i-1} = v_i' B_{i-1}' B_{i-1}.$$

Детерминант системы равен $\det(B_{i-1}' B_{i-1}) = d_{i-1}$. Согласно правилу Крамера компоненты вектора $d_{i-1} v_i$ целые числа.

Оценка для знаменателей

Поскольку из равенства (4) следует соотношение

$$d_{i-1}b_i^* = d_{i-1}b_i + \sum_{j=1}^{i-1}(d_{i-1}v_{i,j})b_j, \qquad (6)$$

знаменатели рациональных векторов b_i^* являются делителями целых чисел d_{i-1}. Представим теперь коэффициенты $\mu_{i,j}$, используемые в процедуре ортогонализации, как дроби:

$$\mu_{i,j} = \frac{(b_i, b_j^*)}{(b_j^*, b_j^*)} = \frac{d_{j-1}(b_i, b_j^*)}{d_{j-1}\|b_j^*\|^2} = \frac{(b_i, d_{j-1}b_j^*)}{d_j},$$

поскольку $d_j = \prod_{k=1}^{j} \|b_k^*\|^2$. Поэтому знаменатели μ_{ij} делят d_j.

Оценка для знаменателей

Ранее было показано, что размер представления начального значения D_0 целозначной функции $D = \prod_{i=1}^{n} d_i$ полиномиален относительно длины входа и эта величина уменьшается в процессе выполнения LLL -алгоритма.

Поэтому знаменатели всех рациональных чисел (поскольку знаменатели μ_{ij} делят d_j) в процессе выполнения алгоритма имеют полиномиальный размер относительно длины входа.

Полиномиальность размеров числителей

Согласно определению выполняются соотношения $|\mu_{i,j}| \leq 1/2$. Отметим, что при $i > 1$ из целочисленности вектора $d_i b_i^*$ (см. соотношение (6)) следует, что $\|b_i^*\| \geq 1/d_i$. Также выполняются соотношения $\|b_1^*\| = \|b_1\| \geq 1$ и $d_i = \prod_{j=1}^{i} \|b_j^*\|^2$.

Следовательно,

$$\|b_i^*\|^2 = \frac{d_i}{\prod_{j=1}^{i-1} \|b_j^*\|^2} \leq d_i \prod_{j=1}^{i-2} d_j^2 \leq D.$$

Полиномиальность размеров числителей

Поэтому

$$\|b_i\|^2 = \|b_i^*\|^2 + \sum_{j=1}^{i-1} \mu_{i,j}^2 \|b_j^*\|^2 \leq D + (n/4)D \leq nD.$$

Следовательно, длины числителей также имеют полиномиальнцю длину относительно длины входа и LLL -алгоритм с $\delta_n = (1/4) + (3/4)^{n/(n-1)}$ выполняется за полиномиальное время от длины входа.

Теорема существования приведенного базиса

Таким образом доказана

Теорема

Существует полиномиальный алгоритм, находящий для базиса B решетки LLL -приведенный базис решетки $\mathcal{L}(B)$ с параметром $\delta_n = (1/4) + (3/4)^{n/(n-1)}$.

Теперь из леммы 3 следует

Теорема

Существует полиномиальный алгоритм, находящий для базиса B решетки ненулевой вектор решетки $x \in \mathcal{L}(B)$ длины не более чем $(2/\sqrt{3})^n \lambda_1$.

Аппроксимация решения задачи CVP

Задача ACVP (Approximate CVP). Пусть задан вектор $b \in \mathbb{R}^m$ и n-мерная решетка с базисом $(b_1, \ldots, b_n)$, $(n \leq m)$. Требуется найти вектор $b_0 \in \mathcal{L}(b_1, \ldots, b_n)$, для которого выполнено соотношение

$$\|b - b_0\| \leq 2(2/\sqrt{3})^n \min_{x \in \mathcal{L}(b_1, \ldots, b_n)} \|x - b\|.$$

ACVP-алгоритм

Вход: Базис решетки $B = (b_1, \ldots, b_n) \in \mathbb{Z}^{m \times n}$ и вектор $t \in \mathbb{Z}^m$

Выход: Вектор решетки $x \in \mathcal{L}(b_1, \ldots, b_n)$, для которого выполняется соотношение

$$\|x - t\| \leq 2(2/\sqrt{3})^n \min_{y \in \mathcal{L}(b_1, \ldots, b_n)} \|y - t\|.$$

Выполнить LLL -алгоритм (найти приведенный базис).

$b := t$

for $j = n, \ldots, 1$

$c_j = \lfloor (b, b_j^*)/(b_j^*, b_j^*) \rceil$

$b := b - c_j b_j$

$x := t - b$ — выход