



Gitter in der Kryptographie

Diplomarbeit

Humboldt-Universität zu Berlin
Mathematisch-Naturwissenschaftliche Fakultät II
Institut für Informatik

eingereicht von: Kay Schönberger

Betreuer: Prof. Dr. Johannes Köbler

Berlin, den 14. Mai 2008

Inhaltsverzeichnis

1	Einleitung	5
2	Grundlegende Begriffe und Sätze	7
2.1	Notation	7
2.2	Einführung in die Gittertheorie	8
2.3	Gitterprobleme	10
2.4	GRAM-SCHMIDT-Orthogonalisierung	11
2.5	Grundlagen der Komplexitätstheorie	14
3	Einfache Reduktionsverfahren	17
3.1	Paarweise Reduktion	21
3.2	Längenreduktion	23
3.3	GAUSS-Reduktion	24
3.4	L^3 -Reduktion	26
4	Anwendungen der Gitterreduktion	31
4.1	BABAI's Round-Off-Methode	31
4.2	Primfaktorzerlegung	36
5	Gitterbasierte Kryptosysteme	39
5.1	Das AJTAI-DWORK-System	39
5.1.1	Variante 1	40
5.1.2	Variante 2	41
5.2	Das GH-Kryptosystem	44
5.2.1	Parameterwahl und Aufwand	48
5.2.2	Angriffsmethoden	49
5.3	MICCIANCIOS Kryptosystem	51
5.4	Das NTRU-Kryptosystem	55
5.4.1	Praktische Parameterwahl	57
5.4.2	Angriffsmethoden	59
6	Komplexität von Gitterproblemen	63
6.1	Reduktion von SVP auf CVP	65
6.2	Randomisierte Reduktion	66
6.3	\mathcal{NP} -Schwere von SVP_∞ und CVP	67
6.4	Nicht-Approximierbarkeit von SVP	70

7	Anhang	73
7.1	Meilensteine der Gittertheorie	73
7.2	Vermutungen und offene Probleme	74
7.3	Implementationen	75

1 Einleitung

Die vorliegende Diplomarbeit befasst sich mit der Theorie von Gittern und ihren Anwendungen in der Kryptographie. Beim Entwurf von Kryptosystemen ist man an Einwegfunktionen interessiert, die die Sicherheit des jeweiligen Kryptosystems gewährleisten sollen. Bei vielen bekannten Systemen wie zum Beispiel RSA ist die Frage nach der genauen Schwierigkeit, das Kryptosystem zu brechen, noch ungeklärt. Die Gittertheorie liefert dagegen Probleme, von denen man zumindest ihre \mathcal{NP} -Vollständigkeit beweisen kann. Zwei typische Gitterprobleme sind das „Shortest Vector Problem“ (SVP) und das „Closest Vector Problem“ (CVP). Die Untersuchung dieser Probleme hat in den vergangenen 25 Jahren eine Reihe von interessanten Ergebnissen hervorgebracht, die in der vorliegenden Arbeit zum Teil behandelt werden.

Die Arbeit konzentriert sich im Einzelnen auf

- die Erarbeitung der Grundlagen der Gittertheorie
- die Vorstellung und Analyse von Reduktionsverfahren für Gitterbasen
- die Approximation von Gitterproblemen wie SVP und CVP
- die Entwicklung gitterbasierter Kryptosysteme, wobei der Frage der praktischen Anwendbarkeit nachgegangen wird
- die Untersuchung komplexitätstheoretischer Fragen in Bezug auf Gitterprobleme

Schwerpunkt bildet das Kapitel über gitterbasierte Kryptosysteme, von denen man sich praktische Einsatzmöglichkeiten erhofft. Ein weiterer wichtiger Punkt ist die Antwort auf die Frage nach der \mathcal{NP} -Schwere der Gitterprobleme und ihre Approximierbarkeit.

1 Einleitung

2 Grundlegende Begriffe und Sätze

In diesem Abschnitt sollen die Grundbegriffe und die zugehörige Notation eingeführt werden. Da wir ohne elementare Sätze aus der linearen Algebra und der Gittertheorie nicht auskommen können, werden diese ebenfalls Erwähnung finden. Es kann aber nicht jeder Satz bewiesen werden, sodass auf die entsprechende Literatur verwiesen wird, z. B. [Art98].

2.1 Notation

Seien nachfolgend \mathbb{R}^m der m -dimensionale reelle Vektorraum und x, y Vektoren aus \mathbb{R}^m .

Definition 2.1.1 $\langle x, y \rangle := \sum_{i=1}^m x_i y_i$ bezeichnet das Standardskalarprodukt des \mathbb{R}^m . x und y heißen orthogonal, falls $\langle x, y \rangle = 0$.

Definition 2.1.2 $\|x\|_2 := \langle x, x \rangle^{\frac{1}{2}}$ bezeichnet die Euklidische Norm und $\|x\|_\infty := \max_{1 \leq i \leq m} \{|x_i|\}$ die Maximumnorm. Allgemein bezeichnet $\|x\|_p := (\sum_{i=1}^m |x_i|^p)^{\frac{1}{p}}$ die \mathcal{L}_p -Norm von x .

Als Standardnorm benutzen wir ab hier die Euklidische Norm und schreiben dafür kurz $\|\cdot\|$.

Satz 2.1.3 (CAUCHY-SCHWARZsche Ungleichung) *Es gilt*

$$|\langle x, y \rangle| \leq \|x\| \cdot \|y\|$$

Satz 2.1.4 *Es gelten die folgenden Rechenregeln:*

$$\begin{aligned}\|x + y\| &= \|x\| + 2\langle x, y \rangle + \|y\| \\ \|x - y\| &= \|x\| - 2\langle x, y \rangle + \|y\|\end{aligned}$$

Satz 2.1.5 *Für den Winkel α zwischen zwei Vektoren x und y gilt*

$$\cos \alpha = \frac{\langle x, y \rangle}{\|x\| \cdot \|y\|}.$$

Für $\langle x, y \rangle = 0$ ergibt sich ein rechter Winkel, für $\langle x, y \rangle > 0$ gilt $|\alpha| < 90^\circ$, für $\langle x, y \rangle < 0$ gilt $|\alpha| > 90^\circ$.

2 Grundlegende Begriffe und Sätze

Definition 2.1.6 Ist $x \in \mathbb{R}$, so wird mit $\lceil x \rceil$ diejenige ganze Zahl bezeichnet, für die $-\frac{1}{2} < x - \lceil x \rceil \leq \frac{1}{2}$ gilt.

Definition 2.1.7 (lineare Hülle) Seien v_1, \dots, v_n Vektoren aus \mathbb{R}^m . Dann heißt die Menge

$$\text{span}(v_1, \dots, v_n) = \left\{ \sum_{i=1}^n x_i v_i \mid x_i \in \mathbb{R} \right\}$$

lineare Hülle der Menge $\{v_1, \dots, v_n\}$.

Bemerkung 2.1.8 Die lineare Hülle der Menge $\{v_1, \dots, v_n\}$ ist der kleinste Untervektorraum des \mathbb{R}^m , der diese Menge enthält.

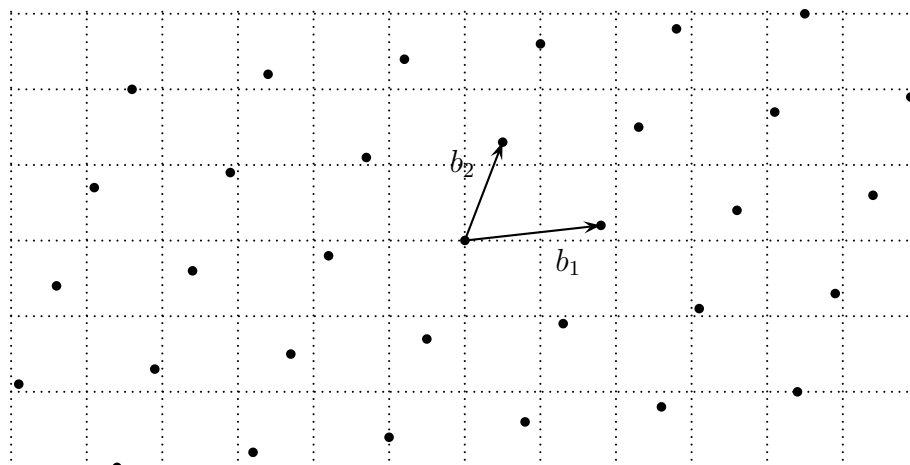
2.2 Einführung in die Gittertheorie

Definition 2.2.1 (Gitter) Seien $b_1, \dots, b_n \in \mathbb{R}^m$ linear unabhängige Vektoren. Die Menge

$$L = L(b_1, \dots, b_n) := \left\{ \sum_{i=1}^n z_i b_i \mid z_i \in \mathbb{Z} \right\}$$

heißt Gitter zur Gitterbasis $\{b_1, \dots, b_n\}$ mit Rang $\text{rg } L := n$. Im Falle $\text{rg } L = m$ heißt das Gitter vollständig.

Bemerkung 2.2.2 Ein Gitter ist eine diskrete additive Untergruppe des \mathbb{R}^m .



Beispiel eines Gitters im \mathbb{R}^2

Definition 2.2.3 (Untergitter, ganzzahliges Gitter) Seien $L, U \subset \mathbb{R}^m$ Gitter. Dann ist U ein Untergitter von L , falls $U \subseteq L$. L heißt ganzzahlig, falls L ein Untergitter von \mathbb{Z}^m ist.

Definition 2.2.4 (Basismatrix) Die Matrix $B := [b_1, \dots, b_n] \in \mathbb{R}^{m \times n}$ wird Basismatrix zum Gitter L genannt. Entsprechend bezeichnet $L(B) := L(b_1, \dots, b_n)$ das von B erzeugte Gitter.

Definition 2.2.5 Eine Matrix $M \in \mathbb{Z}^{n \times n}$ heißt unimodular, falls $\det M = \pm 1$. $\text{GL}_n(\mathbb{Z})$ bezeichnet die multiplikative Gruppe der unimodularen $(n \times n)$ -Matrizen.

Es stellt sich die Frage, wann für zwei Basismatrizen A und B die erzeugten Gitter $L(A)$ und $L(B)$ identisch sind.

Satz 2.2.6 Seien $A, B \in \mathbb{R}^{m \times n}$ zwei Gitterbasismatrizen. Genau dann ist $L(A) = L(B)$, wenn eine unimodulare Matrix $M \in \text{GL}_n(\mathbb{Z})$ mit der Eigenschaft $A = BM$ existiert.

Beweis. Ist M eine unimodulare Matrix und $A = BM$, so gehören die Spaltenvektoren von A zum Gitter $L(B)$. Da A eine Gitterbasismatrix ist, gilt $L(A) \subseteq L(B)$ und wegen $B = AM^{-1}$, $M^{-1} \in \text{GL}_n(\mathbb{Z})$ folgt mit derselben Argumentation $L(B) \subseteq L(A)$, also die Gleichheit beider Gitter. Wenn umgekehrt A und B dasselbe Gitter erzeugen, müssen die Spaltenvektoren von A in $L(B)$ liegen, es muss also eine Matrix $M_1 \in \mathbb{Z}^{n \times n}$ existieren mit $A = BM_1$. Ebenso liegen alle Spaltenvektoren von B in $L(A)$, es gilt also $B = AM_2$ für eine Matrix $M_2 \in \mathbb{Z}^{n \times n}$. Dann gilt jedoch $B = BM_1M_2$ und das Produkt $M_1M_2 = I_n$ ist die n -dimensionale Einheitsmatrix. Das ist nur möglich, wenn $M_1 = M_2^{-1}$ ist und M_1, M_2 unimodulare Matrizen sind. \square

Das bedeutet, dass die Basismatrix eines Gitters nicht eindeutig bestimmt ist. Da es für $n \geq 2$ unendlich viele unimodulare Matrizen M gibt, existieren sogar unendlich viele verschiedene Gitterbasen.

Definition 2.2.7 (Grundmasche) Die Grundmasche zu einer Basismatrix $B = [b_1, \dots, b_n]$ ist das Parallelotop

$$\mathcal{P}(B) := \left\{ \sum_{i=1}^n x_i b_i \mid 0 \leq x_i < 1 \right\}$$

Definition 2.2.8 (Gitterdeterminante) Die Determinante $\det L$ eines Gitters $L(B)$ ist definiert als

$$\det L := \text{vol } \mathcal{P}(B) = \sqrt{\det B^T B}$$

und entspricht dem n -dimensionalen Volumen von $\mathcal{P}(B)$.

Bemerkung 2.2.9 Die Matrix $B^T B \in \mathbb{R}^{n \times n}$ ist die GRAM-Matrix zur Matrix B .

Satz 2.2.10 Die Gitterdeterminante ist unabhängig von der Wahl der Gitterbasis.

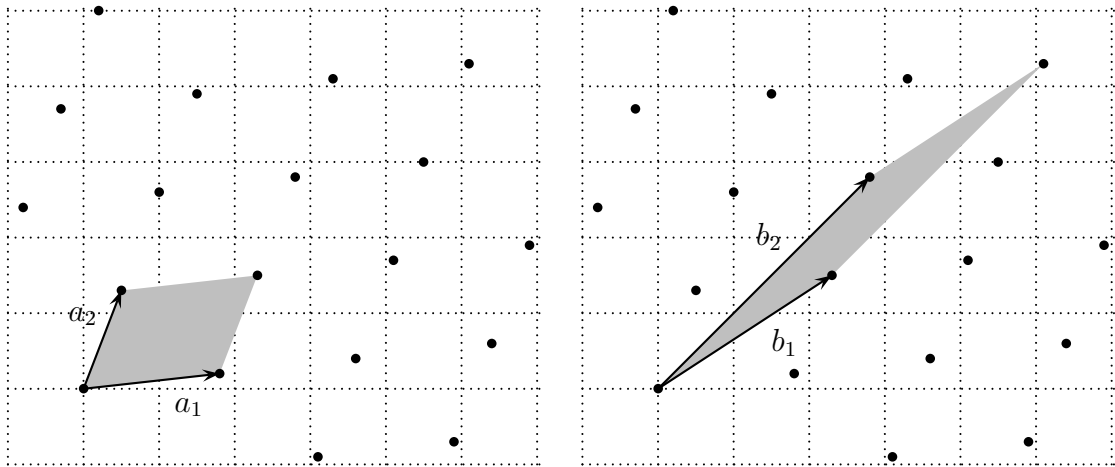
2 Grundlegende Begriffe und Sätze

Beweis. Seien A, B zwei Basismatrizen mit $L(A) = L(B)$. Dann existiert eine Matrix $M \in \text{GL}_n(\mathbb{Z})$ mit der Eigenschaft $A = BM$ und aufgrund der Multiplikativität der Determinante gilt

$$\begin{aligned} \det L(A) &= \sqrt{\det A^T A} = \sqrt{\det (BM)^T BM} = \sqrt{\det M^T B^T B M} \\ &= \sqrt{\det M^T \cdot \det B^T B \cdot \det M} = \sqrt{\det B^T B} = \det L(B) \end{aligned}$$

□

Die untenstehenden Abbildungen geben eine anschauliche Darstellung des Maschenvolumens (grau eingefärbt) eines Gitters für zwei unterschiedliche Basen:



Gitterdeterminante bzw. Volumen sind unabhängig von der Basis

Ist B eine Basismatrix zum Gitter L , so lässt sich jeder Vektor $v \in L$ darstellen als $v = Bz$ mit einem geeigneten Auswahlvektor $z \in \mathbb{Z}^n$. Um zu vorgelegtem $v \in L$ den Vektor z rekonstruieren zu können, betrachten wir die Gleichung

$$B^T v = B^T B z.$$

Da B vollen Rang besitzt, ist die GRAM-Matrix $B^T B$ regulär (das bedeutet, die Gitterdeterminante ist positiv) und man bekommt z durch Lösen eines linearen Gleichungssystems. Es gilt

$$z = (B^T B)^{-1} B^T v.$$

2.3 Gitterprobleme

Kern der Arbeit ist die Betrachtung einer Reihe von Problemen, die im Zusammenhang mit Gittern auftreten. Seit einigen Jahren haben diese in Form von neuen kryptographischen Verfahren Eingang in die Kryptographie gefunden. Ziel soll es sein, solche Verfahren vorzustellen und die Komplexität der Basisprobleme zu analysieren, auf denen diese Verfahren beruhen.

Ist $L \subset \mathbb{R}^m$ ein Gitter mit Basisvektoren b_1, \dots, b_n , so betrachte die folgenden Berechnungsprobleme:

- **Shortest-Vector-Problem (SVP):** Bestimme einen Gittervektor $b \in L, b \neq 0$, für den $\|b\|$ minimal wird.
- **Closest-Vector-Problem (CVP):** Bestimme zu einem Punkt $x \in \mathbb{R}^m$ einen Gittervektor $b \in L$, für den der Abstand $\|b - x\|$ minimal wird.
- **Shortest-Basis-Problem (SBP):** Bestimme eine Gitterbasis b_1, \dots, b_n von L , für die das Produkt $\prod_{i=1}^n \|b_i\|$ minimal wird.

Es handelt sich im Wesentlichen um Probleme der Gitterreduktion, die man auch für andere Normen betrachten kann. Definiert man ein $d \in \mathbb{R}$ und fragt nach einem Gittervektor b mit $\|b\| \leq d$ bzw. $\|b - x\| \leq d$, so gehen das SVP und das CVP in Entscheidungsprobleme über. Wie man zeigen kann, sind das CVP und das SBP \mathcal{NP} -schwer, das SVP ist \mathcal{NP} -schwer unter randomisierten Reduktionen - die Frage nach kryptographischen Anwendungen ist daher naheliegend.

2.4 GRAM-SCHMIDT-Orthogonalisierung

Das nachfolgend vorgestellte GRAM-SCHMIDT-Verfahren liefert zu einer gegebenen Basis $\{b_1, \dots, b_n\}$ Vektoren $\hat{b}_1, \dots, \hat{b}_n$ mit der Eigenschaft $\langle \hat{b}_i, \hat{b}_j \rangle = 0$ für alle $0 \leq i \neq j \leq n$. Erzeugt wird also eine Orthogonalbasis für die lineare Hülle $\text{span}(b_1, \dots, b_n)$ (aber *nicht* für das Gitter $L(b_1, \dots, b_n)$).

Algorithmus (GRAM-SCHMIDT-Verfahren)

EINGABE: Basisvektoren $b_1, \dots, b_n \in \mathbb{R}^m$

1. $\hat{b}_1 := b_1$
2. FOR $i = 2, \dots, n$ DO
 - FOR $j = 1, \dots, i - 1$ DO

$$\mu_{ij} := \frac{\langle b_i, \hat{b}_j \rangle}{\|\hat{b}_j\|^2}$$
$$\hat{b}_i := b_i - \sum_{j=1}^{i-1} \mu_{ij} \hat{b}_j$$

AUSGABE: paarweise orthogonale Basisvektoren $\hat{b}_1, \dots, \hat{b}_n \in \mathbb{R}^m$

Beweis der Korrektheit. Der Beweis wird induktiv geführt: Seien $\hat{b}_1, \dots, \hat{b}_{i-1}$ bereits paarweise orthogonal, d. h. $\langle \hat{b}_j, \hat{b}_k \rangle = 0$ für alle $1 \leq j \neq k < i$. Für den im i -ten Schritt berechneten Vektor \hat{b}_i gilt:

$$\begin{aligned} \langle \hat{b}_i, \hat{b}_k \rangle &= \langle b_i - \sum_{j=1}^{i-1} \mu_{ij} \hat{b}_j, \hat{b}_k \rangle \\ &= \langle b_i, \hat{b}_k \rangle - \sum_{j=1}^{i-1} \mu_{ij} \langle \hat{b}_j, \hat{b}_k \rangle \\ &\stackrel{\text{IV.}}{=} \langle b_i, \hat{b}_k \rangle - \mu_{ik} \langle \hat{b}_k, \hat{b}_k \rangle \\ &= \langle b_i, \hat{b}_k \rangle - \langle b_i, \hat{b}_k \rangle = 0 \end{aligned}$$

2 Grundlegende Begriffe und Sätze

\hat{b}_i ist also orthogonal zu allen vorher erzeugten Vektoren \hat{b}_k . Dadurch sind am Ende alle erzeugten Vektoren paarweise orthogonal. \square

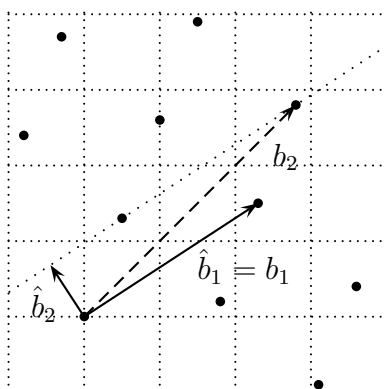
Definition 2.4.1 Die Koeffizienten μ_{ij} heißen GRAM-SCHMIDT-Koeffizienten. Zusätzlich setzt man $\mu_{ii} := 1$ für alle $1 \leq i \leq n$.

Mit Hilfe der GRAM-SCHMIDT-Koeffizienten lassen sich die Basisvektoren b_i und die Orthogonalvektoren \hat{b}_i ineinander überführen. Es gilt

$$\hat{b}_i = b_i - \sum_{j=1}^{i-1} \mu_{ij} \hat{b}_j$$

$$b_i = \hat{b}_i + \sum_{j=1}^{i-1} \mu_{ij} \hat{b}_j = \sum_{j=1}^i \mu_{ij} \hat{b}_j.$$

Anschaulich bedeutet die Orthogonalisierung von b_i die Projektion von b_i auf den zu $\text{span}(\hat{b}_1, \dots, \hat{b}_{i-1})$ orthogonalen Unterraum (siehe Abbildung).



Orthogonalisierung nach GRAM-SCHMIDT

Beispiel 2.4.2 Gegeben sind die drei Gitterbasisvektoren¹ $b_1 = (-1, 3, -5)$, $b_2 = (3, 7, -2)$ und $b_3 = (0, 7, 7)$. Wir nehmen $\hat{b}_1 := b_1 = (-1, 3, -5)$ in die Orthogonalbasis auf und orthogonalisieren b_2 bezüglich b_1 :

Berechnet werden $\|(-1, 3, -5)\|^2 = 35$, $\langle (3, 7, -2), (-1, 3, -5) \rangle = 28$, somit ist $\mu_{21} = \frac{28}{35} = \frac{4}{5}$. Also ergibt sich

$$\hat{b}_2 = (3, 7, -2) - \frac{4}{5} \cdot (-1, 3, -5) = \left(\frac{19}{5}, \frac{23}{5}, 2\right).$$

Um \hat{b}_3 zu erhalten, wird b_3 bezüglich b_1 und b_2 orthogonalisiert: Dazu bestimmen wir $\|(\frac{19}{5}, \frac{23}{5}, 2)\|^2 = \frac{198}{5}$ und die beiden Skalarprodukte $\langle (0, 7, 7), (-1, 3, -5) \rangle = -14$, $\langle (0, 7, 7), (\frac{19}{5}, \frac{23}{5}, 2) \rangle = \frac{231}{5}$. Die gesuchten GRAM-SCHMIDT-Koeffizienten sind also

¹Der Übersichtlichkeit halber werden in den Beispielen Zeilenvektoren benutzt.

$\mu_{31} = \frac{-14}{35} = -\frac{2}{5}$ sowie $\mu_{32} = \frac{231/5}{198/5} = \frac{7}{6}$. Der dritte Orthogonalvektor bestimmt sich demnach zu

$$\hat{b}_3 = (0, 7, 7) + \frac{2}{5} \cdot (-1, 3, -5) - \frac{7}{6} \cdot \left(\frac{19}{5}, \frac{23}{5}, 2\right) = \left(-\frac{29}{6}, \frac{17}{6}, \frac{8}{3}\right).$$

Damit haben wir die gesuchte GRAM-SCHMIDT-Basis

$$\hat{b}_1 = (-1, 3, -5), \quad \hat{b}_2 = \left(\frac{19}{5}, \frac{23}{5}, 2\right), \quad \hat{b}_3 = \left(-\frac{29}{6}, \frac{17}{6}, \frac{8}{3}\right)$$

erhalten. Wie sich leicht nachrechnen lässt, sind alle drei Vektoren paarweise orthogonal.

Ist $\hat{B} = [\hat{b}_1, \dots, \hat{b}_n]$ die zur Orthogonalbasis zugehörige Basismatrix, so lässt sich die Ausgangsmatrix B schreiben als $B = \hat{B}M$ mit einer Transformationsmatrix M , wobei

$$M = \begin{pmatrix} 1 & \mu_{21} & \cdots & \mu_{n-1,1} & \mu_{n,1} \\ 0 & 1 & \cdots & \mu_{n-1,2} & \mu_{n,2} \\ \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & 1 & \mu_{n,n-1} \\ 0 & \cdots & 0 & 0 & 1 \end{pmatrix}.$$

M ist eine rechte obere Dreiecksmatrix. Die Determinante einer solchen Matrix ist das Produkt der Hauptdiagonaleinträge; demzufolge ist $\det M = 1$. Wegen

$$\det L = \sqrt{\det B^T B} = \sqrt{\det M^T \hat{B}^T \hat{B} M} = \sqrt{\det \hat{B}^T \hat{B}}$$

bleibt die Gitterdeterminante unverändert. Da die in \hat{B} enthaltenen Vektoren orthogonal sind, erhält man

$$\det L = \text{vol } \mathcal{P}(\hat{B}) = \prod_{i=1}^n \|\hat{b}_i\|.$$

Da für alle Basisvektoren $\|\hat{b}_i\| \leq \|b_i\|$ gilt, folgt hieraus die HADAMARD-Ungleichung:

Korollar 2.4.3 (HADAMARD-Ungleichung) Jedes Gitter $L = L(b_1, \dots, b_n) \subset \mathbb{R}^m$ erfüllt die Ungleichung

$$\det L \leq \prod_{i=1}^n \|b_i\|.$$

Bemerkung 2.4.4 Der Quotient

$$\text{OrthDefect}(b_1, \dots, b_n) = \frac{\prod_{i=1}^n \|b_i\|}{\det L}$$

wird auch als Orthogonalitätsdefekt bezeichnet und ist ein Maß für die Reduziertheit einer Gitterbasis.

2.5 Grundlagen der Komplexitätstheorie

In diesem Abschnitt werden kurz die wichtigsten aus der Komplexitätstheorie verwendeten Begriffe erläutert. Der Begriff der Berechenbarkeit bezieht sich auf die deterministischen (bzw. nichtdeterministischen) 1-Band-Turingmaschinen (DTMs bzw. NTMs); für genauere Informationen wird auf die entsprechende Literatur verwiesen.

Definition 2.5.1 (Alphabet, Sprache) *Ein Alphabet ist eine endliche Menge Σ von Zeichen. Eine Sprache L ist eine Teilmenge $L \subseteq \Sigma^*$.*

Definition 2.5.2 (Klasse \mathcal{P} , \mathcal{NP}) *\mathcal{P} bezeichnet die Klasse derjenigen Sprachen, welche in Polynomialzeit (bezogen auf die Länge der Eingabe) deterministisch akzeptiert werden können. \mathcal{NP} ist die Klasse der Sprachen, welche in Polynomialzeit nichtdeterministisch akzeptiert werden können.*

Definition 2.5.3 (Polynomialzeitreduktion) *Gegeben seien Sprachen $L', L \subseteq \Sigma^*$. L' heißt polynomiell reduzierbar auf L (kurz $L' \leq_p L$), falls es eine polynomiell berechenbare Funktion $f : \Sigma^* \rightarrow \Sigma^*$ gibt mit*

$$w \in L' \iff f(w) \in L.$$

L' heißt COOK-reduzierbar auf L , falls f polynomiell ist und durch (mehrere) Orakelanfragen entscheidet, ob ein Wort in L liegt oder nicht.

Definition 2.5.4 (\mathcal{NP} -schwer, \mathcal{NP} -vollständig) *Eine gegebene Sprache L heißt \mathcal{NP} -schwer, falls $L' \leq_p L$ für alle Sprachen $L' \in \mathcal{NP}$. L heißt \mathcal{NP} -vollständig, falls L \mathcal{NP} -schwer ist und zusätzlich $L \in \mathcal{NP}$ gilt.*

Das bedeutet insbesondere, dass $\mathcal{P} = \mathcal{NP}$ gilt, falls sich eine \mathcal{NP} -vollständige Sprache L mit $L \in \mathcal{P}$ finden lässt. Dass \mathcal{NP} -vollständige Sprachen überhaupt existieren, wurde erstmals 1971 von COOK anhand des Erfüllbarkeitsproblems (SAT) festgestellt.

Definition 2.5.5 (PTM) *Eine probabilistische Turingmaschine (PTM) ist eine nichtdeterministische Turingmaschine, die in jedem Schritt genau ein oder zwei Nachfolgekonfigurationen besitzt. In letzterem Fall werden beide jeweils mit Wahrscheinlichkeit $1/2$ ausgewählt.*

Die PTMs werden häufig für randomisierte Reduktionen benötigt, da sich diese als nützlich erweisen (so auch für uns in einem späteren Kapitel). Für probabilistische Turingmaschinen wurden auch eigene Komplexitätsklassen eingeführt.

Definition 2.5.6 (Einwegfunktion, Falltür) *Eine Funktion $f : X \rightarrow Y$ heißt Einwegfunktion, falls*

1. zu gegebenem $x \in X$ ist $y = f(x)$ in Polynomialzeit berechenbar

2. zu gegebenem $y \in Y$ gibt es keinen Algorithmus, der in Polynomialzeit ein $x \in X$ findet mit $y = f(x)$

Eine zusätzliche Information I heißt Falltürinformation zu f , falls es einen Polynomialzeitalgorithmus gibt, der bei Eingabe von $y \in Y$ und I ein $x \in X$ liefert mit $y = f(x)$.

Unter Punkt 2 fordert man häufig auch nur probabilistische Polynomialzeitalgorithmen. Die Existenz von Einwegfunktionen konnte noch nicht streng nachgewiesen werden, obwohl von einigen bekannten Funktionen angenommen wird, dass es sich um solche handelt. Einwegfunktionen mit Falltürinformation spielen eine Rolle beim Design von Kryptosystemen; die Falltürinformation wird hier in der Regel für die Entschlüsselung der Kryptotexte benötigt.

Definition 2.5.7 (Promise-Problem) *Unter einem Promise-Problem versteht man ein Paar (L_{ja}, L_{nein}) zweier disjunkter Sprachen $L_{ja}, L_{nein} \subseteq \Sigma^*$. Ein Algorithmus löst das Promise-Problem, falls er bei Eingabe einer Instanz $I \in L_{ja} \cup L_{nein}$ korrekt entscheidet, ob $I \in L_{ja}$ oder $I \in L_{nein}$. Das Verhalten bei $I \notin L_{ja} \cup L_{nein}$ ist undefiniert.*

2 Grundlegende Begriffe und Sätze

3 Einfache Reduktionsverfahren

Wie bereits gezeigt, kann man zu einem Gitter L beliebig viele Basen angeben. In der Praxis ist man aber an Basen interessiert, die möglichst kurze Vektoren enthalten (reduzierte Basen). Wir werden also zunächst der Frage nachgehen, inwiefern sich eine gegebene Gitterbasis verkürzen lässt. Diese Frage spielt später bei der Konstruktion gitterbasierter Kryptosysteme eine wichtige Rolle.

Wir haben gesehen, dass die Gitterdeterminante unabhängig von der konkreten Wahl der Basis ist. Das bedeutet, dass reduzierte Basen Vektoren besitzen, die paarweise orthogonal oder „fast“ orthogonal sind (nicht jedes Gitter besitzt eine Orthogonalbasis). Die Orthogonalität ist hier also ein Maß für die Reduziertheit einer Basis. Mit den GRAM-SCHMIDT-Vektoren bekommt man eine untere Abschätzung für die Länge des kürzesten Gittervektors:

Lemma 3.0.8 *Ist $\{\hat{b}_1, \dots, \hat{b}_n\}$ die GRAM-SCHMIDT-Basis zu $L = L(b_1, \dots, b_n) \subset \mathbb{R}^m$, dann gilt für jeden vom Nullvektor verschiedenen Gittervektor $v \in L$ die Ungleichung*

$$\|v\| \geq \min\{\|\hat{b}_1\|, \dots, \|\hat{b}_n\|\}.$$

Beweis. v lässt sich darstellen durch $v = \sum_{i=1}^n z_i b_i$ mit ganzen Zahlen $z_i \in \mathbb{Z}$. Für die Basisvektoren gilt $b_i = \sum_{j=1}^i \mu_{ij} \hat{b}_j$. Ist k der höchste Index mit $z_k \neq 0$, so gilt

$$\begin{aligned} v &= \sum_{i=1}^k z_i \sum_{j=1}^i \mu_{ij} \hat{b}_j \\ &= z_k \sum_{j=1}^k \mu_{kj} \hat{b}_j + \sum_{i=1}^{k-1} z_i \sum_{j=1}^i \mu_{ij} \hat{b}_j \\ &= z_k \hat{b}_k + \sum_{i=1}^{k-1} x_i \hat{b}_i \end{aligned}$$

für geeignete $x_i \in \mathbb{R}$. Damit folgt

$$\begin{aligned} \|v\|^2 &= \langle v, v \rangle = \left\langle z_k \hat{b}_k + \sum_{i=1}^{k-1} x_i \hat{b}_i, z_k \hat{b}_k + \sum_{i=1}^{k-1} x_i \hat{b}_i \right\rangle \\ &= z_k^2 \langle \hat{b}_k, \hat{b}_k \rangle + \sum_{i=1}^{k-1} x_i^2 \langle \hat{b}_i, \hat{b}_i \rangle \\ &\geq z_k^2 \|\hat{b}_k\|^2 \geq \|\hat{b}_k\|^2 \\ \|v\| &\geq \min\{\|\hat{b}_1\|, \dots, \|\hat{b}_n\|\}. \end{aligned}$$

3 Einfache Reduktionsverfahren

□

Nachfolgend sollen einige elementare Verfahren vorgestellt werden, die zu einer vorgegebenen Basis eine reduzierte Basis berechnen. Um zu beurteilen, ob eine Basis gut reduziert ist, benutzt man den von MINKOWSKI eingeführten Begriff des sukzessiven Minimums.

Definition 3.0.9 (sukzessive Minima) Sei $L \subset \mathbb{R}^m$ ein Gitter mit Rang n . Dann ist für $1 \leq i \leq n$ das i -te sukzessive Minimum von L definiert als

$$\lambda_i(L) := \min \left\{ \max_{1 \leq j \leq i} \|v_j\| \mid v_1, \dots, v_i \in L \text{ sind linear unabhängig} \right\}$$

Den Wert $\lambda_i(L)$ kann man auffassen als kleinsten Radius r einer Kugel um den Nullpunkt, die i linear unabhängige Gittervektoren aus L enthält. Da L diskret ist gibt es auch Gittervektoren, die die sukzessiven Minima realisieren. Man kann sie als untere Schranke für die Länge von Basisvektoren auffassen.

Satz 3.0.10 Die sukzessiven Minima besitzen folgende Eigenschaften:

1. Es ist $\lambda_1(L) \leq \dots \leq \lambda_n(L)$.
2. Für linear unabhängige Gittervektoren $v_1, \dots, v_i \in L$ mit $\|v_1\| \leq \dots \leq \|v_i\|$ gilt $\lambda_i(L) \leq \|v_i\|$.
3. Es existieren linear unabhängige Vektoren $v_1, \dots, v_n \in L$ mit $\lambda_i(L) = \|v_i\|$ für $1 \leq i \leq n$.

Beweis.

1. Sei $i \leq j \leq n$. Es existieren linear unabhängige Vektoren $v_1, \dots, v_j \in L$, wobei $\lambda_j(L) = \max\{\|v_1\|, \dots, \|v_j\|\}$. Nach Entfernen von $j - i$ beliebigen Vektoren $v_{k_{i+1}}, \dots, v_{k_j}$ gilt nach Definition $\lambda_i(L) \leq \max\{\|v_{k_1}\|, \dots, \|v_{k_i}\|\} \leq \lambda_j(L)$.
2. Aus der Definition folgt direkt $\lambda_i(L) \leq \max\{\|v_1\|, \dots, \|v_i\|\} = \|v_i\|$.
3. Beweis durch Induktion: Sind bereits $v_1, \dots, v_{i-1} \in L$ linear unabhängig mit $\lambda_j(L) = \|v_j\|$ und $1 \leq j \leq i - 1$, so gibt es linear unabhängige Vektoren $w_1, \dots, w_i \in L$ mit $\lambda_i(L) = \max_{1 \leq k \leq i} \|w_k\|$. Eines dieser w_k ist linear unabhängig zu v_1, \dots, v_{i-1} , sodass wir dieses hinzunehmen können. Mit $v_i := w_k$ gilt dann $\|v_i\| \leq \lambda_i(L)$ und aufgrund von 2. die Gleichheit $\|v_i\| = \lambda_i(L)$.

□

Gemäß der Definition ist $\lambda_1(L)$ die Länge des kürzesten, vom Nullvektor verschiedenen Gittervektors. Aus Satz 3.0.10 folgt zwar, dass die sukzessiven Minima von linear unabhängigen Gittervektoren v_1, \dots, v_n angenommen werden können, jedoch handelt es sich hierbei i. A. um keine Basis des Gitters, sondern um eine Basis eines kleineren Untergitters. Wie das nachfolgende Beispiel belegt, muss es eine solche Gitterbasis nicht geben.

Beispiel 3.0.11 Wir betrachten das Gitter $L(b_1, \dots, b_5)$ mit den Basisvektoren $b_i := e_i$ für $i = 1, \dots, 4$ und $b_5 := \frac{1}{2} \sum_{i=1}^5 e_i$. L ist ein vollständiges Gitter im \mathbb{R}^5 . Wegen $e_5 = 2b_5 - \sum_{i=1}^4 b_i$ sind alle Einheitsvektoren enthalten, die sukzessiven Minima besitzen also alle den Wert 1. Es gibt nun aber keine Basisvektoren a_1, \dots, a_5 mit Länge 1: Es müsste ein $a_k = \sum_{i=1}^5 z_i b_i$, $z_i \in \mathbb{Z}$ mit ungeradem z_5 geben, d. h. sämtliche Vektoreinträge a_{kj} wären von Null verschiedene ganzzahlige Vielfache von $\frac{1}{2}$. Dann aber wäre $|a_{kj}| \geq \frac{1}{2}$ und somit $\|a_k\| \geq \sqrt{\frac{5}{4}} > 1$.

Dabei handelt es sich um das einfachste Gegenbeispiel, da für Gitter mit Rang ≤ 4 stets eine Basis existiert, die die sukzessiven Minima realisiert:

Satz 3.0.12 Sei $L \subset \mathbb{R}^m$ ein Gitter mit Rang n .

1. Ist $r \leq 3$ und werden die ersten r sukzessiven Minima von linear unabhängigen Gittervektoren b_1, \dots, b_r realisiert, so lassen sich diese zu einer Basis von L ergänzen.
2. L besitzt stets eine Basis b_1, \dots, b_n mit $\|b_i\| = \lambda_i(L)$ für $1 \leq i \leq \min\{4, n\}$.

Beweis. Hier wird auf [PoZas89] verwiesen. □

Die im letzten Abschnitt eingeführte HADAMARD-Ungleichung lässt sich mit Hilfe der sukzessiven Minima noch verschärfen: Realisieren a_1, \dots, a_n die sukzessiven Minima von L , so gilt für das von diesen Vektoren erzeugte Untergitter \tilde{L}

$$\det \tilde{L}(a_1, \dots, a_n) \geq \det L$$

und mit Hilfe der HADAMARD-Ungleichung ergibt sich

$$\det L \leq \det \tilde{L}(a_1, \dots, a_n) \leq \prod_{i=1}^n \|a_i\| = \prod_{i=1}^n \lambda_i(L).$$

Darüberhinaus ist es uns möglich, nicht nur eine untere Schranke für die Länge des kürzesten Gittervektors, sondern sogar untere Schranken für alle $\lambda_i(L)$ anzugeben. Das erledigt nachfolgendes Lemma.

Lemma 3.0.13 Sind b_1, \dots, b_n Gitterbasisvektoren von $L \subset \mathbb{R}^m$ und $\hat{b}_1, \dots, \hat{b}_n$ die zugehörigen GRAM-SCHMIDT-Basisvektoren, so gilt

$$\lambda_i(L) \geq \min_{i \leq j \leq n} \|\hat{b}_j\|.$$

Beweis. v_1, \dots, v_n seien linear unabhängige Gittervektoren, die die sukzessiven Minima realisieren, d. h. $\|v_i\| = \lambda_i(L)$. Für alle $1 \leq k \leq n$ gibt es nun Koeffizienten $z_{kj} \in \mathbb{Z}, x_{kj} \in \mathbb{R}$, sodass

$$v_k = \sum_{j=1}^n z_{kj} b_j = \sum_{j=1}^n x_{kj} \hat{b}_j.$$

3 Einfache Reduktionsverfahren

Damit erhält man die Darstellung

$$v_k = \sum_{i=1}^n z_{ki} b_i = \sum_{i=1}^n z_{ki} \sum_{j=1}^i \mu_{ij} \hat{b}_j = \sum_{j=1}^n \left(\sum_{i=j}^n z_{ki} \mu_{ij} \right) \hat{b}_j.$$

Aufgrund der linearen Unabhängigkeit der \hat{b}_j folgt $x_{kj} = \sum_{i=j}^n z_{ki} \mu_{ij}$. Ist s_k der größte Index, für den $z_{ks_k} \neq 0$ ist, so folgt $x_{ks_k} = z_{ks_k}$ (wegen $\mu_{ii} = 1$) und somit ist x_{ks_k} eine ganze Zahl ungleich Null. Das lässt sich nun benutzen: Für jedes $i \leq n$ existiert ein $k \leq i$ mit der Eigenschaft $s_k \geq i$ (wäre dem nicht so, wäre die Dimension des Unterraums $\text{span}(v_1, \dots, v_i)$ kleiner als i) und es gilt die Ungleichungskette

$$\lambda_i(L)^2 \geq \lambda_k(L)^2 = \|v_k\|^2 \geq x_{ks_k}^2 \|\hat{b}_{s_k}\|^2 \geq \|\hat{b}_{s_k}\|^2 \geq \min_{i \leq j \leq n} \|\hat{b}_j\|^2.$$

□

Ein wichtiges Ergebnis der Gittertheorie ist ein Zusammenhang zwischen $\lambda_1(L)$ und dem Gittervolumen $\det L$. Ist das Gittervolumen bekannt, so lässt sich für die Länge des kürzesten Gittervektors eine obere Schranke angeben.

Definition 3.0.14 (HERMITE-Konstante) Die HERMITE-Konstante γ_n ist definiert als

$$\gamma_n := \max \left\{ \frac{\lambda_1(L)^2}{(\det L)^{2/n}} \mid L \subset \mathbb{R}^m \text{ ist ein Gitter mit Rang } n \right\}.$$

$(\det L)^{1/n}$ kann man als Kantenlänge eines Würfels auffassen, dessen Volumen dem Gittervolumen entspricht. γ_n ist dann die maximal mögliche (quadrierte) Abweichung von λ_1 zu diesem Wert.

Ist die entsprechende HERMITE-Konstante γ_n bekannt, so bekommt man mit

$$\lambda_1(L) \leq \sqrt{\gamma_n} \cdot (\det L)^{1/n}$$

eine obere Schranke für die Länge des kürzesten Gittervektors. Exakte Werte γ_n für $n = 1, \dots, 8$ sind bekannt und wurden erstmals von BLICHFELDT, KORKINE und ZOLOTAREFF angegeben:

Rang n	1	2	3	4	5	6	7	8
HERMITE-Konstante γ_n^n	1	$\frac{4}{3}$	2	4	8	$\frac{2^6}{3}$	2^6	2^8

Bekannte HERMITE-Konstanten

Für große n begnügt man sich mit einer auf GAUSS zurückgehenden Heuristik, die asymptotische Schranken für γ_n liefert. Nach dieser gilt

$$\frac{n}{2\pi e} + o(n) \leq \gamma_n \leq \frac{n}{\pi e} + o(n).$$

Diese Abschätzung spielt bei den später vorgestellten gitterbasierten Angriffen auf das NTRU-Kryptosystem eine Rolle. Eine Schranke für das Produkt aller $\lambda_i(L)$ liefert die MINKOWSKI-Ungleichung.

Satz 3.0.15 (MINKOWSKI-Ungleichung) Sei $L \subset \mathbb{R}^m$ ein Gitter mit Rang n . Dann gilt:

$$\prod_{i=1}^n \lambda_i(L) \leq \sqrt{\gamma_n^n} \cdot \det L.$$

Neben der aus der HADAMARD-Ungleichung folgenden unteren Schranke verfügen wir damit also auch über eine obere Schranke.

3.1 Paarweise Reduktion

Die Idee der paarweisen Reduktion ist es, durch Subtraktion bzw. Addition eines Vielfachen eines Basisvektors zu einem anderen einen möglicherweise kürzeren Basisvektor zu erhalten. Dieser ersetzt dann den größeren der beiden Vektoren.

Definition 3.1.1 (paarweise reduzierte Basis) Seien $b_1, \dots, b_n \in \mathbb{R}^m$ Gitterbasisvektoren mit $\|b_1\| \leq \dots \leq \|b_n\|$. Die Vektoren b_1, \dots, b_n heißen paarweise reduziert, falls $\|b_i\| \leq \|b_i \pm b_j\|$ für $1 \leq j < i \leq n$.

Bei einer paarweise reduzierten Basis führen also Addition oder Subtraktion zweier Basisvektoren zu keiner weiteren Verkürzung.

Lemma 3.1.2 Seien $b_1, \dots, b_n \in \mathbb{R}^m$ Gitterbasisvektoren und $\lambda_{ij} := \frac{\langle b_i, b_j \rangle}{\|b_j\|^2}$ für $1 \leq j < i \leq n$.

1. b_1, \dots, b_n sind genau dann paarweise reduziert, wenn $|\lambda_{ij}| \leq \frac{1}{2}$ für alle $1 \leq j < i \leq n$.
2. Mit $\tilde{b}_i := b_i - \lceil \lambda_{ij} \rceil b_j$ folgt $\|\tilde{b}_i\| \leq \|b_i \pm b_j\|$.

Beweis.

1. Wegen $\|b_i \pm b_j\|^2 = \langle b_i \pm b_j, b_i \pm b_j \rangle = \|b_i\|^2 \pm 2\langle b_i, b_j \rangle + \|b_j\|^2$ gilt die Äquivalenz $\|b_i\|^2 \leq \|b_i \pm b_j\|^2 \Leftrightarrow \pm 2\langle b_i, b_j \rangle + \|b_j\|^2 \geq 0 \Leftrightarrow |\lambda_{ij}| = \frac{|\langle b_i, b_j \rangle|}{\|b_j\|^2} \leq \frac{1}{2}$.
2. Es ist $\frac{|\langle \tilde{b}_i, b_j \rangle|}{\|b_j\|^2} = \frac{|\langle b_i - \lceil \lambda_{ij} \rceil b_j, b_j \rangle|}{\|b_j\|^2} = \frac{|\langle b_i, b_j \rangle - \lceil \lambda_{ij} \rceil \langle b_j, b_j \rangle|}{\|b_j\|^2} = \left| \frac{\langle b_i, b_j \rangle}{\|b_j\|^2} - \lceil \lambda_{ij} \rceil \right| \leq \frac{1}{2}$ und unter Benutzung von 1. folgt die Behauptung.

□

Man erhält somit einen einfachen Reduktionsalgorithmus, indem man paarweise Vektoren b_i, b_j vergleicht und den (größeren) Vektor b_i durch $\tilde{b}_i := b_i - \lceil \lambda_{ij} \rceil b_j$ ersetzt.

Algorithmus (paarweise Reduktion)EINGABE: Basisvektoren $b_1, \dots, b_n \in \mathbb{R}^m$

1. Ordne die Basisvektoren so, dass $\|b_1\| \leq \dots \leq \|b_n\|$
2. FOR $i = 2, \dots, n$ DO
 - FOR $j = 1, \dots, i - 1$ DO

$$\lambda_{ij} := \frac{\langle b_i, b_j \rangle}{\|b_j\|^2}$$
 IF $|\lambda_{ij}| > \frac{1}{2}$ THEN

$$b_i := b_i - \lceil \lambda_{ij} \rceil b_j$$
 GOTO 1.

AUSGABE: paarweise reduzierte Basisvektoren $b_1, \dots, b_n \in \mathbb{R}^m$

Beweis der Korrektheit. Da in jeder Iteration b_i, b_j linear unabhängig sind und $\lceil \lambda_{ij} \rceil$ ganzzahlig, ist \tilde{b}_i ein Gittervektor und ebenfalls linear unabhängig zu b_j . Die Ersetzung ändert also nicht das Gitter und bewirkt eine echte Verkürzung der Basisvektoren. Da das Gitter diskret ist, terminiert der Algorithmus irgendwann. Es existiert also immer eine paarweise reduzierte Basis. \square

Beispiel 3.1.3 Wir gehen wieder von den Basisvektoren $b_1 = (-1, 3, -5)$, $b_2 = (3, 7, -2)$ und $b_3 = (0, 7, 7)$ aus. Es gilt $\|b_1\| \leq \|b_2\| \leq \|b_3\|$, sodass wir die paarweise Reduktion durchführen können.

Im ersten Schritt bekommen wir ähnlich dem GRAM-SCHMIDT-Verfahren die beiden Werte $\|(-1, 3, -5)\|^2 = 35$ und $\langle (3, 7, -2), (-1, 3, -5) \rangle = 28$, woraus sich der Koeffizient $\lambda_{21} = \frac{28}{35} = \frac{4}{5}$ ergibt. Wegen $\lambda_{21} > \frac{1}{2}$ kann eine Verkürzung durchgeführt werden:

$$b_2 = (3, 7, -2) - 1 \cdot (-1, 3, -5) = (4, 4, 3).$$

Nach Anpassen der Basis (die Reihenfolge der Vektoren bleibt unverändert) wird wieder neu begonnen und λ_{21} , λ_{31} , λ_{32} werden berechnet:

$$\begin{aligned} \langle (4, 4, 3), (-1, 3, -5) \rangle &= -7 \Rightarrow \lambda_{21} = \frac{-7}{35} = -\frac{1}{5} \Rightarrow |\lambda_{21}| \leq \frac{1}{2}, \\ \langle (0, 7, 7), (-1, 3, -5) \rangle &= -14 \Rightarrow \lambda_{31} = \frac{-14}{35} = -\frac{2}{5} \Rightarrow |\lambda_{31}| \leq \frac{1}{2}. \end{aligned}$$

Bestimmen von λ_{32} : $\|(4, 4, 3)\|^2 = 41$, $\langle (0, 7, 7), (4, 4, 3) \rangle = 49$, also $\lambda_{32} = \frac{49}{41} > 1$. Das ermöglicht die erneute Reduktion

$$b_3 = (0, 7, 7) - 1 \cdot (4, 4, 3) = (-4, 3, 4)$$

Ein weiterer vollständiger Durchlauf des Verfahrens bringt aber keine Veränderung der Vektoren mehr, sodass wir mit

$$b_1 = (-1, 3, -5), \quad b_2 = (4, 4, 3), \quad b_3 = (-4, 3, 4)$$

eine paarweise reduzierte Basis erhalten haben.

Es kann durchaus vorkommen, dass eine paarweise reduzierte Basis nicht „maximal“ reduziert ist, d. h. einer der Basisvektoren könnte durch einen kürzeren ersetzt werden. Bereits für die Gitterdimension 3 finden sich einfache Beispiele. Zum Beispiel ist die Basis

$$B = \begin{pmatrix} 9 & -3 & -3 \\ 0 & 8 & -5 \\ 0 & 0 & 6 \end{pmatrix}$$

paarweise reduziert. Die Addition aller drei Basisvektoren ergibt aber den reduzierten Vektor $b' = (3, 3, 6)$, welcher kürzer ist als jeder der Ausgangsvektoren. Dieser kann jeden der drei Vektoren in der Basis ersetzen.

Bemerkung 3.1.4 *Die Paarreduktion ist nur eine sehr schwache Reduktion. In [Spra94] wird gezeigt, dass zwischen der Länge des kürzesten Vektors einer paarweise reduzierten Basis und der Länge des kürzesten Gittervektors beliebige Größenunterschiede herrschen können. Es gibt also keine reelle Konstante c_n mit der Eigenschaft*

$$\|b_1\| \leq c_n \lambda_1(L).$$

Des Weiteren ist nicht klar, ob die Paarreduktion ein Polynomialzeitalgorithmus ist. An gleicher Stelle wird als Beispiel die Basismatrix

$$B_k = \begin{pmatrix} -1 & 0 & \cdots & 0 & 0 & 2^k \\ 0 & -1 & \cdots & 0 & \vdots & 2^k \\ 0 & 0 & -1 & 0 & \vdots & \vdots \\ \vdots & \vdots & 0 & \ddots & 0 & \vdots \\ 1 & 1 & \cdots & 1 & -1 & 2^k \\ 0 & 0 & \cdots & 0 & 1 & 0 \end{pmatrix}$$

angegeben, für die der Algorithmus vermutlich weder im Gitterrang n noch in der Bitlänge k der Einträge polynomialzeitbeschränkt ist.

3.2 Längenreduktion

Bei der Längenreduktion versucht man die Basisvektoren so zu modifizieren, dass die Beträge der GRAM-SCHMIDT-Koeffizienten μ_{ij} zwischen $-\frac{1}{2}$ und $\frac{1}{2}$ liegen. Die so erzeugte Basis approximiert also ihre zugehörige GRAM-SCHMIDT-Basis.

Definition 3.2.1 (längenreduzierte Basis) *Die Basisvektoren b_1, \dots, b_n des Gitters $L \subset \mathbb{R}^m$ heißen längenreduziert, falls*

$$|\mu_{ij}| \leq \frac{1}{2} \quad \text{für } 1 \leq j < i \leq n.$$

Das führt zu folgender Idee: Für jedes $1 \leq i \leq n$ werden (aufsteigend) der Orthogonalvektor \hat{b}_i berechnet und b_i durch eine Näherung ähnlich wie bei der Paarreduktion ersetzt. Das geschieht mit Hilfe aller vorher berechneten b_j , für welche

3 Einfache Reduktionsverfahren

jeweils eine Anpassung von b_i vorgenommen wird. Die zugehörigen GRAM-SCHMIDT-Koeffizienten werden dabei laufend angepasst.

Algorithmus (Längenreduktion)

EINGABE: Basisvektoren $b_1, \dots, b_n \in \mathbb{R}^m$

1. Ordne die Basisvektoren so, dass $\|b_1\| \leq \dots \leq \|b_n\|$.
2. $\hat{b}_1 := b_1$
3. FOR $i = 2, \dots, n$ DO
 FOR $j = 1, \dots, i - 1$ DO
 $\mu_{ij} := \frac{\langle b_i, \hat{b}_j \rangle}{\|\hat{b}_j\|^2}$
 $\hat{b}_i := b_i - \sum_{j=1}^{i-1} \mu_{ij} \hat{b}_j$
 FOR $j = i - 1, \dots, 1$ DO
 $b_i := b_i - \lceil \mu_{ij} \rceil b_j$
 FOR $k = 1, \dots, j$ DO
 $\mu_{ik} := \mu_{ik} - \lceil \mu_{ij} \rceil \mu_{jk}$

AUSGABE: längenreduzierte Basisvektoren $b_1, \dots, b_n \in \mathbb{R}^m$

Beweis der Korrektheit. Die Substitution $b_i := b_i - \lceil \mu_{ij} \rceil b_j$ erzeugt einen kürzeren Gittervektor und erhält die lineare Unabhängigkeit. Für das neue μ_{ik} muss gelten

$$\mu_{ik} = \frac{\langle b_i - \lceil \mu_{ij} \rceil b_j, \hat{b}_k \rangle}{\|\hat{b}_k\|^2} = \frac{\langle b_i, \hat{b}_k \rangle}{\|\hat{b}_k\|^2} - \lceil \mu_{ij} \rceil \frac{\langle b_j, \hat{b}_k \rangle}{\|\hat{b}_k\|^2} = \mu_{ik} - \lceil \mu_{ij} \rceil \mu_{jk},$$

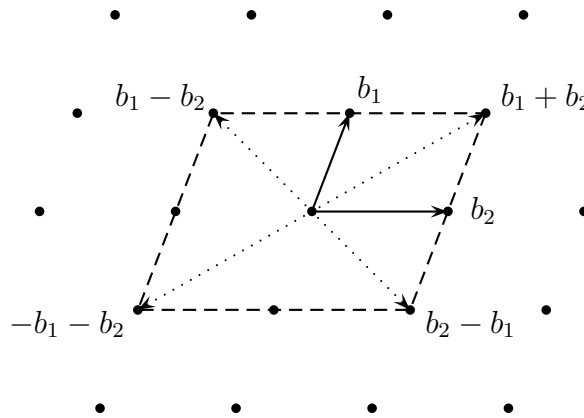
was genau dem im Algorithmus vorgenommenen Update entspricht. Diese Anpassung wird für alle GRAM-SCHMIDT-Koeffizienten μ_{ij} vorgenommen und bewirkt, dass $|\mu_{ij}| \leq \frac{1}{2}$. \square

3.3 GAUSS-Reduktion

Wie bereits erwähnt, existiert für Gitter mit $\text{Rang} \leq 4$ eine Gitterbasis, die die sukzessiven Minima realisiert. Für Gitter vom Rang 2 gab bereits GAUSS einen Reduktionsalgorithmus an, der eine solche Basis in Polynomialzeit bestimmt.

Definition 3.3.1 (GAUSS-reduzierte Basis) Eine Gitterbasis $\{b_1, b_2\}$ heißt GAUSS-reduziert, falls $\|b_1\| \leq \|b_2\|$ und $0 \leq \mu_{21} \leq \frac{1}{2}$.

Eine GAUSS-reduzierte Basis ist also nach Definition längenreduziert. Aufgrund von Lemma 3.1.2 ist sie ebenfalls paarweise reduziert (man beachte hierbei die Gleichheit $\mu_{21} = \lambda_{21}$). Für ihre Basisvektoren b_1, b_2 gilt also $\|b_1\| \leq \|b_2\| \leq \|b_1 \pm b_2\|$.



GAUSS-reduzierte Basen sind paarweise reduziert

Durch die Bedingung $\mu_{21} \geq 0$ ist sichergestellt, dass $\langle b_1, b_2 \rangle \geq 0$ (das bedeutet, der Winkel zwischen b_1 und b_2 ist spitz, siehe Abbildung). Somit gilt sogar die Ungleichung $\|b_1\| \leq \|b_2\| \leq \|b_1 - b_2\| \leq \|b_1 + b_2\|$.

Satz 3.3.2 Sei $\{b_1, b_2\}$ eine GAUSS-reduzierte Basis für $L(b_1, b_2) \subset \mathbb{R}^m$. Dann gilt

$$\|b_i\| = \lambda_i(L) \quad \text{für } i = 1, 2.$$

Beweis. b_1 und b_2 sind zwei linear unabhängige kürzeste Gittervektoren. Gezeigt werden müssen also die Ungleichungen

$$\begin{aligned} \|b_1\| &\leq \|z_1 b_1 + z_2 b_2\| \quad \text{für alle } (z_1, z_2) \in \mathbb{Z}^2 - \{(0, 0)\}, \\ \|b_2\| &\leq \|z_1 b_1 + z_2 b_2\| \quad \text{für alle } (z_1, z_2) \in \mathbb{Z}^2, z_2 \neq 0. \end{aligned}$$

Diese ergeben sich direkt aus den Ungleichungen

$$\begin{aligned} \|b_1\| &\leq \|b_2\|, \\ \|b_1\| &\leq \|z_1 b_1\| \quad \text{für alle } z_1 \in \mathbb{Z} - \{0\}, \\ \|b_2\| &\leq \|x_1 b_1 + x_2 b_2\| \quad \text{für alle } x_1, x_2 \in \mathbb{R} \text{ mit } |x_1|, |x_2| \geq 1. \end{aligned}$$

Um die Korrektheit der letzten Ungleichung zu sehen, betrachten wir obige Abbildung. Alle reellen Vektoren der Form $\|x_1 b_1 + x_2 b_2\|$, $|x_1|, |x_2| \geq 1$ befinden sich innerhalb der vier gepunkteten Gebiete. Die jeweils kürzesten Vektoren dieser Flächen sind von der Form $\pm b_1 \pm b_2$. Aufgrund der Reduktionseigenschaft folgt sofort $\|b_2\| \leq \|\pm b_1 \pm b_2\| \leq \|x_1 b_1 + x_2 b_2\|$, also die Behauptung. \square

Wir können nun einen Algorithmus angeben, der zu einem beliebigen Gitter mit Rang 2 zwei linear unabhängige kürzeste Vektoren liefert.

Algorithmus (GAUSS-Reduktion)EINGABE: Basisvektoren $b_1, b_2 \in \mathbb{R}^m$

1. $b_2 := b_2 - \lceil \mu_{21} \rceil b_1$
2. IF $\mu_{21} < 0$ THEN
 $b_2 := -b_2$
3. IF $\|b_1\| > \|b_2\|$ THEN
 Vertausche b_1 und b_2 .
 GOTO 1.

AUSGABE: GAUSS-reduzierte Basisvektoren $b_1, \dots, b_n \in \mathbb{R}^m$

Beweis der Korrektheit. Am Ende von Schritt 2 gilt die geforderte Bedingung $0 \leq \mu_{21} \leq \frac{1}{2}$. Schritt 3 stellt dabei sicher, dass beide Vektoren geordnet sind. Da in Schritt 1 eine echte Verkürzung vorgenommen wird, terminiert der Algorithmus; im Ergebnis bekommt man also die gewünschte GAUSS-reduzierte Basis. \square

Beispiel 3.3.3 Gegeben sind die beiden Basisvektoren $b_1 = (3, 12, 1)$ und $b_2 = (-10, -19, 5)$. Im ersten Schritt der GAUSS-Reduktion erhalten wir den Koeffizienten $\mu_{21} = -\frac{23}{14}$, d. h. $\lceil \mu_{21} \rceil = -2$. Das führt zum Austausch $b_2 = b_2 - 2b_1 = (-10, -19, 5) + 2(3, 12, 1) = (-4, 5, 7)$. Da μ_{21} negativ ist, wechseln wir das Vorzeichen und bekommen

$$b_2 = (4, -5, -7).$$

Im zweiten Schritt besitzt μ_{21} den Wert $-\frac{5}{14}$. Eine Verkürzung ist also nicht mehr möglich – allerdings wechselt b_2 wieder das Vorzeichen. Im dritten Schritt werden folglich keine Änderungen mehr vorgenommen; die GAUSS-reduzierte Basis lautet also

$$b_1 = (3, 12, 1), \quad b_2 = (-4, 5, 7).$$

3.4 L^3 -Reduktion

Die L^3 -Reduktion (oder auch LLL-Reduktion) wurde 1982 von A. K. Lenstra, H. W. Lenstra und L. Lovász in [LLL82] eingeführt. Die ursprünglich vorgesehene Anwendung dieses Verfahrens bestand in der Zerlegung von Polynomen in irreduzible Faktoren. Man kann es aber ebenfalls zur Gitterbasisreduktion verwenden. Das Verfahren approximiert die sukzessiven Minima eines Gitters und erzeugt in Polynomialzeit eine reduzierte Gitterbasis, die bis auf einen Faktor optimal ist. Der Faktor hängt allerdings exponentiell vom Rang des Gitters ab. Das Verfahren kann als Mischung von Längen- und GAUSS-Reduktion aufgefasst werden.

Die L^3 -Reduktion hat vielfältige Anwendungen in der Zahlentheorie. Neben typischen Einsätzen bei Polynomfaktorisierung und linearer Programmierung gibt es

auch andere interessante Ergebnisse. So wird beispielsweise in [BoDu00] beschrieben, wie sich RSA-Entschlüsselungsexponenten $d < n^{0.292}$ effizient durch L^3 -Reduktion brechen lassen.

Definition 3.4.1 (L^3 -reduzierte Basis) Eine Gitterbasis $\{b_1, \dots, b_n\}$ von $L \subset \mathbb{R}^m$ heißt L^3 -reduziert mit δ (wobei $\frac{1}{4} < \delta \leq 1$), wenn

- $\|b_1\| \leq \dots \leq \|b_n\|$ (Ordnung)
- $|\mu_{ij}| \leq \frac{1}{2}$ für $1 \leq j < i \leq n$ (Längenreduziertheit)
- $\delta \|\hat{b}_{k-1}\|^2 \leq \|\hat{b}_k\|^2 + \mu_{k,k-1}^2 \|\hat{b}_{k-1}\|^2$ für $k = 2, \dots, n$ (L^3 -Eigenschaft)

Eine L^3 -reduzierte Basis ist also geordnet und längenreduziert, wobei δ ein Maß für die Güte der Reduktion darstellt; je größer δ , desto besser ist die Reduktion (desto länger aber auch die Laufzeit des Verfahrens).

Lemma 3.4.2 Ist die Gitterbasis $\{b_1, \dots, b_n\}$ L^3 -reduziert mit δ , dann gilt für $\alpha := (\delta - \frac{1}{4})^{-1}$ die Ungleichung

$$\|\hat{b}_j\|^2 \leq \alpha^{i-j} \|\hat{b}_i\|^2 \quad \text{für } 1 \leq j < i \leq n.$$

Beweis. Der Beweis erfolgt induktiv: Sei $j = i - 1$ für ein $2 < i \leq n$, dann gilt aufgrund der Definition

$$\delta \|\hat{b}_{i-1}\|^2 \leq \|\hat{b}_i\|^2 + \mu_{i,i-1}^2 \|\hat{b}_{i-1}\|^2 \leq \|\hat{b}_i\|^2 + \frac{1}{4} \|\hat{b}_{i-1}\|^2$$

und damit $(\delta - \frac{1}{4}) \|\hat{b}_{i-1}\|^2 \leq \|\hat{b}_i\|^2$ bzw. $\|\hat{b}_{i-1}\|^2 \leq \alpha \|\hat{b}_i\|^2$. Induktion über $i - j$ liefert dann die Behauptung. \square

Satz 3.4.3 Mit $\alpha := (\delta - \frac{1}{4})^{-1}$ gelten die Schranken

$$\frac{\|\hat{b}_i\|}{\lambda_i(L)} \leq \alpha^{(n-i)/2}, \quad 1 \leq i \leq n.$$

Das liefert eine Schranke für alle sukzessiven Minima, insbesondere für die Länge des kürzesten Gittervektors.

Beweis. Offensichtlich gilt $\|\hat{b}_i\|^2 \leq \alpha^{n-i} \|\hat{b}_j\|^2$ für alle $i \leq j \leq n$. Nach Lemma 3.0.13 gilt ebenfalls $\lambda_i(L) \geq \min_{i \leq j \leq n} \|\hat{b}_j\|$. Daraus folgt

$$\lambda_i(L) \geq \min_{i \leq j \leq n} \|\hat{b}_j\| \geq \alpha^{(n-i)/2} \|\hat{b}_i\|$$

und daraus die Behauptung

$$\frac{\|\hat{b}_i\|}{\lambda_i(L)} \leq \alpha^{(n-i)/2}.$$

\square

Nachstehend ist der originale L^3 -Algorithmus abgebildet. Mittlerweile existieren verschiedene Varianten und Erweiterungen desselben.

Algorithmus (L^3 -Reduktion)

EINGABE: Basisvektoren $b_1, \dots, b_n \in \mathbb{R}^m$ und ein $\frac{1}{4} < \delta \leq 1$

1. $k := 2, \hat{b}_1 := b_1$
2. WHILE $k \leq n$ DO
 - Berechne μ_{kj} und \hat{b}_k für $j = 1, \dots, k$
 - Längenreduziere b_k ($\mu_{k,1}, \dots, \mu_{k,k-1}$ werden dabei aktualisiert)
 - IF $\delta \|\hat{b}_{k-1}\|^2 > \|b_k\|^2 + \mu_{k,k-1}^2 \|\hat{b}_{k-1}\|^2$ THEN
 - Vertausche b_k und b_{k-1}
 - IF $k = 2$ THEN
 - Aktualisiere \hat{b}_1
 - $k := \max\{k - 1, 2\}$
 - ELSE $k := k + 1$

AUSGABE: L^3 -reduzierte Basisvektoren $b_1, \dots, b_n \in \mathbb{R}^m$

Beweis der Korrektheit. Man kann induktiv zeigen, dass im Schritt k die Basisvektoren b_1, \dots, b_{k-1} immer L^3 -reduziert sind. Der Fall $k = 1$ ist trivial; sind bereits b_1, \dots, b_{k-1} L^3 -reduziert, so wird für b_k eine Längenreduktion durchgeführt. Die IF-Bedingung garantiert hierbei die dritte L^3 -Eigenschaft von b_k . Ist die Bedingung nicht erfüllt, so wird durch die Zuweisung $k := \max\{k - 1, 2\}$ k zurückgesetzt. Gleichzeitig ist dadurch sichergestellt, dass die Ordnung der Basisvektoren erhalten bleibt. Damit sind auch die Vektoren b_1, \dots, b_k L^3 -reduziert. \square

Lenstra, Lenstra und Lovász konnten zeigen, dass dieser Algorithmus eine polynomielle Laufzeit besitzt und die Längen der L^3 -reduzierten Basisvektoren dieselben oberen Schranken besitzen wie die Orthogonalvektoren aus Satz 3.4.3.

Satz 3.4.4 Sind b_1, \dots, b_n L^3 -reduzierte Basisvektoren mit δ , so gelten mit $\alpha := (\delta - \frac{1}{4})^{-1}$ die Schranken

$$\frac{\|b_i\|}{\lambda_i(L)} \leq \alpha^{(n-i)/2}, \quad 1 \leq i \leq n.$$

Beweis. Zunächst wird die Ungleichung $\alpha^{1-j} \leq \|\hat{b}_i\|^2 \lambda_i(L)^{-2}$ für alle $i \leq n$ gezeigt. Dazu betrachtet man ein $1 \leq k \leq i$ mit der Eigenschaft $\lambda_i \leq \|b_k\|$:

$$\begin{aligned} \lambda_i^2 &\leq \|b_k\|^2 \leq \|\hat{b}_k\|^2 + \frac{1}{4} \sum_{j=1}^{k-1} \|\hat{b}_j\|^2 \\ &\leq \|\hat{b}_i\|^2 (\alpha^{i-k} + \frac{1}{4} \sum_{j=1}^{k-1} \alpha^{i-j}) \quad (\text{siehe Lemma 3.4.2}) \end{aligned}$$

Damit gilt die Ungleichung $\|b_k\|^2 \leq \alpha^{i-1} \|\hat{b}_i\|^2$ für alle $k \leq i$. Es bleibt zu zeigen, dass

$$\alpha^{i-k} + \frac{1}{4} \sum_{j=1}^{k-1} \alpha^{i-j} \leq 1$$

gilt. Die Richtigkeit für $k = 1$ ist offensichtlich. Ist dagegen $k \geq 2$, so bekommt man mit Hilfe von $\alpha^{-1} = \delta - \frac{1}{4} \leq \frac{3}{4}$ und

$$\alpha^{1-k} + \frac{1}{4} \sum_{j=1}^{k-1} \alpha^{1-j} \leq \left(\frac{3}{4}\right)^{k-1} + \frac{1}{4} \frac{1 - \left(\frac{3}{4}\right)^{k-1}}{1 - \frac{3}{4}} = \frac{1}{4} \frac{1}{1 - \frac{3}{4}} = 1$$

ebenfalls das gewünschte Ergebnis. Nach Lemma 3.0.13 gibt es nun ein $k \geq i$ mit der Eigenschaft $\lambda_i \geq \|\hat{b}_k\|$. Es folgt

$$\begin{aligned} \lambda_i^2 &\geq \|\hat{b}_k\|^2 \geq \alpha^{-k+i} \|\hat{b}_i\|^2 \quad (\text{siehe Lemma 3.4.2}) \\ &\geq \alpha^{-k+1} \|\hat{b}_i\|^2 \quad (\text{oben gezeigt}) \\ &\geq \alpha^{-n+1} \|\hat{b}_i\|^2 \quad (\text{wegen } k \leq n \text{ und } \alpha \geq 1) \end{aligned}$$

□

Um zu zeigen, dass das L^3 -Reduktionsverfahren mit polynomiellm Aufwand auskommt, ist erheblich mehr Aufwand nötig. Dabei gehen wir von der Grundannahme aus, dass wir ganzzahlige Gitterbasisvektoren vor uns haben (in praktischen Anwendungen werden prinzipiell ganzzahlige Gitterbasen verwendet). Die GRAM-SCHMIDT-Koeffizienten μ_{ij} und die Längen der GRAM-SCHMIDT-Vektoren besitzen dadurch rationale Werte und können als ganzzahlige Brüche dargestellt werden.

Satz 3.4.5 *Sind $b_1, \dots, b_n \in \mathbb{Z}^m$ ganzzahlige Gitterbasisvektoren und ist ferner $M := \max_{1 \leq i \leq n} \{\|b_i\|^2, \det L(b_1, \dots, b_i)^2\}$, so führt das L^3 -Verfahren $\mathcal{O}(n^2 m \log_{1/\delta} M)$ Operationen auf den Vektoren b_i und den rationalen Zahlen $\mu_{ij}, \|\hat{b}_i\|^2$ durch. Werden $\mu_{ij}, \|\hat{b}_i\|^2$ als Brüche dargestellt, so ist die Bitlänge aller im Algorithmus verwendeten Ganzzahlen durch $\mathcal{O}(n + \log_2 M)$ beschränkt.*

Beweis. Einen vollständigen Beweis kann man in [Schn06] nachlesen. □

Gleichzeitig haben wir damit ein wichtiges theoretisches Ergebnis erhalten: Der kürzeste, von 0 verschiedene Gittervektor lässt sich in Polynomialzeit bis auf einen Faktor $(2/\sqrt{3})^n$ approximieren. Das erreichen wir durch L^3 -Reduktion mit $\delta := 1 - \varepsilon_n$ für hinreichend kleine $\varepsilon_n > 0$ und Ausgabe von b_1 .

Beispiel 3.4.6 *Um die Arbeitsweise transparent zu halten wird ein relativ einfaches Beispiel gewählt. Wir betrachten ein ganzzahliges Gitter mit den Basisvektoren $b_1 = (12, 2)$ und $b_2 = (13, 4)$. Offensichtlich zeigen beide Vektoren in fast dieselbe „Richtung“ und werden sich deshalb vermutlich gut reduzieren lassen. Ferner wählen wir $\delta = \frac{3}{4}$, was auch eine übliche Wahl darstellt.*

Das zunächst berechnete GRAM-SCHMIDT-System besitzt die Vektoren

$$\hat{b}_1 = (12, 2), \quad \hat{b}_2 = \left(-\frac{11}{37}, \frac{66}{37}\right).$$

3 Einfache Reduktionsverfahren

Der nachfolgende Längenreduktionsschritt liefert $(1, 2)$ als neuen Vektor b_2 . Nach dem Austauschschritt haben wir

$$b_1 = (1, 2) \text{ und } b_2 = (12, 2).$$

Im nächsten Iterationsschritt der WHILE-Schleife muss die GRAM-SCHMIDT-Basis aktualisiert werden. Dadurch bekommen wir

$$\hat{b}_1 = (1, 2), \quad \hat{b}_2 = \left(\frac{44}{5}, -\frac{22}{5}\right).$$

Im zweiten Längenreduktionsschritt wird b_2 erneut reduziert, wir erhalten $(9, -4)$ als neuen Wert. Da die Abbruchbedingung mit dem selbstgewählten δ erfüllt ist, stoppt der Algorithmus und es liegt im Ergebnis die L^3 -reduzierte Gitterbasis

$$b_1 = (1, 2), \quad b_2 = (9, -4)$$

vor.

Bei der praktischen Implementation des L^3 -Algorithmus' steht man vor dem Problem der Rechengenauigkeit. Durch die GRAM-SCHMIDT-Orthogonalisierung kommen rationale Zahlen ins Spiel, die man in der Praxis nicht exakt darstellen kann (man beachte, dass das formale Rechnen mit Brüchen a/b nicht praktikabel ist, da die Länge der Zahlen a und b exponentiell in der Anzahl der Rechenoperationen anwächst). Die Verwendung von Fließkommazahlen führt auf der anderen Seite aufgrund der unvermeidbaren Rundungsfehler zu Auslöschungseffekten und lässt das Verfahren numerisch instabil werden. Um diesem Problem zu begegnen wurden Erweiterungen des Verfahrens insbesondere von SCHNORR entwickelt. Eine solche wird etwa in [SchnEu94] beschrieben.

Wie beschrieben kann mit Hilfe des L^3 -Verfahrens in Polynomialzeit ein Gittervektor gefunden werden, der im schlechtesten Fall um einen Faktor $(2/\sqrt{3})^n$ länger ist als der kürzeste. Es hat sich aber herausgestellt, dass diese worst-case-Aussage in der Praxis deutlich übertroffen wird. Selbst bei hohen Gitterdimensionen (z. B. $n \approx 100$) werden viel bessere Ergebnisse erzielt. Ferner konnte der L^3 -Algorithmus durch Modifikationen verbessert werden (siehe z. B. [Schn87]). Neben Varianten desselben wurden aber auch neuere Verfahren entwickelt (hervorzuheben ist hier vielleicht die (Block-)KORKINE-ZOLOTAREFF-Reduktion); Arbeiten dazu stammen von SCHNORR, SEYSEN, LOVÁSZ und SCARF.

4 Anwendungen der Gitterreduktion

In diesem Abschnitt sollen zwei nützliche Anwendungen der Gitterreduktion besprochen werden. Die erste ist ein wichtiges theoretisches Resultat von L. BABAI [Bab86], welches die Approximierbarkeit des CVP bis auf einen konstanten Faktor (der wiederum exponentiell in der Gitterdimension ist) liefert. Die von BABAI vorgestellte Round-Off-Methode spielt außerdem eine wichtige Rolle für die im nächsten Kapitel vorgestellten Kryptosysteme.

Eine weitere Anwendung der Gitterbasenreduktion liegt in der Faktorisierung ganzer Zahlen. Dieses Problem spielt bei Public-Key-Kryptosystemen eine wesentliche Rolle und ist bis heute Gegenstand der Forschung. Vorgestellt werden soll eine Reduktion des Faktorisierungsproblems auf ein Gitterproblem.

4.1 BABAI's Round-Off-Methode

Eine naheliegende Methode, eine Näherungslösung für das CVP zu bestimmen, hat L. BABAI 1986 vorgeschlagen und analysiert. Gegeben sei eine Gitterbasis $B = [b_1, \dots, b_n]$ und ein Vektor $x \in \mathbb{R}^n$, $u \in L(B)$ sei der zu x nächste Gitterpunkt. Die Idee besteht darin, das Gleichungssystem

$$Bz = x, \quad z \in \mathbb{R}^n$$

im Reellen zu lösen und die Vektoreinträge z_i auf die jeweils nächste ganze Zahl zu runden. Dann ist

$$w = B[z] = B[B^{-1}x]$$

ein Gitterpunkt, der (möglicherweise) eine gute Approximation der tatsächlichen Lösung $u \in L(B)$ darstellt.

Die Güte dieser Round-Off-Methode hängt von der Reduziertheit der Gitterbasis B ab. An dieser Stelle können wir die Ergebnisse aus dem letzten Kapitel verwenden, um zu zeigen, dass sich u auf einen Faktor, der exponentiell in der Gitterdimension n ist, approximieren lässt, falls B eine L^3 -reduzierte Gitterbasis ist. Damit bekommen wir analog zum SVP auch für das CVP eine Aussage über dessen Approximierbarkeit.

Nach einigen Vorbereitungen wird das Resultat in Satz 4.1.3 vorgestellt.

Lemma 4.1.1 *Sei $B = [b_1, \dots, b_n]$ eine L^3 -reduzierte Gitterbasis. Für alle $1 \leq k \leq n$ bezeichne U_k den Unterraum $U_k := \text{span}(b_1, \dots, b_{k-1}, b_{k+1}, \dots, b_n)$. Dann gilt mit $c_n := \left(\frac{3}{\sqrt{2}}\right)^n$ die Ungleichung*

$$\|b_k\| \leq c_n \|v - b_k\|$$

für alle $v \in U_k$.

4 Anwendungen der Gitterreduktion

Beweis. Mit der zu B zugehörigen GRAM-SCHMIDT-Basis $\hat{B} = [\hat{b}_1, \dots, \hat{b}_n]$ gilt

$$v = \sum_{\substack{i=1 \\ i \neq k}}^n \alpha_i b_i = \sum_{\substack{i=1 \\ i \neq k}}^n \alpha_i \sum_{j=1}^n \mu_{ij} \hat{b}_j = \sum_{j=1}^n \hat{b}_j \underbrace{\sum_{\substack{i=1 \\ i \neq k}}^n \alpha_i \mu_{ij}}_{=: \beta_j} = \sum_{j=1}^n \beta_j \hat{b}_j,$$

wobei α_i, β_j reelle Koeffizienten sind. Für den Differenzvektor $v - b_k$ bekommt man die Darstellung

$$v - b_k = \sum_{j=1}^n \underbrace{(\beta_j - \mu_{kj})}_{=: \gamma_j} \hat{b}_j = \sum_{j=1}^n \gamma_j \hat{b}_j.$$

Ab hier setzen wir $\alpha_k := -1$; damit gilt

$$\gamma_j = \beta_j - \mu_{kj} = \sum_{i=1}^n \alpha_i \mu_{ij} = \alpha_j + \sum_{i=j+1}^n \alpha_i \mu_{ij}$$

und somit

$$\alpha_j = \gamma_j - \sum_{i=j+1}^n \alpha_i \mu_{ij}.$$

Da die GRAM-SCHMIDT-Vektoren paarweise orthogonal sind, gilt $\|\sum_{j=1}^n \gamma_j \hat{b}_j\|^2 = \sum_{j=1}^n \gamma_j^2 \|\hat{b}_j\|^2$; damit folgt aus der Behauptung die zu zeigende Ungleichung

$$\sum_{j=1}^k \mu_{kj}^2 \|\hat{b}_j\|^2 \leq c_n^2 \sum_{j=1}^n \gamma_j^2 \|\hat{b}_j\|^2. \quad (4.1)$$

Wir zeigen zunächst die Eigenschaft

$$\sum_{j=k}^n \gamma_j^2 \geq \left(\frac{4}{9}\right)^{n-k}. \quad (4.2)$$

Unter Annahme des Gegenteils folgt $|\gamma_j| < \varepsilon$ mit $\varepsilon := \left(\frac{2}{3}\right)^{n-k}$ für $k \leq j \leq n$. Durch (Rückwärts-)Induktion über j bekommen wir die Schranke $|\alpha_j| < \left(\frac{3}{2}\right)^{n-j} \varepsilon$; Für $j = n$ stimmt das sicherlich wegen $|\alpha_n| = |\gamma_n| < \varepsilon$. Für $j < n$ bekommen wir induktiv

$$|\alpha_j| = \left| \gamma_j - \sum_{i=j+1}^n \alpha_i \mu_{ij} \right| \leq |\gamma_j| + \sum_{i=j+1}^n \frac{1}{2} |\alpha_i|,$$

wobei wir die Eigenschaft $|\mu_{ij}| \leq \frac{1}{2}$ für L^3 -reduzierte Basen benutzt haben. Die rechte Seite lässt sich jetzt mit Hilfe der Induktionsannahme nach oben abschätzen:

$$|\alpha_j| < \varepsilon + \sum_{i=j+1}^n \frac{1}{2} \cdot \left(\frac{3}{2}\right)^{n-i} \varepsilon = \left(\frac{3}{2}\right)^{n-j} \varepsilon.$$

Die Gleichheit folgt aus der Summenformel $\sum_{i=0}^k \left(\frac{3}{2}\right)^i = 2 \cdot \left(\frac{3}{2}\right)^{k+1} - 2$. Einsetzen von $j = k$ liefert den Widerspruch

$$1 = |\alpha_k| < \left(\frac{3}{2}\right)^{n-k} \varepsilon = 1.$$

Damit ist 4.2 gezeigt. Für 4.1 benutzen wir die in Lemma 3.4.2 gezeigte Eigenschaft $\|\hat{b}_i\| \geq \frac{1}{2}\sqrt{2} \cdot \|\hat{b}_{i-1}\|$ für L^3 -reduzierte Gitterbasen¹. Es folgt unmittelbar

$$\|\hat{b}_j\|^2 \leq 2^{k-j} \|\hat{b}_k\|^2, \quad j \leq k \quad \text{und} \quad \|\hat{b}_j\|^2 \geq 2^{k-j} \|\hat{b}_k\|^2, \quad j \geq k.$$

Damit bekommen wir die Schranken

$$\sum_{j=1}^k \mu_{kj}^2 \|\hat{b}_j\|^2 \leq \sum_{j=1}^k \mu_{kj}^2 2^{k-j} \|\hat{b}_k\|^2 < 2^k \|\hat{b}_k\|^2$$

sowie

$$\sum_{j=1}^n \gamma_j^2 \|\hat{b}_j\|^2 \geq \sum_{j=k}^n \gamma_j^2 \|\hat{b}_j\|^2 \geq \sum_{j=k}^n \gamma_j^2 2^{k-j} \|\hat{b}_k\|^2 \geq \|\hat{b}_k\|^2 2^{k-n} \sum_{j=k}^n \gamma_j^2 \stackrel{4.2}{\geq} \|\hat{b}_k\|^2 \left(\frac{2}{9}\right)^{n-k}.$$

Zusammen liefern beide Ungleichungen wegen

$$2^k \|\hat{b}_k\|^2 \leq \left(\frac{9}{2}\right)^k \|\hat{b}_k\|^2 = c_n^2 \|\hat{b}_k\|^2 \left(\frac{2}{9}\right)^{n-k}$$

die Ungleichung 4.1 und damit Lemma 4.1.1. \square

Als interessante Folgerung ergibt sich noch das folgende Korollar, das aber sonst nicht weiter relevant ist.

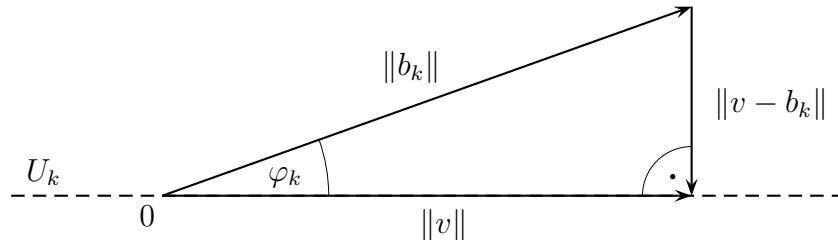
Korollar 4.1.2 Sei $B = [b_1, \dots, b_n]$ eine L^3 -reduzierte Gitterbasis. φ_k bezeichne den Winkel zwischen b_k und dem Unterraum $U_k := \text{span}(b_1, \dots, b_{k-1}, b_{k+1}, \dots, b_n)$. Dann gilt für alle $1 \leq k \leq n$ die Ungleichung

$$\sin \varphi_k \geq \left(\frac{\sqrt{2}}{3}\right)^n.$$

Beweis. Für den Winkel φ_k gilt

$$\sin \varphi_k = \min_{v \in U_k} \frac{\|v - b_k\|}{\|b_k\|},$$

was untenstehende Abbildung veranschaulicht.



¹Dort wurde sogar die schärfere Schranke $\|\hat{b}_i\| \geq \frac{1}{2}\sqrt{3} \cdot \|\hat{b}_{i-1}\|$ bewiesen.

4 Anwendungen der Gitterreduktion

Unter Anwendung von Lemma 4.1.1 folgt die Behauptung. \square

Das Lemma liefert uns nun die Mittel, um den durch die Round-Off-Methode gemachten Fehler abzuschätzen.

Satz 4.1.3 (Approximationssatz für das CVP) *Ist B eine L^3 -reduzierte Gitterbasis, dann liefert die Round-Off-Methode zu einem $x \in \mathbb{R}^n$ einen Gitterpunkt $w \in L(B)$ mit der Eigenschaft $\|x - w\| \leq C_n \|x - u\|$, wobei $C_n := 1 + 2n \left(\frac{3}{\sqrt{2}}\right)^n$ und u der zu x nächste Gitterpunkt ist.*

Beweis. Sei $B = [b_1, \dots, b_n]$ die Basismatrix und $1 \leq k \leq n$. Mit dem Unterraum $U_k := \text{span}(b_1, \dots, b_{k-1}, b_{k+1}, \dots, b_n)$ gilt nach Lemma 4.1.1 für alle $v \in U_k$ die Ungleichung

$$\|b_k\| \leq c_n \|v - b_k\|,$$

wobei $c_n := \left(\frac{3}{\sqrt{2}}\right)^n$. Zunächst gibt es für $1 \leq i \leq n$ reelle Koeffizienten $|\delta_i| \leq \frac{1}{2}$ und ganze Zahlen z_i mit

$$w - x = \sum_{i=1}^n \delta_i b_i \quad \text{und} \quad u - w = \sum_{i=1}^n z_i b_i.$$

Wir zeigen die folgende Ungleichung:

$$\|u - w\| \leq 2nc_n \|u - x\|. \quad (4.3)$$

Zum Beweis können wir von $u \neq w$ ausgehen. Sei $1 \leq k \leq n$ der Index, für den $\|z_k b_k\|$ maximal wird (woraus $z_k \neq 0$ folgt). Dann haben wir

$$\|u - w\| \leq \sum_{i=1}^n \|z_i b_i\| \leq n \|z_k b_k\|$$

und

$$u - x = (u - w) + (w - x) = \sum_{i=1}^n (z_i + \delta_i) b_i = (z_k + \delta_k)(b_k - v)$$

mit dem Vektor

$$v := -\frac{1}{z_k + \delta_k} \sum_{\substack{i=1 \\ i \neq k}}^n (z_i + \delta_i) b_i.$$

Offensichtlich ist $v \in U_k$, sodass wir Lemma 4.1.1 anwenden können und

$$\|u - x\| = |z_k + \delta_k| \cdot \|b_k - v\| \geq \frac{|z_k|}{2c_n} \|b_k\|$$

erhalten. Damit kommen wir zur Abschätzung

$$\|u - w\| \leq n |z_k| \|b_k\| \leq 2nc_n \|u - x\|,$$

also genau 4.3. Mit 4.3 folgt die Behauptung sofort; es ist

$$\|x - w\| \leq \|x - u\| + \|u - w\| \leq \|x - u\|(1 + 2nc_n).$$

□

Damit ist die Approximierbarkeit des CVP bis auf einen Faktor $1 + 2n \left(\frac{3}{\sqrt{2}}\right)^n$ gezeigt. SCHNORR konnte diesen Faktor auf $(1 + \varepsilon)^n$ für alle $\varepsilon > 0$ verbessern (siehe [Schn87]). Es ist aber noch kein Verfahren gefunden worden mit einem Approximationsfaktor polynomiell in n .

Die Rundungsmethode liefert um so schlechtere Resultate, je schlechter die Basismatrix B reduziert ist. Für den berechneten Gitterpunkt w gilt

$$w = B\lceil B^{-1}x \rceil = B(B^{-1}x + \delta) = x + B\delta,$$

wobei $|\delta_i| \leq \frac{1}{2}$. Ist B eine Basismatrix mit sehr großen Einträgen, so sind die Einträge in $B\delta$ ebenfalls groß. Gegenüber einer kürzeren Basis ist damit die Chance, den zu x nächsten Gitterpunkt zu finden, deutlich kleiner. Eine unwesentlich kleine Änderung der Einträge in x kann dann zu einem völlig anderen Gitterpunkt w führen. Diesen Effekt machen sich die im nächsten Kapitel eingeführten Kryptosysteme zu Nutze.

Zum Schluss soll die Rundungsmethode an einem einfachen Beispiel vorgeführt werden, um ein Gefühl für die Problematik zu vermitteln.

Beispiel 4.1.4 *Zu einem Gitter $L \subset \mathbb{Z}^3$ sind die beiden Basismatrizen*

$$A = \begin{pmatrix} 8 & 1 & -2 \\ 0 & 11 & -1 \\ -1 & -3 & 11 \end{pmatrix}, \quad B = \begin{pmatrix} 44 & 31 & -32 \\ -90 & -57 & 67 \\ 39 & 32 & -24 \end{pmatrix}$$

gegeben. A ist eine kurze, L^3 -reduzierte Basis, während B eine künstlich vergrößerte Basis darstellt, die mit Hilfe der Transformationsmatrix

$$M = \begin{pmatrix} 7 & 5 & -5 \\ -8 & -5 & 6 \\ 2 & 2 & -1 \end{pmatrix} \in \text{GL}_n(\mathbb{Z})$$

aus A erzeugt wurde. Nun soll mit Hilfe von BABAI's Round-Off-Methode der Vektor $x = (70, 70, 70)$ approximiert werden. Mit Hilfe der Basis A erhalten wir

$$z_A = A^{-1}x \approx (10.16, 7.20, 9.25), \quad \lceil z_A \rceil = (10, 7, 9)$$

und mit B

$$z_B = B^{-1}x \approx (-60.90, 43.76, -43.53), \quad \lceil z_B \rceil = (-61, 44, -44).$$

Wie wir sehen, liefern die zugehörigen Approximationen unterschiedliche Gitterpunkte, nämlich

$$w_1 = A\lceil z_A \rceil = (69, 68, 68), \quad w_2 = B\lceil z_B \rceil = (88, 34, 85),$$

wobei der zweite deutlich weiter von x entfernt liegt.

4.2 Primfaktorzerlegung

Das Problem, eine ganze Zahl N in ihre Primfaktoren zu zerlegen, wird bereits seit dem Altertum untersucht. Wesentliche Fortschritte auf diesem Gebiet wurden aber erst in den letzten 40 Jahren gemacht. Einige neuere Lösungsmethoden gehen von folgender Idee aus: Gelingt es, zwei ganze Zahlen x, y zu finden, die die Kongruenz

$$x^2 \equiv y^2 \pmod{N}$$

erfüllen, so hat man $(x - y)(x + y) \equiv 0 \pmod{N}$ und man bekommt durch Berechnung von $\gcd(x - y, N)$ und $\gcd(x + y, N)$ echte Faktoren von N . Das gilt unter der Voraussetzung, dass

$$x \not\equiv \pm y \pmod{N},$$

was bei „zufälliger“ Wahl von x, y mit einer Wahrscheinlichkeit von mindestens $\frac{1}{2}$ der Fall ist (die Wahrscheinlichkeit hängt von der Anzahl der Primteiler von N ab).

Erste Ansätze, das Faktorisierungsproblem auf die Gitterreduktion zurückzuführen, gelangen SCHNORR und ADLEMAN 1993 bzw. 1995. Allerdings gelang erst AJTAI [Aj97] eine Reduktion auf das SVP in der Euklidischen Norm ohne zusätzliche zahlentheoretische Annahmen; diese Reduktion wird nun vorgestellt.

Seien $p_0 := -1$ und p_1, \dots, p_k die ersten k Primzahlen, B eine natürliche Zahl mit $p_k \leq B$.

Definition 4.2.1 Eine natürliche Zahl N heißt B -glatt, falls für alle ihre Primteiler p gilt: $p \leq B$.

Die Gitterbasenreduktion erzeugt Kongruenzen der Form

$$\prod_{i=0}^k p_i^{d_i} \equiv \prod_{i=0}^k p_i^{e_i} \pmod{N}.$$

Besitzt man $k + 2$ solcher Kongruenzen, so gibt es eine Auswahl, deren gemeinsames Produkt eine quadratische Kongruenz ergibt (das bedeutet, alle d_i, e_i sind gerade). Eine solche findet man durch Lösen eines linearen Gleichungssystems über \mathbb{F}_2 .

AJTAI benutzt in seiner Reduktion das Gitter $L(B)$ mit der Basismatrix

$$B = [b_1, \dots, b_{k+2}] = \begin{pmatrix} \sqrt{\ln p_1} & \cdots & 0 & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & \cdots & \sqrt{\ln p_k} & 0 & 0 \\ 0 & \cdots & 0 & 0 & N^{-2} \\ N^{11} \ln p_1 & \cdots & N^{11} \ln p_k & N^{11} \ln b & N^{11} \ln(1 + \frac{N}{b}) \end{pmatrix},$$

wobei b eine zufällige natürliche Zahl mit den Eigenschaften

$$N^{10} < b < (2N)^{10} \quad \text{und} \quad b \equiv \prod_{i=1}^k p_i^{d_i} \pmod{N}, \quad d_i \in \{0, 1\}$$

ist (b ist also quadratfrei). Die die Größe der Gitterbasis bestimmende Zahl k wird so gewählt, dass die Ungleichung $(\ln N)^c < k < (\ln N)^{2c}$ erfüllt wird. Ist dabei $c > 0$ hinreichend groß, so gibt es in der Sequenz $b + lN$ mit $|l| \leq N^{\frac{1}{2}}$ wenigstens eine Zahl, die B -glatt und quadratfrei ist; für diese gilt dann

$$b + lN = \prod_{i=1}^k p_i^{e_i}, \quad e_i \in \{0, 1\}.$$

Nun betrachte man den Gittervektor

$$v = \sum_{i=1}^k -e_i b_i + b_{k+1} + l b_{k+2}.$$

Die ersten k Einträge sind 0 oder $-\sqrt{\ln p_i}$. Der $(k+1)$ -te Eintrag hat die Größe $l \cdot N^{-2}$, ist wegen $|l| \leq N^{\frac{1}{2}}$ also kleiner als 1. Für den letzten Eintrag v_{k+2} gilt

$$\begin{aligned} v_{k+2} &= N^{11} \left(- \sum_{i=1}^k e_i \ln p_i + \ln b + l \cdot \ln \left(1 + \frac{N}{b} \right) \right) \\ &= N^{11} \cdot \ln \left(\prod_{i=1}^k p_i^{-e_i} \cdot b \cdot \left(1 + \frac{N}{b} \right)^l \right) \\ &\approx N^{11} \cdot \ln \left(\frac{1}{b + lN} \cdot b \cdot \left(1 + \frac{lN}{b} \right) \right) \\ &= 0. \end{aligned}$$

Die Näherung der vorletzten Zeile beruht auf der Tatsache, dass $N \ll b$ unter Benutzung von $\left(1 + \frac{N}{b} \right)^l = 1 + \frac{lN}{b} + \mathcal{O}\left(\frac{l^2 N^2}{b^2}\right)$. Das bedeutet, dass v ein sehr kurzer Gittervektor sein muss; AJTAI konnte zeigen, dass die Ungleichung

$$\|v\| \leq C + \ln b$$

gilt, wobei $C < 2 \ln 2$ ist. Durch eine Reduktion der Gitterbasis besteht also die Möglichkeit, den Vektor v zu bestimmen. Durch Lösen eines linearen Gleichungssystems bekommt man dann die Exponenten e_i und erhält eine Kongruenz

$$\prod_{i=1}^k p_i^{d_i} \equiv b \equiv b + lN \equiv \prod_{i=1}^k p_i^{e_i} \pmod{N}.$$

Durch Wiederholung ist es somit möglich, die erforderliche Zahl von mindestens $k+2$ Kongruenzen zu erzeugen.

Bemerkung 4.2.2 Die Reduktion arbeitet offensichtlich mit reellen Vektoren b_i . Für praktische Aspekte ist das (aufgrund von Rundungsfehlern) problematisch; die Vektoren werden hier durch hinreichend genaue Vektoren mit rationalen Einträgen angenähert.

4 Anwendungen der Gitterreduktion

5 Gitterbasierte Kryptosysteme

In diesem Kapitel wird das eigentliche Ziel der in den vorherigen Kapiteln eingeführten Grundlagen und Probleme von Gittern behandelt – die gitterbasierten Kryptosysteme. Die Grundlagen gitterbasierter Kryptographie wurden in [Aj96] gelegt. Die hier vorgestellten Systeme sind Public-Key-Systeme und arbeiten meist auf vollständigen ganzzahligen Gittern. Die Sicherheit basiert in der Regel auf der Schwierigkeit, das CVP zu lösen. Gitterbasierte Kryptosysteme bieten sich als Alternative zu den etablierten Systemen wie RSA an. Zum jetzigen Zeitpunkt wird von flächendeckenden Einsätzen aber abgesehen, da ihr gemeinsamer Nachteil – die hohen Schlüssellängen – noch zu stark ins Gewicht fällt.

Die nachfolgend vorgestellten Systeme benutzen als öffentlichen Schlüssel meist eine Gitterbasis $B \in \mathbb{Z}^{n \times n}$. Im Hinblick auf die Schlüssellängen wollen wir daher zunächst eine Aussage über „sinnvolle“ Größen solcher Basen machen.

Sei B Basismatrix eines Gitters $L = L(B) \subset \mathbb{Z}^n$ und sei $u_i \in \mathbb{Z}^n$ derjenige Vektor, dessen i -ter Eintrag den Wert $\det L (= \det B)$ besitzt, wobei die restlichen Einträge 0 sind. Dann besitzt der Vektor

$$z_i := B^{-1}u_i$$

ganzzahlige Einträge, da für jede reguläre Matrix B die Darstellung

$$B^{-1} = \frac{\text{adj}(B)}{\det B}$$

gilt. Hierbei ist $\text{adj}(B)$ die *Adjunkte* der Matrix B , deren Einträge Linearkombinationen der Einträge von B und damit ebenfalls ganzzahlig sind. Ist aber z_i ganzzahlig, so ist der Vektor $u_i = Bz_i$ selbst im Gitter L enthalten. Zu jedem Gitterpunkt v ist damit auch $v + ku_i$ für $k \in \mathbb{Z}$ ein Gitterpunkt (anschaulich bedeutet dies, dass sich das Gitter entlang der Hauptachsen in Abständen $\det L$ wiederholt). Das bedeutet, dass die Einträge einer Gitterbasis B gegebenenfalls modulo $\det L$ reduziert werden können. Bei einem Kryptosystem mit öffentlicher Basis B kann jeder diese Reduktion vornehmen, sodass es keinen Sinn macht, Basen mit Einträgen betragsgrößer als $\det L$ zu veröffentlichen.

5.1 Das AJTAI-DWORK-System

Bei diesem von AJTAI und DWORK in [AD97] vorgestellten System kommt die Idee des Störens von Gittervektoren durch kurze Fehlervektoren zum Einsatz. Es handelt sich um ein probabilistisches System, bei dem theoretisch Verschlüsselungsfehler auftreten können.

Ziel der Arbeit war aber nicht die Einführung eines praxistauglichen Systems, sondern der Beweis einer worst-case/average-case-Äquivalenz zweier Gitterprobleme. Genauer konnte eine Reduktion von worst-case-Instanzen des SVP mit (in der Gitterdimension n) polynomiellem Approximationsfaktor auf average-case-Instanzen eines SVP in speziellen Gittern angegeben werden. Bereits zufällige Instanzen des letztgenannten Problems sind damit so schwierig wie worst-case-Instanzen des erstgenannten Problems.

Das System selbst ist nicht praktikabel, da der Kryptotext um ein Vielfaches länger ist als der Klartext und die Wahrscheinlichkeit für Dechiffrierfehler zu groß ist. Daher wird das System nur vorgestellt und auf die Analyse von Angriffsmethoden verzichtet.

Vom AJTAI-DWORK-System existieren verschiedene Varianten, von denen nachfolgend zwei besprochen werden. Das System verschlüsselt (in jeder Variante) einen Klartext bitweise. Bei der Entschlüsselung wird entschieden, ob das fragliche Bit auf 0 oder 1 gesetzt wird. Möglich ist dabei ein einseitiger Dechiffrierfehler.

5.1.1 Variante 1

Erforderliche Parameter

- die Gitterdimension n sowie $m := n^3$
- die Parameter $R := 2^{n \ln(n)}$ und $r := n^{-3}$, die die Größe der Vektoreinträge bestimmen
- B und S bezeichnen die n -dimensionale Kugeln mit Mittelpunkt 0 und Radien R bzw. r

Schlüsselerzeugung

Als privater Schlüssel dient ein Vektor u , der gleichverteilt aus der n -dimensionalen Einheitskugel gewählt wird. Die Generierung des öffentlichen Schlüssels geschieht in mehreren Schritten:

1. Wähle gleichverteilt „lange“ Vektoren a_1, \dots, a_m aus der Menge $\{x \in B : \langle x, u \rangle \in \mathbb{Z}\}$.
2. Für $i = 1, \dots, m$ und $j = 1, \dots, n$ wähle δ_{ij} gleichverteilt aus S und bilde „kurze“ Vektoren $\delta_i := \sum_{j=1}^n \delta_{ij}$.
3. Bilde die Summe $v_i := a_i + \delta_i$ für $i = 1, \dots, m$.
4. Sei i_0 der kleinste Index i , für den der Durchmesser des von den Vektoren v_{i+1}, \dots, v_{i+n} aufgespannten Parallelotops $\mathcal{P}(v_{i+1}, \dots, v_{i+n})$ mindestens den Wert $n^{-2}R$ besitzt (gemäß den Autoren existiert dieser mit sehr hoher Wahrscheinlichkeit). Setze $w_j := v_{i_0+j}$ für $j = 1, \dots, n$.
5. Der öffentliche Schlüssel besteht aus dem Tupel (v_1, \dots, v_m) und dem Index i_0 .

Verschlüsseln

Die Verschlüsselung erfolgt bitweise. Dazu definieren wir die Reduktion eines Vektors v modulo eines Parallelotops $\mathcal{P} := \mathcal{P}(w_1, \dots, w_n)$. Es gilt

$$v' := v \bmod \mathcal{P} \quad \Leftrightarrow \quad v = v' + \sum_{i=1}^n z_i w_i \quad \text{mit } v' \in \mathcal{P}, z_i \in \mathbb{Z}.$$

Verschiebt man also v in die Grundmasche (wobei die Verschiebung einem ganzzahligen Vielfachen einer Gittermasche entspricht), so erhält man v' .

Um eine 0 zu verschlüsseln, wählen wir gleichverteilt Koeffizienten $b_1, \dots, b_m \in \{0, 1\}$ und bilden den Vektor $c := \sum_{i=1}^m b_i v_i \bmod \mathcal{P}$. Im Falle der 1 wählen wir einfach einen gleichverteilten Zufallsvektor c aus \mathcal{P} . c dient in beiden Fällen als Kryptozeichen, das das entsprechende Bit ersetzt.

Entschlüsseln

Ist c gegeben, so können wir mit Hilfe von u das Skalarprodukt $\tau = \langle c, u \rangle$ berechnen. Ist τ höchstens um $1/n$ von einer ganzen Zahl entfernt, so entschlüsseln wir c als 0, in allen anderen Fällen als 1.

Bemerkung 5.1.1 AJTAI und DWORK konnten zeigen, dass eine 0 des Klartextes immer auch als solche entschlüsselt wird. Dagegen kann eine 1 mit Wahrscheinlichkeit $1/n$ fälschlicherweise als 0 entschlüsselt werden. In [GGH97a] wird eine Methode vorgestellt, mit der diese Dechiffrierfehler eliminiert werden können.

5.1.2 Variante 2

Für die genaue Beschreibung dieser Variante sind zunächst noch einige Vorbereitungen nötig.

Definition 5.1.2 Sei $n \in \mathbb{N}$, $M, d > 0$ reelle Zahlen und $L \subseteq \mathbb{Z}^n$ ein ganzzahliges Gitter, das ein Untergitter L' der Dimension $n - 1$ und folgenden Eigenschaften besitzt:

1. L' besitzt eine Basis der maximalen Länge M .
2. Ist H die lineare Hülle von L' , dann hat jede Nebenklasse H' , die L schneidet, mindestens den Abstand d zu H .

Für L' schreiben wir kurz $L_{d,M}$. Mit d_L wird der minimale Abstand zwischen H und einer Nebenklasse H' , die L schneidet, bezeichnet.

Mit U_n wird der n -dimensionale Einheitswürfel und mit $S_n(R)$ die Einheitskugel mit Radius $R > 0$ bezeichnet. In [Aj96] wird eine Methode angegeben, in Polynomzeit Gitterpunkte zufällig in einem Würfel KU_n , $K > 0$ zu wählen. Die wesentliche Aussage ist der nachfolgende Satz, für dessen vollständigen Beweis aber auf die Arbeit von AJTAI verwiesen wird.

5 Gitterbasierte Kryptosysteme

Satz 5.1.3 Sei $L = L(B) \subseteq \mathbb{Z}^n$ ein ganzzahliges Gitter zur Basismatrix B . Dann gibt es eine Konstante c_0 derart, dass für jedes $c \geq c_0$ eine Konstante c_1 gibt, sodass ein polynomiell beschränkter Algorithmus existiert, der bei Eingabe von B zufällig Gitterpunkte in einem Würfel KU_n mit Kantenlänge $K = n^c \|B\|$ ausgibt und deren Verteilung höchstens $2^{-n^{c_1}}$ von der Gleichverteilung abweicht.

Beweis. Siehe hierzu [Aj96]. Im Beweis wird ein Parallelotop \mathcal{P} mit Gittervektoren als Ecken konstruiert, welches den Würfel KU_n enthält und dessen Anzahl enthaltenen Gitterpunkte die Anzahl der Gitterpunkte in KU_n höchstens polynomiell übersteigt. \square

Des Weiteren benötigen wir zufällige Vektoren auf dem Rand der Kugel $S_n(R)$. In [AD97] wird ein Verfahren angegeben, mit dem solche Vektoren gleichverteilt erzeugt werden können.

Erforderliche Parameter

- Natürliche Zahlen $n, l \in \mathbb{N}$ und $l \geq 4n$. n ist hierbei die Dimension eines Gitters $L \subset \mathbb{R}^n$.
- Zwei reelle Zahlen $K, R > 0$.
- Der Parameter $\xi_{L,K,R}$. Um diesen zu erhalten, wird zunächst ein zufälliger Gittervektor $v \in KU_n \cap L$ mit dem AJTAI-Algorithmus bestimmt. Danach wird ein Störvektor w erzeugt, der sich aus der Addition von l zufälligen Vektoren auf dem Rand der Kugel $S_n(R)$ ergibt. $\xi_{L,K,R}$ ist dann die Summe aus v und w .
- Der Parameter η_K , ein zufälliger Vektor aus KU_n .

Schlüsselerzeugung

Die Generierung des privaten und des öffentlichen Schlüssels geschieht in fünf Schritten:

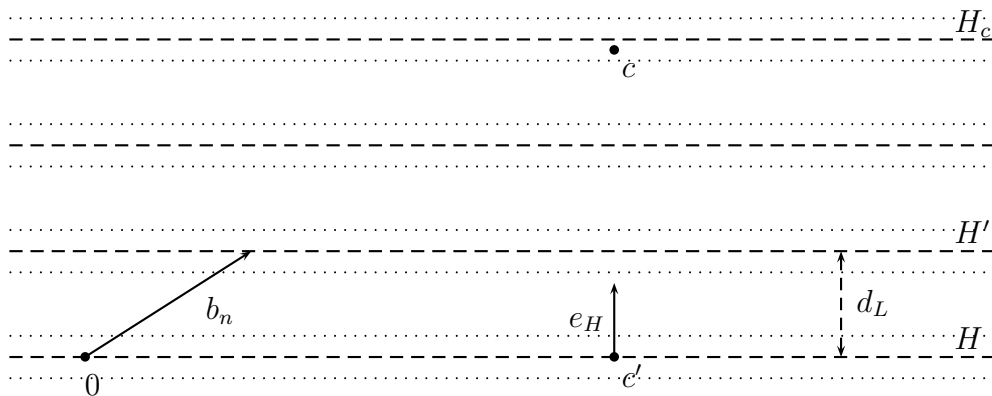
1. Wähle zufällig ein $(n - 1)$ -dimensionales Gitter L' mit den Basisvektoren b_1, \dots, b_{n-1} und der Längenbeschränkung $\|b_i\| \leq M$. H bezeichnet dann die lineare Hülle $\text{span}(b_1, \dots, b_{n-1})$.
2. Wähle Konstanten $c > 5$ und $d \geq n^c M$.
3. Erzeuge einen Zufallsvektor b_n mit einer Entfernung $d \leq d_L \leq 2d$ von H .
4. Der private Schlüssel ist eine Basis des Untergitters $L_{d,M}$.
5. Der öffentliche Schlüssel ist ein Paar (B', M) , wobei B' eine zufällige Basis für das (vollständige) Gitter $L = L(B)$ mit $B = [b_1, \dots, b_n]$ ist.

Verschlüsseln

Ein Klartext p wird bitweise verschlüsselt. Hierzu wird $K \geq 2^n d$ und $R = n^3 M$ gewählt. Um den Schlüsseltext c zu erhalten, wird in p eine 0 durch den Vektor $\xi_{L,K,R}$ und eine 1 durch den Vektor η_K ersetzt.

Entschlüsseln

Zu Dechiffrierung eines Kryptotextes c wird ein zur Hyperebene H orthogonaler Einheitsvektor e_H und der Abstand d_L zwischen zwei Hyperebenen berechnet. e_H zu bestimmen ist leicht möglich, wenn der private Schlüssel, die Basismatrix zu $L_{d,M} \subset H$, bekannt ist. Auch d_L ist dann leicht zu ermitteln: Zu H wird der Vektor b_n des öffentlichen Schlüssels addiert. Dieser ist so beschaffen, dass seine Addition zu H direkt zu einer nächstgelegenen Nebenklasse H' führt. Sind H und H' bekannt, so lässt sich sehr leicht der Abstand d_L berechnen (siehe Abbildung).



Entschlüsseln eines Kryptotextes c

Als nächstes wird der Abstand d_c zwischen c und H bestimmt. Projiziert man c auf die Ebene H , so bekommt man einen Vektor $c' \in H$ und es gilt

$$c' = c - \frac{\langle c, e_H \rangle}{\|e_H\|^2} e_H = c - \langle c, e_H \rangle e_H.$$

Für den gesuchten Abstand d_c gilt

$$d_c = \|c - c'\| = \langle c, e_H \rangle \|e_H\| = \langle c, e_H \rangle.$$

c wird als 0 dekodiert, falls c hinreichend dicht an der nächstgelegenen Hyperebene H_c liegt. Das ist genau dann der Fall, falls das Verhältnis $\delta := d_c/d_L$ hinreichend dicht an einer ganzen Zahl liegt. Die genaue Toleranz ist aufgrund der Parameter bekannt. Der Bereich, in dem c als 0 dekodiert wird, ist in der Abbildung durch die gepunkteten Linien markiert.

Offensichtlich wird eine 0 auch als 0 entschlüsselt. Eine 1 kann aber durchaus als 0 entschlüsselt werden, falls bei der Verschlüsselung ein Punkt zufällig zu dicht zu einer Nebenklasse erzeugt wird. Die Wahrscheinlichkeit für einen solchen Entschlüsselungsfehler ergibt sich aus den Parametern und ist

$$\Pr[\text{Dechiffrierfehler}] = \frac{2lR}{d_L} \leq 4n^{4-c}.$$

5.2 Das GGH-Kryptosystem

Das GGH-Kryptosystem ist ein Public-Key-System und wurde erstmalig von seinen Erfindern GOLDREICH, GOLDWASSER und HALEVI [GGH97] auf der Crypto'97 vorgestellt. Es ist ebenfalls ein Kryptosystem, das von Gitterproblemen Gebrauch macht und lässt sich auch für digitale Signaturen verwenden. Das Ziel der Autoren war in erster Linie die Entwicklung einer auf Gittern basierenden Einwegfunktion mit Falltür. Erst in einer verbesserten, von MICCIANCIO vorgestellten Variante (die wir danach betrachten) gewinnt dieses System Praxisrelevanz. Es gilt als Alternative zu bekannten Verfahren wie RSA und ELGAMAL und besitzt gegenüber diesen den Vorteil der schnelleren Ver- und Entschlüsselung. Aber auch der große Nachteil, die relativ großen Schlüssel, wird nicht verschwiegen (siehe Tabelle).

	Schlüssellänge	Zeitaufwand	Kryptotextlänge
RSA	$\mathcal{O}(n)$	$\mathcal{O}(n^3)$	$\mathcal{O}(n)$
GGH	$\mathcal{O}(n^3 \ln(n))$	$\mathcal{O}(n^3 \ln(n))$	$\mathcal{O}(n^2 \ln(n))$
MICCIANCIO	$\mathcal{O}(n^2 \ln(n))$	$\mathcal{O}(n^2 \ln(n))$	$\mathcal{O}(n \ln(n))$

Vergleich der Kryptosysteme bezogen auf den Sicherheitsparameter n

Der Sicherheitsparameter n ist hier die Gitterdimension, bei RSA entspricht er der Länge der verwendeten Zahlen.

Öffentlicher und privater Schlüssel sind zwei Basismatrizen B und R desselben (vollständigen) Gitters $L \subset \mathbb{Z}^n$. Den Chiffretext c erhält man aus dem Klartext p durch Bilden von

$$c = Bp + e,$$

wobei e ein (kurzer) Störvektor ist. Die Matrix R ist dabei so beschaffen, dass sich mit ihrer Hilfe p sehr leicht und mit sehr hoher Erfolgsrate aus c bestimmen lässt. Ohne die Kenntnis von R – so die Annahme – lässt sich p nur ermitteln, indem man eine Instanz des CVP löst. Darüberhinaus wird angenommen, dass sich R nicht einfach aus B berechnen lässt, was einer Instanz des SBP entspricht. Die Addition des genannten Störvektors kann man in diesem Sinne als Einwegfunktion auffassen. Die Basis R spielt dabei die Rolle einer Falltürinformation.

Praktische Experimente haben gezeigt, dass als Gitterrang $n \geq 400$ gewählt werden sollte, damit nicht eines dieser Probleme (beispielsweise mit dem L^3 -Verfahren oder Varianten desselben) gelöst werden kann (dieser Wert basiert auf neueren Experimenten und liegt bereits deutlich über der ursprünglichen Schätzung der Autoren). Eine solche Wahl macht das Verfahren aber sehr unhandlich und wenig praktikabel.

Erforderliche Parameter

Nachfolgend werden die Parameter aufgeführt, durch die das Kryptosystem bestimmt wird. Möglichkeiten für die konkrete Wahl der Parameter werden weiter unten beschrieben. Sie basieren im Wesentlichen auf den von den Autoren in [GGH97] durchgeführten Experimenten sowie späteren Versuchen, das Verschlüsselungssystem zu brechen. Benötigt werden

- der Rang des Gitters n , wobei n hinreichend groß sein sollte

- der öffentliche Schlüssel B und der private Schlüssel R , die beide Gitterbasen desselben Gitters darstellen
- der Störparameter $\sigma \in \mathbb{N}_+$, der das Aussehen des Vektors e bestimmt
- der Parameter $l \in \mathbb{N}$, der zur Erzeugung von R benötigt wird

Schlüsselerzeugung

Zunächst wird der private Schlüssel R bestimmt, wofür es verschiedene Möglichkeiten gibt. Die einfachste Möglichkeit besteht darin, die Einträge r_{ij} gleichverteilt aus der Menge $\{-l, \dots, +l\}$ zu wählen. l ist dabei relativ klein (z. B. $l = 4$). Die Autoren favorisieren dagegen folgende Methode: Zunächst wird eine Störmatrix $S \in \mathbb{Z}^{n \times n}$ erzeugt, deren Einträge s_{ij} gleichverteilt aus $\{-l, \dots, +l\}$ sind. Die private Basis ergibt sich dann zu

$$R := kI_n + S, \quad k \in \mathbb{N}$$

mit der Einheitsmatrix I_n und einer natürlichen Zahl k , die groß ist gegenüber l . Damit sind die zugehörigen Basisvektoren nahezu orthogonal (die „Masse“ der Vektoren liegt auf der Hauptdiagonalen). Statt ein konstantes k zu wählen, empfehlen die Autoren eine Größenordnung $k \approx l\sqrt{n}$.

Der öffentliche Schlüssel B geht aus R durch eine zufällige Basistransformation hervor. Dazu werden Matrizen

$$M_j = \begin{pmatrix} 1 & 0 & \cdots & m_{1j} & \cdots & 0 & 0 \\ 0 & 1 & & m_{2j} & & & 0 \\ \vdots & & 1 & \vdots & & & \vdots \\ & & & 1 & & & \\ \vdots & & & \vdots & 1 & & \vdots \\ 0 & & & \vdots & & 1 & 0 \\ 0 & 0 & \cdots & m_{nj} & \cdots & 0 & 1 \end{pmatrix}.$$

für $1 \leq j \leq n$ definiert. Die Einträge m_{ij} werden zufällig aus der Menge $\{-1, 0, +1\}$ gewählt. Man prüft leicht nach, dass alle $M_j \in \text{GL}_n(\mathbb{Z})$ unimodulare Matrizen sind. Die Produktmatrix

$$M := \prod_{j=1}^n M_j$$

ist also selbst unimodular. Die öffentliche Matrix B ergibt sich durch die Basistransformation

$$B := RM,$$

wobei, wie zu Beginn des Kapitels erwähnt, die Einträge von B modulo $\det B$ reduziert werden. Die oben beschriebene Erzeugung von M kann auch mehrmals wiederholt werden. Die Konstruktion von B bewirkt, dass die veröffentlichten Basisvektoren viel größer sind als die privaten. Um R zu bekommen, muss also eine Basisreduktion durchgeführt werden.

Verschlüsseln

Ist $B \in \mathbb{Z}^{n \times n}$ die veröffentlichte Basismatrix, so kann man jedem Klartext $p \in \mathbb{Z}^n$ eindeutig einen Gittervektor

$$x := Bp$$

zuordnen. Damit der Klartext nicht ohne weiteres rekonstruiert werden kann, wird ein Störvektor $e \in \{\pm\sigma\}^n$ addiert, wobei das Vorzeichen pro Eintrag zufällig gleichverteilt gewählt wird. Der resultierende Schlüsseltext c ist also von der Form

$$c := x + e = Bp + e.$$

Entschlüsseln

Die Entschlüsselung basiert auf der Annahme, dass x der zu c nächste Gittervektor ist. Den Klartext erhält man über die private Basismatrix R mittels der vorgestellten Round-Off-Methode von BABAI. Diese geht von der (notwendigen) Bedingung aus, dass $\lceil R^{-1}e \rceil = 0$ ist. Dann gilt

$$\begin{aligned} p &= B^{-1}x \\ &= B^{-1}RR^{-1}x \\ &= B^{-1}R(R^{-1}x + \underbrace{\lceil R^{-1}e \rceil}_{=0}) \\ &= B^{-1}R\lceil R^{-1}c \rceil \end{aligned}$$

und p lässt sich aus B , R und c rekonstruieren. Die letzte Gleichung folgt aus $R^{-1}x = \lceil R^{-1}x \rceil$, da

$$R^{-1}x = \underbrace{R^{-1}B}_{\in \text{GL}_n(\mathbb{Z})} \underbrace{B^{-1}x}_{=p}$$

und somit $R^{-1}x$ nur ganzzahlige Einträge enthält.

Es gibt nun keine Garantie, dass tatsächlich $\lceil R^{-1}e \rceil = 0$ gilt. Diese Annahme beruht, wie bereits bei der Analyse von BABAI's Round-Off-Methode gesehen, auf der Beobachtung $|\det R| \gg 1$, $\det R^{-1} = (\det R)^{-1}$, wonach die Einträge von R^{-1} und somit auch die von $R^{-1}e$ sehr klein sein müssen. Ist $\lceil R^{-1}e \rceil \neq 0$, so kommt es zu einem Dechiffrierfehler. Das Auftreten eines solchen Fehlers hängt entscheidend von der Größe der Einträge in der Matrix R^{-1} ab. Von den Autoren stammt folgende Aussage über den Dechiffrierfehler:

Satz 5.2.1 *Sei $\rho := \max_{1 \leq i \leq n} \|s_i\|_1$, wobei s_i die i -te Spalte der Matrix R^{-1} darstellt. Solange $\sigma < \frac{1}{2\rho}$ gewählt wird, tritt kein Dechiffrierfehler auf.*

Mit einem kleinen σ und geeigneter Wahl von R lassen sich Dechiffrierfehler demnach ganz vermeiden. Dazu müssen aber die Einträge von R recht groß gewählt werden; damit der Satz greift, mindestens in der Größe von n – was sich aber nachteilig auf die Laufzeit auswirkt. Weiterhin impliziert ein kleiner Wert für σ auch einen kurzen Störvektor e . Je kürzer dieser ist, desto größer ist aber die Chance eines Angreifers, mit Hilfe der Round-Off-Methode (angewandt auf die öffentliche

Basis B) den Klartext p zu erhalten, weswegen man aus Gründen der Sicherheit an möglichst großen Werten für σ interessiert ist. Die Autoren geben daher noch eine Analyse für die *Wahrscheinlichkeit* eines Dechiffrierfehlers für ein gewähltes σ an.

Satz 5.2.2 *Ist R die geheime Gitterbasis und sind die Einträge von R^{-1} im Betrag durch $\frac{\gamma}{\sqrt{n}}$ beschränkt, so gilt*

$$Pr[\text{Dechiffrierfehler}] \leq 2n \cdot \exp\left(-\frac{1}{8\sigma^2\gamma^2}\right).$$

Beweis. Für den Beweis brauchen wir die Hoeffding-Ungleichung [Hoe63]. Diese besagt folgendes: Sind X_1, \dots, X_n unabhängige Zufallsvariablen und gilt

$$a_i \leq X_i - E[X_i] \leq b_i,$$

dann gilt für jede positive Konstante c die Schranke

$$Pr\left[\sum_{i=1}^n (X_i - E[X_i]) \geq c\right] \leq \exp\left(-\frac{2c^2}{\sum_{i=1}^n (b_i - a_i)^2}\right).$$

Sei $d := R^{-1}e$ und d_i, e_i seien die jeweils i -ten Einträge der Vektoren d und e . s_i bezeichne die i -te Zeile der Matrix R^{-1} sowie s_{ij} den dazugehörigen j -ten Eintrag.

Wir betrachten jetzt die Zufallsvariablen $s_{ij}e_j$. Die Einträge e_j des Fehlervektors nehmen mit gleicher Wahrscheinlichkeit die beiden Werte $\pm\sigma$ an, es ist also $E[e_j] = 0$. Damit gilt ebenfalls $E[s_{ij}e_j] = 0$. Nach Voraussetzung ist $|s_{ij}| \leq \frac{\gamma}{\sqrt{n}}$ und somit $-\sigma\frac{\gamma}{\sqrt{n}} \leq s_{ij}e_j \leq \sigma\frac{\gamma}{\sqrt{n}}$. Wir können jetzt die Hoeffding-Ungleichung auf einen Eintrag d_i anwenden und erhalten

$$Pr\left[|d_i| > \frac{1}{2}\right] = 2 \cdot Pr\left[d_i > \frac{1}{2}\right] = 2 \cdot Pr\left[\sum_{j=1}^n s_{ij}e_j > \frac{1}{2}\right] \leq 2 \cdot \exp\left(-\frac{1}{8\sigma^2\gamma^2}\right).$$

Ein Dechiffrierfehler tritt bereits auf, falls für irgendein $1 \leq i \leq n$ die Bedingung $|d_i| > \frac{1}{2}$ erfüllt ist. Eine einfache Abschätzung liefert

$$Pr[\text{Dechiffrierfehler}] \leq n \cdot Pr\left[|d_i| > \frac{1}{2}\right] \leq 2n \cdot \exp\left(-\frac{1}{8\sigma^2\gamma^2}\right)$$

und damit das gesuchte Ergebnis. □

Um die Wahrscheinlichkeit eines Dechiffrierfehlers unter ε zu halten ist eine Wahl von $\sigma \leq \left(\gamma\sqrt{8\ln(2n/\varepsilon)}\right)^{-1}$ also ausreichend. Wichtig ist hier natürlich der genaue Wert für γ . Testläufe der Autoren ergaben z. B. für $n = 120$, $\varepsilon = 10^{-5}$ und Matrizen R mit Einträgen zwischen -4 und $+4$ den Wert $\gamma = \frac{1}{30}$, woraus man $\sigma \leq 2.5$ erhält. Um die „erlaubte“ Fehlerwahrscheinlichkeit weiter zu senken oder mehr Spielraum für den Parameter σ zu bekommen ist eine Verkleinerung von γ erforderlich, was sich nur mit größeren Einträgen in der Matrix R realisieren lässt.

5.2.1 Parameterwahl und Aufwand

Für den privaten Schlüssel R ergibt sich ein Speicheraufwand von $\mathcal{O}(n^2)$, für den öffentlichen Schlüssel B dagegen $\mathcal{O}(n^2 \ln(\det L))$. Wegen $\det R = \det B = \det L$ bekommt man mit Hilfe der LEIBNIZ-Formel für die Determinante (alternativ kann man auch die HADAMARD-Ungleichung heranziehen) die Schranke

$$\det L \leq n \cdot c^n$$

für die Determinante, falls die Einträge in R im Betrag durch c beschränkt sind. Mit $c \approx \sqrt{n}$ folgt ein Speicheraufwand von $\mathcal{O}(n^3 \ln(n))$ für B .

Nehmen wir an, dass der Klartext in einen Vektor p transformiert wird, dessen Einträge durch eine Konstante beschränkt sind (die Autoren schlagen pro Eintrag in p eine konstante Zahl zufälliger Bits vor, wobei das niederwertigste Bit aus der zu verschlüsselnden Nachricht stammt), so wird die Laufzeit für die Verschlüsselung durch die Multiplikation dieses Vektors mit der Basismatrix B bestimmt und entspricht daher einem Aufwand von $\mathcal{O}(n^3 \ln(n))$.

Ein Problem ergibt sich bei der Entschlüsselung des Kryptotextes. Dazu benötigen wir die inverse Matrix R^{-1} und die unimodulare Matrix $B^{-1}R$ (das ist gerade die Inverse zur Transformationsmatrix M aus der Schlüsselgenerierung). Letztere kann man vorberechnen und speichern. Die Matrix R^{-1} zu speichern wäre aber sehr teuer, da sie um einiges größer ist als R selbst. Diesem Problem kann man begegnen, indem man von R^{-1} jeweils nur die ersten k Bit abspeichert und mit der so erhaltenen gerundeten Matrix R' weiterrechnet. Diese ist von der Form

$$R' := R^{-1} + E,$$

wobei die Einträge e_{ij} der Fehlermatrix E durch $|e_{ij}| \leq 2^{-k}$ beschränkt sind¹. Bei der Dekodierung bekommen wir dann ein p' mit

$$\begin{aligned} p' &= B^{-1}R[R'c] \\ &= B^{-1}R[(R^{-1} + E)(Bp + e)] \\ &= B^{-1}R[R^{-1}Bp + R^{-1}e + E(Bp + e)] \\ &= p + B^{-1}R[R^{-1}e + E(Bp + e)] \end{aligned}$$

(da $R^{-1}B$ unimodular ist, ist der Vektor $R^{-1}Bp$ ganzzahlig und lässt sich ausklammern). Die Dekodierung geht schief, falls $[R^{-1}e + E(Bp + e)]$ nicht der Nullvektor ist. Um festzustellen, wie wir diesen Fall vermeiden können, müssen wir uns die Einträge des Vektors anschauen. Die Einträge von $R^{-1}e$ wurden schon analysiert, sodass wir uns auf den Term $E(Bp + e)$ beschränken können. Gegenüber Bp fällt der Vektor e nicht ins Gewicht (dessen Einträge $\pm\sigma$ sind vernachlässigbar klein), wir betrachten also nur EBp . Sind die Einträge von B bzw. p durch s bzw. t Bit beschränkt, so enthält EBp Einträge der Größe $n^2 \cdot 2^{s+t-k}$. Ist diese Größe hinreichend klein (z. B. $\approx 2^{-30}$), so fällt der durch die Matrix E eingeschleppte Fehler nicht ins

¹So wie R gewählt wurde, sollten die Einträge von R^{-1} immer kleiner als 1 sein, die ersten k Bit sind also Nachkommastellen.

Gewicht. Die Autoren geben ein „typisches“ Beispiel mit $n = 200$, $s = 16$ und $t = 4$ an, für welches $k = 64$ ausreichend ist. Selbst wenn man hier n auf 400 erhöhen würde, bliebe noch genügend Spielraum. Für deutlich größere n ergibt sich aber schon ein schlechteres Bild: Da s von der Größenordnung $n \ln n$ ist, muss k ebenfalls asymptotisch zu $n \ln n$ vergrößert werden – was einen Gesamtspeicheraufwand von $\mathcal{O}(n^3 \ln n)$ nach sich zieht.

Das GGH-System lässt sich auch als Signaturverfahren verwenden. Um eine Nachricht $p \in \mathbb{Z}^n$ zu signieren, wird mit Hilfe der kurzen, geheimen Gitterbasis R ein Gitterpunkt u in der Nähe zu p bestimmt. Dieser wird als Linearkombination der Vektoren der öffentlichen Basis B dargestellt (d. h. $u = Bz$ mit $z \in \mathbb{Z}^n$), wobei z als Unterschrift dient. Bei der Verifikation wird die Unterschrift akzeptiert, falls u ein Gitterpunkt und $\|p - u\|$ hinreichend klein ist, es also praktisch unmöglich ist, mit der öffentlichen Basis B einen entsprechend nahegelegenen Gitterpunkt u zu erzeugen.

5.2.2 Angriffsmethoden

Nachfolgend werden einige Methoden beschrieben, das GGH-System zu brechen. Einige Angriffe benutzen eine heuristische Reduktion des zugrundeliegenden CVP auf ein SVP in einem erweiterten Gitter. Der Grund ist die Hoffnung auf die „leichtere“ Lösbarkeit des SVP gegenüber dem CVP.

Angriff bei Mehrfachkodierung

Vermeiden sollte man in jedem Fall die Kodierung derselben Nachricht p unter Verwendung unterschiedlicher Störvektoren e_i . Bekommt ein Angreifer insgesamt k solcher Schlüsseltexte $c_i = Bp + e_i$ in die Hände, so kann er den Mittelwert

$$c^* = \frac{1}{k} \sum_{i=1}^k c_i = \frac{1}{k} \sum_{i=1}^k (Bp + e_i) = Bp + \underbrace{\frac{1}{k} \sum_{i=1}^k e_i}_{=: \varepsilon_k}$$

berechnen. Wegen $E[e_i] = 0$ gilt $\varepsilon_k \rightarrow 0$ für $k \rightarrow \infty$ und der Angreifer wird bei hinreichend großem k mit BABAI's Round-Off-Methode, angewandt auf c^* und die öffentliche Basis B , Erfolg haben.

Round-Off-Attacke

Der Angreifer kann versuchen, die öffentliche Basis B zum Entschlüsseln zu verwenden. Er bekommt so einen Vektor $v := B^{-1}c = p + B^{-1}e$. Dass der gerundete Vektor $\lceil v \rceil$ wohl kaum die gesuchte Nachricht p ergibt, wurde schon diskutiert. Da aber p ganzzahlig ist, könnte eine Suche auf allen ganzzahligen Punkten in der Nähe von v Erfolg versprechen. Unter vereinfachten Annahmen konnten die Autoren belegen, dass der zugehörige Suchraum exponentiell in der Gitterdimension n ist. Bereits bei $n = 100$ soll diese Attacke aufwändiger sein als das Durchprobieren aller möglicher Störvektoren.

Nearest-Plane-Attacke

BABAI hat in [Bab86] eine weitere Möglichkeit beschrieben, den nächsten Gitterpunkt zu approximieren. Sei $B = \{b_1, \dots, b_n\}$ eine L^3 -reduzierte Basis eines Gitters L . Bezeichne U den Unterraum $U := \text{span}(b_1, \dots, b_{n-1})$ und $L' := U \cap L$ das enthaltene Untergitter. Um zu $x \in \mathbb{R}^n$ einen nahegelegenen Gitterpunkt u zu bestimmen, wird x auf die zu x nächstgelegene Hyperebene $U + v$, $v \in L$ orthogonal projiziert, was einen Punkt x' liefert. Zu $x' - v \in U$ wird nun rekursiv ein nahegelegener Gitterpunkt w in dem Untergitter L' berechnet, mit dessen Hilfe man $u := w + v$ erhält. In der Rekursion fährt man dann mit der Basis $B' := \{b_1, \dots, b_{n-1}\}$ fort.

BABAI zeigte, dass die Nearest-Plane-Methode ebenfalls einen Approximationsfaktor exponentiell in n besitzt. Experimente haben aber ergeben, dass ein Angriff mit dieser Methode bei $n \geq 150$ fast aussichtslos ist.

Einbettungsattacke

Die chiffrierte Nachricht c ist von der Form $c = x + e = Bp + e$, wobei e ein gegenüber x sehr kurzer Vektor ist. Das führt auf die Idee, e über eine geeignete SVP-Instanz eines modifizierten Gitters zu bestimmen. Ist e berechnet, so erhält man wegen

$$p = B^{-1}(c - e)$$

direkt den Klartext p . Hierzu definieren wir das erweiterte Gitter $L_c = L(B_c)$ mit der Basismatrix

$$B_c := \begin{pmatrix} b_1 & \cdots & b_n & c \\ 0 & \cdots & 0 & 1 \end{pmatrix},$$

wobei b_1, \dots, b_n die öffentlichen Basisvektoren sind. Dann sind die Vektoren $\begin{pmatrix} x \\ 0 \end{pmatrix} = B_c \begin{pmatrix} p \\ 0 \end{pmatrix}$ und $\begin{pmatrix} c \\ 1 \end{pmatrix} = B_c \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ Gittervektoren von L_c . Somit liegt auch der Differenzvektor

$$\begin{pmatrix} c \\ 1 \end{pmatrix} - \begin{pmatrix} x \\ 0 \end{pmatrix} = \begin{pmatrix} c - x \\ 1 \end{pmatrix} = \begin{pmatrix} e \\ 1 \end{pmatrix} \in L_c$$

in L_c und ist nach Annahme ein kürzester Gittervektor. Dieser lässt sich möglicherweise mit dem L^3 -Algorithmus bzw. Varianten desselben finden.

Diese Form von Einbettung ist eine allgemeine Heuristik für das CVP und scheint nach numerischen Experimenten bei Gittern mit $n \leq 120$ recht gut zu funktionieren.

Angriff nach Nguyen

Ein ähnlicher, von Nguyen [Ngu99] vorgeschlagener Angriff nutzt die Kenntnis des Störparameters σ aus. Zum Störvektor $e \in \{\pm\sigma\}^n$ wird zunächst der Vektor $\{\sigma\}^n$ addiert. Dann ist

$$e + \{\sigma\}^n =: e' \in \{0, 2\sigma\}^n$$

und wir bekommen einen modifizierten Chiffriertext

$$c' := Bp + e'.$$

Die nun folgenden Rechnungen werden modulo 2σ durchgeführt. Es ist

$$c' \equiv Bp \pmod{2\sigma}.$$

Dieses Gleichungssystem lässt sich lösen; man erhält eine Lösung $p_{2\sigma}$, die kongruent ist zu p modulo 2σ . Daher gilt für die Differenz

$$p' := \frac{p - p_{2\sigma}}{2\sigma} \in \mathbb{Z}^n.$$

Der Chiffriertext wird nochmals modifiziert; wir erhalten:

$$c'' = Bp' + \frac{e}{2\sigma} = \frac{c' - p_{2\sigma}B}{2\sigma}.$$

Analog zum letzten vorgestellten Angriff erfolgt nun eine Einbettung in ein Gitter der Dimension $n + 1$. Definiere dazu

$$B_{c''} := \begin{pmatrix} b_1 & \cdots & b_n & c'' \\ 0 & \cdots & 0 & 1 \end{pmatrix}.$$

Die Reduktion des zugehörigen Gitters $L(B_{c''})$ liefert den Fehlervektor $\frac{e}{2\sigma}$, woraus sich e und der gesuchte Klartext p rekonstruieren lassen. Der Mehrwert dieser Attacke besteht also in einer Reduktion des Fehlers um den Faktor 2σ .

5.3 MICCIANCIOS Kryptosystem

Das 2001 von MICCIANCIO in [Mic01] vorgeschlagene System stellt eine Verbesserung des GGH-Systems dar. Die Größe für die benutzten Schlüssel und die Länge des erzeugten Kryptotextes sind hier um eine Größenordnung kleiner. Möglich wird dies durch die Verwendung von Matrizen in der HERMITE-Normalform.

Wie auch beim GGH-System dient eine Matrix R als privater und eine Matrix B als öffentlicher Schlüssel, wobei auch hier der Orthogonalitätsdefekt von B hoch und der von R klein ist. Im Unterschied zum GGH-System ergibt sich der Chiffriertext c aus dem Klartext p durch

$$c = Bz + p,$$

wobei z ein zufälliger ganzzahliger Vektor ist.

Definition 5.3.1 (HERMITESche Normalform) *Eine Matrix $B \in \mathbb{R}^{n \times n}$ liegt in HERMITEScher Normalform (HNF) vor, wenn*

- B eine rechte obere Dreiecksmatrix ist und
- $0 \leq b_{ij} < b_{ii}$ für alle $0 \leq i < j \leq n$

Für uns ist die folgende Eigenschaft der HERMITESchen Normalform relevant: Ist $L(R) \subset \mathbb{R}^n$ ein vollständiges Gitter zur Basismatrix R , so existiert eine eindeutig bestimmte Matrix B in HNF mit $L = L(B)$. Zu jedem Gitter gibt es also eine eindeutig bestimmte Basismatrix in HERMITE-Normalform.

MICCIANCIO schlug vor, als öffentliche Basis B für das GGH-Kryptosystem die HNF zu wählen – im Gegensatz zur dortigen zufälligen Wahl. Die angeführten Gründe sind

5 Gitterbasierte Kryptosysteme

- ein deutlich verringerter Speicherplatzverbrauch
- die Vermeidung möglicher Schwächen einer zufällig gewählten Matrix B

Unter dem letzten Punkt ist zu verstehen, dass man durch die Veröffentlichung der HNF einem potentiellen Angreifer keine Informationen preisgibt, die dieser sich nicht sowieso verschaffen könnte (was bei einer Wahl von B wie im GGH-System durchaus passieren könnte). Es ist nämlich möglich, ausgehend von einer beliebigen Basismatrix B' , die zum Gitter zugehörige Basis in HNF in Polynomialzeit zu berechnen (eine solche Methode wird z. B. in [MicWar01]) beschrieben.

Der geringere Speicherplatzverbrauch ergibt sich aus folgender Argumentation: Ist B eine ganzzahlige Basis in HERMITE-Normalform, so entspricht das Produkt der Hauptdiagonaleinträge gerade der Gitterdeterminante $\det L$ und der Platzverbrauch dieser Einträge ist von der Größe $\sum_{i=1}^n \ln(b_{ii}) = \ln(\det L)$. Der Speicherplatzbedarf l für die Matrix B lässt sich daher durch

$$l \leq \sum_{i=1}^n n \ln(b_{ii}) = n \ln(\det L)$$

nach oben abschätzen. Wählt man nun das Gitter L analog zum GGH-System, so ergibt sich ein Aufwand von $\mathcal{O}(n^2 \ln(n))$ für den öffentlichen Schlüssel. Dieser ist also um einen Faktor n kleiner als der im GGH-System vorgeschlagene.

Schlüsselerzeugung

Zur Erzeugung der privaten Basis R kann ähnlich vorgegangen werden wie beim GGH-System. Im System von MICCIANCIO wird zusätzlich darauf geachtet, den Wert $h(R) := \min_i \|\hat{r}_i\|$ möglichst groß zu halten, da dieser die Wahlmöglichkeiten für den Klartext p einschränkt (siehe unten). $h(R)$ ist umso größer, je „orthogonaler“ die Vektoren in R sind, da das GRAM-SCHMIDT-Verfahren bei sehr kleinen bzw. großen Winkeln eine stärkere Verkürzung vornimmt. Die Motivation hierfür ist die Beseitigung möglicher Dechiffrierfehler. Wie wir nachfolgend sehen werden, lässt sich das CVP *exakt* lösen, falls bestimmte Kriterien erfüllt sind.

Wie oben erwähnt, entspricht der öffentliche Schlüssel B der zugehörigen Basismatrix in HNF, um den Speicheraufwand zu reduzieren.

Definition 5.3.2 (orthogonale Projektion) Ist $\{b_1, \dots, b_n\} \in \mathbb{R}^m$ eine (geordnete) Gitterbasis, so heißt die Zuordnung

$$\pi_i : \mathbb{R}^m \rightarrow \text{span}(b_1, \dots, b_{i-1})^\perp$$

orthogonale Projektion, falls für alle $v \in \mathbb{R}^m$

$$v - \pi_i(v) \in \text{span}(b_1, \dots, b_{i-1})$$

gilt.

Bemerkung 5.3.3 Für die GRAM-SCHMIDT-Vektoren gilt $\hat{b}_i = \pi_i(b_i)$. Mit Hilfe der \hat{b}_i ist es möglich, die Projektion eines beliebigen Vektors $v \in \mathbb{R}^m$ zu bestimmen. Es gilt

$$\pi_i(v) = v - \sum_{j=1}^{i-1} \frac{\langle v, \hat{b}_j \rangle}{\|\hat{b}_j\|^2} \hat{b}_j.$$

Satz 5.3.4 Sei $L = L(b_1, \dots, b_n) \subset \mathbb{R}^m$ ein Gitter und $h(B) := \min_i \|\hat{b}_i\|$. Ist ein Vektor der Form $v = Bz + p$ mit $z \in \mathbb{Z}^n, p \in \mathbb{R}^m$ und der Bedingung $\|p\| < \frac{1}{2}h(B)$ gegeben, so kann das CVP-Problem für v in polynomieller Zeit durch die Round-Off-Methode gelöst werden.

Beweis. Bz ist ein Gittervektor und nach Voraussetzung gilt $\|Bz - v\| < \frac{1}{2}h(B)$. Dann gilt ebenso für die Projektion $\|\pi_n(Bz - v)\| < \frac{1}{2}h(B)$. v ist bekannt und kann mit Hilfe der GRAM-SCHMIDT-Basis in Polynomialzeit dargestellt werden als $v = \sum_{i=1}^n x_i \hat{b}_i$ mit $x_i \in \mathbb{R}$. Weiterhin ist $Bz = \sum_{i=1}^n z_i b_i$, wobei die $z_i \in \mathbb{Z}$ zu bestimmen sind. Für die Projektion bekommt man

$$\begin{aligned} \pi_n(Bz - v) &= \pi_n(Bz) - \pi_n(v) \\ &= \sum_{i=1}^n z_i b_i - \sum_{j=1}^{n-1} \frac{\langle \sum_{i=1}^n z_i b_i, \hat{b}_j \rangle}{\|\hat{b}_j\|^2} \hat{b}_j - x_n \hat{b}_n \\ &= \sum_{i=1}^n z_i \left(b_i - \sum_{j=1}^{n-1} \frac{\langle b_i, \hat{b}_j \rangle}{\|\hat{b}_j\|^2} \hat{b}_j \right) - x_n \hat{b}_n \\ &= z_n \hat{b}_n - x_n \hat{b}_n. \end{aligned}$$

Als Folgerung ergibt sich $|z_n - x_n| \cdot \|\hat{b}_n\| < \frac{1}{2}h(B)$, d. h. $|z_n - x_n| < \frac{1}{2}$. Den Koeffizienten z_n erhält man somit durch Runden von x_n , da x_n bekannt ist. Mit derselben Argumentation für die Projektionen π_i bekommt man die restlichen Koeffizienten z_i . \square

Verschlüsseln

Für den zu verschlüsselnden Klartext p muss die Bedingung $\|p\| < \frac{1}{2}h(R)$ erfüllt sein. Es wird ein zufälliger Vektor $z \in \mathbb{Z}^n$ gewählt, mit dessen Hilfe der Chiffretext

$$c = Bz + p$$

bestimmt wird. Nach MICCIANCIO wird dieser nach folgendem Schema erzeugt:

Dazu betrachten wir ein beliebiges Gitter L . Sind $v, w \in \mathbb{R}^m$ beliebige reelle Vektoren, so lässt sich die Äquivalenzrelation

$$v \equiv_L w \iff v - w \in L$$

definieren. Sind c, c' zwei Kryptotexte, so gilt offensichtlich $c \equiv_L c'$; es genügt also, einen Repräsentanten zu wählen. Man kann leicht zeigen, dass zu jedem $v \in \mathbb{Z}^n$ ein

5 Gitterbasierte Kryptosysteme

eindeutiger Vektor $w \in \mathcal{P}(\hat{B})$ ($\mathcal{P}(\hat{B})$ ist das zur GRAM-SCHMIDT-Basis zugehörige Parallelotop) existiert mit $v \equiv_L w$. w kann leicht mit folgender Methode bestimmt werden: Ist v gegeben, so berechne rekursiv absteigend

$$v_n := v$$

$$v_{i-1} := v_i - \left\lfloor \frac{\langle v_i, \hat{b}_i \rangle}{\|\hat{b}_i\|^2} \right\rfloor b_i \quad \text{für } i = n, \dots, 1.$$

Dann finden wir

$$w := v_0, \quad w = (v \bmod B).$$

Da sich v und w nur um ganzzahlige Vielfache von Gittervektoren unterscheiden, gilt die Relation $w \equiv_L v$. Darüberhinaus ist leicht nachzuprüfen, dass $\frac{\langle w, \hat{b}_i \rangle}{\|\hat{b}_i\|^2} \in [0, 1)$ für alle i , sodass $w \in \mathcal{P}(\hat{B})$. Man beachte, dass die Äquivalenzrelation basisunabhängig ist, der reduzierte Vektor $w = (v \bmod B)$ aber vom Aussehen der Basismatrix B abhängt. Anschaulich wird der Vektorraum \mathbb{R}^n in Parallelotope der Form $\{\mathcal{P}(\hat{B}) + u \mid u \in L(B)\}$ partitioniert; der reduzierte Vektor w ist dann der zu v korrespondierende Vektor in $\mathcal{P}(\hat{B})$ (im Gegensatz zur Reduktion modulo $\mathcal{P}(B)$ kann sich hier aber eine andere relative Position des Gitterpunktes ergeben). Ist $v \in \mathbb{Z}^n$, so ist auch $(v \bmod B) \in \mathbb{Z}^n$.

Zu einem Klartext p wird also $c = p \bmod B$ bestimmt und veröffentlicht. Aufgrund der obigen Überlegungen gilt $\|p \bmod B\| < \frac{1}{2}h(B)$, eine schnelle Entschlüsselung mit Kenntnis von R ist also möglich. Da der reduzierte Vektor $p \bmod B$ eindeutig bestimmt ist, arbeitet das System deterministisch.

Entschlüsseln

Ein gegebener Schlüsseltext c wird entschlüsselt, indem die entsprechende CVP-Instanz für das Gitter $L(B)$ gelöst wird. Mit Kenntnis der geheimen Basis R lässt sich Satz 5.3.4 anwenden und ein Gittervektor in Polynomialzeit bestimmen. Wegen $\|Bz - c\| < \frac{1}{2}h(R) \leq \lambda_1(L)$ ist sogar *garantiert* (im Unterschied zum GGH-System), dass Bz der zu c nächste Gittervektor ist. Man erhält somit Bz und hieraus den Klartext p .

Platz- und Zeitverbrauch

Die Größe der Schlüssel und Kryptotexte sowie die benötigte Zeit zum Ver- und Entschlüsseln haben wir schon beim GGH-System analysiert. Im Unterschied dazu haben wir hier die HERMITE-Normalform mit dem erwähnten, um den Faktor n geringeren Platzverbrauch, zur Verfügung. Für B und den Schlüsseltext c ergeben sich hier die Schranken $\mathcal{O}(n^2 \ln n)$ und $\mathcal{O}(n \ln n)$ – also eine deutliche Verbesserung.

Die Verschlüsselungszeit ist etwa proportional zur Größe des öffentlichen Schlüssels und spielt – da die Schlüssellänge deutlich geringer als bei GGH oder RSA ist – eine untergeordnete Rolle. Die Schlüsselgenerierung spielte bei GGH eine entscheidende Rolle, da dort durch die vielen Mixrunden die Berechnung der Transformationsmatrix relativ aufwändig war. Im System von MICCIANCIO wird nur die Berechnung der HNF vorgenommen, was sehr zeiteffizient vonstatten geht.

Sicherheit

Die Sicherheit des Systems beruht auf dem Orthogonalitätsdefekt: Die Basisvektoren in B besitzen untereinander sehr spitze bzw. sehr stumpfe Winkel. Eine GRAM-SCHMIDT-Orthogonalisierung erzeugt in einem solchen Fall sehr kurze Vektoren (wegen $\hat{b}_1 = b_1$ bleibt der erste Vektor dagegen sehr lang). Das ist bei der privaten Basis R nicht der Fall. Da die Gitterdeterminante (das Produkt der $\|\hat{b}_i\|$ bzw. der $\|\hat{r}_i\|$) unveränderlich ist, gilt daher für den „Entschlüsselungsradius“

$$h(B) = \min_i \|\hat{b}_i\| \ll \min_i \|\hat{r}_i\| = h(R).$$

Somit ist mit hoher Wahrscheinlichkeit sichergestellt, dass Satz 5.3.4 für die öffentliche Basis B nicht anwendbar ist.

Es gibt bis heute nur wenige Untersuchungen die Sicherheit und demnach die Wahl der Sicherheitsparameter betreffend. Eine etwas umfangreichere Untersuchung machte C. Ludwig [Lud04], in der verschiedene Angriffsmethoden (L^3 -Reduktion, BKZ-Reduktion, Lenstra-Verheul-Extrapolation) mit unterschiedlichen Parametern durchgeführt wurden. Im Ergebnis steht die Feststellung, dass mindestens Gitterdimension $n \geq 800$ erforderlich ist, um die notwendige Sicherheit zu gewährleisten.

5.4 Das NTRU-Kryptosystem

Das Kryptosystem NTRU („Number Theory Research Unit“) wurde 1996 von J. Hoffstein, J. Pipher und J. H. Silverman [HPS96] vorgestellt. Die Verschlüsselung basiert bei diesem System auf einer Mischung aus polynomieller Algebra und der Reduktion modulo zweier Zahlen p und q unterschiedlicher Größenordnung, während die Korrektheit der Entschlüsselung auf probabilistischen Überlegungen beruht. Ähnlich den anderen Kryptosystemen basiert die Sicherheit auf dem Finden möglichst kurzer Vektoren in einem Gitter. Die von den Autoren hervorgehobenen Vorteile liegen in der schnellen Ver- und Entschlüsselung: Ist N die Blocklänge für einen Klartext, so liegt die Schlüssellänge in $\mathcal{O}(N)$, während die Laufzeit für Ver- und Entschlüsselung durch $\mathcal{O}(N^2)$ beschränkt ist.

Erforderliche Parameter

Das NTRU-System besitzt als Parameter ein Tupel (N, p, q) natürlicher Zahlen und vier Mengen $\mathcal{L}_F, \mathcal{L}_{G_1}, \mathcal{L}_{G_2}, \mathcal{L}_m$ bestehend aus Polynomen vom Grad $N-1$. Die Zahlen p und q sind nicht notwendigerweise prim, werden aber als teilerfremd vorausgesetzt, wobei $q \gg p$. Die Beträge der Koeffizienten der Polynome aus den genannten Mengen sind durch p nach oben beschränkt.

NTRU arbeitet ringbasiert, d. h. zu gegebener natürlicher Zahl N werden alle Operationen im Polynomring $R = \mathbb{Z}[X]/(X^N - 1)$ ausgeführt. Das bedeutet, dass bei der Multiplikation die Exponenten der Monome im Produkt modulo N reduziert werden. Ist $F \in R$ ein Polynom, so definieren wir die Vektordarstellung

$$F = \sum_{i=0}^{N-1} F_i X^i = [F_0, \dots, F_{N-1}].$$

5 Gitterbasierte Kryptosysteme

Sind $F, G, H \in R$ Polynome und $F \star G = H$ das Produkt, so gilt für den Sternoperator \star

$$H_k = \sum_{i=0}^k F_i G_{k-i} + \sum_{i=k+1}^{N-1} F_i G_{n+k-i} = \sum_{i+j \equiv k \pmod{N}} F_i G_j.$$

Das Produkt lässt sich effizient in $\mathcal{O}(N^2)$ berechnen.

Das Ergebnis einer Multiplikation von Polynomen modulo p (q) erhält man, indem die Koeffizienten des Produktes $F \star G$ modulo p (q) reduziert werden. Für die Stärke der Sicherheit und das erfolgreiche Durchführen einer Entschlüsselung ist die Größe der Koeffizienten in den verwendeten Polynomen wichtig, welche durch p und q bestimmt werden.

Schlüsselerzeugung

Um einen NTRU-Schlüssel zu erzeugen, wählt man zufällig zwei Polynome $F \in \mathcal{L}_F$ und $G_1 \in \mathcal{L}_{G_1}$. Das Polynom F muss zusätzlich die Eigenschaft aufweisen, modulo p und q invertierbar zu sein. Bei einer „normalen“ Wahl von F ist das auch meistens der Fall. Die Inversen benennen wir mit F_p und F_q , es gilt also

$$F_q \star F \equiv 1 \pmod{q}, \quad F_p \star F \equiv 1 \pmod{p}.$$

Da die Ringe $\mathbb{Z}_p[X]/(X^N - 1)$ und $\mathbb{Z}_q[X]/(X^N - 1)$ Euklidische Ringe sind, kann eine Invertierung effizient mit dem Erweiterten Euklidischen Algorithmus durchgeführt werden; die Inversen müssen darüberhinaus nur einmalig berechnet werden. Der öffentliche Schlüssel ist ein Polynom $H \in R$ und ergibt sich zu

$$H \equiv F_q \star G_1 \pmod{q}.$$

Der private Schlüssel ist das Polynom F ; für die Entschlüsselung wird F_p benötigt, sodass F_p dauerhaft gespeichert werden sollte.

Verschlüsseln

Als Klartext m wird ein Polynom aus der Klartextmenge \mathcal{L}_m gewählt; darüberhinaus wird noch ein zufälliges Polynom $G_2 \in \mathcal{L}_{G_2}$ erzeugt. Nun kann der öffentliche Schlüssel H benutzt werden, um

$$c \equiv p \cdot G_2 \star H + m \pmod{q}$$

zu berechnen. c ist dann der Kryptotext, der an den Empfänger geschickt wird.

Entschlüsseln

Der Empfänger erhält c und kann nun den geheimen Schlüssel F (und das gespeicherte Inverse F_p) zur Dechiffrierung benutzen. Dazu berechnet er zunächst

$$a \equiv F \star c \pmod{q},$$

wobei die Koeffizienten von a aus dem Intervall $[-\frac{q}{2}, \frac{q}{2}]$ gewählt werden. Im zweiten Schritt wird

$$b \equiv F_p \star a \pmod{p}$$

berechnet, wobei b mit sehr hoher Wahrscheinlichkeit dem Klartext m entspricht.

Korrektheit

Warum wird c korrekt dekodiert? Dazu betrachten wir die Umformungen

$$\begin{aligned} a &\equiv F \star c \pmod{q} \\ &\equiv p \cdot F \star G_2 \star H + F \star m \pmod{q} \\ &\equiv p \cdot F \star G_2 \star F_q \star G_1 + F \star m \pmod{q} \\ &\equiv p \cdot G_1 \star G_2 + F \star m \pmod{q} \end{aligned}$$

Die Polynome G_1 , G_2 , F und m wurden gerade so gewählt, dass ihre Koeffizienten sehr klein gegenüber q sind. Das bewirkt, dass auch die Koeffizienten in den Produkten $G_1 \star G_2$ und $F \star m$ klein gegenüber q sind, was somit die Gleichheit $a = p \cdot G_1 \star G_2 + F \star m$ im gesamten Ring R impliziert. Damit darf der Empfänger modulo p rechnen und bekommt

$$\begin{aligned} b &\equiv F_p \star a \pmod{p} \\ &\equiv \underbrace{F_p \star p \cdot G_1 \star G_2}_{=0} + \underbrace{F_p \star F \star m}_{=1} \pmod{p} \\ &\equiv m \pmod{p} \end{aligned}$$

und damit den Klartext. Ist nun wie anfangs angenommen $p \ll q$, so kommt es nur mit einer vernachlässigbar geringen Wahrscheinlichkeit zu einem Dechiffrierfehler. Um auch den Dechiffrierfehler zu vermeiden, schlagen die Autoren das Erweitern der Nachricht um zusätzliche Checkbits vor. Natürlich könnte man bei einem entsprechenden Größenunterschied zwischen p und q Dechiffrierfehler ganz ausschließen. Dadurch steigt aber der Rechenaufwand für die Ringoperationen, sodass man eher an einem Kompromiss interessiert ist (siehe nächsten Abschnitt).

5.4.1 Praktische Parameterwahl

Bei der Wahl der Parameter geht es im Wesentlichen um die Koeffizienten der verwendeten Polynome, welche die Wahrscheinlichkeit für das Auftreten eines Dechiffrierfehlers und die Erfolgsaussichten von Angriffen auf das Kryptosystem bestimmen.

Definition 5.4.1 (Weite) Die Weite eines Polynoms $P \in R$ ist definiert als

$$|F|_\infty := \max_{1 \leq i \leq N} \{F_i\} - \min_{1 \leq i \leq N} \{F_i\}.$$

Die Weite kann als Maximumnorm auf R aufgefasst werden. Wir definieren zusätzlich die Norm

$$|F|_2 := \sqrt{\sum_{i=1}^N (F_i - \hat{F})^2}, \quad \text{wobei } \hat{F} = \frac{1}{N} \sum_{i=1}^N F_i,$$

5 Gitterbasierte Kryptosysteme

welche ein Maß für die Varianz der Koeffizienten darstellt (der Quotient $|F|_2/\sqrt{N}$ ist die Standardabweichung der Koeffizienten von F). Die nachfolgende Abschätzung der Weite des Produktpolynoms $F \star G$ für $F, G \in R$ wird in [HPS96] angegeben.

Satz 5.4.2 *Für jedes $\varepsilon > 0$ existieren (von N und ε abhängige) Konstanten $\gamma_1, \gamma_2 > 0$, sodass für zufällig gewählte Polynome $F, G \in R$ die Eigenschaft*

$$\gamma_1 |F|_2 |G|_2 \leq |F \star G|_\infty \leq \gamma_2 |F|_2 |G|_2$$

mit einer Wahrscheinlichkeit größer als $1 - \varepsilon$ erfüllt ist.

Die Autoren betonen hierbei, dass das bei diesem Satz wesentliche Verhältnis γ_2/γ_1 auch noch für moderat große N und sehr kleine ε - wie durch Experimente gezeigt - nicht allzu groß gerät.

Praktisch wählt man für den Nachrichtenraum \mathcal{L}_m die Menge

$$\mathcal{L}_m := \left\{ m \in R : -\frac{1}{2}(p-1) \leq m_i \leq \frac{1}{2}(p-1) \right\},$$

wobei p als ungerade angenommen wird. Mit der Kurznotation

$\mathcal{L}(d_1, d_2) := \{F \in R : d_1 \text{ Koeffizienten sind } 1, d_2 \text{ Koeffizienten sind } -1, \text{ die restlichen Koeffizienten sind } 0\}$

können wir die restlichen Mengen beschreiben. Dazu wählen wir drei natürliche Zahlen d_F, d_{G_1}, d_{G_2} und setzen

$$\mathcal{L}_F := \mathcal{L}(d_F, d_F - 1), \quad \mathcal{L}_{G_1} := \mathcal{L}(d_{G_1}, d_{G_1}), \quad \mathcal{L}_{G_2} := \mathcal{L}(d_{G_2}, d_{G_2}).$$

(Polynome aus $\mathcal{L}(d_F, d_F)$ sind nicht invertierbar.) Die Wahl bewirkt eine möglichst kleine Varianz der Koeffizienten; wir erhalten

$$|F|_2 = \sqrt{2d_F - 1 - N^{-1}}, \quad |G_1|_2 = \sqrt{2d_{G_1}}, \quad |G_2|_2 = \sqrt{2d_{G_2}}.$$

Für die konkrete Wahl der Parameter d_F, d_{G_1}, d_{G_2} müssen noch Sicherheitsüberlegungen angestellt werden.

Um Dechiffrierfehler zu vermeiden, sollte die Bedingung

$$|p \cdot G_1 \star G_2 + F \star m|_\infty < q$$

erfüllt sein. Wie besprochen ist das fast immer der Fall, wenn

$$|p \cdot G_1 \star G_2|_\infty \leq \frac{q}{4} \quad \text{und} \quad |F \star m|_\infty \leq \frac{q}{4}$$

gelten. In Anwendung von Satz 5.4.2 führt das für ein hinreichend kleines ε zur Wahl

$$|G_1|_2 |G_2|_2 \approx \frac{q}{4p\gamma_2} \quad \text{und} \quad |F|_2 |m|_2 \approx \frac{q}{4\gamma_2}.$$

Eine geeignete Wahl, basierend auf Experimenten der Autoren, gibt die untenstehende Übersicht.

N	107	167	503
γ_2	0.35	0.27	0.17

5.4.2 Angriffsmethoden

Brute-Force-Attacke

Brute-Force-Attacken auf das NTRU-System zielen im Wesentlichen darauf ab, den Privatschlüssel durch Durchprobieren aller möglichen Belegungen der Polynome F, G_1, G_2 zu erlangen. Praktisch können alle Schlüssel $F \in \mathcal{L}_F$ getestet werden; finden sich im Produkt $F \star H \pmod{q}$ nur kleine Koeffizienten, hat man sehr wahrscheinlich den richtigen Schlüssel gefunden. Ebenso kann man indirekt sämtliche Belegungen der Hilfspolynome G_1 und G_2 durchprobieren. Wegen $F \equiv G_1 \star H^{-1} \pmod{q}$ und $m \equiv c - p \cdot G_2 \star H \pmod{q}$ können dann analog auch F und m auf kleine Koeffizienten hin untersucht werden. Der Schlüsselraum \mathcal{L}_F wird in der Praxis größer gewählt als \mathcal{L}_{G_1} , sodass die Sicherheit des Schlüssels F durch $\#\mathcal{L}_{G_1}$ bestimmt wird; die Sicherheit der Nachricht m hängt demzufolge von $\#\mathcal{L}_{G_2}$ ab.

Geburtstagsangriff

Ein Geburtstagsangriff kann gegen die Polynome F und G_2 geführt werden. Bei einem Angriff auf den Schlüssel F wird man eine Spaltung $F = F_1 + F_2$ vornehmen. Es sei daran erinnert, dass das Polynom $a \equiv F \star c \pmod{q}$ kleine Koeffizienten besitzt. Ein Angreifer kann also untersuchen, ob die korrespondierenden Koeffizienten der Polynome $F_1 \star c$ und $-F_2 \star c$ vergleichbare Größe aufweisen. Wählt man F_1 und F_2 zufällig, so wird man bereits nach $k \approx \sqrt{\#\mathcal{L}_F}$ Versuchen ein Paar (F_1, F_2) erzeugt haben, dass diese Eigenschaft erfüllt. Aufgrund der Ordnung der Koeffizienten genügen auch nur $\mathcal{O}(k)$ Tests, um dieses Paar zu finden. In der Praxis bedeutet dies z. B. einen Schlüsselraum \mathcal{L}_F der Größe 2^{160} zu verwenden, um eine Sicherheit von etwa 2^{80} zu erreichen.

Angriff bei mehrfacher Nachrichtenübermittlung

Wird eine Nachricht m unter Verwendung desselben öffentlichen Schlüssels H mehrfach versendet, so ist ein Angriff denkbar, falls unterschiedliche Zufallspolynome G_2 benutzt werden. Werden also r verschiedene Kryptotexte $c_i \equiv p \cdot G_{2i} \star H + m \pmod{q}$ für $i = 1, \dots, r$ übermittelt, kann ein Angreifer aus den Differenzen $c_i - c_1$ die Werte

$$G_{2i} - G_{21} \equiv p^{-1} \cdot (c_i - c_1) \star H^{-1} \pmod{q}$$

berechnen. Bereits bei moderaten Werten von r (meist reichen vier oder fünf Versuche) lassen sich die Koeffizienten von G_{21} durch eine einfache Häufigkeitsanalyse rekonstruieren, woraus man die versteckte Nachricht m erhält.

Gitterbasierte Angriffe

Die eigentliche Verbindung von NTRU zur Gittertheorie besteht in der Möglichkeit, gitterbasierte Angriffe auf die Nachricht m unter Verwendung des öffentlichen Schlüssels H zu fahren. Hier wie auch bei den anderen zuvor behandelten Kryptosystemen nutzt man dabei die unterschiedlichen Größenordnungen der öffentlichen bzw. privaten Schlüssel aus.

Wir betrachten nun ein vollständiges, $2N$ -dimensionales Gitter L mit der Basismatrix

$$B = [b_1, \dots, b_N] = \begin{pmatrix} \alpha & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 0 & \alpha & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & \alpha & 0 & 0 & \cdots & 0 \\ H_0 & H_{N-1} & \cdots & H_1 & q & 0 & \cdots & 0 \\ H_1 & H_0 & \cdots & H_2 & 0 & q & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ H_{N-1} & H_{N-2} & \cdots & H_0 & 0 & 0 & \cdots & q \end{pmatrix}.$$

α ist dabei ein hinreichend klein gewählter Parameter. Da es sich um eine linke untere Dreiecksmatrix handelt, gilt für die Gitterdeterminante $\det L = \alpha^N q^N$. Aufgrund der Gleichheit $G_1 = F \star H$ bewirkt die spezielle Wahl der Basismatrix, dass der Vektor $v = (\alpha F, G_1)$ im Gitter $L(B)$ enthalten ist. Man überzeugt sich leicht, dass $v = B(F, F)^T$.

Während die Basis aus sehr langen Vektoren b_i besteht, ist der Vektor v offensichtlich deutlich kürzer. Nach der Abschätzung von GAUSS (siehe Kapitel 2) können wir für die Länge des kürzesten Gittervektors $\lambda_i(L)$ eines (zufälligen) Gitters mit Rang n erwarten, dass

$$(\det L)^{1/n} \cdot \sqrt{\frac{n}{2\pi e}} \leq \lambda_i(L) \leq (\det L)^{1/n} \cdot \sqrt{\frac{n}{\pi e}}$$

gilt. In unserem Fall folgt nach Einsetzen von $n = 2N$ und $\det L = \alpha^N q^N$

$$\lambda_1(L) \approx \sqrt{\frac{N\alpha q}{\pi e}}.$$

Ein Angreifer wird versuchen, den Vektor v oder einen Vektor vergleichbarer Größe durch Gitterreduktion zu finden. Gemäß den Autoren sind die Erfolgsaussichten für einen solchen Angriff am größten, falls das Verhältnis

$$r_H := \frac{|v|_2}{\lambda_1(L)}$$

durch geschickte Wahl von α minimiert wird. Das ist genau bei $\alpha = |G_1|_2/|F|_2$ der Fall. Man beachte hierbei, dass $|G_1|_2$ und $|F|_2$ bekannte Werte sind. Eingesetzt ergibt sich

$$r_H = \sqrt{\frac{2\pi e |F|_2 |G_1|_2}{Nq}}.$$

Je kleiner dieser Wert wird, umso leichter wird es einem Angreifer fallen, den Zielvektor v zu ermitteln. r_H kann als Maß dafür angesehen werden, wie stark das zugehörige Gitter von einem „zufälligen“ Gitter abweicht. Ein zufälliges Gitter liegt vor, falls r_H einen Wert dicht bei 1 annimmt, im anderen Fall ist r_H deutlich kleiner, was die Laufzeit der Gitterreduktion verkürzt. Nach den Autoren ist die erwartete Laufzeit exponentiell in N und einer Konstanten, die sich proportional zu r_H verhält.

Ein analoger Angriff lässt sich gegen die private Nachricht m führen. Das zu lösende Gitterproblem ähnelt dem für H , der Zielvektor ist hier dagegen von der Form $v = (\alpha m, G_2)$. Der Angreifer sollte hier $\alpha = |G_2|_2/|m|_2$ wählen; analog zum Angriff auf H ergibt sich ein Verhältnis

$$r_m = \sqrt{\frac{2\pi e|m|_2|G_2|_2}{Nq}}$$

als Maß für die Anfälligkeit von m gegenüber Gitterreduktionsverfahren. Auch hier ist bei kleinerem r_m ein Angriff leichter.

Da der Angreifer die Wahl zwischen einem Angriff auf H und einem auf m besitzt, sollten die Größen r_H und r_m derart ausbalanciert werden, dass beide Methoden etwa gleich schwer erscheinen. In diesem Fall nehmen wir $r_H \approx r_m$ an, was $|F|_2|G_1|_2 \approx |m|_2|G_2|_2$ zur Folge hat. Zum Beispiel hätten bei $p = 3$ Nachrichten m Koeffizienten aus $\{-1, 0, 1\}$, bei „normalen“ Nachrichten etwa gleichverteilt, d. h. $|m|_2 \approx \sqrt{2N/3}$. Gemäß der Standardwahl für G_2 gilt $|G_2|_2 = \sqrt{2d_{G_2}}$. Bei der Schlüsselgenerierung sollte also auf die Bedingung

$$|F|_2|G_1|_2 \approx 2\sqrt{Nd_{G_2}/3}$$

geachtet werden.

Eine dritte Angriffsmöglichkeit besteht in der Berechnung „falscher“ Schlüssel. Erzeugt die Gitterreduktion einen Vektor v (der nicht mit dem „echten“ Lösungsvektor übereinstimmen muss), so hat dieser die Gestalt $v = (\alpha F', G'_1)$. F' und G'_1 lösen dabei ebenfalls die Gleichung

$$F' \star c \equiv p \cdot G'_1 \star G_2 + F' \star m \pmod{q}.$$

Die Hoffnung besteht nun darin, dass beide Polynome hinreichend kleine Einträge besitzen, sodass sie zur Entschlüsselung eingesetzt werden können. Untersuchungen der Autoren haben aber ergeben, dass falsche Schlüssel keine ernstzunehmende Schwächung der Sicherheit bedeuten.

Nachfolgend noch einige in [HPS96] aufgelistete Ergebnisse, die die Autoren durch Experimente mit unterschiedlichen Größen der Sicherheitsparameter erhalten haben. Im Mittelpunkt stand dabei die Laufzeit für die damals bekannten Gitterreduktionsverfahren. In der nachfolgenden Tabelle eine kurze Übersicht der Ergebnisse.

Sicherheit	q	N	Zeit (in Sekunden)
mittel	64	107	780.230 (9 Tage)
hoch	128	167	$1.198 \cdot 10^{10}$ (380 Jahre)
höchst	256	503	$1.969 \cdot 10^{35}$ ($6.2 \cdot 10^{27}$ Jahre)

Die Zeiten sind Schätzungen und beziehen sich auf einen Pentium Pro 200.

5 Gitterbasierte Kryptosysteme

6 Komplexität von Gitterproblemen

In diesem Kapitel sollen komplexitätstheoretische Eigenschaften von Gittern untersucht werden. Motivation ist u. a. die Frage, wie schwer Gitterprobleme wie SVP und CVP tatsächlich sind und welche Reduktionen zwischen den Problemen möglich sind. Besprochen wird einerseits die Frage nach der \mathcal{NP} -Schwere der bekannten Gitterprobleme sowie die Frage nach deterministischen bzw. randomisierten Reduktionen der Probleme aufeinander. Als Grundlage dienen klassische Arbeiten wie die von VAN EMDE BOAS sowie neuere Resultate u. a. von MICCIANCIO.

Bisher haben wir die Berechnungsprobleme SVP und CVP untersucht. Wir formulieren jetzt dazu die zugehörigen Entscheidungsprobleme SVP bzw. CVP. Eine SVP-Instanz besteht aus einem Tupel (B, d) und ist genau dann eine Ja-Instanz, falls es im Gitter $L(B)$ einen Vektor $v \neq 0$ mit der Eigenschaft $\|v\| \leq d$ gibt. Eine Instanz des CVP-Problems ist ein Tupel (B, u, d) und bildet genau dann eine Ja-Instanz, falls es einen Gittervektor $v \in L(B)$ mit der Eigenschaft $\|u - v\| \leq d$ gibt.

Die Länge einer Instanz von SVP bzw. CVP wird damit durch die folgenden drei Parameter bestimmt:

- die Vektorraumdimension m (bestimmt die Zeilenzahl der Matrix B)
- die Gitterdimension n (bestimmt die Spaltenzahl der Matrix B)
- die Länge k der verwendeten Zahlen (Einträge in B , u und d)

Nach Ergebnissen von LENSTRA und KANNAN (siehe [Kan83]) lassen sich SVP und CVP auf lineare Programmierung zurückführen. Die Laufzeit ist polynomiell in m und k , die Abhängigkeit zur Gitterdimension n ist allerdings $n^{n+o(n)}$ (eine obere Schranke, die mittlerweile auf $n^{0.184n+o(n)}$ verbessert werden konnte). Erst 2001 konnte ein (probabilistischer) Algorithmus mit $2^{O(n)}$ Operationen vorgestellt werden [AKS01]. Die eigentliche Schwierigkeit der Gitterprobleme resultiert demzufolge aus der Gitterdimension, sodass man die Komplexität dieser Probleme in Abhängigkeit von n untersucht.

Neben den „exakten“ Problemen SVP und CVP benötigen wir weiterhin noch die Approximationsprobleme SVP_γ und CVP_γ .

Definition 6.0.3 *Das Promise-Problem SVP_γ besteht aus Instanzen (B, d) , wobei $B \in \mathbb{Z}^{m \times n}$ eine Gitterbasismatrix, $\gamma : n \rightarrow \mathbb{R}_{\geq 1}$ eine Funktion in der Gitterdimension und $d \in \mathbb{R}$ eine positive Konstante ist. Es ist*

1. (B, d) ist eine Ja-Instanz, falls $\|Bz\| \leq d$ für ein $z \in \mathbb{Z}^n \setminus \{0\}$.

6 Komplexität von Gitterproblemen

2. (B, d) ist eine Nein-Instanz, falls $\|Bz\| > \gamma d$ für alle $z \in \mathbb{Z}^n \setminus \{0\}$.

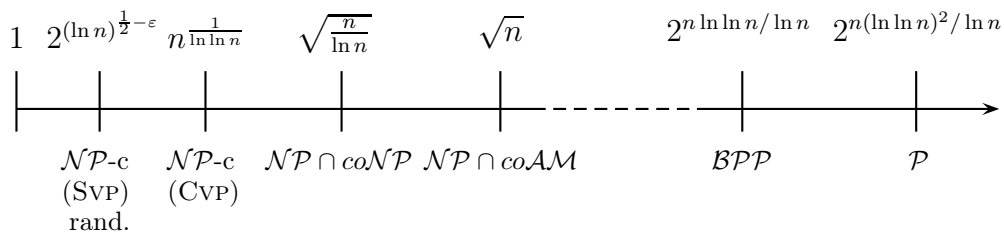
Definition 6.0.4 Das Promise-Problem CVP_γ besteht aus Instanzen (B, x, d) , wobei $B \in \mathbb{Z}^{m \times n}$ eine Gitterbasismatrix, $\gamma : n \rightarrow \mathbb{R}_{\geq 1}$ eine Funktion in der Gitterdimension, $x \in \mathbb{Z}^m$ ein Vektor und $d \in \mathbb{R}$ eine positive Konstante ist. Es ist

1. (B, x, d) ist eine Ja-Instanz, falls $\|Bz - x\| \leq d$ für ein $z \in \mathbb{Z}^n$.

2. (B, x, d) ist eine Nein-Instanz, falls $\|Bz - x\| > \gamma d$ für alle $z \in \mathbb{Z}^n$.

Die Probleme SVP und CVP kann man als Spezialfälle für $\gamma = 1$ auffassen. In [ABSS97] werden einige Ergebnisse zu SVP_γ und CVP_γ zusammengetragen.

Die interessante Frage ist unter anderem, bei welchen Funktionen $\gamma(n)$ die Probleme SVP_γ und CVP_γ welche Komplexitäten aufweisen. Eine kurze Zusammenfassung des heutigen Wissensstandes gibt die untenstehende Skala.



Komplexitäten der Probleme SVP_γ und CVP_γ
bei bestimmten Approximationsfaktoren γ

Man wusste bereits 1990, dass beide Probleme für einen bestimmten polynomiellen Approximationsfaktor in $\mathcal{NP} \cap \text{co}\mathcal{NP}$ liegen (siehe auch [AhReg05]). Da man heute eher von $\mathcal{NP} \neq \text{co}\mathcal{NP}$ ausgeht, gilt es als wahrscheinlich, dass die Probleme bei diesen Approximationsfaktoren nicht \mathcal{NP} -schwer sind.

Betrachten wir nun noch das Problem, einen Wert $d' \in [\lambda_1(L), \gamma \lambda_1(L)]$ zu bestimmen (für $\gamma = 1$ entspricht das der Berechnung von $\lambda_1(L)$). Zwischen diesem Problem und SVP_γ besteht folgende Beziehung: Haben wir ein $d' \in [\lambda_1(L), \gamma \lambda_1(L)]$ gegeben, so können wir durch Test auf $d' \leq \gamma d$ entscheiden, ob eine Ja-Instanz und durch Test auf $d' > \gamma d$, ob eine Nein-Instanz vorliegt (beachte, dass die gegebene Instanz (B, d) in jedem Fall eine Ja- oder Nein-Instanz sein muss) und damit das SVP_γ lösen. Sei umgekehrt $z \in \mathbb{Z}$ eine obere Schranke für $\lambda_1(L)$ (erhält man z. B. durch einen beliebigen Basisvektor). Durch eine Binärsuche lässt sich ein $d \in \{0, \dots, z^2\}$ finden, für das ein SVP_γ -Orakel (B, \sqrt{d}) als Ja- und $(B, \sqrt{d-1})$ als Nein-Instanz ausgibt. Dann liegt $\lambda_1(L)$ im Intervall $[\sqrt{d}, \gamma \sqrt{d}]$ und mit $d' := \gamma \sqrt{d}$ haben wir eine Lösung des Berechnungsproblems.

6.1 Reduktion von SVP auf CVP

Wir wollen uns jetzt mit der Frage beschäftigen, ob es möglich ist, eine Instanz des SVP unter Zuhilfenahme einer Instanz von CVP zu lösen. Wie schon früher angedeutet, wird das SVP tendenziell leichter zu lösen sein. Es ist daher naheliegend, nach einer Reduktion auf das CVP zu suchen. Der naive Ansatz, ein Problem (B, d) auf eine CVP-Instanz $(B, 0, d)$ abzubilden schlägt fehl, da in diesem Falle ein Orakel für das entsprechende SVP-Problem immer den Nullvektor zurückliefert – was als Lösung des SVP-Problems aber explizit ausgeschlossen wird. Eine einfache Reduktion liegt also zunächst nicht auf der Hand.

Nachfolgend wird daher eine Reduktion einer SVP-Instanz auf mehrere CVP-Instanzen vorgestellt. Die grundlegende Idee ist die folgende: Eine Basismatrix $B = [b_1, \dots, b_n]$ für das Gitter $L(B)$ dient als Ausgangspunkt der SVP-Instanz (B, d) . Wir definieren nun n CVP-Instanzen, wobei die j -te Instanz auf der Basis

$$B_j := [b_1, \dots, b_{j-1}, 2b_j, b_{j+1}, \dots, b_n]$$

beruht. Entschieden werden sollen dann die insgesamt n Probleme (B_j, b_j, d) mit deren Hilfe das Ausgangsproblem entschieden wird. Analog dazu lässt sich die Berechnungsvariante modellieren, in der der kürzeste der Differenzvektoren $v_j - b_j$ (mit v_j als j -te CVP-Lösung) als Lösung zurückgegeben wird.

Lemma 6.1.1 Sei $v = \sum_{i=1}^n z_i b_i$, ($z_i \in \mathbb{Z}$) ein kürzester Gittervektor des Gitters $L(B)$ zur Basismatrix $B = [b_1, \dots, b_n]$. Dann ist eines der z_i ungerade.

Beweis. Wären alle z_i gerade, so würde der Vektor $\frac{1}{2}v = \sum_{i=1}^n \frac{1}{2}z_i b_i$ im Gitter $L(B)$ liegen und einen echt kürzeren Gittervektor darstellen. \square

Lemma 6.1.2 Sei $v = \sum_{i=1}^n z_i b_i$ ein Gittervektor von $L(B)$ mit ungeradem z_j . Dann ist der Vektor $u = \frac{z_j+1}{2}(2b_j) + \sum_{i \neq j} z_i b_i$ ein Gittervektor in $L(B_j)$ und es gilt $v = u - b_j$.

Beweis. Da z_j ungerade ist, ist $\frac{z_j+1}{2}$ ganzzahlig und u ist somit ein Gittervektor von $L(B_j)$. Dann prüft man nach, dass

$$u - b_j = \frac{z_j+1}{2}2b_j + \sum_{i \neq j} z_i b_i - b_j = z_j b_j + \sum_{i \neq j} z_i b_i = v,$$

es folgt die Behauptung. \square

Lemma 6.1.3 Sei $u = z'_j(2b_j) + \sum_{i \neq j} z_i b_i$ ein Gittervektor von $L(B_j)$. Dann ist $v = (2z'_j - 1)b_j + \sum_{i \neq j} z_i b_i$ ein Gittervektor in $L(B)$ ungleich dem Nullvektor und es gilt $v = u - b_j$.

Beweis. Es ist $v \neq 0$, da $2z'_j - 1$ ungerade ist. Des weiteren gilt

$$v = (2z'_j - 1)b_j + \sum_{i \neq j} z_i b_i = z'_j(2b_j) + \sum_{i \neq j} z_i b_i - b_j = u - b_j.$$

□

Lemma 6.1.2 besagt, dass die Lösung einer der CVP-Instanzen mindestens so gut ist wie die Lösung der SVP-Instanz mit v als kürzestem Gittervektor. Lemma 6.1.3 liefert dagegen eine untere Schranke: Die Länge des kürzesten Vektors in $L(B)$ ist kleiner oder gleich der Länge der Lösungen der einzelnen CVP-Instanzen. Mit diesen Voraussetzungen können wir nun den Reduktionssatz beweisen.

Satz 6.1.4 *Für jede Funktion $\gamma : \mathbb{N}_+ \rightarrow \mathbb{R}_{\geq 1}$ ist SVP_γ COOK-reduzierbar auf CVP_γ .*

Beweis. Sei (B, d) eine Instanz von SVP_γ und seien (B_j, b_j, d) mit $j = 1, \dots, n$ insgesamt n CVP-Instanzen wie oben definiert. Gezeigt wird nun, dass genau dann (B, d) eine Ja/Nein-Instanz ist, falls ein $1 \leq j \leq n$ existiert, für das (B_j, b_j, d) eine Ja/Nein-Instanz bildet.

„ \Rightarrow “: Ist (B, d) eine Ja-Instanz von SVP_γ , so existiert im Gitter $L(B)$ ein von Null verschiedener kürzester Vektor v mit $\|v\| \leq d$. Nach Lemma 6.1.1 gibt es ein j mit ungeradem Koeffizienten z_j . Lemma 6.1.2 liefert dann einen Gittervektor $u \in L(B_j)$ mit $\|v\| = \|u - b_j\|$. Somit ist (B_j, b_j, d) ebenfalls eine Ja-Instanz für CVP_γ .

„ \Leftarrow “: Angenommen, für ein j sei (B_j, b_j, d) keine Nein-Instanz von CVP_γ . Dann existiert ein Gittervektor $u \in L(B_j)$ mit der Eigenschaft $\|u - b_j\| \leq \gamma d$. Nach Lemma 6.1.3 lässt sich ein $v \in L(B)$, $v \neq 0$ konstruieren, sodass $\|v\| = \|u - b_j\|$ gilt. Dann aber ist auch (B, d) keine Nein-Instanz von SVP_γ . □

6.2 Randomisierte Reduktion

Die Reduktion aus dem letzten Abschnitt hat den Nachteil, dass mehrere CVP_γ -Instanzen erzeugt werden. Es stellt sich die Frage, ob eine Reduktion auf eine einzelne Instanz möglich ist. Wie wir gleich sehen werden, besteht diese Möglichkeit unter Verwendung einer randomisierten Reduktion. Dabei handelt es sich um eine probabilistische Reduktion, bei der ein einseitiger Fehler erlaubt ist. Es gibt mehrere Reduktionen solchen Typs, wir benötigen hier die RUR-Reduktion (Reverse Unfaithful Random Reduction).

Definition 6.2.1 (RUR-Reduktion) *Ein Problem A ist RUR-reduzierbar auf ein Problem B , falls es eine probabilistische Turingmaschine gibt, die eine Nein-Instanz von A immer auf eine Nein-Instanz von B und eine Ja-Instanz von A mindestens mit einer Wahrscheinlichkeit $1/p(n)$ auf eine Ja-Instanz von B abbildet. n ist hierbei die Länge der Eingabe und $p(n)$ ein Polynom, dessen Grad unabhängig von n ist.*

Die COOK-Reduktion aus dem letzten Abschnitt lässt sich leicht in eine RUR-Reduktion transformieren: Auf die Eingabe (B, d) wählen wir zufällig einen Index

$1 \leq j \leq n$ und geben (B_j, b_j, d) aus. Damit haben wir eine Ja-Instanz mit einer Wahrscheinlichkeit von mindestens $1/n$ auf eine Ja-Instanz abgebildet.

Die Wahrscheinlichkeit von $1/n$ für eine erfolgreiche Reduktion lässt sich sogar noch auf $1/2$ verbessern.

Satz 6.2.2 *Für jede Funktion $\gamma : \mathbb{N}_+ \rightarrow \mathbb{R}_{\geq 1}$ ist SVP_γ RUR-reduzierbar auf CVP_γ , wobei Ja-Instanzen mit einer Wahrscheinlichkeit größer oder gleich $1/2$ auf Ja-Instanzen abgebildet werden.*

Beweis. Die SVP_γ -Instanz sei (B, d) mit $B = [b_1, \dots, b_n]$, die ausgegebene CVP_γ -Instanz sei (B', b_1, d) , wobei die Basismatrix $B' = [b'_1, \dots, b'_n]$ wie folgt definiert wird: Wir erzeugen Koeffizienten $c_1 := 1$ und c_2, \dots, c_n unabhängig gleichverteilt aus $\{0, 1\}$ und berechnen $b'_i := b_i + c_i b_1$ für alle i .

Als erstes zeigen wir indirekt, dass (B', b_1, d) eine Nein-Instanz ist, falls vorher auch (B, d) eine Nein-Instanz war. Ist (B', b_1, d) keine Nein-Instanz, dann gibt es einen Gittervektor $u \in L(B')$ mit der Eigenschaft $\|u - b_1\| \leq \gamma d$. Aus der Definition geht hervor, dass $L(B')$ ein Untergitter zu $L(B)$ ist und b_1 nicht in diesem enthalten ist. Der Vektor $u - b_1$ ist demzufolge in $L(B)$ enthalten, ist nicht der Nullvektor und besitzt eine Länge von höchstens γd .

Jetzt betrachten wir den Fall, dass (B, d) eine Ja-Instanz ist. Ist $v = \sum_{i=1}^n z_i b_i$ ein kürzester Gittervektor von $L(B)$, so gibt es nach Lemma 6.1.1 ein j , für welches z_j ungerade ist. Definiere $\alpha := z_1 + 1 - \sum_{i=2}^n c_i z_i$. α ist mit einer Wahrscheinlichkeit von mindestens $1/2$ gerade (α ist sicher gerade, falls alle z_i für $i \geq 2$ gerade sind, in allen anderen Fällen mit Wahrscheinlichkeit $1/2$), sodass der Vektor $u := \frac{1}{2}\alpha b'_1 + \sum_{i=2}^n z_i b'_i$ mit einer Wahrscheinlichkeit größer oder gleich $1/2$ ein Gittervektor von $L(B')$ ist. Andererseits gilt

$$\begin{aligned} u - b_1 &= \left(\alpha b_1 + \sum_{i=2}^n z_i (b_i + c_i b_1) \right) - b_1 \\ &= \left(z_1 - \sum_{i=2}^n c_i z_i \right) b_1 + \sum_{i=2}^n z_i b_i + \sum_{i=2}^n z_i c_i b_1 \\ &= v, \end{aligned}$$

womit wir auch $\|u - b_1\| \leq d$ bewiesen haben. Damit ist auch (B', b_1, d) mit mindestens Wahrscheinlichkeit $1/2$ eine Ja-Instanz. \square

6.3 \mathcal{NP} -Schwere von SVP_∞ und CVP

Wie VAN EMDE BOAS 1981 zeigte, sind die Probleme SVP_∞ und CVP \mathcal{NP} -schwer. Der Beweis basiert auf einer zweistufigen Reduktion des PARTITION-Problems (das als \mathcal{NP} -vollständig bekannt ist) auf SVP_∞ bzw. CVP . Die verwendete Beweistechnik funktioniert allerdings nicht für das SVP in der Euklidischen Norm. Wir zeigen hier eine etwas neuere Reduktion von 3-SAT auf SVP_∞ . Für CVP wird auf den Originalartikel [EB81] verwiesen.

6 Komplexität von Gitterproblemen

Beim 3-SAT-Problem ist zu einem booleschen Ausdruck in konjunktiver Normalform mit höchstens drei Literalen pro Klausel zu entscheiden, ob eine erfüllende Belegung existiert. In der Berechnungsvariante von 3-SAT soll eine solche Belegung bestimmt werden. Da das Problem \mathcal{NP} -vollständig ist, folgt durch eine Reduktion auf das SVP_∞ auch die \mathcal{NP} -Schwere des SVP_∞ .

Definition 6.3.1 (Klausel, konjunktive Normalform) Seien x_1, \dots, x_n logische Variablen und $x_i^{-1} := \neg x_i, x_i^0 := \text{false}, x_i^1 := x_i$. Eine Klausel ist ein Ausdruck der Form

$$K = K(x_1, x_2, \dots, x_n) = x_1^{e_1} \vee x_2^{e_2} \vee \dots \vee x_n^{e_n},$$

wobei $e_1, \dots, e_n \in \{-1, 0, 1\}$ und $e_i = 0$, falls x_i nicht in K auftritt. Ein boolescher Ausdruck $H = H(x_1, \dots, x_n)$ liegt in konjunktiver Normalform (KNF) vor, falls H von der Gestalt

$$H(x_1, \dots, x_n) = \bigwedge_{j=1}^m K_j(x_1, \dots, x_n)$$

ist, wobei K_1, \dots, K_m Klauseln sind.

Definition 6.3.2 (3-SAT) Sei $H(x_1, \dots, x_n)$ eine KNF, bestehend aus höchstens drei Literalen pro Klausel, d. h. $\sum_{i=1}^n |e_{ij}| \leq 3$ für alle $j = 1, \dots, m$. Das Problem 3-SAT besteht darin, ein Tupel $(y_1, \dots, y_n) \in \{0, 1\}^n$ mit der Eigenschaft

$$H(y_1, \dots, y_n) = 1$$

zu finden oder zu zeigen, dass ein solches nicht existiert.

Nachfolgend werden wir eine geeignete Gitterbasis definieren und das 3-SAT-Problem auf eine Gitterreduktion in der Maximumnorm transformieren. Als Zwischenschritt wird zunächst 3-SAT auf ein $\{0, 1\}$ -ILP-Problem (Integer Linear Programming) reduziert. Dazu definieren wir die Variablen

$$c_j := 2 - \#\{i \mid e_{ij} = -1\}, \quad j = 1, \dots, m.$$

Eine entsprechende Instanz des ILP bekommt man durch das Ungleichungssystem

$$\sum_{i=1}^n e_{ij} y_i \geq 1 - c_j \quad \text{für } j = 1, \dots, m,$$

was äquivalent ist zu

$$\left| \sum_{i=1}^n e_{ij} y_i - c_j \right| \leq 1 \quad \text{für } j = 1, \dots, m,$$

wobei $y_1, \dots, y_n \in \{0, 1\}$ die gesuchten Unbekannten sind. Die Beträge lassen sich durch Aufspalten in jeweils zwei betragslose Ungleichungen eliminieren. Es folgt

Lemma 6.3.3 *Das angegebene $\{0, 1\}$ -ILP hat genau dann eine Lösung $y \in \{0, 1\}^n$, wenn $H(y) = 1$ ist.*

Beweis. Nach Konstruktion ist die j -te Ungleichung genau dann erfüllt, wenn die zugehörige Klausel K_j erfüllt wird. Der Ausdruck $H(y)$ ist eine Konjunktion aller Klauseln K_j und genau dann 1, wenn alle Ungleichungen erfüllt werden. \square

Mit Hilfe der Exponenten e_{ij} und der Koeffizienten c_j lässt sich nun die zugehörige Gitterbasis aufstellen. Sie wird beschrieben durch die Basismatrix

$$B = [b_1, \dots, b_{n+1}] = \begin{pmatrix} 2 & 0 & \cdots & 0 & -1 \\ 0 & 2 & \cdots & 0 & -1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 2 & -1 \\ e_{11} & e_{12} & \cdots & e_{1n} & -c_1 \\ e_{21} & e_{22} & \cdots & e_{2n} & -c_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ e_{m1} & e_{m2} & \cdots & e_{mn} & -c_m \\ 0 & 0 & \cdots & 0 & 1 \end{pmatrix}.$$

Wir benötigen also insgesamt $n + 1$ Vektoren $b_1, \dots, b_{n+1} \in \mathbb{Z}^{n+m+1}$. Ist $y \in \{0, 1\}^n$ derart, dass $H(y) = 1$, so sind die Basisvektoren gerade so gewählt, dass die Einträge des Lösungsvektors

$$b(y) = \sum_{i=1}^n y_i b_i + b_{n+1}$$

nur die Werte -1, 0 oder 1 annehmen können. Die genaue Wahl der Basis bewirkt sogar, dass es sich in dem Fall um einen kürzesten Gittervektor handelt.

Satz 6.3.4 *Sei $L = L(B)$ ein Gitter mit der Basismatrix B wie oben definiert. Genau dann ist ein Vektor $z \in L$ kürzester Gittervektor mit $\|z\|_\infty = 1$, wenn z von der Gestalt $z = \pm b(y)$ und $y \in \{0, 1\}^n$ eine erfüllende Belegung ist, d. h. $H(y) = 1$.*

Beweis.

„ \Rightarrow “: Da y eine erfüllende Belegung ist, sind nach Konstruktion alle Einträge von z aus $\{0, \pm 1\}$, also $\|z\|_\infty = 1$.

„ \Leftarrow “: Sei $z \in L$ mit $\|z\|_\infty = 1$ gegeben. Dann besitzt z eine Darstellung

$$z = \sum_{i=1}^{n+1} y'_i b_i,$$

wobei $y' = (y'_1, \dots, y'_{n+1}) \in \mathbb{Z}^{n+1}$. Da z ein kürzester Gittervektor ist, gilt $y_{n+1} = \pm 1$. Untersucht man die ersten n Einträge, so ergibt sich offensichtlich

$$(y'_1, \dots, y'_n) \in \{0, y_{n+1}\}^n,$$

6 Komplexität von Gitterproblemen

je nach Belegung von y_{n+1} . Die Vereinbarung

$$y := y'_{n+1} \cdot (y'_1, \dots, y'_n)$$

bewirkt $y \in \{0, 1\}^n$ und $(y, 1) = y'_{n+1} \cdot y'$. Wegen $\|z\|_\infty = 1$ und $y'_{n+1} = \pm 1$ kommen wir auf

$$\left| \sum_{i=1}^n e_{ij} y_i - c_j \right| = |y'_{n+1}| \cdot \left| \sum_{i=1}^n e_{ij} y'_i - c_j \right| \leq \|z\|_\infty \leq 1 \quad \text{für } j = 1, \dots, m.$$

Aufgrund von Lemma 6.3.3 ist y eine erfüllende Belegung. □

Beispiel 6.3.5 *Wir betrachten*

$$H(x_1, x_2, x_3) := (\neg x_1 \vee \neg x_2) \wedge (x_1 \vee x_2) \wedge (\neg x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_2 \vee \neg x_3)$$

mit den booleschen Variablen x_1, x_2, x_3 . Die oben beschriebene Konstruktion angewendet führt auf die Matrix

$$B = [b_1, b_2, b_3, b_4] = \begin{pmatrix} 2 & 0 & 0 & -1 \\ 0 & 2 & 0 & -1 \\ 0 & 0 & 2 & -1 \\ -1 & -1 & 0 & 0 \\ 1 & 1 & 0 & -2 \\ -1 & 1 & -1 & 0 \\ 0 & -1 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Durch Gitterreduktion bekommen wir den „kurzen“ Vektor

$$b = 0 \cdot b_1 + 1 \cdot b_2 + 0 \cdot b_3 + 1 \cdot b_4 = (-1, 1, -1, -1, -1, 1, -1, 1).$$

Die ersten drei Koeffizienten ergeben damit die Lösung $x_1 = 0, x_2 = 1, x_3 = 0$.

6.4 Nicht-Approximierbarkeit von SVP

In diesem Abschnitt werden wir eines der wichtigsten Resultate zur Komplexität von Gitterproblemen besprechen. 1998 gelang AJTAI (siehe [Aj97]) der Beweis, dass das SVP in der Euklidischen Norm \mathcal{NP} -schwer unter RUR-Reduktionen ist. Darüberhinaus konnte sogar gezeigt werden, dass ein $\varepsilon > 0$ existiert, für welches auch die Approximation des SVP auf einen Faktor $1 + 2^{-n^\varepsilon}$ unter RUR-Reduktionen \mathcal{NP} -schwer ist. Ein weiteres Resultat, dass nachfolgend vorgestellt werden soll, gelang MICCIANCIO vier Jahre später (siehe [Mic01a]). Demnach ist das SVP_γ in einer beliebigen \mathcal{L}_p -Norm (mit $p \geq 1$) \mathcal{NP} -schwer unter RUR-Reduktionen, wobei $\gamma \in [1, \sqrt[p]{2}]$.

Definition 6.4.1 *Das Problem CVP_γ besteht aus Instanzen (B, x, d) , wobei $B \in \mathbb{Z}^{m \times n}$ eine Gitterbasismatrix, $\gamma : n \rightarrow \mathbb{R}_{\geq 1}$ eine Funktion in der Gitterdimension, $x \in \mathbb{Z}^m$ ein Vektor und $d \in \mathbb{R}$ eine positive Konstante ist. Es ist*

1. (B, x, d) ist eine Ja-Instanz, falls $\|Bz - x\| \leq d$ für ein $z \in \{0, 1\}^n$.
2. (B, x, d) ist eine Nein-Instanz, falls $\|Bz - x\| > \gamma d$ für alle $z \in \mathbb{Z}^n$.

Der Beweis besteht in einer randomisierten Reduktion von CVP_γ ' (welches in [ABSS97] als \mathcal{NP} -schwer nachgewiesen wurde) auf das gegebene SVP_γ . Er folgt der folgenden Idee: Ist eine Gitterbasis B und ein Vektor x gegeben, so kann man nach dem nächsten Gitterpunkt u zu x fragen. Erweitert man jetzt das Gitter um den Vektor x , d. h. betrachtet $L' := L([B|x])$, und ist der kürzeste Vektor in L' von der Form $Bz - x$, so bekommt man mit $u := Bz$ eine Lösung des Originalproblems. Natürlich kann es passieren, dass bereits das Gitter $L(B)$ kürzere Vektoren enthält. Um diesem Problem zu begegnen wird das Gitter L' geeignet modifiziert. Dazu wird ein Gitter $L(A)$ und eine Kugel $\mathcal{B}(s, r)$ konstruiert, sodass die Abstände der Gitterpunkte in $L(A)$ immer größer sind als $\gamma' r$, wobei r der Radius der Kugel und $\gamma' > \gamma$ eine Konstante ist.

Lemma 6.4.2 *Für jede \mathcal{L}_p -Norm (mit $p \geq 1$) und jede Konstante $\gamma' \in [1, \sqrt[p]{2}]$ gibt es eine probabilistische Turingmaschine, die bei Eingabe einer Zahl $k \in \mathbb{Z}_+$ (polynomialzeitbeschränkt in k) zwei Ganzzahlen $m, r \in \mathbb{Z}_+$, eine Gitterbasis $A \in \mathbb{Z}^{(m+1) \times m}$, einen Vektor $s \in \mathbb{Z}^{m+1}$ und eine Transformationsmatrix $T \in \mathbb{Z}^{k \times m}$ ausgibt, wobei*

1. $\lambda_1(L(A)) > \gamma' r$
2. für alle $y \in \{0, 1\}^k$ existiert mit einer Wahrscheinlichkeit von mindestens $1 - 1/q(k)$ ein $z \in \mathbb{Z}^m$, sodass $y = Tz$ und $Az \in \mathcal{B}(s, r)$. $q(k)$ ist ein Polynom, dessen Grad unabhängig ist von k

Beweis. Der Beweis erfordert einige Vorbereitungen, weswegen auf [Mic01a] verwiesen wird. \square

Die Boolean-Vektoren y führen hier auf die (potentiellen) Lösungen des CVP_γ -Problems. Damit lässt sich nun der Hauptsatz beweisen.

Satz 6.4.3 *Für jede \mathcal{L}_p -Norm (mit $p \geq 1$) und jede Konstante $\gamma \in [1, \sqrt[p]{2}]$ ist das Problem SVP_γ \mathcal{NP} -schwer unter RUR-Reduktion.*

Beweis. Seien γ' und γ'' reelle Zahlen mit $\gamma < \gamma' < \sqrt[p]{2}$ und $\gamma'' > (\gamma^{-p} - \gamma'^{-p})^{-1/p}$, wobei wir ohne Einschränkung annehmen können, dass die Verhältnisse γ'/γ und γ''/γ rationale Zahlen darstellen. Wir beweisen den Satz durch RUR-Reduktion des Problems $\text{CVP}_{\gamma''}$ auf SVP_γ .

Sei (B, x, d) eine Instanz von $\text{CVP}_{\gamma''}$, wobei $B \in \mathbb{Z}^{n \times k}$, $x \in \mathbb{Z}^n$ und $d \in \mathbb{Z}$. Die zugehörige SVP_γ -Instanz ist (V, t) , wobei V und t nachstehend geeignet konstruiert werden.

Lemma 6.4.2 liefert uns bei Eingabe von k Zahlen $m, r \in \mathbb{Z}_+$, eine Gitterbasis $A \in \mathbb{Z}^{(m+1) \times m}$, einen Vektor $s \in \mathbb{Z}^{m+1}$ und eine Transformationsmatrix $T \in \mathbb{Z}^{k \times m}$ mit den Eigenschaften

1. $\|Az\|_p > \gamma' r$ für beliebige $z \in \mathbb{Z}^m \setminus \{0\}$

6 Komplexität von Gitterproblemen

2. für alle $y \in \{0, 1\}^k$ existiert mit einer Wahrscheinlichkeit von mindestens $1 - 1/q(k)$ ein $z \in \mathbb{Z}^m$, sodass $y = Tz$ und $\|Az - s\|_p \leq r$

Seien a und b zwei ganzzahlige Skalierungsfaktoren mit der Eigenschaft $\frac{a}{b} = \frac{r\gamma'}{d\gamma''}$ (diese gibt es, da γ'/γ'' als rational vorausgesetzt wurden). Wir definieren die Basismatrix V und die Zahl t als

$$V := \begin{bmatrix} a \cdot BT & a \cdot x \\ b \cdot A & b \cdot s \end{bmatrix}, \quad t := ad \cdot \frac{\gamma''}{\gamma}.$$

Durch geeignetes „Erweitern“ können wir a und b so wählen, dass t ganzzahlig ist.

Nehmen wir jetzt an, dass mit (B, x, d) eine Nein-Instanz von $\text{CVP}_{\gamma''}$ vorliegt. Zu zeigen ist, dass die korrespondierende SVP_{γ} -Instanz (V, t) ebenfalls eine Nein-Instanz ist. Wir betrachten einen von 0 verschiedenen ganzzahligen Vektor

$$u := \begin{bmatrix} z \\ w \end{bmatrix} \in \mathbb{Z}^{m+1}, \quad w \in \mathbb{Z}.$$

Zu zeigen ist $\|Vu\|_p^p > (\gamma t)^p$. Es gilt

$$\|Vu\|_p^p = (a \cdot \|By + wx\|_p)^p + (b \cdot \|Az + ws\|_p)^p,$$

wobei $y = Cz$ mit einer Matrix $C \in \mathbb{Z}^{k \times m}$. Es genügt, einen der beiden Fälle

$$a \cdot \|By + wx\|_p > \gamma t \quad \text{oder} \quad b \cdot \|Az + ws\|_p > \gamma t$$

zu zeigen. Dazu machen wir eine Fallunterscheidung:

1. $w \neq 0$. Nach Definition von $\text{CVP}_{\gamma''}$ gilt

$$a \cdot \|By + wx\|_p > a \cdot \gamma'' d = \gamma t.$$

2. $w = 0$. Dann muss $z \neq 0$ sein und wir haben

$$b \cdot \|Az + ws\|_p = b \cdot \|Az\|_p > b \cdot \gamma' r = \gamma t.$$

Jetzt nehmen wir an, (B, x, d) ist eine Ja-Instanz. Dann wissen wir, dass ein Boolean-Vektor $y \in \{0, 1\}^k$ existiert, für den $\|By - x\|_p \leq d$ erfüllt ist. Gemäß Lemma 6.4.2 haben wir einen Vektor $z \in \mathbb{Z}^m$ mit $y = Tz$ und $\|Az - s\| \leq r$. Definiere

$$u := \begin{bmatrix} z \\ -1 \end{bmatrix} \in \mathbb{Z}^{m+1}.$$

Für die Länge des zugehörigen Gittervektors bekommen wir damit

$$\begin{aligned} \|Vu\|_p^p &= (a \cdot \|By - x\|_p)^p + (b \cdot \|Az - s\|_p)^p \\ &\leq (ad)^p + (br)^p \\ &= \left(\frac{\gamma t}{\gamma''}\right)^p + \left(\frac{\gamma t}{\gamma'}\right)^p \\ &\leq t^p \gamma^p ((\gamma^{-p} - \gamma'^{-p}) + \gamma'^{-p}) \\ &= t^p. \end{aligned}$$

Damit ist auch (V, t) eine Ja-Instanz von SVP_{γ} . □

7 Anhang

7.1 Meilensteine der Gittertheorie

Hier wird noch einmal eine kurze Zusammenfassung der wesentlichen Resultate aus der Gittertheorie und ihren Anwendungen gegeben. Aufgeführt sind auch Ergebnisse, die in der Diplomarbeit noch nicht erwähnt wurden.

Jahr	Entdeckung/Ergebnis
1981	VAN EMDE BOAS beweist die \mathcal{NP} -Vollständigkeit der Probleme CVP und SVP_∞ .
1982	Der L^3 -Algorithmus wird vorgestellt. Mit ihm ergibt sich das wichtige Resultat der Approximierbarkeit des kürzesten Gittervektors auf einen Faktor $(2/\sqrt{3})^n$.
1983	KANNAN gibt einen Algorithmus an, mit dem sich SVP und CVP bei fixierter Gitterdimension in Polynomialzeit lösen lassen. Die Abhängigkeit von der Gitterdimension n ist $n^{n+o(n)}$.
1986	BABAI zeigt die Approximierbarkeit des CVP auf einen Faktor $2^{n/2}$ durch Vorstellung der Round-Off- sowie der Nearest-Plane-Methode. Beides sind die wichtigsten Methoden zur näherungsweise Lösung des CVP und bestimmen heute wesentlich die Sicherheit gitterbasierter Kryptosysteme.
1987	SCHNORR zeigt die Approximierbarkeit des CVP auf einen Faktor $(1 + \varepsilon)^n$ für alle $\varepsilon > 0$. Ein von SCHNORR vorgestellter verbesserter L^3 -Algorithmus erreicht eine Approximationsgüte von $2^{\mathcal{O}(n(\ln \ln n)^2 / \ln n)}$.
1990	LAGARIAS, LENSTRA und SCHNORR zeigen $\text{SVP}_\gamma, \text{CVP}_\gamma \in \mathcal{NP} \cap \text{coNP}$ für $\gamma(n) = n^{1.5}$.
1993	Nach einem Ergebnis von BANASZCZYK gilt $\text{SVP}_n, \text{CVP}_n \in \mathcal{NP} \cap \text{coNP}$. ARORA et al. zeigen, dass die Approximation von CVP auf jeden konstanten Faktor $c \geq 1$ \mathcal{NP} -schwer ist.
1997	AJTAI weist die \mathcal{NP} -Schwere von SVP unter RUR-Reduktionen nach. Auch die Approximation des kürzesten Gittervektors auf $1 + n^{-\varepsilon}$ ist \mathcal{NP} -schwer unter RUR-Reduktionen (CAI/NERUKAR). AJTAI und DWORK zeigen die Worst-Case/Average-Case-Äquivalenz zweier Gitterprobleme Auf der Crypto'97 wird das GH-System vorgestellt.

1998	Für einen Approximationsfaktor $\gamma(n) = \sqrt{\frac{n}{\ln n}}$ liegen die Probleme SVP_γ und CVP_γ in $\mathcal{NP} \cap \text{coAM}$, wie GOLDREICH und GOLDWASSER zeigen.
1999	DINUR, KINDLER, RAZ und SAFRA zeigen die \mathcal{NP} -Schwere von CVP_γ für einen polylogarithmischen Faktor $\gamma(n) = n^{\frac{c}{\ln \ln n}}$ ($c > 0$ ist eine beliebige Konstante).
2001	MICCIANCIO erweitert das Ergebnis von AJTAI für das SVP auf jede beliebige \mathcal{L}_p -Norm. Von AJTAI, KUMAR und SIVAKUMAR wird ein probabilistischer Siebalgorithmus vorgestellt, der einen kürzesten Gittervektor in $2^{\mathcal{O}(n)}$ Schritten findet. Weiterhin kann ein (allerdings probabilistischer) Approximationsfaktor von $2^{\mathcal{O}(n \ln \ln n / \ln n)}$ für das SVP gezeigt werden. MICCIANCIO stellt eine verbesserte Variante des GGH-Systems mit deutlich reduzierten Schlüssellängen vor.
2004	Unter quasi-polynomiellen randomisierten Reduktionen ist SVP_γ \mathcal{NP} -schwer für $\gamma(n) = 2^{(\ln n)^{\frac{1}{2}-\varepsilon}}$ für jedes $\varepsilon > 0$ (Beweis durch KHOT).
2005	AHARONOV, REGEV: Es existiert eine Konstante $c > 0$, sodass für $\gamma(n) = c\sqrt{n}$ die Probleme SVP_γ und CVP_γ in $\mathcal{NP} \cap \text{coNP}$ liegen. KHOT: Aus $\mathcal{NP} \neq \mathcal{RP}$ folgt für jedes $p > 1$ die Nichtapproximierbarkeit von SVP in der \mathcal{L}_p -Norm auf einen beliebigen konstanten Faktor.

7.2 Vermutungen und offene Probleme

Trotz beachtlicher Fortschritte der letzten Jahre verbleiben immer noch einige offene Fragen. Nachfolgend eine kleine Übersicht.

- Unbeantwortet ist die Frage nach der Komplexität der Approximationsprobleme SVP_γ und CVP_γ für einen polynomiellen Approximationsfaktor $\gamma(n) = n^c$, $c > 0$.
- Damit verbunden ist die Frage, ob es ein $c > 0$ gibt, sodass SVP_γ und CVP_γ \mathcal{NP} -schwer sind für $\gamma(n) = n^c$.
- Vermutlich ist die Approximation auf einen Faktor $\gamma(n) = \sqrt{n}$ nicht \mathcal{NP} -schwer (beide Probleme liegen in $\mathcal{NP} \cap \text{coAM}$).
- Interessant wäre ein Zusammenhang zu zahlentheoretischen Problemen wie dem Faktorisierungsproblem oder dem Diskrete-Logarithmus-Problem (z. B.

„Lassen sich ganze Zahlen unter Verwendung eines SVP_n -Orakels effizient faktorisieren?“). Damit wäre ein Vergleich von Kryptosystemen wie RSA mit gitterbasierten gegeben.

7.3 Implementationen

Die folgenden bekannten Softwarepakete enthalten u. a. Routinen zur Gitterbasisreduktion, Berechnung nächster Gitterpunkte bzw. sukzessiver Minima und zur Berechnung der HERMITE-Normalform:

- MAGMA: *LLL()*, *PairReduce()*, *Seysen()*, *ShortestVectors()*, *ClosestVectors()*, *SuccessiveMinima()*
- PARI/GP: *linddep()*, *mathnf()*, *matkerint()*, *qflll()*
- LiDIA: Paket „LT“, *lll()*
- Mathematica: *LatticeReduce()*

Das Paket LiDIA wird von einer mehrköpfigen Arbeitsgruppe (u. a. der TH Darmstadt und der Universität Saarbrücken) gepflegt und enthält auch Realisierungen von Public-Key-Systemen wie dem GGH-System. L^3 -Gitterreduktionen können hier optional parallelisiert (durch Aufteilung der Gitterbasis in Blöcke) durchgeführt werden.

Eine eigene einfache C-Implementation (Paar-, Längen-, L^3 -Reduktion) kann man von <http://www2.informatik.hu-berlin.de/~schoenbe/gitter> herunterladen. Die Paarreduktion arbeitet mit beliebig großen Integern. Längen- und L^3 -Reduktion arbeiten dagegen mit fester (double-)Genauigkeit für die Orthogonalisierungsschritte. Bei großer Gitterdimension kann es daher aufgrund der Rundungsfehler theoretisch zu falschen Ergebnissen kommen.

Aus eigenen Experimenten hat sich die Beobachtung ergeben, dass die Paarreduktion bei großen Matrixeinträgen mitunter sehr lange Laufzeiten aufweist – auch bei kleiner Gitterdimension. Viele Reduktionsschritte erreichen eine nur minimale Verkürzung der Basisvektoren, was die These der (in der Länge der Einträge) exponentiellen Laufzeit untermauert.

Literaturverzeichnis

- [ABSS97] S. Arora, L. Babai, J. Stern, E. Z. Sweedyk: *The Hardness of Approximate Optima in Lattices, Codes, and Systems of Linear Equations*. Journal of Computer and System Sciences, Vol. 54, 317-331, 1997.
- [AD97] Miklós Ajtai, Cynthia Dwork: *A Public-Key Cryptosystem with Worst-Case-/Average-Case Equivalence*. Technischer Bericht Revision 01 of ECCC Report TR96-065, Electronic Colloquium on Computational Complexity, Trier 1997.
- [AhReg05] Dorit Aharonov, Oded Regev: *Lattice Problems in $\mathcal{NP} \cap \text{co}\mathcal{NP}$* . Journal of the ACM 52(5), 749-765, 2005.
- [Aj96] Miklós Ajtai: *Generating Hard Instances of Lattice Problems*. Technischer Bericht ECCC Report TR96-007, Electronic Colloquium on Computational Complexity, Trier 1996.
- [Aj97] Miklós Ajtai: *The Shortest Vector Problem in L_2 is \mathcal{NP} -hard for Randomized Reductions*. Electronic Colloquium on Computational Complexity, ECCC TR 97-047, 1997.
- [AKS01] Miklos Ajtai, Ravi Kumar, D. Sivakumar: *A Sieve Algorithm for the Shortest Lattice Vector Problem*. ACM Symposium on Theory of Computing, 601-610, 2001.
- [Art98] Michael Artin: *Algebra*. Birkhäuser 1998.
- [Bab86] László Babai: *On Lovász' lattice reduction and the nearest lattice point problem*. Combinatorica, Vol. 6 (1), 1-13, 1986.
- [BoDu00] Dan Boneh, Glenn Durfee: *Cryptanalysis of RSA with private key d less than $N^{0.292}$* . IEEE Transactions on Information Theory, Vol 46 (4), 1339-1349, 2000.
- [EB81] Peter van Emde Boas: *Another NP-complete Partition Problem and the Complexity of Computing Short Vectors in Lattices*. TR 81-04, Mathematics Department, University of Amsterdam, 1981.
- [Fil02] Bartol Filipović: *Implementierung der Gitterbasenreduktion in Segmenten*. Diplomarbeit, J. W. Goethe-Universität Frankfurt am Main 2002.

Literaturverzeichnis

- [GaGe03] Joachim von zur Gathen, Jürgen Gerhard: *Modern Computer Algebra*. Cambridge University Press 2003.
- [GGH97] Oded Goldreich, Shafi Goldwasser, Shai Halevi: *Public-Key Cryptosystems from Lattice Reduction Problems*. Proc. of Crypto'97, Lecture Notes in Computer Science, Vol. 1294, 112-131, Springer 1997.
- [GGH97a] Oded Goldreich, Shafi Goldwasser, Shai Halevi: *Eliminating Decryption Errors in the Ajtai-Dwork Cryptosystem*. Proc. of Crypto'97, Lecture Notes in Computer Science, Vol. 1294, 105-111, Springer 1997.
- [Hoe63] Wassily Hoeffding: *Probability Inequalities for Sums of Bounded Random Variables*. Journal of the American Statistical Association, Vol. 58 (301), 13-30, 1963.
- [HPS96] Jeffrey Hoffstein, Jill Pipher, Joseph H. Silverman: *NTRU: A Ring-Based Public Key Cryptosystem*. Brown University, Providence (USA) 1996.
- [Kan83] Ravi Kannan: *Improved Algorithms for Integer Programming and Related Lattice Problems*. Proceedings of the 15. Annual ACM Symposium on Theory of computing, 193-206, 1983.
- [Koy99] Henrik Koy: *Angriffe auf das GGH-Kryptosystem mittels Gitterreduktion in Blöcken*. Diplomarbeit, J. W. Goethe-Universität Frankfurt am Main 1999.
- [LLL82] A. K. Lenstra, H. W. Lenstra, L. Lovász: *Factoring Polynomials with Rational Coefficients*. Mathematische Annalen 261, 515-534, 1982.
- [Lud04] Christoph Ludwig: *The Security and Efficiency of Micciancio's Cryptosystem*. Technische Universität Darmstadt, 2004.
- [Mic98] Daniele Micciancio: *On the Hardness of the Shortest Vector Problem*. Massachusetts Institute of Technology, 1998.
- [Mic01] Daniele Micciancio: *Improving Lattice Based Cryptosystems Using the Hermite Normal Form*. University of California, San Diego 2001.
- [Mic01a] Daniele Micciancio: *The Shortest Vector in a Lattice is Hard to Approximate to within Some Constant*. SIAM Journal on Computing, Vol. 30 (6), 2008-2035, 2001.
- [MicGw02] Daniele Micciancio, Shafi Goldwasser: *Complexity of Lattice Problems: A Cryptographic Perspective*. Springer 2002.
- [MicWar01] Daniele Micciancio, Bogdan Warinschi: *A Linear Space Algorithm for Computing the Hermite Normal Form*. ISSAC, 231-236, 2001.

- [Ngu99] Phong Q. Nguyen: *Cryptoanalysis of the Goldreich-Goldwasser-Halevi Cryptosystem from Crypto'97*. Advances in Cryptology - Crypto'99, 1999.
- [Pape04] Sebastian Pape: *Sicherheitsmodelle für das Ajtai-Dwork Kryptosystem*. Diplomarbeit, Technische Universität Darmstadt 2004.
- [PoZas89] Michael Pohst, Hans Zassenhaus: *Algorithmic Algebraic Number Theory*. Cambridge University Press 1989.
- [Schn06] Claus-Peter Schnorr: *Gitter und Kryptographie*. Vorlesung, J. W. Goethe-Universität Frankfurt am Main 2006.
- [Schn87] Claus-Peter Schnorr: *A Hierarchy of Polynomial Time Lattice Basis Reduction Algorithms*, Theoretical Computer Science, Vol. 53, 201-224, 1987.
- [SchnEu94] C. P. Schnorr, M. Euchner: *Lattice Basis Reduction: Improved Practical Algorithms and Solving Subset Sum Problems*, Mathematical Programming 66, 181-199, 1994.
- [Spra94] Oliver van Sprang: *Basisreduktionsalgorithmen für Gitter kleiner Dimension*. Dissertation, Universität des Saarlandes, Saarbrücken 1994.

Literaturverzeichnis

Selbständigkeitserklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel angefertigt habe.

Berlin, den 14. Mai 2008

Datum, Unterschrift

Einverständniserklärung

Ich erkläre hiermit mein Einverständnis, dass die vorliegende Arbeit in der Bibliothek des Institutes für Informatik der Humboldt-Universität zu Berlin ausgestellt werden darf.

Berlin, den 14. Mai 2008

Datum, Unterschrift