Effiziente Algorithmen für Probleme in Gittern über $\mathbb{R}^d$

Diplomarbeit
von
Sandro Amendola

nach einem Thema von
Prof. Dr. Buchmann
Fachbereich 14 – Informatik –
Universität des Saarlandes

25. September 1995

Hiermit versichere ich an Eides Statt, daß ich diese Arbeit selbständig verfaßt und keine anderen als die angegebenen Quellen benutzt habe.

Saarbrücken, den 8. Mai 1995

Sandro Amendola

An dieser Stelle möchte ich mich bei Herrn Prof. Dr. Buchmann für das interessante Thema und die gute Betreuung bedanken.

Desweiteren danke ich Patrik Bentz und Dr. Oliver van Sprang für die aufschlußreichen Diskussionen und das Korrekturlesen der Arbeit.

Meinen Eltern gilt mein besonderer Dank, denn sie haben mir das Studium erst ermöglicht.

Mit der Annahme, das menschliche Leben könne nach den Grundsätzen des Verstandes geleitet werden, verneint man die Möglichkeit des Lebens selbst.

Leo N. Tolstoj
Krieg und Frieden

Inhaltsverzeichnis

Vorwort

Seien $n, d \in \mathbb{N}$ und $\mathbb{R}^d$ der d-dimensionale euklidische Raum mit dem Standardskalarprodukt $\langle \cdot, \cdot \rangle$ und der euklidischen Norm $\| \cdot \| = \langle \cdot, \cdot \rangle^{1/2}$. Die vorliegende Arbeit befaßt sich mit algorithmischen Problemen in n-dimensionalen Gittern über $\mathbb{R}^d$. Es sei stets eine Basis $B = (\underline{b}_1, \ldots, \underline{b}_n)$ des Gitters L, d.h. linear unabhängige $\underline{b}_1, \ldots, \underline{b}_n \in L$ mit

$$L = L(B) = \sum_{i=1}^{n} \mathbb{Z} \cdot \underline{b}_i,$$

gegeben. Eine solche Basis ist bis auf unimodulare Transformation von rechts eindeutig bestimmt. Gegenstand der Arbeit ist die Analyse von Algorithmen zur Berechnung von kurzen und kürzesten Gittervektoren, reduzierten Gitterbasen, d.h. Basen aus kurzen Gittervektoren und nahen und nächsten Gitterpunkten zu Punkten in $\mathbb{R}^d$. Die Ermittlung kurzer Gittervektoren wird mit der Berechnung reduzierter Basen einhergehen.

Die Auswahl dieser Probleme ist darin begründet, daß mit deren Lösung auch verschiedene Aufgaben aus anderen Bereichen der Informatik und der Mathematik leicht bewältigt werden können (Hinweise dazu in [Ka87b]). So gibt es Fragen in der Kryptographie und der diskreten Optimierung, die sich über die Berechnung reduzierter Gitterbasen beantworten lassen.

Probleme, wie die Ermittlung einer Basis $\underline{b}_1, \ldots, \underline{b}_n$ von L mit minimalem Produkt $\|\underline{b}_1\| \cdot \ldots \cdot \|\underline{b}_n\|$ oder eines nächsten Gitterpunktes in L von $\underline{b}_0 \in \mathbb{Z}^d \backslash \{\underline{0}\}$ sind $\mathcal{NP}$-hart, selbst wenn vorausgesetzt wird, daß $L \subseteq \mathbb{Z}^d$ ist. Für die in dieser Arbeit behandelten Probleme in Gittern über $\mathbb{Z}^d$ wurden in den achtziger Jahren erstmals Algorithmen gefunden, deren Laufzeiten allein von der Gitterdimension, nicht mehr jedoch von der maximalen Länge eines Vektors der gegebenen Basis, exponentiell abhängen ([He85], [Ka87a] und [Sch87]). Zu deren Entstehung hat wesentlich der LLL-Algorithmus [LeLeLo82] beigetragen, der in Polynomzeit eine reduzierte Basis eines Gitters über $\mathbb{Z}^d$ berechnet, die im ungünstigen Fall allerdings noch „lange“ Gittervektoren enthalten kann. Mit dem Algorithmus lassen sich auch nahe Gitterpunkte berechnen [Ba86]. Kapitel 2 wird die angesprochenen Algorithmen zum Thema haben, dabei werden besonders die Teile des Korrektheitsbeweises und der Laufzeitanalyse im Mittelpunkt der Untersuchung stehen, die für das Verständnis des Algorithmus von Bedeutung sind.

Die genannten Probleme können auch in Gittern über $\mathbb{Q}^d$ durch Multiplikation der Basis mit einem geeigneten ganzzahligen Vielfachen auf entsprechende Probleme in Gittern über $\mathbb{Z}^d$ zurückgeführt und über die Division der Ergebnisvektoren durch das gewählte Vielfache exakt gelöst werden.

Für Gitter, die Gitterpunkte mit echt reellwertigen Einträgen enthalten, vermag man die Probleme nur in einem Näherungsgitter zu lösen. Nun lassen sich mit den Algorithmen zusätzlich die Transformationsvektoren ermitteln, die multipliziert mit der Basis des Gitters die reduzierten Basisvektoren bzw. den nächsten Gitterpunkt ergeben. Wir werden in Kapitel 3 zeigen, von welcher Genauigkeit das Näherungsgitter sein muß, damit die darin gewonnenen Transformationsvektoren auch mit der Basis des Ausgangsgitters über $\mathbb{R}^d$ Gittervektoren ausreichender Güte erzeugen. Darüber hinaus wird untersucht werden, welche Zeit für die Berechnung der Transformationsvektoren notwendig ist. Den Überlegungen wird die Arbeit [Bu94] zu Grunde liegen.

Bezeichnungen

$\mathbb{N}$	Menge der natürlichen Zahlen
$\mathbb{Z}$	Menge der ganzen Zahlen
$\mathbb{Q}$	Menge der rationalen Zahlen
$\mathbb{R}$	Menge der reellen Zahlen
$M_{\mathrm{op}\ a_0}$	Menge der Zahlen $a \in M$ für die a op a_0 gilt, op $\in \{>, \geq\}$
M^d	Menge der d-stelligen Vektoren mit Einträgen aus M, Vektoren werden durch unterstrichene Kleinbuchstaben notiert
$\underline{a}^\top$	Transponierter Vektor zu $\underline{a}$
$M^{d \times n}$	Menge der $d \times n$-Matrizen mit Einträgen aus M, Matrizen werden durch Großbuchstaben notiert
$A^\top$	Transponierte Matrix zu A
L	Gitter
$B = (\underline{b}_1, \ldots, \underline{b}_n)$	Basis eines Gitters L bzw. Matrix mit Basisvektoren als Spaltenvektoren
$L(B)$	von B erzeugtes Gitter
$V(L)$	von den Gittervektoren aus L erzeugter Unterraum des Vektorraums $\mathbb{R}^d$
$\langle\langle \underline{b}_1, \ldots, \underline{b}_n \rangle\rangle$	von $\underline{b}_1, \ldots, \underline{b}_n$ erzeugter Unterraum des Vektorraums $\mathbb{R}^d$
$\langle\langle \underline{b}_1, \ldots, \underline{b}_n \rangle\rangle^\perp$	zu $\langle\langle \underline{b}_1, \ldots, \underline{b}_n \rangle\rangle$ orthogonaler Unterraum im Vektorraum $\mathbb{R}^d$
$\underline{a}(k)$	Projektion von $\underline{a}$ auf das orthogonale Komplement von $\langle\langle \underline{b}_1, \ldots, \underline{b}_{k-1} \rangle\rangle$
$L_k, L_k(\underline{b}_1, \ldots, \underline{b}_n)$	von $\underline{b}_k(k), \ldots, \underline{b}_n(k)$ erzeugtes Gitter
$\underline{b}_1^\star, \ldots, \underline{b}_n^\star$	Gram-Schmidt-Zerlegung von $\underline{b}_1, \ldots, \underline{b}_n$
$\lceil a \rceil$	nächste obere ganze Zahl zu $a \in \mathbb{R}$
$\lfloor a \rfloor$	nächste untere ganze Zahl zu $a \in \mathbb{R}$
$\lceil a \rfloor$	$\lceil a - 1/2 \rceil$
$\lfloor a \rceil$	$\lfloor a + 1/2 \rfloor$
$\log a$	Logarithmus von a zur Basis 2
γ_n	n-te Hermitesche Konstante

$\lambda_i(A)$	Länge des i-ten kleinsten Spaltenvektors einer Matrix A
$\lambda_i(L)$	i-tes sukzessives Minimum des Gitters L
$\xi_i(L)$	kleinste positive Größe, die von den Längen genau i linear unabhängiger Gittervektoren in L nicht überschritten wird
$\mu(\underline{b}_0, L)$	Abstand von $\underline{b}_0$ zu einem nächsten Gitterpunkt in L
$\mu(L)$	maximaler Abstand, den ein $\underline{b}_0 \in V(L)$ zu einem nächsten Gitterpunkt haben kann
$\nu(\underline{b}_0, L)$	Abstand von $\underline{b}_0$ zum zweitnächsten Gitterpunkt in L
$\mathrm{dft}(B)$	Defekt einer Basis B
$\det B$	Determinante einer quadratischen Matrix B
$\mathrm{d}(L)$	Gitterdeterminante von L
$\langle \underline{a}, \underline{b} \rangle$	Standardskalarprodukt von $\underline{a}$ und $\underline{b}$
$\lVert \underline{a} \rVert$	euklidische Norm eines Vektors $\underline{a}$
$\lvert \underline{a} \rvert$	Betrag des betragsgrößten Eintrags eines Vektors $\underline{a}$
$\lVert A \rVert$	Frobeniusnorm einer Matrix A
$\lvert A \rvert$	Betrag des betragsgrößten Eintrags einer Matrix A

Kapitel 1

Grundlagen

Zunächst wollen wir festlegen, was wir unter dem Begriff Algorithmus verstehen. Ein Algorithmus ist eine endliche Folge von Anweisungen, die bei Eingabe einer Folge ganzer Zahlen, der Eingabe des Algorithmus, eine Folge ganzer Zahlen ausgibt. Diese Folge nennen wir die Ausgabe des Algorithmus [Bu92]. Ziel dieser Arbeit ist es, Algorithmen anzugeben, mit denen sich Probleme in Gittern über $\mathbb{R}^d$ durch Rechnen auf ganzen Zahlen *exakt* lösen lassen. D.h. wir wollen Computeralgebra betreiben und nicht Numerik, wo man sich mit Näherungslösungen begnügt. Dieser Ansatz führt zu Schwierigkeiten, wenn das Problem oder dessen Lösung echt reellwertige Zahlen enthält. Kritisch ist in unserem Fall, wenn das Gitter, welches dem Problem zu Grunde liegt, Gitterpunkte mit echt reellwertigen Einträgen beinhaltet. In der Computeralgebra begnügt man sich dann mit einer *symbolischen* Lösung, welche man durch exaktes Rechnen, z.B. auf ganzen Zahlen, erhalten kann.

So werden wir zeigen, wie sich ein $\underline{t} \in \mathbb{Z}^d$ exakt ermitteln läßt, so daß von einem Gitter $L(B) \subset \mathbb{R}^d$ gerade $B\underline{t}$ ein kürzester Gittervektor ist. Den Algorithmen genügt hierzu als Eingabe eine rationale Näherung genügend großer Genauigkeit für die Basis des Gitters (die Definition dieser Begriffe erfolgt erst im entsprechenden Abschnitt).

Ein Grundgedanke wird sein, eine Frage auf ein Problem zurückzuführen, für dessen Lösung es einen Algorithmus gibt, um bereits entwickelte Verfahren erneut anwenden zu können.

1.1 Komplexität von Algorithmen über $\mathbb{Z}$

Wir wollen das Verhalten der Algorithmen in Abhängigkeit der Eingabe analysieren. Weil die Algorithmen ausschließlich mit ganzen Zahlen beliebiger Länge arbeiten, ist es sinnvoll, einige Begriffe entsprechend zu definieren. Mit Algorithmen lassen sich Probleme lösen. Es gibt unendlich viele *Fragestellungen (Problemstellungen)* zu einem Problem. In unserem Fall ist beispielsweise eine Fragestellung des Problems einen kürzesten Gittervektor in einem Gitter $L(B)$ über $\mathbb{R}^d$ zu berechnen: Bestimme den kürzesten Gittervektor in $L = \mathbb{Z}\cdot\binom{1345}{159} + \mathbb{Z}\cdot\binom{-112}{315}$.

Einer Problemstellung ordnen wir ein Tupel nichtnegativer Zahlen, die *Problemgrößen*, zu. Sie bestimmen die Eingabe und ergeben sich aus der Problemstellung. Wir kommen weiter unten darauf zurück.

Für die Ausführung eines Algorithmus auf einer Rechenmaschine ist ein *Rechenaufwand* notwendig. Von Bedeutung ist dabei vor allem Rechenzeit und Speicherplatz. Damit die Untersuchung des Rechenaufwandes unabhängig von speziellen Computern ist, wird ein formales Berechnungsmodell zu Grunde gelegt. Wir definieren die *Komplexität* eines Algorithmus als den erforderlichen Rechenaufwand bei einer konkreten Realisierung des Algorithmus innerhalb des Berechnungsmodells [Duden]. Als Berechnungsmodell wählen wir die in ([Me84] Seite 1-12) vorgestellte Maschine RASP. Dieses Modell erlaubt es, daß jede ganze Zahl in eine Speicherzelle paßt. Die Speicherverwaltung ist dadurch sehr einfach.

Bei der Ermittlung der Rechenzeit müssen wir allerdings die Größe der in einem Rechenschritt vorkommenden Operanden mitberücksichtigen. Wir werden nur die Rechenzeit analysieren, der Speicherplatz ergibt sich aus den Abschätzungen der bei den Berechnungen auftretenden Operanden. Zuvor muß jedoch eine Übereinkunft getroffen werden, wie eine ganze Zahl zu repräsentieren ist. Hierzu benutzen wir die *Binärdarstellung*. Eine ganze Zahl $z \in \mathbb{Z}\backslash\{0\}$ wird dargestellt als $(\alpha_{n-1}, \ldots, \alpha_0, \beta) \in \{0,1\}^{n+1}$, wobei $z = (-1)^\beta \sum_{i=0}^{n-1} \alpha_i 2^i$, $\alpha_{n-1} \neq 0$. Die Zahl 0 wird dargestellt als $(0,0)$. Wir definieren die *binäre Länge* einer ganzen Zahl $z \neq 0$ als $\lfloor \log |z| \rfloor + 2$, für $z = 0$ als 2.

Die Addition, Subtraktion, Multiplikation, Division und den Vergleich zweier Zahlen bezeichnen wir als *arithmetische Operationen*. Wir legen fest, daß wir das Komplexitätsverhalten von Algorithmen im schlimmsten Fall untersuchen, und definieren die *maximale Laufzeit* oder kurz *Laufzeit* eines Algorithmus als die maximale Anzahl an arithmetischen Operationen auf ganzen Zahlen einer maximalen binären Länge, die bei der Ausführung des Algorithmus für eine Eingabe einer bestimmten Länge gemacht werden. Wir werden nicht darauf eingehen, welchen Zeitbedarf eine bestimmte arithmetische Operation in Abhängigkeit der binären Länge der Operanden hat. Wir verweisen dazu auf [Bu92] oder [Mu94].

Die *Eingabelänge* ist die Summe der binären Längen ihrer Elemente, die durch die Problemgrößen einer Problemstellung bestimmt sind. Die beiden oberen Schranken, die bei der Laufzeitanalyse ermittelt werden, können sowohl als Funktionen über der Eingabelänge als auch als Funktionen über den Problemgrößen betrachtet werden. Wir werden oft die letztere Anschauungsweise vorziehen. Wir interessieren uns nicht für die genauen Werte der Schranken, sondern nur für den qualitativen Verlauf, die *Ordnung*, der Funktionen. Für dessen Darstellung ist die O- und o-Notation geeignet.

Definition 1.1 (O- und o-Notation) *Seien $k_1, k_2 \in \mathbb{Z}_{\geq 0}$, $k_1 \geq 1$ oder $k_2 \geq 1$, $M_i \subseteq \mathbb{R}_{\geq 0}$ für $1 \leq i \leq k_1$ und $f, g : M_1 \times \ldots \times M_{k_1} \times \mathbb{Z}_{\geq 0}^{k_2} \mapsto \mathbb{R}_{\geq 0}$.* f **ist gleich groß** o **von** g, *$f = O(g)$, genau dann wenn es ein $c > 0$ und $m_1, \ldots, m_{k_2} \in \mathbb{Z}_{\geq 0}$ gibt, so daß für alle $a_i \in M_i$, $1 \leq i \leq k_1$ und $n_i \geq m_i$, $1 \leq i \leq k_2$, gilt*

$$f(a_1, \ldots, a_{k_1}, n_1, \ldots, n_{k_2}) \leq c \cdot g(a_1, \ldots, a_{k_1}, n_1, \ldots, n_{k_2}).$$

f **ist gleich klein** o **von** g, *$f = o(g)$, genau dann wenn es für alle $\epsilon > 0$ $m_1, \ldots, m_{k_2} \in \mathbb{Z}_{\geq 0}$ gibt, so daß für alle $a_i \in M_i$, $1 \leq i \leq k_1$ und $n_i \geq m_i$, $1 \leq i \leq k_2$, gilt*

$$f(a_1, \ldots, a_{k_1}, n_1, \ldots, n_{k_2}) < \epsilon \cdot g(a_1, \ldots, a_{k_1}, n_1, \ldots, n_{k_2}).$$

Für die Darstellung der Algorithmen benutzen wir Sprachelemente aus C und PROSA ([LoMeWi86] Seite 149-64 und Seite 226-32). An einigen Stellen enthalten die Algorithmen auch informalen Text, aus dem jedoch klar hervorgeht, was genau zu tun ist oder wo Informationen zu finden sind, mit denen eine Folge von formalen Anweisungen konstruiert werden kann. Wir begnügen uns oft damit, nur eine Beschreibung dafür anzugeben, wie ein Algorithmus zu konstruieren ist.

1.2 Grundbegriffe zu Gittern

Definition 1.2 (Gitter) *Ein* **Gitter** *L ist eine Menge von Punkten im $\mathbb{R}^d$, die eine diskrete, additive Untergruppe des $\mathbb{R}^d$ bildet. L genügt also zwei Bedingungen:*
1. Additive Untergruppe. Ist $\underline{b}_1$, $\underline{b}_2 \in L$, dann ist $-\underline{b}_1 \in L$ und $\underline{b}_1 + \underline{b}_2 \in L$.
2. Es gibt ein $\delta > 0$ derart, daß für beliebige $\underline{b}_1$, $\underline{b}_2$ mit $\underline{b}_1 \neq \underline{b}_2$ gilt: $\|\underline{b}_1 - \underline{b}_2\| \geq \delta$.
Elemente aus L heißen **Gitterpunke** *oder* **Gittervektoren**.

Bemerkung 1.3 Ist eine Punktmenge diskret, dann enthält jede Kugel im $\mathbb{R}^d$ nur endlich viele Punkte dieser Menge.

Durch folgende Aussage wird ein Gitter anschaulicher:

Satz 1.4 *Sei $L \subset \mathbb{R}^d$. Es gibt über $\mathbb{R}$ linear unabhängige Vektoren $\underline{b}_1, \ldots, \underline{b}_n \in \mathbb{R}^d$ mit*

$$L = \sum_{i=1}^{n} \mathbb{Z} \cdot \underline{b}_i.$$

Genau dann ist L ein Gitter.

Beweis: Siehe Abschnitt III.1.2 (Lemma 1) und Abschnitt III.4 (Satz III.4) in [Ca59]. □

In Abbildung 1.1 auf der folgenden Seite wird ein Ausschnitt aus einem Beispielgitter widergegeben. Die Schnittpunkte der gepunkteten Linien stellen die Gitterpunkte dar. Die Linien sind in Richtung der erzeugenden Basisvektoren gezeichnet.

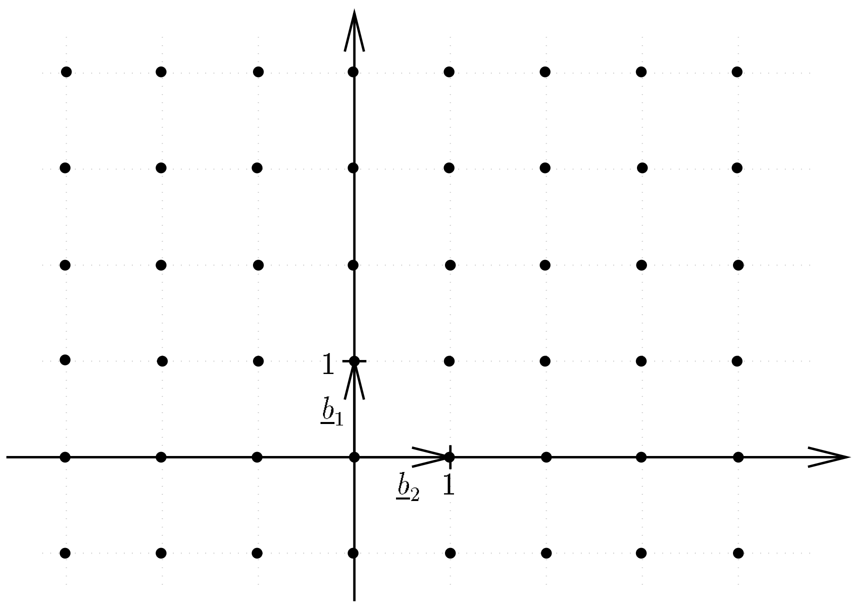

Abbildung 1.1: Das Gitter $L = \mathbb{Z} \cdot (1,0)^\top + \mathbb{Z} \cdot (0,1)^\top$

Die Notation für den von einer Teilmenge aus $\mathbb{R}^d$ aufgespannten linearen Unterraum sei $\langle\langle \cdots \rangle\rangle$.

Definition 1.5 (Gitterbasis, Dimension) $\underline{b}_1, \ldots, \underline{b}_n$ *aus Satz 1.4 heißen* **Gitterbasis** *oder kurz* **Basis** *von L; die Dimension n von $V(L) := \langle\langle \underline{b}_1, \ldots, \underline{b}_n \rangle\rangle$ heißt die* **Dimension des Gitters** *L. Sei L ein Gitter mit $L = \sum_{i=1}^{k} \mathbb{Z} \cdot \underline{b}_i$, dann nennen wir L* **das von $\underline{b}_1, \ldots, \underline{b}_k$ erzeugte Gitter** *und schreiben $L = L(\underline{b}_1, \ldots, \underline{b}_k)$.*

Für $\underline{b}_1, \ldots, \underline{b}_k$ benutzen wir auch die Matrixschreibweise $(\underline{b}_1, \ldots, \underline{b}_k) \in \mathbb{R}^{d \times k}$.

Bemerkung 1.6 Sind $\underline{b}_1, \ldots, \underline{b}_k$ linear unabhängig über $\mathbb{R}$, dann stellt sich die Frage, wann eine Punktmenge $M \subset \mathbb{R}^d$ mit $M = \sum_{i=1}^{k} \mathbb{Z} \cdot \underline{b}_i$ ein Gitter ist. Ist $\underline{b}_1, \ldots, \underline{b}_k \in \mathbb{Q}^d$, dann genügt M Definition 1.2. Ist aber ein Eintrag in einem Vektor $\underline{b}_i$ echt reell, also nicht rational, dann muß M schon kein Gitter mehr sein. Das hängt damit zusammen, daß M in diesem Fall nicht immer diskret ist.

Beispiel 1.7 Sei $M = \mathbb{Z}\sqrt{2} + \mathbb{Z}$. Dann ist für alle $p \in \mathbb{N}$ $|10^p\sqrt{2} - \lceil 10^p\sqrt{2} \rfloor| < 1$ und $|\{10^p\sqrt{2} - \lceil 10^p\sqrt{2} \rfloor \mid p \in \mathbb{N}\}| = \infty$. Mit Bemerkung 1.3 folgt, daß M kein Gitter ist.

Bemerkung 1.8 Die Dimension eines Gitters ist eindeutig bestimmt, die Basis nicht.

Unimodulare Transformation

Sei $B \in \mathbb{R}^{d \times n}$ eine Basis von L. Es stellt sich die Frage, welche Bedingungen eine $n \times n$-Transformationsmatrix T erfüllen muß, damit $B \cdot T$ eine Basis von L ist. Wir sagen, eine $n \times n$-Matrix T ist **unimodular**, wenn sie nur ganzzahlige Einträge hat und $|\det(T)| = 1$.

Bemerkung 1.9 Die Inverse T^{-1} von $T \in \mathbb{Z}^{n \times n}$ existiert und hat ganzahlige Einträge genau dann, wenn T unimodular ist (siehe dazu [La80], Lemma 6.5).

Es gilt nun folgender Zusammenhang:

Lemma 1.10 (Unimodulare Transformation) *Sei $B \in \mathbb{R}^{d \times n}$ eine Basis eines Gitters L und $T \in \mathbb{R}^{n \times n}$ regulär. $B \cdot T$ ist genau dann eine Gitterbasis von L, wenn T unimodular ist.*

Beweis: „$\Leftarrow$“: Sei T unimodular. Folgende Aussagen sind zueinander äquivalent:

$$\begin{aligned} \underline{b} \in L \quad &\Leftrightarrow \quad \text{es existiert ein } \underline{v} \in \mathbb{Z}^n \text{ mit } \underline{b} = B \cdot \underline{v} \\ &\Leftrightarrow \quad \text{es existiert ein } \underline{v}' \in \mathbb{Z}^n \text{ mit } \underline{b} = B \cdot T \cdot \underline{v}' \\ &\Leftrightarrow \quad \underline{b} \text{ ist ein Gittervektor des Gitters, das von } B \cdot T \text{ erzeugt wird.} \end{aligned}$$

„$\Rightarrow$“: Sei $A = (\underline{a}_1, \ldots, \underline{a}_n) = B \cdot T$ eine Basis von L. $T = (\underline{t}_1, \ldots, \underline{t}_n) \in \mathbb{Z}^{n \times n}$, denn B ist Basis von L und $\underline{a}_i \in L$, $1 \leq i \leq n$. Da A Basis von L, gibt es $\underline{s}_i \in \mathbb{Z}^n$ mit $A \cdot \underline{s}_i = \underline{b}_i$, $1 \leq i \leq n$. Mit $S = (\underline{s}_1, \ldots, \underline{s}_n)$ ist

$$A \cdot S = B \text{ und } B \cdot T = A \Rightarrow T \cdot S = I \Rightarrow T^{-1} = S.$$

d.h. T^{-1} existiert und hat ganzzahlige Einträge. Mit Bemerkung 1.9 ist T unimodular. □

Wir haben gezeigt, daß es mehr als eine Basis geben kann. Durch Multiplikation mit einer unimodularen Matrix können wir aus einer Basis eine neue berechnen. Haben wir zwei Basen A und B, dann gibt es eine unimodulare Matrix T mit $A = B \cdot T$.

Projektionen von Vektoren

Wir sind oft an der Projektion von Vektoren interessiert. Die Projektion von $\underline{b}$ auf $\underline{v}$ ist der Vektor $\langle \underline{b}, \underline{v} \rangle / \langle \underline{v}, \underline{v} \rangle \cdot \underline{v}$. Projizieren wir $\underline{b}$ auf den Raum V mit $\underline{v}$ als einen Normalenvektor, d.h. für alle $\underline{w} \in V$ gilt $\langle \underline{w}, \underline{v} \rangle = 0$, so erhalten wir $\underline{b} - \langle \underline{b}, \underline{v} \rangle / \langle \underline{v}, \underline{v} \rangle \cdot \underline{v}$. Wir sprechen dann von der orthogonalen (senkrechten) Projektion von $\underline{b}$ auf $\underline{v}$. Unter der Projektion einer Menge M von Vektoren verstehen wir die Menge der Projektionen jedes Vektors aus M.

Sei $\underline{b}_1, \ldots, \underline{b}_n$ eine Basis von $V \subseteq \mathbb{R}^d$. Oft ist es notwendig, eine Basis für V zu finden, deren Basisvektoren orthogonal zueinander sind, d.h. das Skalarprodukt von zwei beliebigen, verschiedenen Basisvektoren ist Null. Eine solche Basis $\underline{b}_1^\star, \ldots, \underline{b}_n^\star$ läßt sich wie folgt mit der bekannten Gram-Schmidt-Orthogonalisierung aus der Ausgangsbasis berechnen:

$$\underline{b}_1^\star = \underline{b}_1, \quad \underline{b}_i^\star = \underline{b}_i - \sum_{j=1}^{i-1} \mu_{ij} \cdot \underline{b}_j^\star \quad \text{für} \quad 1 < i \leq n, \tag{1.1}$$

wobei

$$\mu_{ij} = \frac{\langle \underline{b}_i, \underline{b}_j^\star \rangle}{\langle \underline{b}_j^\star, \underline{b}_j^\star \rangle} \quad \text{für} \quad 1 \leq j < i \leq n. \tag{1.2}$$

Man sieht durch Einsetzen, daß

$$\langle\langle \underline{b}_1^\star, \ldots, \underline{b}_n^\star \rangle\rangle = \langle\langle \underline{b}_1, \ldots, \underline{b}_n \rangle\rangle$$

gilt. $\underline{b}_1^\star, \ldots, \underline{b}_n^\star$ nennen wir auch die Gram-Schmidt-Zerlegung von $\underline{b}_1, \ldots, \underline{b}_n$. Wir verallgemeinern jetzt den Begriff der senkrechten Projektion. Wollen wir $\underline{b} \in \mathbb{R}^d$ senkrecht auf einen Unterraum projizieren, dann suchen wir eine Orthogonalbasis des Unterraums mit Hilfe der Gram-Schmidt-Orthogonalisierung und führen sukzessive eine senkrechte Projektion von $\underline{b}$ auf jeden Vektor der Orthogonalbasis aus. Genau dies geschieht bei der Gram-Schmidt-Orthogonalisierung, wenn wir die $\underline{b}_i^\star$ berechnen. Es genügt nicht $\underline{b}$ senkrecht auf die $\underline{b}_i$ zu projizieren, denn $\langle \underline{b} - \sum_{i=1}^n \langle \underline{b}, \underline{b}_i \rangle / \langle \underline{b}_i, \underline{b}_i \rangle \; \underline{b}_i, \underline{b}_j \rangle = 0$ für $1 \leq j \leq n$ genau dann, wenn $\langle \underline{b}_i, \underline{b}_j \rangle = 0$ für $1 \leq i \neq j \leq n$. Die Projektion eines Vektors auf einen Unterraum kann man auch als senkrechte Projektion auf das orthogonale Komplement des Unterraums ansehen.

Die Gitter L_k

Seien $\underline{b}_1, \ldots, \underline{b}_n \in \mathbb{R}^d$ linear unabhängig. Wir definieren jetzt allgemein

$$\underline{b}_k(i) = \underline{b}_k - \sum_{j=1}^{i-1} \mu_{kj} \cdot \underline{b}_j(j) \quad \text{für} \quad 1 \leq i \leq k \leq n \tag{1.3}$$

mit

$$\mu_{kj} = \frac{\langle \underline{b}_k, \underline{b}_j(j) \rangle}{\langle \underline{b}_j(j), \underline{b}_j(j) \rangle} \quad \text{für} \quad 1 \leq j < k \leq n. \tag{1.4}$$

Der Vollständigkeit halber sei außerdem $\mu_{ii} = 1$ für $1 \leq i \leq n$ und $\mu_{ij} = 0$ für $1 \leq i < j \leq n$. Wie man sieht, ist $\underline{b}_i(i) = \underline{b}_i^\star$ aus der Gram-Schmidt-Orthogonalisierung. $\underline{b}_k(i)$ ist die Projektion von $\underline{b}_k$ auf $\langle\langle \underline{b}_1, \ldots, \underline{b}_{i-1} \rangle\rangle^\perp$, den zu $\langle\langle \underline{b}_1, \ldots, \underline{b}_{i-1} \rangle\rangle$ orthogonalen Unterraum. Aufbauend auf den Vektoren $\underline{b}_k(i)$ mit $1 \leq i \leq k \leq n$ definieren wir jetzt spezielle Gitter. Unter $L_k(\underline{b}_1, \ldots, \underline{b}_n)$ verstehen wir die Projektion von L auf $\langle\langle \underline{b}_1, \ldots, \underline{b}_{k-1} \rangle\rangle^\perp$. $L_k(\underline{b}_1, \ldots, \underline{b}_n) = \sum_{j=k}^n \mathbb{Z} \cdot \underline{b}_j(k)$ und folglich ein Gitter der Dimension $n - k + 1$. Wenn klar ist, was gemeint ist, schreiben wir L_k für $L_k(\underline{b}_1, \ldots, \underline{b}_n)$.

Liften von Vektoren

Wir wollen uns jetzt mit der Umkehroperation der Projektion befassen. Suchen wir einen Vektor aus einem n-dimensionalen Raum, der projiziert in einen $n-j$-dimensionalen Teilraum ein vorgegebener Vektor ist, dann erfüllt auf Grund des Dimensionsunterschiedes nicht nur ein Vektor die gewünschte Bedingung. In Abschnitt 2.6 stellt sich die Aufgabe, einen *kurzen* Gittervektor zu finden, dessen Projektion ein bestimmter Vektor ist. Wir wollen deshalb jetzt zeigen, wie man einen Vektor auf einen eindeutig bestimmbaren kurzen Vektor rückprojizieren kann. Sei $\underline{v}$ ein Nichtnullvektor des Gitters L. Sei $\hat{L}$ die senkrechte Projektion von L auf $\underline{v}$, d.h.

$$\hat{L} = \{\underline{y} - \langle \underline{y}, \underline{v} \rangle / \langle \underline{v}, \underline{v} \rangle \cdot \underline{v} \mid \underline{y} \in L\}.$$

Sei nun $\hat{\underline{w}} \in \hat{L}$ beliebig. Wir suchen $\underline{w} \in L$, so daß die Projektion von $\underline{w}$ auf $\hat{L}$ gerade $\hat{\underline{w}}$ ist und außerdem

$$-1/2 < \langle \underline{w}, \underline{v} \rangle / \langle \underline{v}, \underline{v} \rangle \leq 1/2$$

gilt. Damit ist $\underline{w}$ dann kurz. Wie findet man ein solches $\underline{w}$? Wir nehmen einen beliebigen Vektor $\underline{u} \in L$, der projiziert auf $\hat{L}$ den Vektor $\hat{\underline{w}}$ ergibt. Wie man ein solches $\underline{u}$ bestimmen kann, werden wir erst später im Zusammenhang zu dem beschriebenen Problem sehen (Lemma 2.29). Es gilt $\underline{w} \in \{\underline{u} - k \cdot \underline{v} \mid k \in \mathbb{Z}\}$. Da

$$\begin{array}{rrcccl} & -1/2 & < & \langle \underline{u} - k \cdot \underline{v}, \underline{v} \rangle / \langle \underline{v}, \underline{v} \rangle & \leq & 1/2 \\ \Leftrightarrow & -1/2 & < & \langle \underline{u}, \underline{v} \rangle / \langle \underline{v}, \underline{v} \rangle - k & \leq & 1/2 \end{array}$$

ist k eindeutig bestimmt, nämlich k ist die nächste ganze Zahl zu $\langle \underline{u}, \underline{v} \rangle / \langle \underline{v}, \underline{v} \rangle$. Folglich ist auch $\underline{w}$ eindeutig bestimmt. Wir bezeichen diesen Vorgang mit „Liften von $\hat{\underline{w}}$ nach $\underline{w}$“ (engl. to lift: heben).

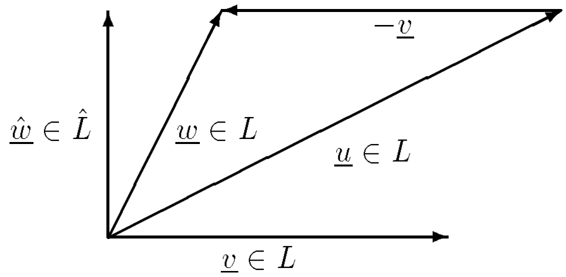

Abbildung 1.2: „Liften von $\hat{\underline{w}}$ auf $\underline{w}$“

Die Determinante von L

Definition 1.11 (Determinante) *Sei $L \subset \mathbb{R}^d$ ein Gitter und $B = (\underline{b}_1, \ldots, \underline{b}_n)$ eine Basis von L. Wir definieren die* **Gitterdeterminante** *oder kurz* **Determinante von L**

$$\mathrm{d}(L) = \det((\langle \underline{b}_i, \underline{b}_j \rangle)_{1 \le i,j \le n})^{1/2}. \tag{1.5}$$

Für den Fall $n = d$ gilt:

$$\mathrm{d}(L) = |\det B|, \tag{1.6}$$

denn

$$\det((\langle \underline{b}_i, \underline{b}_j \rangle)_{1 \le i,j \le n}) = (\det B)^2 = \mathrm{d}(L)^2.$$

Die Gitterdeterminante stellt geometrisch veranschaulicht das Volumen des kleinsten aufspannenden Parallelotops dar. Aus Lemma 1.10 folgt sofort, daß sie eine Invariante des Gitters ist. Für die Determinante des Gitters L gilt außerdem

$$\mathrm{d}(L) = \|\underline{b}_1^\star\| \cdot \ldots \cdot \|\underline{b}_n^\star\|. \tag{1.7}$$

Da für $1 \le i \le n$

$$\|\underline{b}_i\|^2 = \|\underline{b}_i^\star\|^2 + \sum_{j=1}^{i-1} \mu_{ij}^2 \, \|\underline{b}_j^\star\|^2,$$

ist für $1 \le i \le n$

$$\|\underline{b}_i^\star\| \le \|\underline{b}_i\|.$$

Damit gilt folgendes Lemma:

Lemma 1.12 (Hadamardsche Ungleichung) *Sei $L \subset \mathbb{R}^d$ ein Gitter und $\underline{b}_1, \ldots, \underline{b}_n$ eine Basis von L. Dann ist*

$$\mathrm{d}(L) \le \|\underline{b}_1\| \cdot \ldots \cdot \|\underline{b}_n\|. \tag{1.8}$$

Lemma 1.13 *Sei $L \subset \mathbb{R}^d$ ein Gitter und $\underline{b}_1, \ldots, \underline{b}_n$ eine Basis von L. Dann gilt*

$$\mathrm{d}(L) = \|\underline{b}_1\| \cdot \mathrm{d}(L_2(\underline{b}_1, \ldots, \underline{b}_n)).$$

Beweis: Es ist $L_2(\underline{b}_1, \ldots, \underline{b}_n) = L(\underline{b}_2(2), \ldots, \underline{b}_n(2))$. Da die Gram-Schmidt-Zerlegung von $\underline{b}_2(2), \ldots, \underline{b}_n(2)$ gerade $\underline{b}_2^\star, \ldots, \underline{b}_n^\star$ ist, folgt die Aussage mit (1.7). □

Der Defekt einer Basis von L

Definition 1.14 (Defekt) *Sei $L \subset \mathbb{R}^d$ ein Gitter und $B = (\underline{b}_1, \ldots, \underline{b}_n)$ eine Basis von L. Mit* **Defekt von** B *bezeichnen wir den Quotienten*

$$\mathrm{dft}(B) := \|\underline{b}_1\| \cdot \ldots \cdot \|\underline{b}_n\| / \mathrm{d}(L).$$

Bemerkung 1.15 Der Defekt ist um so kleiner, je kürzer die Basisvektoren sind. Der Defekt einer Basis macht auch eine Aussage darüber, wie weit die Basisvektoren davon entfernt sind, orthogonal aufeinander zu stehen. Wir wollen uns anhand eines zweidimensionalen Gitters den Zusammenhang zwischen der Länge der Basisvektoren und dem eingeschlossenen Winkel klar machen (siehe dazu auch [Ka87b]). Sei $L \subset \mathbb{R}^2$ ein Gitter und $\underline{b}_1, \underline{b}_2$ eine Basis von L. Dann haben wir

$$\mathrm{d}(L) = \|\underline{b}_1\| \cdot \|\underline{b}_2\| \cdot \sin \alpha,$$

wobei α der Winkel zwischen $\underline{b}_1$ und $\underline{b}_2$ ist. Da $\mathrm{d}(L)$ unabhängig von der Basis des Gitters ist, ist $\sin \alpha$ groß, wenn die Basisvektoren kurz sind. Ist aber $\sin \alpha$ groß, dann stehen die Basisvektoren „nahezu“ senkrecht aufeinander. Ähnlich kann man für n-dimensionale Gitter argumentieren. Wir können also sagen, daß die Basisvektoren um so kürzer sind, je „orthogonaler“ sie aufeinander stehen. Aus der Hadamardschen Ungleichung (Lemma 1.12) folgt, daß $\mathrm{dft}(B) \geq 1$ ist. Insbesondere gilt, daß $\mathrm{dft}(B) = 1$ genau dann, wenn die Basisvektoren paarweise orthogonal zueinander sind.

Sukzessive Minima

Definition 1.16 (Sukzessive Minima) *Sei $L \subset \mathbb{R}^d$ ein Gitter der Dimension n. Für $1 \leq i \leq n$ definieren wir das i-te* **sukzessive Minimum von** L

$$\lambda_i(L) = \min\{r > 0 \mid \textit{Es gibt } i \textit{ linear unabhängige Gittervektoren in } L, \textit{ deren euklidische Länge kleiner oder gleich } r \textit{ ist.}\}$$

$\lambda_i(L)$ ist also der Radius der kleinsten Kugel im $\mathbb{R}^d$, die wenigstens i linear unabhängige Gittervektoren enthält; $\lambda_1(L)$ ist die Länge des kürzesten Nichtnullvektors aus L. Ist eine Gitterbasis $\underline{b}_1, \ldots, \underline{b}_n$ der Länge nach geordnet, also $\|\underline{b}_i\| \leq \|\underline{b}_{i+1}\|$ für $1 \leq i < n$, dann gilt

$$\lambda_i(L) \leq \|\underline{b}_i\| \quad \text{für} \quad 1 \leq i \leq n.$$

Beispiel 1.17 Sei $L = \mathbb{Z} \cdot \begin{pmatrix} 4 \\ 3 \end{pmatrix} + \mathbb{Z} \cdot \begin{pmatrix} 9 \\ 3 \end{pmatrix}$ ein Gitter. Dann ist $\lambda_1(L) = 3$ und $\lambda_2(L) = 4$ (siehe Abbildung nächste Seite).

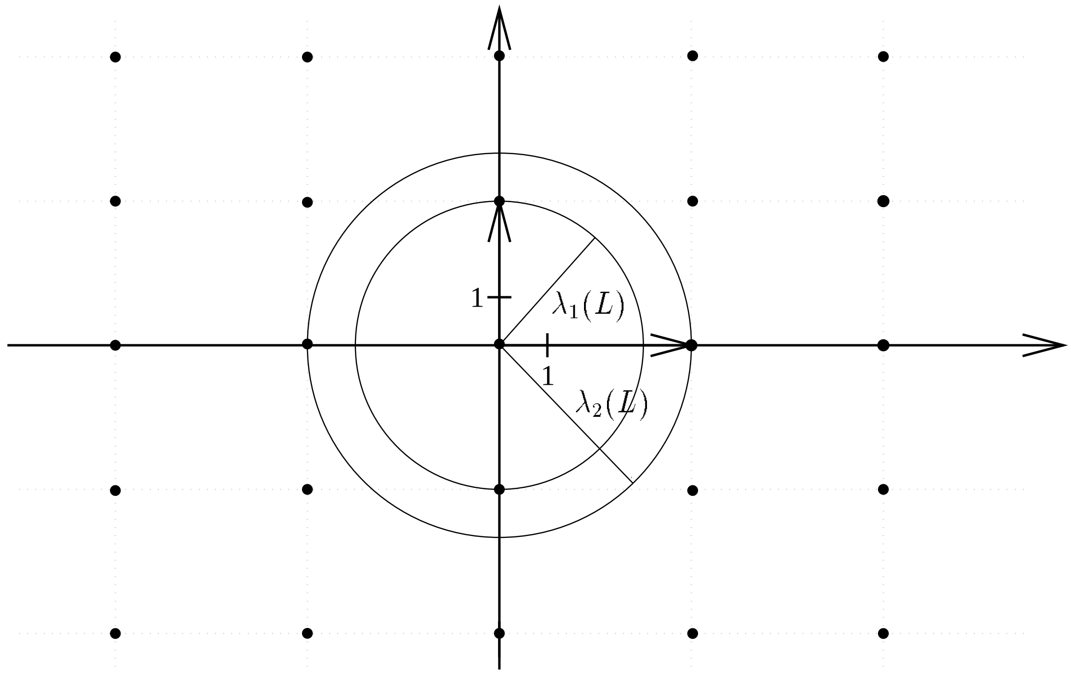

Abbildung 1.3: Sukzessive Minima

In der Regel gibt es allerdings keine Basis von L mit $\|\underline{b}_i\| = \lambda_i(L)$ für $1 \leq i \leq n$ (siehe dazu Abschnitt 2.3). Um bequem mit den Längen der Spaltenvektoren einer Matrix $A = (\underline{a}_1, \ldots, \underline{a}_n) \in \mathbb{R}^{d \times n}$ rechnen zu können, definieren wir analog dazu

$$\lambda_i(A) = \min\{r > 0 \mid \text{Es gibt } i \text{ Vektoren in } A \text{ (Spalten der Matrix } A) \\ \text{deren euklidische Länge kleiner oder gleich } r \text{ ist.}\}$$

Die Hermitesche Konstante und eine Folgerung aus dem Satz von Minkowski

Wir wollen jetzt die Beziehung zwischen den sukzessiven Minima und der Gitterdeterminante untersuchen. Die Hermitesche Konstante γ_n ist für $n \in \mathbb{N}$ definiert als

$$\gamma_n = \sup\{\lambda_1(L)^2 \cdot \mathrm{d}(L)^{-2/n} \mid L \text{ ist Gitter von der Dimension } n\}. \tag{1.9}$$

Ihre Werte sind nur für $n \leq 8$ exakt bekannt ([Ca59] Anhang). Es ist $\gamma_1 = 1$, $\gamma_2 = 2/\sqrt{3}$ und $\gamma_3 = 2^{1/3}$. Außerdem gilt [LaLeSch90]

$$\gamma_n \leq 2/3 \cdot n \quad \text{für } n \geq 2. \tag{1.10}$$

Man nimmt an, daß γ_n monoton mit n steigt. Dies konnte aber noch nicht bewiesen werden. Das Produkt der sukzessiven Minima läßt sich mit Hilfe von γ_n abschätzen.

Satz 1.18 (Folgerung aus dem Satz von Minkowsi) *Sei $\underline{b}_1, \ldots, \underline{b}_n$ eine Basis eines Gitters $L \subset \mathbb{R}^d$. Dann gilt:*

$$\mathrm{d}(L) \leq \lambda_1(L) \cdot \ldots \cdot \lambda_n(L) \leq (\gamma_n)^{n/2} \cdot \mathrm{d}(L), \tag{1.11}$$

$$\lambda_1(L) \leq \gamma_n^{1/2} \cdot \mathrm{d}(L)^{1/n}. \tag{1.12}$$

Beweis: Für $1 \leq i \leq n$ sei $\underline{v}_i \in L$ mit $\|\underline{v}_i\| = \lambda_i(L)$. Solche $\underline{v}_i$ existieren (siehe dazu [Ca59] VIII.1). Es gibt eine Transformationsmatrix $T \in \mathbb{Z}^{n \times n}$ mit $B \cdot T = V = (\underline{v}_1, \ldots, \underline{v}_n)$ und es ist: $\det T \in \mathbb{Z}\backslash\{0\}$, also $|\det(T)| \geq 1$. Es gilt

$$\begin{aligned}
\mathrm{d}(L) &\leq (\det(T^\top) \cdot \det(B^\top \cdot B) \cdot \det(T))^{1/2} \\
&= (\det((B \cdot T)^\top \cdot B \cdot T))^{1/2} = (\det(V^\top \cdot V))^{1/2} \\
&= \prod_{i=1}^{n} \|\underline{v}_i^\star\| \leq \prod_{i=1}^{n} \|\underline{v}_i\| = \lambda_1(L) \cdot \ldots \cdot \lambda_n(L),
\end{aligned}$$

womit die erste Ungleichung in (1.11) gezeigt wurde. Die zweite Ungleichung in (1.11) ist eine Folgerung aus dem 2. Gitterpunktsatz von Minkowski (siehe dazu [Ca59] VIII.2). Ungleichung (1.12) folgt leicht aus (1.11). □

Bemerkung 1.19 Statt (1.12) in Satz 1.18 verwenden wir häufig

$$\lambda_1(L) \leq \sqrt{n} \cdot \mathrm{d}(L)^{1/n}. \tag{1.13}$$

Diese Ungleichung folgt aus (1.10) und (1.12)

Kapitel 2

Algorithmen für Gitterprobleme über $\mathbb{Z}^d$ ($\mathbb{Q}^d$)

Die folgenden Abschnitte in diesem Kapitel werden Algorithmen zur Lösung der im Vorwort genannten Probleme in Gittern über $\mathbb{Z}^d$ zum Thema haben. Je nach Schwierigkeit wird der Korrektheitsbeweis oder die Laufzeitanalyse im Mittelpunkt unseres Interesses stehen. Daneben werden auch Eigenschaften gezeigt werden, denen speziell die Basen genügen, die sich mit den vorzustellenden Reduktionsalgorithmen berechnen lassen. Doch zuvor werden wir sehen, wie vorzugehen ist, wenn $L \subset \mathbb{Q}^d$ ist. Für ein weiteres Problem, nämlich den Fall, daß das Gitter nicht durch eine Basis gegeben ist, werden wir außerdem auf Literatur hinweisen.

2.1 Grundsätzliche Beziehungen in Gittern

Rückführung von $\mathbb{Q}^d$ auf $\mathbb{Z}^d$

Mit den in den nächsten Abschnitten aufgeführten Algorithmen lassen sich die jeweiligen Probleme durch geeignete Transformation der Eingabe auch in Gittern über $\mathbb{Q}^d$ lösen. Ist ein Gitter $L \subset \mathbb{Q}^d$ durch eine Basis

$$B = \left(\begin{pmatrix} \frac{p_{11}}{q_{11}} \\ \vdots \\ \frac{p_{1d}}{q_{1d}} \end{pmatrix} \cdots \begin{pmatrix} \frac{p_{n1}}{q_{n1}} \\ \vdots \\ \frac{p_{nd}}{q_{nd}} \end{pmatrix} \right) \in \mathbb{Q}^{d \times n}$$

gegeben und

$$q = \prod_{i=1}^{n} \prod_{j=1}^{d} q_{ij},$$

dann ist $Y = Bq \in \mathbb{Z}^{d\times n}$ Basis eines Gitters $L(Y) \subseteq \mathbb{Z}^d$. Durch Multiplikation mit q bleibt in $L(Y)$ die Struktur von L erhalten. Berechnen wir die gesuchten Gittervektoren in $L(Y)$, und dividieren deren Einträge durch q, dann haben wir damit auch die Lösung des Problems in $L \subset \mathbb{Q}^d$.

Durch Multiplikation mit q wächst die Länge der Basisvektoren polynomiell. Sei

$$N := \max\{\log|q_{ij}| \mid 1 \le i \le n, 1 \le j \le d\}$$

und

$$Z := \max\{\log|p_{ij}| \mid 1 \le i \le n, 1 \le j \le d\}.$$

Dann ist

$$\log|Y| = O(n \cdot d \cdot N + Z),$$

wobei mit $|Y|$ der Betrag des betragsgrößten Eintrags der Matrix Y bezeichnet wird.

Bemerkung 2.1 Ist die Laufzeit eines Algorithmus für $L \subseteq \mathbb{Z}^d$ polynomiell in der Länge der Basisvektoren, dann auch für $L \subset \mathbb{Q}^d$.

Elementare Berechnungsprobleme

Folgende grundsätzliche Fragen sind in Polynomzeit beantwortbar, wenn die Eingabe aus rationalen Zahlen besteht.

- *Zugehörigkeit:* Sei eine Menge von (möglicherweise) linear abhängigen Vektoren $\underline{b}_1, \ldots, \underline{b}_k$ und ein weiterer Vektor $\underline{v}$ gegeben. Ist $\underline{v} \in \sum_{i=1}^k \mathbb{Z} \cdot \underline{b}_i$?
- *Homogene Gleichungen:* Gegeben sei ein lineares Gleichungssystem $A\underline{u} = \underline{0}$. Finde eine Basis des Gitters $L = \{\underline{u} \mid A\underline{u} = \underline{0}, \underline{u} = (u_1, \ldots, u_d), u_i \in \mathbb{Z} \text{ für } 1 \le i \le d\}$. Man mache sich an Hand von Definition 1.2 klar, daß L ein Gitter ist.
- *Basisberechnung:* Gegeben sei eine Menge von (möglicherweise) linear abhängigen Vektoren $\underline{b}_1, \ldots, \underline{b}_k$. Finde eine Basis von $L = \sum_{i=1}^k \mathbb{Z} \cdot \underline{b}_i$.

Algorithmen mit denen sich diese Probleme lösen lassen findet man erstmals in [GaSi76] (1. Problem) und [VoFr76] oder [KaBa79] (2. und 3. Problem). Eine Arbeit, in der alle obigen Probleme ausführlich bearbeitet sind, ist [Mu94].

2.2 Kürzeste Gittervektoren

Das vielleicht am einfachsten zu formulierende Problem ist die Berechnung eines kürzesten Gittervektors ungleich dem Nullvektor in einem Gitter. An dieser Stelle sei angemerkt, daß wir von nun an den Nullvektor stets ausschließen, wenn wir von kurzen Gittervektoren reden. Wir suchen also ein $\underline{v} \in L \backslash \{\underline{0}\}$ mit

$$\|\underline{v}\| \leq \|\underline{b}\| \text{ für alle } \underline{b} \in L.$$

Äquivalent dazu ist die Forderung $\|\underline{v}\| = \lambda_1(L)$. Wir sprechen nicht von *dem* kürzesten Gittervektor, weil es, wie wir im Beispiel 2.2 sehen, mehr als einen geben kann. Wir wollen dieses Problem im folgenden kurz mit SVP bezeichnen (engl.: **S**hortest **V**ector **P**roblem). Nach (1.13) kennen wir eine obere Schranke für die Länge eines kürzesten Gittervektors, die nur von der Dimension n und der Determinante $\mathrm{d}(L)$ abhängt, nämlich

$$\lambda_1(L) \leq n^{1/2} \cdot \mathrm{d}(L)^{1/n}. \tag{2.1}$$

Somit können wir den Radius einer Kugel um den Nullpunkt im $\mathbb{R}^d$ angeben, in der sich kürzeste Gittervektoren befinden, wenn wir die Determinante und die Dimension des Gitters kennen.

Beispiel 2.2 Sei $L = \mathbb{Z} \cdot \underline{b}_1 + \mathbb{Z} \cdot \underline{b}_2 = \mathbb{Z} \cdot \begin{pmatrix} 1 \\ 2 \end{pmatrix} + \mathbb{Z} \cdot \begin{pmatrix} 2 \\ 1 \end{pmatrix} \subseteq \mathbb{Z}^2$ ein Gitter. Die kürzesten Gittervektoren sind $\underline{v}_1 = \begin{pmatrix} -1 \\ 1 \end{pmatrix}$ und $\underline{v}_2 = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$. Der Radius des Kreises (gemäß (2.1)), in dessen Innerem die kürzesten Gittervektoren liegen, hat die Länge $\sqrt{6}$.

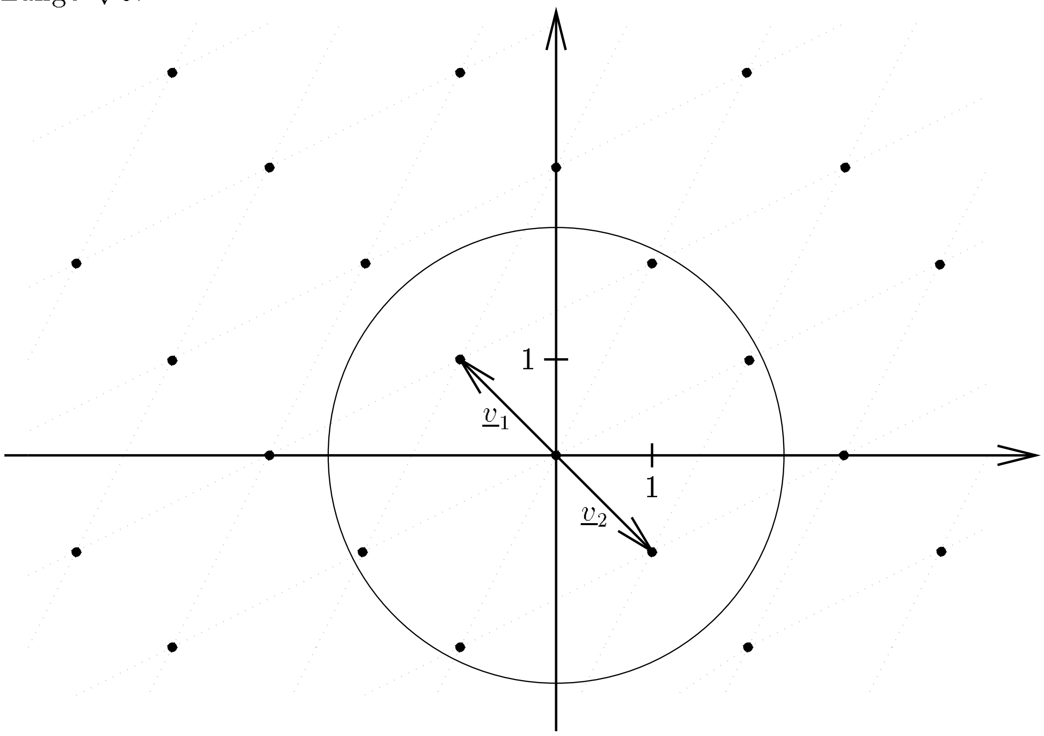

Abbildung 2.1: Kürzeste Gittervektoren

Einige schwierige Berechnungsprobleme aus anderen Bereichen der Mathematik lassen sich auf das SVP zurückführen. Dies läßt vermuten, daß das SVP nicht leicht zu lösen ist. In der Tat konnte bisher kein Polynomzeitalgorithmus für das SVP gefunden werden. Es ist auch nicht bekannt, ob das SVP $\mathcal{NP}$-vollständig ist oder nicht. Ersetzen wir aber die euklidische Norm durch die Maximumsnorm, dann wissen wir, daß das Problem $\mathcal{NP}$-hart ist [Bo81]. Eine Erklärung für die Begriffe $\mathcal{NP}$-vollständig oder $\mathcal{NP}$-hart findet man z.B. in [GrLoSchr88].

Schwächen wir das Problem ab, kommen wir doch noch zu einem Polynomzeitalgorithmus. Wie wir in Abschnitt 2.5 sehen werden, können wir in Polynomzeit aus einer beliebigen Ausgangsbasis eines Gitters $L \subseteq \mathbb{Z}^d$ eine Basis $\underline{b}_1, \ldots, \underline{b}_n$ berechnen [LeLeLo82] mit

$$\|\underline{b}_1\| \leq 2^{(n-1)/2} \cdot \lambda_1(L).$$

Geben wir uns also mit einem Gittervektor zufrieden, der möglicherweise „exponentiell weit" (in der Dimension des Gitters) vom kürzesten Gittervektor entfernt ist, dann können wir das abgeschwächte Problem auf die Berechnung einer im Sinne von [LeLeLo82] konstruierten Basis zurückführen. Schnorr [Sch84] zeigte, daß es für alle $c > 0$ möglich ist, einen Polynomzeitalgorithmus zu konstruieren, der einen Vektor $\underline{v} \in L \backslash \{\underline{0}\}$ mit

$$\|\underline{v}\| \leq (1+c)^{n/2} \cdot \lambda_1(L)$$

berechnet. Es wäre interessant einen Polynomzeitalgorithmus zu entwickeln, bei dem der Faktor vor $\lambda_1(L)$ in den subexponentiellen Bereich fällt.

Haben wir eine Basis mit kleinem orthogonalem Defekt, also $\|\underline{b}_1\| \cdot \ldots \cdot \|\underline{b}_n\| / \mathrm{d}(L)$ möglichst klein, dann können wir schnell einen kürzesten Gittervektor berechnen. Sei $\underline{v} = \sum_{i=1}^{n} \alpha_i \cdot \underline{b}_i$ ein kürzester Vektor. Dann folgt mit der Cramerschen Regel (siehe z.B. in [La78], S. 174) und der Ungleichung von Hadamard (Lemma 1.12)

$$|\alpha_i| \leq \|\underline{b}_1\| \cdot \ldots \cdot \|\underline{b}_{i-1}\| \cdot \|\underline{v}\| \cdot \|\underline{b}_{i+1}\| \cdot \ldots \cdot \|\underline{b}_n\| / \mathrm{d}(L). \tag{2.2}$$

Da $\|\underline{v}\| \leq \|\underline{b}_i\|$ für $1 \leq i \leq n$, läßt sich das SVP folglich mit einem primitiven Suchalgorithmus lösen. Durch Aufzählen aller α_i für $1 \leq i \leq n$, die (2.2) genügen, läßt sich ein kürzester Gittervektor finden und je kleiner der orthogonale Defekt der Basis, um so weniger Kandidaten kommen für die α_i in Frage.

In Abschnitt 2.6 werden wir einen Algorithmus kennenlernen, der eine ähnliche Beziehung wie (2.2) benutzt, um eine Basis zu berechnen, die aus kurzen Vektoren besteht und dessen erster Basisvektor ein kürzester Gittervektor ist. Die Laufzeit des in Abschnitt 2.6 vorgestellten Algorithmus ist für $L \subseteq \mathbb{Z}^d$ exponentiell in der Dimension des Gitters und zwar $n^{n/2+o(n)}$, aber polynomiell in der maximalen Länge eines Vektors der gegebenen Basis.

2.3 Reduzierte Gitterbasen

Wie wir in Abschnitt 2.2 gesehen haben, können wir dann einen kürzesten Gittervektor effizient berechnen, wenn wir eine Basis kennen, die aus möglichst kurzen Vektoren besteht. Neben dem SVP gibt es auch andere Probleme in Gittern, die darin bestehen, Gitterpunkte mit bestimmten Eigenschaften zu finden. Sie werden wie das SVP meist auch durch Suchalgorithmen gelöst. Wir werden sehen, daß bei solchen Algorithmen der Suchraum wesentlich kleiner ist, wenn eine Basis mit kurzen Basisvektoren bekannt ist. Eine solche Basis vermindert auch die Laufzeit von Algorithmen beträchtlich, die intensiv mit Basisvektoren arbeiten.

Beispiel 2.3 Sei L durch die Basis

$$\begin{pmatrix} 60408853613420 \\ -214926455217486 \\ 62343913872 \end{pmatrix}, \begin{pmatrix} -39214636629539 \\ 13952032305640 \\ -4047078798 \end{pmatrix}, \begin{pmatrix} 25531978081724 \\ -90839292844591 \\ 26349836945 \end{pmatrix}$$

gegeben. Mit einem später vorgestellten Algorithmus können wir daraus schnell

$$\begin{pmatrix} 2 \\ 2 \\ 0 \end{pmatrix}, \begin{pmatrix} 7 \\ -5 \\ 1 \end{pmatrix}, \begin{pmatrix} -1 \\ 0 \\ 8 \end{pmatrix}$$

berechnen, eine andere Basis des Gitters. Die Basiseinträge sind bei der Ausgangsbasis rund 10^{12}-mal größer als bei der neuen Basis und lassen sich in der Praxis insbesondere ohne spezielle Langzahlarithmetik darstellen und verarbeiten.

Ohne eine exakte Definition anzugeben, wollen wir eine Basis „reduziert“ nennen, wenn das Produkt

$$\|\underline{b}_1\| \cdot \ldots \cdot \|\underline{b}_n\|$$

„möglichst klein“ ist, und das Problem, mit dem wir uns jetzt im folgenden beschäftigen, soll darin bestehen, eine reduzierte Basis zu berechnen. Wir nehmen das Produkt, und nicht etwa die Summe, weil dann unter anderem ein enger Zusammenhang zur Determinante des Gitters besteht (siehe Lemma 1.12 (Hadamardsche Ungleichung).

Im Zuge der Entwicklung von Algorithmen zur Berechnung von reduzierten Basen ergaben sich aus den Berechnungsvorschriften heraus genau definierbare Reduktionsarten. Bevor wir darauf eingehen, wollen wir die Komplexität des Problems angeben, das Verhältnis zwischen sukzessiven Minima und reduzierten Basisvektoren untersuchen und ein Maß für die Reduktionsgüte einführen.

Lovász behauptet, daß das Problem, aus n linear unabhängigen Vektoren eines n-dimensionalen Gitters $L \subseteq \mathbb{Z}^d$ eine Basis $\underline{b}_1, \ldots, \underline{b}_n$ zu berechnen, für die

$$\|\underline{b}_1\| \cdot \ldots \cdot \|\underline{b}_n\| \leq \|\underline{a}_1\| \cdot \ldots \cdot \|\underline{a}_n\|$$

für alle Basen $\underline{a}_1, \ldots, \underline{a}_n$ von L gilt, $\mathcal{NP}$-hart ist.

Wir wollen nun die Beziehung zwischen den sukzessiven Minima und den euklidischen Längen der Basisvektoren untersuchen. Wir wissen, daß für jede beliebige Basis $\underline{b}_1, \ldots, \underline{b}_n$ von L mit $\|\underline{b}_i\| \leq \|\underline{b}_{i+1}\|$ für $1 \leq i < n$ die Beziehung

$$\lambda_i(L) \leq \|\underline{b}_i\| \text{ für } 1 \leq i \leq n$$

gilt. Besonders interessant ist nun, daß es zwar stets linear unabhängige $\underline{v}_i$ mit $\|\underline{v}_i\| = \lambda_i(L)$ für $1 \leq i \leq n$ gibt (siehe dazu [Ca59], VIII.1), diese aber nicht unbedingt eine Gitterbasis von L bilden müssen, was folgendes Beispiel verdeutlichen soll.

Beispiel 2.4 Sei $L = \sum_{i=1}^{5} \mathbb{Z} \cdot \underline{b}_i$ mit

$$\underline{b}_1 = \begin{pmatrix} 2 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad \underline{b}_2 = \begin{pmatrix} 0 \\ 2 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad \underline{b}_3 = \begin{pmatrix} 0 \\ 0 \\ 2 \\ 0 \\ 0 \end{pmatrix}, \quad \underline{b}_4 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 2 \\ 0 \end{pmatrix}, \quad \text{und } \underline{b}_5 = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}.$$

Da

$$\underline{v} = 2 \cdot \underline{b}_5 - \sum_{i=1}^{4} \underline{b}_i = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 2 \end{pmatrix},$$

haben wir

$$\lambda_1(L) = \ldots = \lambda_5(L) = 2.$$

Sei nun $B' = (\underline{b}_1, \ldots, \underline{b}_4, \underline{v})$. Die Vektoren $\underline{b}_1, \ldots, \underline{b}_4, \underline{v}$ sind linear unabhängig, aber für $T \in \mathbb{Z}^{5 \times 5}$ mit $B' = B \cdot T$ gilt: $2 = |\det(T)| \neq 1$. Also ist T nicht unimodular, folglich B' auch keine Gitterbasis von L.

Trotzdem wollen wir die sukzessiven Minima verwenden, um bei den Reduktionsverfahren obere bzw. untere Schranken nachstehender Art für die maximale bzw. minimale euklidische Länge jedes einzelnen reduzierten Basisvektors anzugeben:

$$\mathrm{c}(i,n) \cdot \lambda_i(L) \leq \|\underline{b}_i\| \leq \mathrm{C}(i,n) \cdot \lambda_i(L) \text{ für } 1 \leq i \leq n,$$

wobei $\mathrm{c}(i,n)$, $\mathrm{C}(i,n)$ Funktionen von $\mathbb{N} \times \mathbb{N}$ nach $\mathbb{R}_{>0}$ sind. Wir werden aber die Schranken erst in den entsprechenden Abschnitten vorstellen.

Sind die sukzessiven Minima im folgenden dazu da, um die Kürze der einzelnen Basisvektoren abzuschätzen, dann ist der Defekt einer Basis $B = (\underline{b}_1, \ldots, \underline{b}_n)$

$$\mathrm{dft}(B) = \|\underline{b}_1\| \cdot \ldots \cdot \|\underline{b}_n\| / \mathrm{d}(L),$$

geeignet, um den Grad der Reduktion der ganzen Basis anzugeben. Der Defekt läßt sich nämlich bei den verschiedenen Reduktionsarten stets durch Terme nach oben abschätzen, die nur von der Dimension abhängen. Zum anderen gibt er, wie wir in Abschnitt 1 gesehen haben, auch Auskunft darüber, inwieweit die Basisvektoren davon entfernt sind, orthogonal aufeinander zu stehen.

Wir wollen nun die bekanntesten Reduktionsarten definieren und aus den eben genannten Gründen den Defekt als Maß für die Reduktionsgüte benutzen. Der Reduktionsalgorithmus spielt bei der Benennung nur eine implizite Rolle.

Wir beginnen mit einer „schwachen“ Reduktionsart. Sie wurde erstmals von Lenstra, Lenstra Jr. und Lovász [LeLeLo82] vorgestellt. Der von ihnen entwickelte Algorithmus berechnet eine in ihrem Sinne reduzierte Basis eines Gitters $L \subseteq \mathbb{Z}^d$ in einer Zeit, die sowohl polynomiell von der maximalen Länge eines Vektors der gegebenen Basis als auch polynomiell von der Dimension des Gitters abhängt. Im folgenden seien $\underline{b}_1^\star, \ldots, \underline{b}_n^\star$ und die dazugehörigen μ_{ij}, $1 \leq j < i \leq n$, wie in Kapitel 1 definiert.

Definition 2.5 (LLL-reduziert) *Eine Basis* $\underline{b}_1, \ldots, \underline{b}_n$ *eines Gitters* $L \subset \mathbb{R}^d$ *heißt* **LLL-reduziert**, *wenn*

$$|\mu_{ij}| \leq 1/2 \text{ für } 1 \leq j < i \leq n, \tag{2.3}$$

$$\|\underline{b}_i^\star + \mu_{i\,i-1} \cdot \underline{b}_{i-1}^\star\|^2 \geq 3/4 \cdot \|\underline{b}_{i-1}^\star\|^2 \text{ für } 1 < i \leq n. \tag{2.4}$$

Eine Basis eines Gitters, die Eigenschaft (2.3) erfüllt, nennt man **schwach reduziert**. Am Anfang von Abschnitt 2.5 werden wir diese Eigenschaft näher untersuchen. Auch die Bedeutung der Eigenschaft (2.4) wird zu Beginn von Abschnitt 2.5 erklärt werden. An dieser Stelle sei nur soviel gesagt: Diese Eigenschaft gewährleistet, daß eine LLL-reduzierte Basis durch Addition eines Vielfachen eines Basisvektors zu einem anderen und durch anschließendes Vertauschen der beiden Vektoren nicht wesentlich reduziert werden kann. Da LLL-reduzierte Gitterbasen nur schwachen Bedingungen genügen, hat der Defekt einer LLL-reduzierte Basis jedoch die Größenordnung $2^{O(n^2)}$. Mit LLL-reduzierten Basen werden wir uns intensiv in Abschnitt 2.5 befassen.

Eine weitere Möglichkeit zu definieren, wann eine Gitterbasis „reduziert“ ist, geht auf Korkine und Zolotarev zurück [KoZo1873].

Definition 2.6 (Korkine-Zolotarev-reduziert) *Eine Basis* $\underline{b}_1, \ldots, \underline{b}_n$ *eines Gitters* $L \subset \mathbb{R}^d$ *heißt* **Korkine-Zolotarev-reduziert**, *wenn*

$$|\mu_{ij}| \leq 1/2 \text{ für } 1 \leq j < i \leq n, \tag{2.5}$$

$$\|\underline{b}_i^\star\| = \lambda_1(L_i) \text{ für } 1 \leq i \leq n. \tag{2.6}$$

Der Vektor $\underline{b}_i^\star = \underline{b}_i(i)$ ist in einer Korkine-Zolotarev-reduzierten Basis also ein kürzester Gittervektor des Gitters $L(\underline{b}_i(i), \underline{b}_{i+1}(i), \ldots, \underline{b}_n(i))$. Insbesondere ist $\underline{b}_1$ ein kürzester Gittervektor von L. Mit Korkine-Zolotarev-reduzierten Basen werden wir uns intensiv in Abschnitt 2.6 beschäftigen. Korkine-Zolotarev-reduzierte Gitterbasen haben einen Defekt der Größenordnung $O(n^n)$ [Sch84]. In Abschnitt 2.6 zählen wir die wichtigsten strukturellen Eigenschaften auf, die Korkine-Zolotarev-reduzierte Basen besitzen [LaLeSch90]. Desweiteren geben wir einen Algorithmus an, der aus einer beliebigen Basis eine Korkine-Zolotarev-reduzierte Basis berechnet. Ist $L \subseteq \mathbb{Z}^d$, dann kann gezeigt werden, daß die Laufzeit zwar exponentiell in der Dimension des Gitters, aber polynomiell in der maximalen Länge eines Vektors der gegebenen Basis ist.

Zum Schluß wollen wir noch auf die vielleicht natürlichste Art definieren, wann eine Basis als reduziert bezeichnet werden kann. Die Definition geht auf Minkowski zurück [Mi10]:

Definition 2.7 (Minkowski-reduziert) *Eine Basis $\underline{b}_1, \ldots, \underline{b}_n$ eines Gitters $L \subset \mathbb{R}^d$ heißt* **Minkowski-reduziert**, *wenn $\underline{b}_1$ ein kürzester Vektor ist und für alle $1 \leq i < n$ $\underline{b}_{i+1}$ von allen Vektoren, mit denen sich die bereits gewählten $\underline{b}_1, \ldots, \underline{b}_i$ zu einer Basis erweitern lassen, der kürzeste ist.*

Minkowski-reduzierte Basen haben einen Defekt der Größenordnung $2^{O(n^2)}$ ([Mi67], Kapitel 5.5-5.8). Der bisher schnellste, mir bekannte Algorithmus, der aus einer beliebigen Basis eines Gitters $L \subseteq \mathbb{Z}^d$ eine Minkowski-reduzierte Basis berechnet, hat eine Laufzeit, die wie der Algorithmus zur Berechnung einer Korkine-Zolotarev-reduzierten Basis exponentiell in der Dimension des Gitters und polynomiell in der maximalen Länge eines Vektors der gegebenen Basis ist [He85]. Weil der Defekt einer Minkowski-reduzierten Basis schlechter ist als der einer Korkine-Zolotarev-reduzeirten Basis, gehen wir nicht weiter auf Minkowski-reduzierte Basen ein. Es sei noch angemerkt, daß sie in der Zahlengeometrie eine große Rolle spielen (siehe dazu [Ca59] und [Le69]).

2.4 Basisreduktion im Zweidimensionalen

Die oben angesprochenen Algorithmen berechnen reduzierte Basen beliebig dimensionaler Gitter. Bevor wir diese beschreiben, wollen wir ein spezielles Verfahren vorstellen, das eine reduzierte Basis $\underline{b}_1, \underline{b}_2$ für ein zweidimensionales Gitter L findet, wenn das Gitter durch eine Basis gegeben ist. Wir bekommen damit einen Eindruck, wie die später vorgestellten, komplizierteren Algorithmen arbeiten. Das Verfahren geht auf Gauß [Ga1889] zurück. In der reduzierten Basis ist $\underline{b}_1$ der kürzeste Vektor von L, d.h $\|\underline{b}_1\| = \lambda_1(L)$ und $\underline{b}_2$ der kürzeste Gittervektor aus L, der mit $\underline{b}_1$ eine Basis bildet. Also berechnet der Gaußalgorithmus eine Minkowski-reduzierte Basis.

Die Basis ist auch Korkine-Zolotarev-reduziert. Denn in zweidimensionalen Gittern sind Minkowski-reduzierte Basen auch Korkine-Zolotarev-reduziert und umgekehrt: Der Basisvektor $\underline{b}_1$ ist in beiden Fällen ein kürzester Gittervektor. Aus

$$\|\underline{b}_2\|^2 = \|\underline{b}_2^\star\|^2 + \mu_{21}^2\,\|\underline{b}_1^\star\|^2$$

und

$$|\mu_{21}| \leq 1/2$$

folgt, daß $\underline{b}_2^\star$ genau dann ein kürzester Gittervektor von L_2 ist (L_2 ist die orthogonale Projektion von L auf $\langle\langle \underline{b}_1\rangle\rangle$), wenn $\underline{b}_2$ ein kürzester Gittervektor von L ist, der mit $\underline{b}_1$ eine Basis bildet. Der Algorithmus zur Berechnung LLL-reduzierter Basen in Abschnitt 2.5.2 stellt eine Verallgemeinerung des Gaußalgorithmus auf n-dimensionale Gitter dar.

Wir machen uns nun klar, wie der Gaußalgorithmus die Ausgangsbasis transformiert und welche Überlegungen hinter jeder Transformation stehen. Sei $B = (\underline{b}_1, \underline{b}_2)$ eine beliebige Basis von L. $B \cdot T$ ist genau dann eine Basis, wenn T unimodular ist (Lemma 1.10). Folgende unimodulare Operationen werden in dem Algorithmus angewendet:

a) Addition eines ganzzahligen Vielfachen des ersten Basisvektors zum zweiten Basisvektor: $\underline{b}_2 := \underline{b}_2 + k \cdot \underline{b}_1$.

b) Vertauschen der Basisvektoren.

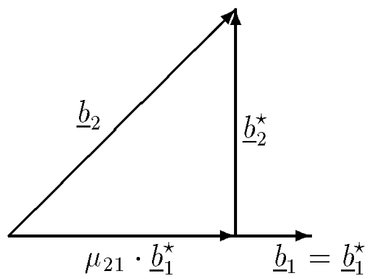

Abbildung 2.2: Lage von $\underline{b}_2^\star$ zu $\underline{b}_1$, $\underline{b}_2$.

Wir wollen durch geeignete Wahl von k gewährleisten, daß nach Ausführung von Operation a) die Projektion des neuen Basisvektors $\underline{b}_2$ auf das alte und neue $\underline{b}_1$, d.h. der Vektor $\mu_{21} \cdot \underline{b}_1^\star$, so klein wie möglich wird. Wie wir in Abbildung 2.2 sehen, ist die Länge von $\underline{b}_2^\star$ durch Addition eines Vielfachen von $\underline{b}_1$ zu $\underline{b}_2$ nicht veränderbar. Deshalb hätten wir dann $\underline{b}_2$ soweit verkürzt, wie es Operation a) maximal zuläßt.
Wie wählen wir nun k? Ist in der neuen Gitterbasis $|\mu_{21}| \leq 1/2$, dann befindet sich $\underline{b}_2$ links von der Mittelsenkrechten von $\underline{b}_1$ und rechts von der Mittelsenkrechten von $-\underline{b}_1$. Weil wir nur ganzzahlige Vielfache von $\underline{b}_1$ addieren dürfen, ist dies das Beste, was man durch Operation a) erreichen kann. Aus

$$\langle \underline{b}_2 + k \cdot \underline{b}_1, \underline{b}_1\rangle / \langle \underline{b}_1, \underline{b}_1\rangle \leq 1/2$$

$$\Leftrightarrow \quad |\mu_{21} + k| \leq 1/2$$

folgt, daß $k = -\lceil \mu_{21} \rfloor$ gewählt werden sollte. Nachdem wir Operation a) durchgeführt haben, wollen wir jetzt $\underline{b}_1$ mit Hilfe von $\underline{b}_2$ verkürzen. Dazu vertauschen wir die beiden Vektoren und wenden erneut Operation a) an. Dann vertauschen wir wieder usw. Wie lange ist es sinnvoll, diese beiden Operationen durchzuführen? Stellen wir uns dazu vor, welche Situation vorliegt, wenn nach Operation a) $\|\underline{b}_1\| \leq \|\underline{b}_2\|$ ist. Wegen dieser Bedingung und weil der Winkel α zwischen $\underline{b}_1$ und $\underline{b}_2$ zwischen 60° und 120° liegen muß (dies ergibt sich aus $|\mu_{21}| \leq 1/2$ und der Beziehung $\langle \underline{b}_1, \underline{b}_2 \rangle = \|\underline{b}_1\| \, \|\underline{b}_2\| \cos\alpha$), hat Operation a) nach der Vertauschung keine Wirkung mehr. Aus diesen beiden Bedingungen folgt nämlich, daß die Länge der Projektion von $\underline{b}_1$ auf $\underline{b}_2$ kleiner oder gleich $\|\underline{b}_2\|/2$ und folglich durch Addition eines k-fachen von $\underline{b}_2$ keine Verkürzung von $\underline{b}_1$ mehr erreicht werden kann. In diesem Fall brechen wir das Verfahren ab. Ist nach der ersten

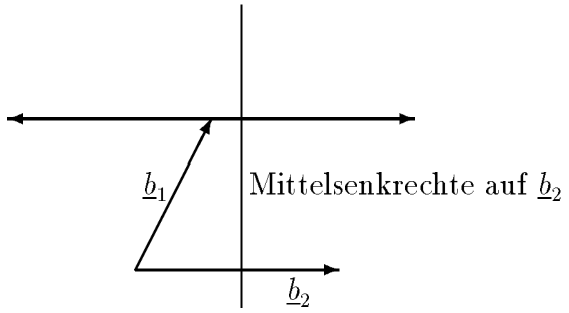

Abbildung 2.3: $\|\underline{b}_1\| \leq \|\underline{b}_2\|$ nach Operation a)

Ausführung von Operation a) $\|\underline{b}_1\| > \|\underline{b}_2\|$ dann wird solange Operation b) und Operation a) durchgeführt bis $\|\underline{b}_1\| \leq \|\underline{b}_2\|$. Das Verfahren terminiert, weil in jedem Schritt verkürzt wird und Kugeln in Gittern nur endlich viele Gittervektoren enthalten.

Algorithmus 2.8 GAUSS($\underline{b}_1$, $\underline{b}_2$)

EINGABE: Basisvektoren $\underline{b}_1$, $\underline{b}_2$ mit $\|\underline{b}_1\| \leq \|\underline{b}_2\|$.
AUSGABE: Reduzierte Basisvektoren $\underline{b}_1$, $\underline{b}_2$.

(1) $\underline{b}_2 := \underline{b}_2 - \lceil \mu_{21} \rfloor \cdot \underline{b}_1$;
(2) **while** ($\|\underline{b}_1\| > \|\underline{b}_2\|$) **do**
(3) Vertausche $\underline{b}_1$ und $\underline{b}_2$.
(4) $\underline{b}_2 := \underline{b}_2 - \lceil \mu_{21} \rfloor \cdot \underline{b}_1$;
/* $\underline{b}_2$ wird mit $\underline{b}_1$ reduziert, so daß $|\mu_{21}| \leq 1/2$. */
(5) **od**

Ist $L \subseteq \mathbb{Z}^d$, dann ist die Laufzeit des Algorithmus $O(d \log(\max\{2, \|\underline{b}_1\|, \|\underline{b}_2\|\}))$. Eine Beweisskizze für diese Aussage findet man in [Ka87b].

2.5 LLL-reduzierte Basen

Wir wollen damit beginnen, die Bedingungen, die eine LLL-reduzierte Basis gemäß Definition erfüllt, zu interpretieren. Warum verlangen wir, daß eine reduzierte Basis schwach reduziert sein soll? Kleine μ_{ij} sorgen dafür, daß die Basisvektoren nahezu orthogonal aufeinander stehen und damit sind die Basisvektoren auch kurz, wie wir in Kapitel 1 gesehen haben. Ist

$$|\mu_{ij}| \leq 1/2, \quad 1 \leq j < i \leq n,$$

dann ist $|\mu_{ij}|$ für $1 \leq j < i \leq n$ „klein" und da

$$\mu_{ij} = \frac{\langle \underline{b}_i, \underline{b}_j^\star \rangle}{\langle \underline{b}_j^\star, \underline{b}_j^\star \rangle} \text{ für } 1 \leq j < i \leq n,$$

ist $\underline{b}_i$ „nahezu" orthogonal zu $\underline{b}_j^\star$. Ist $\underline{b}_1, \ldots, \underline{b}_n$ schwach reduziert, dann gilt

$$|\langle \underline{b}_i, \underline{b}_j^\star \rangle| \leq 1/2\, \|\underline{b}_j^\star\|, \quad 1 \leq j < i \leq n.$$

Damit ist für $1 \leq j < i \leq n$

$$\begin{aligned} |\langle \underline{b}_i, \underline{b}_j \rangle| &= |\langle \underline{b}_i, \underline{b}_j^\star \rangle + \sum_{k=1}^{j-1} \mu_{jk} \langle \underline{b}_i, \underline{b}_k^\star \rangle| \\ &\leq |\langle \underline{b}_i, \underline{b}_j^\star \rangle| + \sum_{k=1}^{j-1} |\mu_{jk}|\, |\langle \underline{b}_i, \underline{b}_k^\star \rangle| \\ &\leq \|\underline{b}_j^\star\|^2 + \sum_{k=1}^{j-1} 1/4\, \|\underline{b}_k^\star\|^2. \end{aligned}$$

Wie wir sehen, führen die „kleinen" μ-Werte dazu, daß auch $|\langle \underline{b}_i, \underline{b}_j \rangle|$ klein ist, also $\underline{b}_i$ „nahezu" orthogonal zu $\underline{b}_j$ ist. Mit Algorithmus 2.14 können wir eine schwach reduzierte Basis aus einer beliebigen Basis berechnen, ohne daß sich die Gram-Schmidt-Zerlegung ändert (Bemerkung 2.19). Daß eine schwach reduzierte Basis aus kurzen Gittervektoren besteht, sehen wir dann mit Hilfe der Gleichung

$$\|\underline{b}_j\|^2 = \|\underline{b}_j^\star\|^2 + \sum_{i=1}^{j-1} \mu_{ij}^2 \|\underline{b}_i^\star\|^2, \quad 1 \leq j \leq k. \tag{2.7}$$

Wie können wir die zweite Eigenschaft interpretieren? Sei $1 < i \leq n$. Wir stellen zunächst einmal fest, daß die Vektoren $\underline{b}_{i-1}^\star$ bzw. $\underline{b}_i^\star + \mu_{i\,i-1} \cdot \underline{b}_{i-1}^\star$ die Projektionen von $\underline{b}_i$ bzw. $\underline{b}_{i-1}$ auf das orthogonale Komplement von $\langle\langle \underline{b}_1, \ldots, \underline{b}_{i-2} \rangle\rangle$ sind. Für $\underline{b}_i$ und $\underline{b}_{i-1}$ gelten gemäß dem Verfahren von Gram-Schmidt

$$\underline{b}_{i-1} = \underline{b}_{i-1}^\star + \sum_{j=1}^{i-2} \mu_{i-1j} \cdot \underline{b}_j^\star, \tag{2.8}$$

$$\underline{b}_i = \underline{b}_i^\star + \sum_{j=1}^{i-1} \mu_{ij} \cdot \underline{b}_j^\star. \tag{2.9}$$

Vertauschen wir $\underline{b}_{i-1}$ und $\underline{b}_i$ und wenden auf die Basis

$$(\underline{\tilde{b}}_1, \ldots, \underline{\tilde{b}}_{i-1}, \underline{\tilde{b}}_i, \ldots, \underline{\tilde{b}}_n) := (\underline{b}_1, \ldots, \underline{b}_{i-2}, \underline{b}_i, \underline{b}_{i-1}, \underline{b}_{i+1}, \ldots, \underline{b}_n)$$

das Verfahren von Gram-Schmidt an, dann erhalten wir

$$\underline{b}_i = \underline{\tilde{b}}_{i-1} = \underline{\tilde{b}}^\star_{i-1} + \sum_{j=1}^{i-2} \mu_{ij} \cdot \underline{b}^\star_j, \tag{2.10}$$

$$\underline{b}_{i-1} = \underline{\tilde{b}}_i = \underline{\tilde{b}}^\star_{i-1} + \frac{\langle \underline{b}_{i-1}, \underline{\tilde{b}}^\star_{i-1} \rangle}{\langle \underline{\tilde{b}}^\star_{i-1}, \underline{\tilde{b}}^\star_{i-1} \rangle} \cdot \underline{\tilde{b}}^\star_{i-1} + \sum_{j=1}^{i-2} \mu_{i-1j} \cdot \underline{b}^\star_j. \tag{2.11}$$

Aus (2.9) folgt mit (2.10)

$$\underline{\tilde{b}}^\star_{i-1} = \underline{b}^\star_{i-1} + \mu_{i\,i-1} \cdot \underline{b}^\star_{i-1},$$

was der linken Seite der Ungleichung (2.4) entspricht. Diese Bedingung besagt also, daß in einer LLL-reduzierten Basis durch Vertauschen von $\underline{b}_{i-1}$ und $\underline{b}_i$ der Vektor $\underline{\tilde{b}}^\star_{i-1}$ nicht wesentlich kürzer wird als $\underline{b}^\star_{i-1}$. Wegen (2.8), (2.9) und weil $|\mu_{ij}| \leq 1/2$ für $1 \leq j < i \leq n$ ist, hat damit eine Vertauschung und eine anschließende Reduktion auch keinen großen Einfluß auf $\underline{b}_{i-1}$ und $\underline{b}_i$. Die Zahl 3/4 tritt auf, damit die Basis in Polynomzeit reduziert werden kann (siehe dazu Abschnitt 2.5.2).

2.5.1 Eigenschaften LLL-reduzierter Basen

Wir wollen nun die Längen der Basisvektoren einer LLL-reduzierten Basis mit Hilfe der sukzessiven Minima und der Determinante des Gitters L abschätzen.

Satz 2.9 *Sei $L \subset \mathbb{R}^d$ ein Gitter und $\underline{b}_1, \ldots, \underline{b}_n$ eine LLL-reduzierte Basis von L. Dann gelten folgende Abschätzungen:*

$$\|\underline{b}_1\| \cdot \ldots \cdot \|\underline{b}_n\| \leq 2^{n(n-1)/4}\, \mathrm{d}(L), \tag{2.12}$$
$$\|\underline{b}_1\| \leq 2^{(n-1)/4}\, \mathrm{d}(L)^{1/n}, \tag{2.13}$$
$$2^{(1-i)/2}\, \lambda_i(L) \leq \|\underline{b}_i\| \leq 2^{(n-1)/2}\, \lambda_i(L) \text{ für } 1 \leq i \leq n. \tag{2.14}$$

Zunächst beweisen wir einige Hilfsaussagen, um anschließend damit den Satz zu zeigen. Die Behauptungen und die dazugehörigen Beweise sind weitgehend dem Originalartikel entnommen [LeLeLo82]. Wir wollen sie hier ausführlich widergeben, damit wir später auch einzelne Beweisschritte mühelos heranziehen können.

Lemma 2.10 *Sei $L \subset \mathbb{R}^d$ ein Gitter und $\underline{b}_1, \ldots, \underline{b}_n$ eine LLL-reduzierte Basis von L. Dann gilt:*

$$\|\underline{b}_j\| \leq 2^{(i-1)/2}\, \|\underline{b}^\star_i\| \text{ für } 1 \leq j \leq i \leq n. \tag{2.15}$$

Beweis: Da $\langle \underline{b}_i^\star, \underline{b}_j^\star \rangle = 0$ für $1 \leq i \neq j \leq n$ folgt mit (2.3) und (2.4) in Definition 2.5

$$\begin{array}{rc} & \|\underline{b}_i^\star\|^2 + \|\mu_{i\,i-1} \cdot \underline{b}_{i-1}^\star\|^2 = \|\underline{b}_i^\star + \mu_{i\,i-1} \cdot \underline{b}_{i-1}^\star\| \geq 3/4\,\|\underline{b}_{i-1}^\star\|^2 \\ \Rightarrow & \|\underline{b}_i^\star\|^2 \geq (3/4 - (\mu_{i\,i-1})^2)\,\|\underline{b}_{i-1}^\star\|^2 \geq 1/2\,\|\underline{b}_{i-1}^\star\|^2 \end{array}$$

für $1 < i \leq n$. Induktiv erhalten wir

$$\|\underline{b}_j^\star\|^2 \leq 2^{i-j}\,\|\underline{b}_i^\star\|^2 \text{ für } 1 \leq j \leq i \leq n. \tag{2.16}$$

Mit (1.1) und (2.3) gilt für $1 < i \leq n$:

$$\begin{aligned} \|\underline{b}_i\|^2 &= \|\underline{b}_i^\star\|^2 + \sum_{j=1}^{i-1} \mu_{ij}^2\,\|\underline{b}_j^\star\|^2 \\ &\leq \|\underline{b}_i^\star\|^2 + \sum_{j=1}^{i-1} 1/4 \cdot 2^{i-j}\,\|\underline{b}_i^\star\|^2 \\ &= (1 + 1/4\,(2^i - 2))\,\|\underline{b}_i^\star\|^2 \\ &\leq 2^{i-1}\,\|\underline{b}_i^\star\|^2. \end{aligned}$$

Für $1 \leq j \leq i \leq n$ haben wir also

$$\|\underline{b}_j\|^2 \leq 2^{j-1}\,\|\underline{b}_j^\star\|^2 \leq 2^{i-1}\,\|\underline{b}_i^\star\|^2.$$

□

Lemma 2.11 *Sei $L \subset \mathbb{R}^d$ ein Gitter und $\underline{b}_1, \ldots, \underline{b}_n$ eine Basis von L, $\underline{b}_1^\star, \ldots, \underline{b}_n^\star$ die Gram-Schmidt-Zerlegung von $\underline{b}_1, \ldots, \underline{b}_n$. Dann gilt für einen beliebigen Nichtnullvektor $\underline{b}$ aus L, daß*

$$\|\underline{b}\| \geq \min\{\|\underline{b}_1^\star\|, \ldots, \|\underline{b}_n^\star\|\}.$$

Beweis: Sei

$$\underline{b} = \sum_{i=1}^{n} \alpha_i \cdot \underline{b}_i,\ \alpha_i \in \mathbb{Z} \text{ für } 1 \leq i \leq n. \tag{2.17}$$

Sei l der größte Index mit $\alpha_l \neq 0$. Ersetzen wir $\underline{b}_i$ in (2.17) wie folgt

$$\underline{b}_i = \sum_{j=1}^{i} \mu_{ij} \cdot \underline{b}_j^\star, \quad 1 \leq i \leq l,$$

so erhalten wir

$$\underline{b} = \sum_{i=1}^{l} \alpha_i^\star \cdot \underline{b}_i^\star, \quad \alpha_i^\star \in \mathbb{R} \text{ für } 1 \leq i \leq l.$$

Es gilt dann

$$\underline{b} = \alpha_l^\star \cdot \underline{b}_l + \sum_{i=1}^{l-1} (\alpha_i^\star - \mu_{li}) \cdot \underline{b}_i^\star.$$

Somit ist $\alpha_l^\star = \alpha_l$, also $\alpha_l^\star$ eine ganze Zahl und ungleich Null. Es folgt also

$$\|\underline{b}\|^2 = \sum_{i=1}^{l} (\alpha_i^\star)^2\,\|\underline{b}_i^\star\|^2 \geq \alpha_l^2\,\|\underline{b}_l^\star\|^2 \geq \|\underline{b}_l^\star\|^2,$$

womit die Behauptung gezeigt wurde. □

Lemma 2.12 *Sei $L \subset \mathbb{R}^d$ ein Gitter und $\underline{b}_1, \ldots, \underline{b}_n$ eine LLL-reduzierte Basis von L. Seien $\underline{a}_1, \ldots, \underline{a}_k \in L$ linear unabhängig. Dann gilt*

$$\|\underline{b}_j\| \leq 2^{(n-1)/2} \cdot \max\{\|\underline{a}_1\|, \ldots, \|\underline{a}_k\|\} \tag{2.18}$$

für $j = 1, \ldots, k$.

Beweis: Wir stellen die $\underline{a}_j$, $1 \leq j \leq k$, als Erzeugnisse der LLL-reduzierten Basis dar. Sei also $\underline{a}_j = \sum_{l=1}^{n} \alpha_{lj} \cdot \underline{b}_l$, $\alpha_{lj} \in \mathbb{Z}$ mit $1 \leq l \leq n$ und $1 \leq j \leq k$. Für festes j sei l_j der größte Index l mit $\alpha_{lj} \neq 0$. Mit Hilfe des Beweises von Lemma 2.11 sehen wir, daß dann

$$\|\underline{a}_j\| \geq \|\underline{b}^\star_{l_j}\| \tag{2.19}$$

gilt. Wir numerieren nun die $\underline{a}_j$ so um, daß $l_1 \leq l_2 \leq \ldots \leq l_k$ gilt. Dann ist

$$j \leq l_j, \qquad \text{für } 1 \leq j \leq k. \tag{2.20}$$

Würde dies nicht gelten, dann wäre $\dim\langle\langle \underline{a}_1, \ldots, \underline{a}_j \rangle\rangle = j-1$, woraus sich ein Widerspruch zur linearen Unabhängigkeit von $\underline{a}_1, \ldots, \underline{a}_k$ ergeben würde. Mit (2.20), Lemma 2.10 und (2.19) erhalten wir

$$\|\underline{b}_j\| \leq 2^{(l_j-1)/2}\, \|\underline{b}^\star_{l_j}\| \leq 2^{(n-1)/2}\, \|b^\star_{l_j}\| \leq 2^{(n-1)/2}\, \|\underline{a}_j\|$$

für $j = 1, 2, \ldots, k$. □

Wir wollen nun Satz 2.9 beweisen.

Beweis: Zu (2.14): Mit

$$\|\underline{b}_i\|^2 = \|\underline{b}^\star_i\|^2 + \sum_{j=1}^{i-1} (\mu_{ij})^2 \|\underline{b}^\star_j\|^2, \quad 1 \leq i \leq n,$$

haben wir

$$\|\underline{b}^\star_i\| \leq \|\underline{b}_i\|, \quad 1 \leq i \leq n,$$

und damit folgt aus Lemma 2.10

$$\|\underline{b}_j\| \leq 2^{(i-1)/2}\, \|\underline{b}_i\| \text{ für } 1 \leq j \leq i \leq n.$$

Da

$$\lambda_i(L) \leq \max\{\|\underline{b}_1\|, \ldots, \|\underline{b}_i\|\},$$

gilt

$$\lambda_i(L) \leq 2^{(i-1)/2}\, \|\underline{b}_i\|,$$

womit die erste Ungleichung folgt.
Die zweite Ungleichung $\|\underline{b}_i\| \leq 2^{(n-1)/2}\, \lambda_i(L)$ für $1 \leq i \leq n$ folgt, wenn wir in

Lemma 2.12 $k = i$ setzen und $\underline{a}_1, \ldots, \underline{a}_k$ derart wählen, daß $\|\underline{a}_j\| = \lambda_j(L)$. Solche $\underline{a}_j$ gibt es ([Ca59] VIII.1). Zu (2.12) und (2.13):
Die Gleichung (1.7), besagt daß

$$\mathrm{d}(L) = \prod_{i=1}^{n} \|\underline{b}_i^\star\|.$$

Aus $\|\underline{b}_i^\star\| \leq \|\underline{b}_i\|$ und $\|\underline{b}_i\| \leq 2^{(i-1)/2}\|\underline{b}_i^\star\|$ folgt (2.12). Wenn wir bei festem $j = 1$ das Produkt der n Ungleichungen (2.15) für $i = 1, 2, \ldots, n$ bilden, erhalten wir (2.13). □

2.5.2 Der LLL-Algorithmus

Der LLL-Algorithmus imitiert den Gaußalgorithmus auf den Gitterbasisvektoren $\underline{b}_i$ und $\underline{b}_{i-1}$. Um festzustellen, ob ein Vertauschen der beiden Vektoren und eine anschließende Reduktion sinnvoll sind, vergleichen wir allerdings nicht direkt $\|\underline{b}_i\|$ mit $\|\underline{b}_{i-1}\|$, sondern bilden die Projektionen von $\underline{b}_{i-1}$ und $\underline{b}_i$ auf $\langle\langle \underline{b}_1, \ldots, \underline{b}_{i-2}\rangle\rangle$. Die Projektionen erhalten wir als $\underline{b}_{i-1}^\star$ und $\underline{b}_i^\star + \mu_{i\,i-1} \cdot \underline{b}_{i-1}^\star$. Falls nun

$$\|\underline{b}_i^\star + \mu_{i\,i-1} \cdot \underline{b}_{i-1}^\star\|^2 < 3/4 \cdot \|\underline{b}_{i-1}^\star\|^2$$

ist, vertauschen wir $\underline{b}_{i-1}$ und $\underline{b}_i$, denn dann ist eine erneute Reduktion sinnvoll. Der Faktor $3/4$ kann ersetzt werden durch einen beliebigen Wert, der größer ist als $1/4$ und kleiner ist als 1; der Algorithmus läuft dann immer noch in Polynomzeit und berechnet weiterhin eine reduzierte Basis, die Definition 2.5 mit entsprechend geändertem Faktor genügt. Nehmen wir allerdings 1 als Faktor, dann ist die Laufzeit nicht mehr polynomiell.

Beschreibung und Beweisskizze für die Korrektheit

Wir werden jetzt den LLL-Algorithmus beschreiben und dabei einen Beweis für seine Korrektheit skizieren (siehe dazu [LeLeLo82]). Sei $\underline{b}_1, \ldots, \underline{b}_n$ eine beliebige Basis von L. Der nun vorgestellte Algorithmus berechnet daraus eine LLL-reduzierte Basis. Der Algorithmus beginnt mit der Berechnung der $\|\underline{b}_i^\star\|^2$ für $1 \leq i \leq n$ und der μ_{ij} für $1 \leq j < i \leq n$ gemäß dem Verfahren von Gram-Schmidt. Die Kenntnis der $\underline{b}_i^\star$ ist nicht erforderlich. Wir werden sehen, daß wir μ_{ij} für $1 \leq j < i$ bei einer Änderung von $\underline{b}_i$ während der Abarbeitung des Algorithmus neu berechnen können, ohne das Verfahren von Gram-Schmidt auf $\underline{b}_1, \ldots, \underline{b}_i$ erneut anwenden zu müssen.

Wir initialisieren nun eine Variable k mit 2. An dieser Stelle im Algorithmus und nach einer Reihe von noch zu beschreibenden Operationen gilt stets die folgende Invariante

$$|\mu_{ij}| \leq 1/2, \quad 1 \leq j < i < k, \tag{2.21}$$

$$\|\underline{b}_i^\star + \mu_{i\,i-1} \cdot \underline{b}_{i-1}^\star\|^2 \geq 3/4 \cdot \|\underline{b}_{i-1}^\star\|^2 \text{ für } 1 < i < k. \tag{2.22}$$

Sie besagt, daß $\underline{b}_1, \ldots, \underline{b}_{k-1}$ LLL-reduziert sind. Ist $k = n+1$, dann endet der Algorithmus, denn dann haben wir eine vollständig LLL-reduzierte Basis. Für $k = 2$ ist die Invariante trivialerweise gültig.

Wir beschreiben nun die Wirkung einer Reihe von Operationen, nach denen die Invariante wieder gilt. Wir werden uns erst weiter unten damit befassen, wie die aufgerufenen Prozeduren die Änderungen an der Basis erreichen und welche Überlegungen den Berechnungen in den Prozeduren zu Grunde liegen.

Sei $2 \leq k \leq n$. Mit der ersten Operation sorgen wir dafür, daß

$$|\mu_{kk-1}| \leq 1/2. \tag{2.23}$$

Wir führen dies mit der Prozedur REDUCE durch. Die weitere Vorgehensweise hängt nun von zwei verschiedenen Fällen ab, die auftreten können:

1. Fall: Gilt $k \geq 2$ und ist

$$\|\underline{b}_k^\star + \mu_{kk-1} \cdot \underline{b}_{k-1}^\star\|^2 < 3/4 \cdot \|\underline{b}_{k-1}^\star\|^2, \tag{2.24}$$

dann rufen wir die Prozedur SWAP auf. In ihr werden $\underline{b}_{k-1}$ und $\underline{b}_k$ vertauscht und einige der μ_{ij}, $1 \leq i < j \leq n$, neu berechnet, damit wir weiterhin eine Darstellung der Gitterbasis als Erzeugnis einer Orthogonalbasis haben. Das Entscheidende an diesem Tausch ist, daß $\|\underline{b}_{k-1}^\star\|^2$ durch $\|\underline{b}_k^\star + \mu_{kk-1} \cdot \underline{b}_{k-1}^\star\|^2$ ersetzt wurde und damit der neue Wert $\|\underline{b}_{k-1}^\star\|^2$ weniger als 3/4 mal so groß ist wie der alte. Bei der Vertauschung handelt es sich um eine unimodulare Transformation. Da sich durch die Vertauschung die Darstellung der Vektoren bzgl. der Orthogonalbasis geändert hat, die Invariante aber weiterhin gelten soll, setzen wir k auf $k-1$ und fahren mit der ersten Operation fort.

2. Fall: Sei $k = 1$ oder

$$\|\underline{b}_k^\star + \mu_{kk-1} \cdot \underline{b}_{k-1}^\star\|^2 \geq 3/4 \cdot \|\underline{b}_{k-1}^\star\|^2, \tag{2.25}$$

dann sorgen wir mit der Prozedur REDUCE dafür, daß

$$|\mu_{kj}| \leq 1/2, \quad \text{für } 1 \leq j \leq k-1, \tag{2.26}$$

gilt. Wir setzen k auf $k+1$. Die Invarinate gilt nun für $k+1$. Ist das neue k gleich $n+1$, dann sind wir fertig, ansonsten fahren wir auch hier mit unserer ersten Operation fort. Wir zeigen in der Laufzeitanalyse, daß der Algorithmus terminiert.

Das Hauptprogramm des Algorithmus sieht also folgendermaßen aus:

Algorithmus 2.13 LLL(n; $\underline{b}_1, \ldots, \underline{b}_n$)

EINGABE: Dimension n und eine Basis $\underline{b}_1, \ldots, \underline{b}_n$ des Gitters L.
AUSGABE: LLL-reduzierte Basis $\underline{b}_1, \ldots, \underline{b}_n$ von L.

Berechne $\|\underline{b}_1^\star\|^2, \ldots, \|\underline{b}_n^\star\|^2$ und die dazugehörigen μ_{ij} für $1 \leq j < i \leq n$ mit dem Verfahren von Gram-Schmidt.
$k := 2$;
while ($k \leq n$) **do**
  REDUCE(μ_{kk-1});
  if ($\|\underline{b}_k^\star\|^2 < (3/4 - \mu_{kk-1}) \cdot \|\underline{b}_{k-1}^\star\|^2$) **then**
    /* $\Leftrightarrow \|\underline{b}_k^\star + \mu_{kk-1}\underline{b}_{k-1}^\star\|^2 < 3/4 \cdot \|\underline{b}_{k-1}^\star\|^2$ */
    SWAP($\underline{b}_{k-1}, \underline{b}_k$);
    $k := \max\{k-1, 2\}$;
  else
    for ($l := k-2$; $l \geq 1$; $l := l-1$) **do**
      REDUCE(μ_{kl});
    od
    $k := k+1$;
  fi
od

Beschreibung der Prozeduren REDUCE und SWAP

Durch die Prozedur REDUCE erreichen wir, daß für die Basis am Ende des Algorithmus $|\mu_{ij}| \leq 1/2$ für $1 \leq j < i \leq n$ gilt. Analog zum Gaußalgorithmus ziehen wir das $\lceil \mu_{ij} \rfloor$-fache von $\underline{b}_j$ von $\underline{b}_i$ ab, wenn wir $\underline{b}_i$ mit Hilfe von $\underline{b}_j$ verkürzen wollen. Wir machen uns an dieser Stelle mit (2.7) noch einmal klar, daß $\underline{b}_i$ in Richtung von $\underline{b}_j$ wirklich verkürzt wird, wenn $|\mu_{ij}| > 1/2$ vor Beginn dieser Prozedur war, denn die Orthogonalbasis $\underline{b}_1^\star, \ldots, \underline{b}_n^\star$ bleibt unverändert. Eine Verkürzung von $\underline{b}_i$ in Richtung von $\underline{b}_j$ muß aber zu keiner echten Verkürzung von $\underline{b}_i$ führen! Betrachte dazu Beispiel 2.20 am Ende des Abschnitts.

Algorithmus 2.14 REDUCE(μ_{ij})

EINGABE: μ_{ij}, μ_{il} und μ_{jl} für $1 \leq l \leq j$ ($\mu_{jj} = 1$).
AUSGABE: $\underline{b}_i$ derart, daß $|\mu_{ij}| \leq 1/2$; neue μ_{il} für $1 \leq l \leq j$.

if ($|\mu_{ij}| > 1/2$) **then**
  $\underline{b}_i := \underline{b}_i - \lceil \mu_{ij} \rfloor \cdot \underline{b}_j$;
  for ($l := 1$; $l \leq j$; $l := l+1$) **do**
    $\mu_{il} := \mu_{il} - \lceil \mu_{ij} \rfloor \cdot \mu_{jl}$;
  od
fi

Mit der Prozedur SWAP vertauschen wir zwei Basisvektoren und führen eine Reihe von Anweisungen durch, die gewährleisten, daß wir, wie vor der Vertauschung, wieder eine korrekte Darstellung der beiden Basisvektoren als Erzeugnis der Orthogonalbasis erhalten. Wir verzichten dabei auf eine erneute Anwendung des Verfahrens von Gram-Schmidt. Statt alle $\mu_{ij},\ 1 \leq j < i \leq n,$ neu zu berechnen, finden wir mit einer speziellen Rechenvorschrift nur die neuen Werte für μ_{kk-1}, $\mu_{k-1,j}$, μ_{kj} für $1 \leq j \leq k-2$ und μ_{ik-1}, μ_{ik} für $k < i$, denn nur diese könnten sich bei der Vertauschung verändert haben. Außerdem werden noch die neuen Beträge $\|\underline{b}_{k-1}^\star\|^2$ und $\|\underline{b}_k^\star\|^2$ berechnet. Eine Herleitung der Berechnungsvorschrift findet man im Originalartikel.

Algorithmus 2.15 SWAP$(\underline{b}_{k-1}, \underline{b}_k)$

EINGABE: Basisvektoren $\underline{b}_k$, $\underline{b}_{k-1}$ und μ_{kk-1}, μ_{k-1j}, μ_{kj}, μ_{ik-1} und μ_{ik} für $j < k-1$ und $i > k$, $B_i := \|\underline{b}_i^\star\|^2$ für $i = k-1, k$.
AUSGABE: $\underline{b}_{k-1}$ und $\underline{b}_k$ vertauscht und die dazugehörigen neuen μ_{k-1j}, μ_{kj} für $1 \leq j \leq k-2$ und μ_{kk-1}.

(1) $\mu := \mu_{kk-1}$;
(2) $B := B_k + \mu^2 \cdot B_{k-1}$;
(3) $\mu_{kk-1} := \mu \cdot B_{k-1}/B$;
(4) $B_k := B_{k-1} \cdot B_k/B$;
(5) $B_{k-1} := B$;
(6) $\begin{pmatrix} \underline{b}_{k-1} \\ \underline{b}_k \end{pmatrix} := \begin{pmatrix} \underline{b}_k \\ \underline{b}_{k-1} \end{pmatrix}$;
(7) **for** $(j := 1;\ j \leq k-2;\ j := j+1)$ **do**
(8) $\quad \begin{pmatrix} \mu_{k-1j} \\ \mu_{kj} \end{pmatrix} := \begin{pmatrix} \mu_{kj} \\ \mu_{k-1j} \end{pmatrix}$
(9) **od**
(10) **for** $(i := k+1;\ i \leq n;\ i := i+1)$ **do**
(11) $\quad \begin{pmatrix} \mu_{ik-1} \\ \mu_{ik} \end{pmatrix} := \begin{pmatrix} 1 & \mu_{kk-1} \\ 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 \\ 1 & -\mu \end{pmatrix} \cdot \begin{pmatrix} \mu_{ik-1} \\ \mu_{ik} \end{pmatrix}$
(12) **od**

Laufzeitanalyse

Proposition 2.16 *Sei $\mathcal{B} \geq 2$ mit $\|\underline{b}_i\|^2 \leq \mathcal{B}$ für $1 \leq i \leq n$. Der LLL-Algorithmus terminiert, denn die Anzahl der Schleifendurchläufe ist durch $O(n^2 \log \mathcal{B})$ beschränkt.*

Beweis: Wir definieren $d_0 = 1$ und für $1 \leq l \leq n$

$$d_l = \det(\langle \underline{b}_i, \underline{b}_j \rangle_{1 \leq i,j \leq l}) = \mathrm{d}(L(\underline{b}_1, \ldots, \underline{b}_l))^2. \tag{2.27}$$

Mit (1.7) ist

$$d_l = \prod_{i=1}^{l} \|\underline{b}_i^\star\|^2. \tag{2.28}$$

Insbesondere ist $d_n = \mathrm{d}(L)^2$. Wir untersuchen nun, welchen Einfluß die Anweisungen des Algorithmus auf die d_l haben. Da REDUCE die Orthogonalbasis $\underline{b}_1^\star, \ldots, \underline{b}_n^\star$ nicht verändert, haben die Anweisungen innerhalb der Prozedur auch keinen Einfluß auf die d_l. Die Prozedur SWAP wird nur dann ausgeführt, wenn $\|\underline{b}_k^\star + \mu_{kk-1} \cdot \underline{b}_{k-1}^\star\|^2 < 3/4\,\|\underline{b}_{k-1}^\star\|^2$ ist. Ist dies der Fall, wird $\underline{b}_{k-1}^\star$ durch $\underline{b}_k^\star + \mu_{kk-1} \cdot \underline{b}_{k-1}^\star$ ersetzt und damit d_{k-1} um 3/4 verkürzt. Da innerhalb der Basis des Gitters außerdem nur $\underline{b}_k$ und $\underline{b}_{k-1}$ vertauscht werden, bleiben wegen (2.27) alle anderen d_l unverändert. Wir definieren

$$D = \prod_{l=1}^{n} d_l. \tag{2.29}$$

Da die Vektoren $\underline{b}_i$ stets ganzzahlige Einträge haben und weil wegen (2.28) die d_l immer positiv sind, ist während der gesamten Laufzeit $D \in \mathbb{N}$, also insbesondere nach unten durch 1 beschränkt.

Wie wir gesehen haben, wird andererseits mit jeder Abarbeitung der Prozedur SWAP D um 3/4 kleiner. Wir können also ausrechnen, wie oft SWAP maximal aufgerufen wird. Es ist

$$\begin{aligned} D &= \|\underline{b}_1^\star\|^{2n} \cdot \|\underline{b}_2^\star\|^{2(n-1)} \cdot \ldots \cdot \|\underline{b}_n^\star\|^2 \\ &\le \|\underline{b}_1\|^{2n} \cdot \|\underline{b}_2\|^{2(n-1)} \cdot \ldots \cdot \|\underline{b}_n\|^2 \\ &\le (\|\underline{b}_1\| \cdot \ldots \cdot \|\underline{b}_n\|)^{2n}. \end{aligned}$$

Sei s die Anzahl an Aufrufen von SWAP. Dann gilt für s

$$(3/4)^s \cdot (\|\underline{b}_1\| \cdot \ldots \cdot \|\underline{b}_n\|)^{2n} \ge 1$$

$$\Leftrightarrow s \cdot \log(3/4) + 2n(\log \|\underline{b}_1\| + \ldots + \log \|\underline{b}_n\|) \ge 0$$

$$\Leftrightarrow s \le \left(2n \sum_{i=1}^{n} \log \|\underline{b}_i\|\right) / \log(4/3).$$

Damit ist s von der Größenordnung $O(n^2 \log \mathcal{B})$. Ebenso oft wird k um 1 dekrementiert. Da am Anfang $k = 2$ und während der gesamten Laufzeit $k \le n+1$ ist, wird der else-Teil des Algorithmus, in dem k um 1 inkrementiert wird, höchstens $s + n - 1$-mal durchlaufen. Damit ist aber die Anzahl der Schleifendurchläufe durch $O(n^2 \log \mathcal{B})$ beschränkt und der Algorithmus terminiert. $\square$

Das folgende Lemma ist notwendig, um die Größe der auftretenden Zahlen abzuschätzen, wenn $L \subseteq \mathbb{Z}^d$ ist.

Lemma 2.17 *Sei $L \subseteq \mathbb{Z}^d$ ein Gitter und $\underline{b}_1, \ldots, \underline{b}_n$ eine Basis von L, $\underline{b}_1^\star, \ldots, \underline{b}_n^\star$ die Gram-Schmidt-Zerlegung von L mit den entsprechenden μ_{ij}, $1 \leq j < i \leq n$, d_i, $0 \leq i \leq n$, gemäß (2.28) definiert. Dann gilt*

$$\|\underline{b}_i^\star\|^2 = d_i / d_{i-1}, \quad 1 \leq i \leq n, \tag{2.30}$$

$$d_{i-1} \cdot \underline{b}_i^\star \in L \subseteq \mathbb{Z}^d, \quad 1 \leq i \leq n, \tag{2.31}$$

$$d_j\, \mu_{ij} \in \mathbb{Z}, \quad 1 \leq j < i \leq n. \tag{2.32}$$

Beweis: (2.30) folgt direkt aus (2.28). Da $\underline{b}_j^\star \in \langle\langle \underline{b}_1, \ldots, \underline{b}_j \rangle\rangle$ für $1 \leq j \leq n$, gibt es $\delta_{ij} \in \mathbb{R}$ mit $\underline{b}_i^\star = \underline{b}_i - \sum_{j=1}^{i-1} \delta_{ij} \cdot \underline{b}_j$, $1 \leq i \leq n$. Betrachten wir das i-te Gleichnungssystem in den Variablen $\delta_{i1}, \ldots, \delta_{i\,i-1}$, also für $1 \leq l \leq i-1$

$$\langle \underline{b}_i, \underline{b}_l \rangle = \sum_{j=1}^{i-1} \delta_{ij} \, \langle \underline{b}_j, \underline{b}_l \rangle.$$

Beachte, daß $\langle \underline{b}_i^\star, \underline{b}_l \rangle = 0$ für $1 \leq l < i \leq n$. Dann folgt mit der Cramerschen Regel (siehe z.B. [La78], S. 174), daß $d_{i-1}\, \delta_{ij} \in \mathbb{Z}$. Damit aber haben wir (2.31) bewiesen, denn $d_{i-1} \in \mathbb{Z}$ und

$$d_{i-1} \cdot \underline{b}_i^\star = d_{i-1} \cdot \underline{b}_i - \sum_{j=1}^{i-1} d_{i-1}\, \delta_{ij} \cdot \underline{b}_j, \ 1 \leq i \leq n.$$

Analog kann man zeigen, daß

$$d_{j-1} \cdot \underline{b}_i(j) \in \mathbb{Z}^d, \ 1 \leq j < i \leq n.$$

Mit (2.30) und (2.31) folgt

$$d_j \mu_{ij} = d_j \langle \underline{b}_i, \underline{b}_j^\star \rangle / \langle \underline{b}_j^\star, \underline{b}_j^\star \rangle = d_{j-1} \langle \underline{b}_i, \underline{b}_j^\star \rangle = \langle \underline{b}_i, d_{j-1} \underline{b}_j^\star \rangle \in \mathbb{Z}, \text{ also (2.32).} \qquad \square$$

Satz 2.18 *Sei $L \subseteq \mathbb{Z}^d$ ein Gitter und $\underline{b}_1, \ldots, \underline{b}_n$ eine Basis von L; $\mathcal{B} \geq 2$ derart, daß $\|\underline{b}_i\|^2 \leq \mathcal{B}$ für $1 \leq i \leq n$. Rufen wir den LLL-Algorithmus mit $\underline{b}_1, \ldots, \underline{b}_n$ als Eingabe auf, dann terminiert er nach $O(n^3 d \log \mathcal{B})$ arithmetischen Operationen auf ganzen Zahlen der binären Länge $O(n \log \mathcal{B})$.*

Beweis: Man sieht leicht ein, daß nur rationale Zahlen auftreten. Wie wir weiter unten sehen werden, genügt es, die Anzahl an maximal notwendigen arithmetischen Operationen unter der Annahme abzuschätzen, daß sie auf rationalen Zahlen ausgeführt werden.

Die Anwendung des Verfahrens von Gram-Schmidt erfordert $O(n^2 d)$ Operationen. Aus Proposition 2.16 folgt, daß die Schleife $O(n^2 \log \mathcal{B})$-mal durchlaufen wird. In REDUCE werden $O(n+d)$ Operationen ausgeführt, in SWAP $O(d)$. Damit sind für den else-Fall $O(nd)$ Operationen notwendig, da $d \geq n$. Insgesamt werden also $O(n^3 d \log \mathcal{B})$ Operationen ausgeführt.

Wir wollen jetzt erläutern, wie man alle Berechnungen nur durch Operationen auf ganzen Zahlen durchführen kann. Gemäß Lemma 2.17 können $\|\underline{b}_i^\star\|^2$, $1 \le i \le n$, und μ_{ij}, $1 \le j < i \le n$, als Brüche mit entsprechenden ganzzahligen Nenner dargestellt werden. Aus der Gram-Schmidt-Zerlegung $\underline{b}_1^\star, \ldots, \underline{b}_n^\star$ berechnet man die d_i am Anfang des Algorithmus für $1 \le i \le n$ und nur während der Abbarbeitung der Prozedur SWAP ändert sich ein d_i, nämlich aus d_{k-1} wird $d_{k-2}(\|\underline{b}_k^\star\|^2 + \mu_{kk-1}^2 \|\underline{b}_{k-1}^\star\|^2)$, wogegen alle anderen d_i unverändert bleiben. Die $\|\underline{b}_i^\star\|^2$ und μ_{ij} bestimmt man entsprechend den Anweisungen neu, indem man Zähler und Nenner der rationalen Zahlen durch Operationen auf ganzen Zahlen neu berechnet, diese dann aber mit Hilfe der bekannten, ganzzahligen d_i sofort gekürzt werden.

Von welcher Größenordnung sind nun die binären Längen aller auftetenden ganzen Zahlen? In [LeLeLo82], S. 524, wird gezeigt, daß $\max\{\|\underline{b}_i^\star\|^2 \mid 1 \le i \le n\}$ sich nie erhöht, damit

$$\|\underline{b}_i^\star\|^2 \le \mathcal{B}, \quad 1 \le i \le n \tag{2.33}$$

und daß

$$|\mu_{ij}| \le 2^{n-1}(n\mathcal{B}^{n-1})^{1/2}, \quad 1 \le j < i \le n, \tag{2.34}$$

$$\|\underline{b}_i\|^2 \le n^2(4\mathcal{B})^n, \quad 1 \le i \le n, \tag{2.35}$$

während der gesamten Laufzeit des Algorithmus gilt. Damit ist aber die Größenordnung der binären Länge der Zähler und Nenner bestimmt, denn für $1 \le j < i \le n$ gilt

$$\begin{aligned} |\text{ZÄHLER}(\mu_{ij})| &\le |\text{NENNER}(\mu_{ij})|\, 2^{n-1}\,(n\mathcal{B}^{n-1})^{1/2} \\ &\le d_j\, 2^{n-1}(n\mathcal{B})^{1/2}. \end{aligned}$$

Analog kann man für $\|\underline{b}_i^\star\|^2$ argumentieren. Die während der Berechnungen der neuen Zähler und Nenner auftretenden ganzen Zahlen sind von der gleichen Größenordung wie d_i, bzw. $d_j\mu_{ij}$, da nur konstant viele arithmetische Operationen pro Anweisung durchgeführt werden. □

Bemerkungen

Bemerkung 2.19 Eine schwach reduzierte Basis können wir, wenn eine Ausgangsbasis des Gitters bekannt ist, durch mehrfaches Aufrufen der Prozedur REDUCE wie folgt berechnen:

(1) **for** $(i = 2;\ i \le n;\ i = i + 1)$ **do**
(2) **for** $(j = i - 1;\ j \ge 1;\ j = j - 1)$ **do**
(3) REDUCE(μ_{ij});
(4) **od**
(5) **od**

Bezeichnen wir mit $\tilde{\underline{b}}_i$ das neue $\underline{b}_i$ nach der Reduktion, also $\tilde{\underline{b}}_i = \underline{b}_i - \lceil \mu_{ij} \rfloor \cdot \underline{b}_j$, dann kann gelten $\|\tilde{\underline{b}}_i\| > \|\underline{b}_i\|$. Hier nun ein Beispiel:

Beispiel 2.20 Sei

$$\underline{b}_1 = \begin{pmatrix} 30 \\ 0 \\ 0 \end{pmatrix}, \ \underline{b}_2 = \begin{pmatrix} -12 \\ 10 \\ 10 \end{pmatrix} \text{ und } \underline{b}_3 = \begin{pmatrix} 3 \\ 2 \\ 10 \end{pmatrix}$$

eine Basis eines Gitters $L \subseteq \mathbb{Z}^d$ und

$$\underline{b}_1^\star = \begin{pmatrix} 30 \\ 0 \\ 0 \end{pmatrix}, \ \underline{b}_2^\star = \begin{pmatrix} 0 \\ 10 \\ 10 \end{pmatrix} \text{ und } \underline{b}_3^\star = \begin{pmatrix} 0 \\ -4 \\ 4 \end{pmatrix}$$

mit $\mu_{21} = -4/10$, $\mu_{31} = 1/10$ und $\mu_{32} = 6/10$ sei die Gram-Schmidt-Orthogonalisierung von $\underline{b}_1$, $\underline{b}_2$ und $\underline{b}_3$. Da $|\mu_{32}| = 6/10 > 1/2$ erscheint uns eine Reduktion von $\underline{b}_3$ in Richtung von $\underline{b}_2$ sinnvoll, also

$$\tilde{\underline{b}}_3 = \underline{b}_3 - \lceil \mu_{32} \rfloor \cdot \underline{b}_2 = \begin{pmatrix} 3 \\ 2 \\ 10 \end{pmatrix} - \begin{pmatrix} -12 \\ 10 \\ 10 \end{pmatrix} = \begin{pmatrix} 15 \\ -8 \\ 0 \end{pmatrix}.$$

Dann gilt

$$\|\tilde{\underline{b}}_3\|^2 = 289 > 113 = \|\underline{b}_3\|^2.$$

Was ist geschehen? Galt vor der Reduktion von $\underline{b}_3$

$$\begin{aligned} \|\underline{b}_3\|^2 &= \|\underline{b}_3^\star\|^2 + \mu_{31}^2 \|\underline{b}_1^\star\|^2 + \mu_{32}^2 \|\underline{b}_2^\star\|^2 \\ &= \|\underline{b}_3^\star\|^2 + (1/10)^2 \|\underline{b}_1^\star\|^2 + (6/10)^2 \|\underline{b}_2^\star\|^2, \end{aligned}$$

dann haben wir mit $\tilde{\mu}_{31} = \mu_{31} - \lceil \mu_{32} \rfloor \mu_{21}$ und $\tilde{\mu}_{32} = \mu_{32} - \lceil \mu_{32} \rfloor$ nach der Reduktion

$$\begin{aligned} \|\tilde{\underline{b}}_3\|^2 &= \|\underline{b}_3^\star\|^2 + \tilde{\mu}_{31}^2 \|\underline{b}_1^\star\|^2 + \tilde{\mu}_{32} \|\underline{b}_2^\star\|^2 \\ &= \|\underline{b}_3^\star\|^2 + (1/2)^2 \|\underline{b}_1^\star\|^2 + (4/10)^2 \|\underline{b}_2^\star\|^2. \end{aligned}$$

Wie wir sehen, ist nun zwar $|\tilde{\mu}_{32}| \leq 1/2$, aber statt $\mu_{31} = 1/10$ gilt nun $\tilde{\mu}_{31} = 1/2$.

Bemerkung 2.21 Wir können also nicht ausschließen, daß der Defekt der LLL-reduzierten Basis größer ist als der Defekt der Ausgangsbasis.

2.6 Korkine-Zolotarev-reduzierte Basen

Wir interessieren uns für Korkine-Zolotarev-reduzierte Basen, weil sie, verglichen mit Basen, die in einem anderen Sinne reduziert sind, den kleinsten orthogonalen Defekt haben.

Bevor wir darauf in Satz 2.24 eingehen, wo wir auch die Länge der Basisvektoren mit Hilfe der sukzessiven Mininima abschätzen, wollen wir versuchen, die Bedingungen, die eine Korkine-Zolotarev-reduzierte Basis im Sinne der Definition 2.6 erfüllt, zu interpretieren. Sei $\underline{b}_1, \ldots, \underline{b}_n$ Korkine-Zolotarev-reduziert.

Die Bedingung (2.5), nämlich, daß $|\mu_{ij}| \leq 1/2$ für $1 \leq j < i \leq n$ (schwache Reduktion), wurde schon zu Beginn von Abschnitt 2.5 eingehend untersucht. Wir erinnern uns, daß auch LLL-reduzierte Basen diese Eigenschaft haben.

Wir wollen nun versuchen, Bedingung (2.6) zu interpretieren. Formal lautet sie nochmals

$$\|\underline{b}_i^\star\| = \lambda_1(L_i) \text{ für } 1 \leq i \leq n.$$

Folglich ist $\underline{b}_i^\star$ in einer Korkine-Zolotarev-reduzierten Basis ein kürzester Gittervektor des Gitters $L_i = L(\underline{b}_i^\star, \underline{b}_{i+1}(i), \ldots, \underline{b}_n(i))$. Das Gitter L_i erhalten wir, wenn wir L orthogonal auf $\langle\langle \underline{b}_1, \ldots, \underline{b}_{i-1} \rangle\rangle$ projizieren (siehe dazu Kapitel 1). Diese Bedingung ist gleichbedeutend mit der Forderung, daß $\underline{b}_i^\star$ so *kurz wie möglich* sei. Da wegen der schwachen Reduktion für $1 \leq i \leq n$

$$\|\underline{b}_i\|^2 \leq \|\underline{b}_i^\star\|^2 + 1/4 \sum_{j=1}^{i-1} \|\underline{b}_j^\star\|^2,$$

ist $\underline{b}_i$ dann kurz. Zu Definition 2.6 gibt es eine äquivalente Definition, von der Korkine-Zolotarev-reduzierte Basen ihren Namen haben [KoZo1873]:

Satz 2.22 *Eine Basis $\underline{b}_1, \ldots, \underline{b}_n$ eines Gitters $L \subset \mathbb{R}^d$ ist genau dann Korkine-Zolotarev-reduziert, wenn sie folgende Bedingungen erfüllt:*
a) $\|\underline{b}_1\| = \lambda_1(L)$, d.h. $\underline{b}_1$ ist ein kürzester Vektor von L.
b) $|\mu_{i1}| \leq 1/2$ für $2 \leq i \leq n$.
c) Sei L_2 die orthogonale Projektion von L auf $\underline{b}_1$, dann bilden die orthogonalen Projektionen $\underline{b}_i - \mu_{ij} \cdot \underline{b}_1$ von $\underline{b}_2, \ldots, \underline{b}_n$ eine Korkine-Zolotarev-reduzierte Basis von L_2.

Einen Beweis für diese Aussage findet man in [LaLeSch90]. Interessant ist Satz 2.22 deshalb, weil er eine rekursive und deshalb konstruktive Beschreibung einer Korkine-Zolotarev-reduzierten Basis liefert. Der in Abschnitt 2.6.2 vorgestellte Algorithmus ist auf dieser Beschreibung aufgebaut.

2.6.1 Eigenschaften Korkine-Zolotarev-reduzierter Basen

Wir wollen nun das Verhältnis zwischen sukzessiven Minima und den Längen Korkine-Zolotarev-reduzierter Basisvektoren untersuchen. Die folgenden Behauptungen sind [LaLeSch90] entnommen, wo auch eine Fülle von weiteren Aussagen über Korkine-Zolotarev-reduzierte Basen zu finden sind.

Lemma 2.23 *Sei $L \subset \mathbb{R}^d$ ein n-dimensionales Gitter. Es gilt*

$$\lambda_1(L_i) \leq \lambda_i(L) \text{ für } 1 \leq i \leq n. \tag{2.36}$$

Ist $\underline{b}_1, \ldots, \underline{b}_n$ eine Basis von L und Korkine-Zolotarev-reduziert, dann ist

$$\|\underline{b}_i^\star\| \leq \lambda_i(L) \text{ für } 1 \leq i \leq n. \tag{2.37}$$

Beweis: Es gibt i linear unabhängige Vektoren $\underline{v}_j \in L$ mit $\|\underline{v}_j\| = \lambda_j(L)$ für $1 \leq j \leq i$ ([Ca59] VIII.1). Es gilt $\underline{v}_{j_0}(i) \neq \underline{0}$ für wenigstens ein $1 \leq j_0 \leq i$. Wäre dies nicht so, dann müßten die linear unabhängigen Vektoren $\underline{v}_1, \ldots, \underline{v}_i$ in $\langle\langle \underline{b}_1, \ldots, \underline{b}_{i-1} \rangle\rangle$ liegen, was nicht möglich ist. Also ist

$$\lambda_1(L_i) \leq \|\underline{v}_{j_0}(i)\| \leq \|\underline{v}_j\| \leq \lambda_i(L).$$

(2.37) folgt aus (2.36) mit Eigenschaft (2.5) aus Definition 2.6 (Korkine-Zolotarev-reduzierte Basis). □

Der folgende Satz zeigt, daß die Basisvektoren einer Korkine-Zolotarev-reduzierten Basis sehr kurz sind.

Satz 2.24 *Sei $L \subset \mathbb{R}^d$ ein Gitter und $\underline{b}_1, \ldots, \underline{b}_n$ eine Korkine-Zolotarev-reduzierte Basis von L. Dann gelten folgende Aussagen (γ_n ist die Hermitesche Konstante):*

$$\sqrt{4/(i+3)}\,\lambda_i(L) \leq \|\underline{b}_i\| \leq \sqrt{(i+3)/4}\,\lambda_i(L) \text{ für } 1 \leq i \leq n, \tag{2.38}$$

$$\prod_{i=1}^{n} \|\underline{b}_i\| \leq \left((\gamma_n)^n \prod_{i=1}^{n} (i+3)/4\right)^{1/2} \mathrm{d}(L) \leq O(n^n)\,\mathrm{d}(L). \tag{2.39}$$

Beweis: Zu (2.38): Wir schätzen $\|\underline{b}_i\|^2$ mit Hilfe der Gram-Schmidt-Darstellung (1.1) ab, verwenden dabei (2.37) und benutzen, daß eine Korkine-Zolotarev-reduzierte Basis schwach reduziert ist, also daß $|\mu_{ij}| \leq 1/2$ für $1 \leq j < i \leq n$. Somit erhalten wir

$$\begin{aligned}
\|\underline{b}_i\|^2 &\leq \|\underline{b}_i^\star\|^2 + 1/4 \sum_{j=1}^{i-1} \|\underline{b}_j^\star\|^2 \\
&\leq \lambda_i(L)^2 + 1/4 \sum_{j=1}^{i-1} \lambda_j(L)^2 \\
&\leq (i+3)/4\,\lambda_i(L)^2,
\end{aligned}$$

womit die rechte Seite von (2.38) folgt. Da $\underline{b}_i(j) \neq \underline{0}$, haben wir für $j \leq i$

$$\|\underline{b}_j^\star\| = \lambda_1(L_j) \leq \|\underline{b}_i(j)\| \leq \|\underline{b}_i\|.$$

Damit gilt für $j \leq i$

$$\|\underline{b}_j\|^2 \leq \|\underline{b}_j^\star\|^2 + 1/4 \sum_{k=1}^{j-1} \|\underline{b}_k^\star\|^2 \leq (j+3)/4\, \|\underline{b}_i\|^2.$$

Deshalb haben wir

$$\lambda_i(L) \leq \max\{\|\underline{b}_j\| \mid 1 \leq j \leq i\} \leq \sqrt{(i+3)/4}\, \|\underline{b}_i\|,$$

womit auch die linke Seite gezeigt wurde.
Zu (2.39): Die erste Ungleichung folgt direkt mit (2.38) und der Folgerung (1.11) aus dem Satz von Minkowski. Zur zweiten Ungleichung siehe [Sch84]. Es sei hier nur erwähnt, daß nach [LaLeSch90] für genügend großes n gilt

$$(\gamma_n)^n \prod_{i=1}^{n} (i+3)/4 \leq n^{2n}/(4\pi e^2 + o(1))^n. \tag{2.40}$$

□

Bevor wir uns nun dem Algorithmus zuwenden, zeigen wir noch ein Lemma, das eine Aussage darüber macht, wann sich ein $\underline{v} \in L$ zu einer Basis von L erweitern läßt [Ka83].

Lemma 2.25 *Sei $\underline{v} \in L$ mit $\underline{v} \neq \underline{0}$ und $\epsilon \cdot \underline{v} \notin L$ für $0 < \epsilon < 1$. Dann gibt es eine Basis von L, die $\underline{v}$ enthält.*

Beweis: Sei n die Dimension des Gitters. Für $n = 1$, muß das Gitter der Form sein $L = \mathbb{Z} \cdot \underline{b}_1$, also ist das einzige $\underline{v}$, das die Behauptung erfüllt, $\underline{b}_1$.
Sei nun $n \geq 2$ und $\hat{L}$ die senkrechte Projektion von L auf $\underline{v}$, also

$$\hat{L} = \{\underline{b} - \langle \underline{b}, \underline{v}\rangle / \langle \underline{v}, \underline{v}\rangle \cdot \underline{v} \mid \underline{b} \in L\}.$$

$\hat{L}$ hat die Dimension $n-1$. Sei $\hat{\underline{b}}_2, \ldots, \hat{\underline{b}}_n$ eine Basis von $\hat{L}$ und $\underline{b}_2, \ldots, \underline{b}_n$ Vektoren aus L, die, projiziert in $\hat{L}$, $\hat{\underline{b}}_2, \ldots, \hat{\underline{b}}_n$ ergeben. Wir wollen zeigen, daß $\underline{v}, \underline{b}_2, \ldots, \underline{b}_n$ eine Basis von L bildet. Wir führen einen Widerspruchsbeweis. Die Vektoren $\underline{v}, \underline{b}_2, \ldots, \underline{b}_n$ sind in L. Es genügt zu zeigen, daß sich ein beliebiges $\underline{w} \in L$ schreiben läßt als

$$\underline{w} = \sum_{i=1}^{n} \alpha_i \cdot \underline{b}_i$$

mit $\underline{v} = \underline{b}_1, \alpha_i \in \mathbb{Z}$. Sei $\hat{\underline{w}}$ die senkrechte Projektion von $\underline{w}$ auf $\underline{v}$. Der Vektor $\hat{\underline{w}}$ ist in $\hat{L}$, also ist

$$\hat{\underline{w}} = \sum_{i=2}^{n} \alpha_i \cdot \hat{\underline{b}}_i.$$

Betrachte nun

$$\underline{w}' = \underline{w} - \sum_{i=2}^{n} \alpha_i \cdot \underline{b}_i.$$

Der Vektor $\underline{w}'$ muß ein Vielfaches von $\underline{v}$ sein, denn $\underline{w} \in \langle\langle \underline{v}, \underline{b}_2, \ldots, \underline{b}_n \rangle\rangle$. Sei also $\underline{w}' = \delta \underline{v}$. Wir halten fest, daß $\underline{w}'$ in L ist, denn $\underline{w}, \underline{b}_2, \ldots, \underline{b}_n$ sind in L. Wir nehmen an, δ sei nun nicht ganzzahlig. Da $\underline{w}'$ und $\underline{v}$ in L sind, wäre dann $\underline{w}' - \lceil \delta \rfloor \cdot \underline{v} = (\delta - \lceil \delta \rfloor) \cdot \underline{v}$ in L. Da aber $0 < \delta - \lceil \delta \rfloor < 1$, ist dies ein Widerspruch zu Voraussetzung. $\square$

Bemerkung 2.26 Ist $\underline{v}$ ein kürzester Gittervektor in L, also $\|\underline{v}\| = \lambda_1(L)$, dann gilt $\epsilon \cdot \underline{v} \notin L$ für $0 < \epsilon < 1$.

2.6.2 Der Algorithmus

Beschreibung

In diesem Abschnitt wollen wir einen Algorithmus beschreiben, der eine Korkine-Zolotarev-reduzierte Basis eines n-dimensionalen Gitters $L \subseteq \mathbb{Z}^d$ berechnet, wenn L durch eine Basis gegeben ist. Die Grundidee des Algorithmus stammt von Kannan ([Ka83], siehe auch [Ka87a]). Nach dem Korrektheitsbeweis werden wir zeigen, daß die Laufzeit des Algorithmus polynomiell von der maximalen Länge eines Vektors der gegebenen Basis und exponentiell von der Dimension des Gitters abhängt.

Der Algorithmus arbeitet rekursiv. Sei $n = 1$. Ist $\underline{b}_1$ eine Basis eines eindimensionalen Gitters, dann gibt der Algorithmus $\underline{b}_1$ zurück, denn $\underline{b}_1$ ist schon Korkine-Zolotarev-reduziert.

Sei $n \geq 2$ und nehmen wir an, wir könnten Korkine-Zolotarev-reduzierte Basen $n-1$-dimensionaler Gitter berechnen. Alle nun folgenden Operationen dienen dazu, die Suche nach einem kürzesten Gittervektor in L zu beschleunigen. Ganz zu Beginn wird die Eingabebasis mit dem LLL-Algorithmus reduziert. Damit soll erreicht werden, daß die noch folgenden Operationen auf (in der Regel) kurzen Basisvektoren ausgeführt werden.

Die nun folgenden Schritte werden in Abhängigkeit von einer bestimmten Bedingung, deren Bedeutung erst weiter unten erklärt wird, wiederholt ausgeführt. Es werden die senkrechten Projektionen von $\underline{b}_2, \ldots, \underline{b}_n$ auf $\underline{b}_1$ berechnet. Sie bilden eine Basis des $n-1$-dimensionalen Gitters L_2. Durch rekursiven Aufruf des Algorithmus mit eben diesen Projektionen als Eingabebasis wird eine Korkine-Zolotarev-reduzierte Basis von L_2 berechnet, die anschließend von L_2 nach L geliftet wird.

Gilt für die ersten beiden Basisvektoren der neuen Basis die oben angesprochene Bedingung

$$\|\underline{b}_2\|^2 \leq 3/4\,\|\underline{b}_1\|^2,$$

dann werden die eben beschriebenen Operationen erneut ausgeführt, nachdem aber vorher $\underline{b}_1$ und $\underline{b}_2$ der neuen Basis mittels Gaußalgorithmus reduziert wurden.

Bemerkung 2.27 Nur durch diese Reduktion von $\underline{b}_1$ und $\underline{b}_2$ wird erreicht, daß nach *logarithmisch* (in der Dimension des Gitters) vielen Aufrufen $\|\underline{b}_2\|^2 > 3/4\,\|\underline{b}_1\|^2$ ist. Durch diesen Trick wurde die Laufzeit des ursprünglichen Algorithmus von Kannan deutlich verbessert (siehe dazu [He85] und [Sch87]). Die Vektoren $\underline{b}_1$ und $\underline{b}_2$ wurden von Kannan lediglich vertauscht und deshalb der rekursive Aufruf und das anschließende Vertauschen *polynomiell* (in der Dimension des Gitters) oft ausgeführt.

Anschließend reduzieren wir $\underline{b}_3, \ldots, \underline{b}_n$ mit $\underline{b}_1, \underline{b}_2$, um erneut eine schwach reduzierte Basis zu erhalten. Nach dem Abbruch der Schleife hat die vorliegende Basis folgende Eigenschaften:

$$\|\underline{b}_i^\star\| = \lambda_1(L_i) \text{ für } 2 \leq i \leq n, \tag{2.41}$$

$$|\mu_{ij}| \leq 1/2 \text{ für } 1 \leq j < i \leq n, \tag{2.42}$$

$$\|\underline{b}_1\|^2 < 4/3\,\|\underline{b}_2\|^2. \tag{2.43}$$

Wie man sieht, wurde noch keine Korkine-Zolotarev-reduzierte Basis von L berechnet, denn $\underline{b}_1$ muß noch kein kürzester Vektor von L sein! Von nun an ist das Ziel, einen Vektor $\underline{v}$ mit $\|\underline{v}\| = \lambda_1(L)$ zu berechnen und diesen zu einer Basis von L zu erweitern, was in diesem Fall nach Lemma 2.25 möglich ist. Deshalb wird eine Menge $S \subset L$ berechnet, die ein $\underline{v} \in L$ mit $\|\underline{v}\| = \lambda_1(L)$ enthält.

Hier spielt Eigenschaft (2.43), die einen mehrmaligen Aufruf von KORKZOLO gekostet hat, eine entscheidende Rolle. Denn nur mit (2.41) und (2.43) kann gezeigt werden, daß $|S| \leq (18n)^{n/2}$ ist, d.h. die Größe des Suchraumes nur von der Dimension n abhängt.

Alle bisher durchgeführten Schritte hatten nur das Ziel, S klein zu halten. Trotzdem wird die Laufzeit des Algorithmus immer noch von der Suche nach einem kürzesten Gittervektor dominiert! Die Mächtigkeit von S kann dabei deutlich unter dieser Zahl liegen, da es meist genügt, einen kürzesten Gittervektor nur in einem Gitter zu suchen, das von weniger als n Basisvektoren aufgespannt wird.

Mit einem geringfügig geänderten LLL-Algorithmus läßt sich ein kürzester Gittervektor zu einer Basis von L erweitern [Sch87], was wir in Schritt 12 mit $\underline{v}$ auch tun. Abschließend werden noch einmal alle Schritte aus der zu Beginn beschriebenen Schleife ausgeführt, und, da $\underline{b}_1$ jetzt ein kürzester Gittervektor von L ist, damit eine Korkine-Zolotarev-reduzierte Basis von L berechnet.

Algorithmus 2.28 KORKZOLO$(n; \underline{b}_1, \ldots, \underline{b}_n)$

EINGABE: Dimension n und beliebige Basis $\underline{b}_1, \ldots, \underline{b}_n$ des Gitters L.
AUSGABE: Korkine-Zolotarev-reduzierte Basis $\underline{b}_1, \ldots, \underline{b}_n$ des Gitters L.

(1) **if** $(n = 1)$ **then return** $\underline{b}_1$;
(2) Berechne die LLL-Reduktion von $\underline{b}_1, \ldots, \underline{b}_n$.

(3) **for** $(i = 2, \ldots, n)$ **do**
    $\underline{b}_i(2) := \underline{b}_i - \langle \underline{b}_i, \underline{b}_1 \rangle / \langle \underline{b}_1, \underline{b}_1 \rangle \cdot \underline{b}_1$;
    /* $\underline{b}_i(2)$ ist die senkrechte Projektion von $\underline{b}_i$ auf $\underline{b}_1$. */
  od
(4) $\underline{b}'_2, \ldots, \underline{b}'_n :=$ KORKZOLO$(n-1; \underline{b}_2(2), \ldots, \underline{b}_n(2))$;

  /* Die folgenden drei Schritte liften die $\underline{b}'_i$ nach L. */
(5) Berechne unimodulare Matrix $T \in \mathbb{Z}^{n-1 \times n-1}$ mit
  $(\underline{b}'_2, \ldots, \underline{b}'_n) = (\underline{b}_2(2), \ldots, \underline{b}_n(2)) \cdot T$.
(6) Transformiere die Ausgangsbasis $(\underline{b}_2, \ldots, \underline{b}_n) := (\underline{b}_2, \ldots, \underline{b}_n) \cdot T$.
(7) Stelle sicher, daß $|\mu_{i1}| \leq 1/2$ für $2 \leq i \leq n$.
  /* Dazu verwenden wir REDUCE aus Abschnitt 2.5 */

(8) **if** $(\|\underline{b}_2\|^2 < 3/4\, \|\underline{b}_1\|^2)$ **then**
    Reduziere $\underline{b}_1$ und $\underline{b}_2$ mit dem Gaußalgorithmus und stelle sicher (mit REDUCE), daß $|\mu_{ij}| \leq 1/2$ auch für $3 \leq i \leq n$, $j = 1, 2$ ist.
    /* Der Gaußalgorithmus berechnet eine Korkine-Zolotarev-reduzierte Basis des zweidimensionalen Gitters $L(\underline{b}_1, \underline{b}_2)$. */
    goto (3)
  fi

(9) Sei j_0 das kleinste $j > 1$ mit $\|\underline{b}_j(j)\|^2 \geq \|\underline{b}_1\|^2$, gibt es kein solches j, dann sei $j_0 = n + 1$.
(10) Berechne durch Aufruf der Prozedur ENUMERATE(j_0) mit der Basis $\underline{b}_1, \ldots, \underline{b}_{j_0 - 1}$ und $r^2 = \|\underline{b}_1\|^2$ die Menge $S = \{\underline{b} \in L \mid \|\underline{b}\|^2 \leq r^2\}$.
(11) Nimm $\underline{v} \in S$ mit $\|\underline{v}\|^2 = \min\{\|\underline{b}\|^2 \mid \underline{b} \in S\}$.
(12) Sei $\underline{a}_1 := \underline{v}$, $\underline{a}_i := \underline{b}_{i-1}$, $2 \leq i \leq n+1$. Ist $\|\underline{a}_1\| \neq \|\underline{a}_2\|$, dann rufe den LLL-Algorithmus mit dem linear abhängigen Erzeugendensystem $\underline{a}_1, \ldots, \underline{a}_n, \underline{a}_{n+1}$ auf. Sobald $\underline{a}_1 = \underline{0}$ ist, nehme $\underline{b}_1 := \underline{a}_2, \ldots, \underline{b}_n := \underline{a}_{n+1}$ als neue Basis für L. War während der Ausführung des LLL-Algorithmus erstmals $\underline{a}^\star_\nu = \underline{0}$, dann stelle sicher, daß auch $|\mu_{ij}| \leq 1/2$ für $\nu < i \leq n$ und $1 \leq j < \nu$.

(13) Führe Schritte (3) bis (8) durch.
(14) **return** $\underline{b}_1, \ldots, \underline{b}_n$;

Korrektheitsbeweis

Nehmen wir an, daß nach der Ausführung von Schritt 4 $\underline{b}'_2, \ldots, \underline{b}'_n$ eine Korkine-Zolotarev-reduzierte Basis von L_2 ist. Wir wollen zunächst zeigen, daß durch die Anweisungen in den Schritten 5 bis 7 die Vektoren $\underline{b}'_2, \ldots, \underline{b}'_n$ von L_2 nach L im Sinne von Kapitel 1 geliftet werden. Dazu beweisen wir folgende Hilfsaussage:

Lemma 2.29 *Sei $B = (\underline{b}_1, \ldots, \underline{b}_n)$ eine Basis eines Gitters $L \subset \mathbb{R}^d$, $\underline{b}_2(2), \ldots, \underline{b}_n(2)$ die Projektion von $\underline{b}_2, \ldots, \underline{b}_n$ auf $\underline{b}_1$ und $T = (\underline{t}_1, \ldots, \underline{t}_{n-1}) = (t_{ij}) \in \mathbb{Z}^{(n-1)\times(n-1)}$ unimodular. Es bezeichne*

$$(\overline{\underline{b}}_2, \ldots, \overline{\underline{b}}_n) = (\underline{b}_2, \ldots, \underline{b}_n) \cdot T.$$

Dann ist $\underline{b}_1, \overline{\underline{b}}_2, \ldots, \overline{\underline{b}}_n$ eine Basis von L und für $2 \leq j \leq n$ gilt

$$\overline{\underline{b}}_j(2) = \overline{\underline{b}}_j - \langle \overline{\underline{b}}_j, \underline{b}_1 \rangle / \langle \underline{b}_1, \underline{b}_1 \rangle \cdot \underline{b}_1 = \sum_{i=2}^{n} t_{i-1j-1} \cdot \underline{b}_i(2).$$

Beweis: Da $\begin{pmatrix} 1 & 0 \ldots 0 \\ 0 & \\ \vdots & T \\ 0 & \end{pmatrix}$ unimodular, ist $\underline{b}_1, \overline{\underline{b}}_2, \ldots, \overline{\underline{b}}_n$ Basis von L.
Für $2 \leq j \leq n$ ist

$$\begin{aligned}
\overline{\underline{b}}_j(2) &= B\underline{t}_{j-1} - \langle B\underline{t}_{j-1}, \underline{b}_1 \rangle / \|\underline{b}_1\|^2 \cdot \underline{b}_1 \\
&= \sum_{i=2}^{n} t_{i-1j-1} \cdot \underline{b}_i - \sum_{i=2}^{n} t_{i-1j-1} \langle \underline{b}_i, \underline{b}_1 \rangle / \|\underline{b}_1\|^2 \cdot \underline{b}_1 \\
&= \sum_{i=2}^{n} t_{i-1j-1} \cdot (\underline{b}_i - \langle \underline{b}_i, \underline{b}_1 \rangle / \|\underline{b}_1\|^2 \cdot \underline{b}_1) \\
&= \sum_{i=2}^{n} t_{i-1j-1} \cdot \underline{b}_i(2).
\end{aligned}$$

□

Proposition 2.30 *Ist $\underline{b}'_2, \ldots, \underline{b}'_n$ nach dem rekursiven Aufruf in Schritt 4 eine Korkine-Zolotarev-reduzierte Basis des Gitters L_2, dann erfüllt $\underline{b}_1, \ldots, \underline{b}_n$ nach der Ausführung der Schritte 5-7 Eigenschaften (2.41) und (2.42).*

Beweis: Mit Lemma 2.29 gilt, daß nach Schritt 6 $\underline{b}'_2, \ldots, \underline{b}'_n$ die senkrechten Projektionen von $\underline{b}_2, \ldots, \underline{b}_n$ auf $\underline{b}_1$ sind. Nach Schritt 7 gilt für die neuen $\underline{b}_2, \ldots, \underline{b}_n$ zusätzlich $|\mu_{i1}| \leq 1/2$ für $2 \leq i \leq n$, d.h. $\underline{b}'_2, \ldots, \underline{b}'_n$ wurden nach L im Sinne von Kapitel 1 geliftet. Da $\underline{b}_1, \ldots, \underline{b}_n$ Bedingungen b) und c) in Satz 2.22 erfüllen, gelten mit der Definition 2.6 auch die Eigenschaften (2.41) und (2.42). □

Proposition 2.31 *In Schritt 4 werde eine Korkine-Zolotarev-reduzierte Basis von L_2 berechnet. Seien $\underline{\tilde{b}}_1$ und $\underline{\tilde{b}}_2$ die Bezeichnungen der Vektoren $\underline{b}_1$ und $\underline{b}_2$ vor der Ausführung von Schritt 3. Nach den Berechnungen in den Schritten 3 bis 8 gilt für das neue $\underline{b}_1$ die Ungleichung*

$$\|\underline{b}_1\|^2/\lambda_1(L)^2 \leq (4/3)^{1/2}\,\|\underline{\tilde{b}}_1\|/\lambda_1(L). \tag{2.44}$$

Der Sprung in 8 nach 3 wird maximal $\lceil\log(n-1)\rceil + 5$-mal ausgeführt.

Beweis: Der Beweis orientiert sich an [Sch87]. Ist $\|\underline{\tilde{b}}_1\| > \lambda_1(L)$, dann gilt nach Ausführung der Schritte 3 bis 7 für $\underline{b}_2^\star$ des neuen Vektors $\underline{b}_2$ die Ungleichung

$$\|\underline{b}_2^\star\| = \lambda_1(L_2) \leq \lambda_1(L),$$

denn für $\underline{v}$ mit $\|\underline{v}\| = \lambda_1(L)$ gilt $\lambda_1(L_2) \leq \|\underline{v} - \langle\underline{v},\underline{\tilde{b}}_1\rangle/\langle\underline{\tilde{b}}_1,\underline{\tilde{b}}_1\rangle\,\underline{\tilde{b}}_1\| \leq \|\underline{v}\| = \lambda_1(L)$. Wir wenden den Gaußalgorithmus in Schritt 8 auf die Basis $\underline{\tilde{b}}_1, \underline{b}_2$ des Gitters $L' := \mathbb{Z}\cdot\underline{\tilde{b}}_1 + \mathbb{Z}\cdot\underline{b}_2$ an. Da das neue $\underline{b}_1$ der kürzeste Vektor von L' ist, gilt

$$\|\underline{b}_1\|^2 = \lambda_1(L')^2 \leq \gamma_2\,\mathrm{d}(L') = 2/\sqrt{3}\,\|\underline{\tilde{b}}_1\|\,\|\underline{b}_2^\star\|. \tag{2.45}$$

Mit $\|\underline{b}_2^\star\| \leq \lambda_1(L)$ folgt (2.44). Bevor wir zum ersten Mal zu Schritt 3 kommen, wird eine LLL-reduzierte Basis von L berechnet. Es gilt also nach Schritt 2

$$\|\underline{b}_1\|/\lambda_1(L) \leq 2^{(n-1)/2}.$$

Gehen wir nun m-mal von Schritt 3 nach Schritt 8, dann gilt

$$\|\underline{b}_1\|/\lambda_1(L) \leq 2^{(n-1)2^{-m-1}}\,(4/3)^{(1+2^{-1}+\ldots+2^{-m})/4},$$

also mit $m = \lceil\log(n-1)\rceil - 1$ haben wir

$$\|\underline{b}_1\|/\lambda_1(L) \leq 2\,(4/3)^{1/2}.$$

Wir wollen nun zeigen, daß die Schritte 3 bis 8, dann noch maximal sechs Mal durchlaufen werden. Es gilt folgende Ungleichung für den neuen Vektor $\underline{b}_1$ nach Schritt 8

$$\|\underline{b}_1\|^2 = \|\underline{b}_2^\star\|^2 + \mu_{21}^2\,\|\underline{\tilde{b}}_1\|^2 \leq \|\underline{\tilde{b}}_2^\star\|^2 + \mu_{21}^2\,\|\underline{\tilde{b}}_1\|^2 \leq (1/2+1/4)\,\|\underline{\tilde{b}}_1\|^2.$$

Gegenüber dem alten $\underline{\tilde{b}}_1$ ist also das neue $\underline{b}_1$ um einen Faktor $(3/4)^{1/2}$ kürzer. Für $j = 6$ gilt erstmals

$$(3/4)^{1/2\,j}\,2\,(4/3)^{1/2} \leq 1$$

und da $\|\underline{b}_1\| \geq \lambda_1(L)$, werden also die Schritte 3 bis 8 insgesamt maximal $\lceil\log(n-1)\rceil + 5$ Mal durchlaufen. □

Bemerkung 2.32 Der LLL-Algorithmus, den wir in Schritt 2 und Schritt 12 aufrufen, arbeitet mit der Konstante 3/4 (siehe Abschnitt 2.5). Schnorr schlägt vor, statt dessen 100/101 zu verwenden, da sich dadurch ein Laufzeitgewinn ergibt. Da die Größenordnung der Laufzeit bei beiden Konstanten gleich ist und eine einheitliche Darstellung des Themas im Vordergrund stand, wurde auf die Wahl von 100/101 verzichtet.

Wir wollen nun zeigen, daß es in Schritt 10 genügt, einen kürzesten Vektor in $L(\underline{b}_1, \ldots, \underline{b}_{j_0-1})$ zu suchen.

Lemma 2.33 *Wählen wir j_0 wie in Schritt 9 beschrieben, dann ist ein kürzester Vektor von $L(\underline{b}_1, \ldots, \underline{b}_{j_0-1})$ auch ein kürzester Vektor von $L(\underline{b}_1, \ldots, \underline{b}_n)$.*

Beweis: Sei $\underline{v} = \sum_{i=1}^n \alpha_i \cdot \underline{b}_i \in L(\underline{b}_1, \ldots, \underline{b}_n)$ mit $\|\underline{v}\| = \lambda_1(L)$. Ist $\alpha_{j_0} = \alpha_{j_0+1} = \ldots = \alpha_n = 0$, dann ist die Aussage trivial. Sei also mindestens ein $\alpha_i \neq 0$, $j_0 \leq i \leq n$. Daraus folgt, daß die Projektion $\underline{v}'$ von $\underline{v}$ in das orthogonale Komplement von $\langle\langle \underline{b}_1, \ldots, \underline{b}_{j_0-1} \rangle\rangle$, nämlich

$$\underline{v}' = \underline{v} - \sum_{i=1}^{j_0-1} \langle \underline{v}, \underline{b}_i^\star \rangle / \langle \underline{b}_i^\star, \underline{b}_i^\star \rangle \cdot \underline{b}_i^\star$$

ungleich Null ist. Mit (2.41) gilt

$$\|\underline{v}'\| \geq \lambda_1(L_{j_0}(\underline{b}_1, \ldots, \underline{b}_n)) = \|\underline{b}_{j_0}^\star\|.$$

Dann gilt wegen der speziellen Wahl von j_0

$$\|\underline{v}\| \geq \|\underline{v}'\| \geq \|\underline{b}_{j_0}^\star\| \geq \|\underline{b}_1\|, \text{ also } \|\underline{v}\| = \|\underline{b}_1\|,$$

womit $\underline{b}_1$ gerade schon ein kürzester Vektor von L. □

Die Gram-Schmidt-Orthogonalisierung sei analog zu Kapitel 1 für linear abhängige Vektoren definiert. Sei $\underline{a}_1 = \underline{v}$ und $\underline{a}_i = \underline{b}_{i-1}$ für $2 \leq i \leq n+1$. Wir wollen nun zeigen, daß sich $\underline{a}_1$ mit dem LLL-Algorithmus zu einer Basis von L erweitern läßt. Wir denken uns an den Stellen im Programm wo $\|\underline{b}_i^\star\|^2$ steht, die Variable B_i aus der Prozedur SWAP. In Schritt 1 werden die B_i mit $\|\underline{a}_i^\star\|^2$ für $1 \leq i \leq n+1$ initialisiert.

Lemma 2.34 *Sei $\underline{a}_1 \in L$ mit $\|\underline{a}_1\| = \lambda_1(L)$ und $\underline{a}_2, \ldots, \underline{a}_{n+1}$ eine Basis von L. Rufen wir den LLL-Algorithmus mit $\underline{a}_1, \ldots, \underline{a}_n, \underline{a}_{n+1}$ auf und stoppen wir den LLL-Algorithmus, sobald $\underline{a}_1 = \underline{0}$ ist, dann gelten folgende Aussagen:*

1. *Der Vektor $\underline{a}_1$ wird genau dann mit $\underline{a}_2$ vertauscht, wenn $\underline{a}_2^\star = \underline{0}$ ist.*
2. *Sobald in der Prozedur SWAP des LLL-Algorithmus die Variable $B_k = 0$ für ein k mit $2 \leq k \leq n+1$ wird, nimmt die Laufvariable, die die while-Schleife steuert, nicht mehr den Wert $k+1$ an.*
3. *Nach endlich vielen Schritten sind $\underline{a}_1, \underline{a}_2$ linear abhängig, und für die Laufvariable gilt $k = 2$.*

Beweis: Im folgenden handelt es sich stets um die Schrittnumerierung des LLL-Algorithmus.
zu 1:
$\Leftarrow$: Ist $\underline{a}_2^\star = \underline{0}$, dann gilt nach Schritt 4

$$(\langle \underline{a}_1, \underline{a}_2\rangle / \|\underline{a}_1\|)^2 \|\underline{a}_1\|^2 \leq 1/4 \|\underline{a}_1\|^2 < 3/4 \|\underline{a}_1\|^2$$

und es wird vertauscht.
$\Rightarrow$: Sei $\underline{a}_2^\star \neq \underline{0}$, also $\underline{a}_2 \in L$ und $\underline{a}_2 \notin \mathbb{Z} \cdot \underline{a}_1$. Dann ist $\|\underline{a}_2\|^2 = \|\underline{a}_2\|^2 - 2\mu_{21} \langle \underline{a}_2, \underline{a}_1\rangle + 2\mu_{21}^2 \|\underline{a}_1\|^2 = \|\underline{a}_2 - \mu_{21} \cdot \underline{a}_1\|^2 + \mu_{21}^2 \|\underline{a}_1\|^2 = \|\underline{a}_2^\star\|^2 + \mu_{21}^2 \|\underline{a}_1\|^2 = \|\underline{a}_2^\star + \mu_{21} \cdot \underline{a}_1\|^2$, damit $\|\underline{a}_2\|^2 \geq 3/4 \|\underline{a}_1\|^2$ genau dann, wenn $\|\underline{a}_2^\star + \mu_{21} \cdot \underline{a}_1^\star\|^2 \geq 3/4\, \|\underline{a}_1^\star\|^2$ womit wegen Schritt 5 nicht vertauscht wird.

zu 2: Wir zeigen die Aussage durch Induktion über die Laufvariable k. Ist $k = 2$, und $B_2 = 0$. Dann sind $\underline{a}_1, \underline{a}_2$ linear abhängig und der Algorithmus stoppt, da nach der Vertauschung $\underline{a}_1 = \underline{0}$ ist. Sei $k > 2$ und $B_k = 0$. Dann kann die Laufvariable nur dann den Wert $k+1$ annehmen, wenn $B_{k-1} = 0$ und sie aktuell den Wert k hat. Den Wert k kann sie aber nach Induktionsvoraussetzung nicht annehmen, womit die Induktionsbehauptung folgt.

zu 3: Wir führen den Beweis induktiv. Sei $k = k_0$ und k_0 derart, daß $k_0 \leq n+1$ mit $\underline{a}_1 = \sum_{i=2}^{k_0} \alpha_i \cdot \underline{a}_i$, $\alpha_i \in \mathbb{Z}$, $\alpha_{k_0} \neq 0$, *bevor* wir die zweite Anweisung im LLL-Algorithmus ausführen. Dann ist $B_{k_0} = 0$ und $\mu_{k_0+1\,k_0} = 0$, wenn $k_0 < n+1$, und $\underline{a}_1, \ldots, \underline{a}_{k_0}$ sind linear abhängig. Da $\underline{a}_1, \ldots, \underline{a}_{k-1}$ linear unabhängig sind, wird nach endlich vielen Schritten $k = k_0$ und wir können Schritt 4 ausführen.

Sei $k < k_0$. Seien $\underline{a}_1, \ldots, \underline{a}_k$ linear abhängig, also $\mu_{k+1\,k} = 0$ und $B_k = 0$; die Laufvariable habe den Wert k und wir haben Schritt 4 ausgeführt. Nach Aussage 2 wird k dekrementiert. Solange $\mu_{k\,k-1} \neq 0$ wird $D = \|\underline{a}_1^\star\|^{2k-1} \cdot \ldots \cdot \|\underline{a}_{k-1}^\star\|^2$ (siehe auch (2.29)) bei jeder Reduktion um 1/4 bzw. 3/4 kleiner. Da D nach unten beschränkt ist, muß nach der Ausführung von Schritt 3 und während die Laufvariable den Wert $k-1$ hat, nach endlich vielen Schritten $\mu_{k\,k-1} = 0$ sein. Dann sind aber schon $\underline{a}_1, \ldots, \underline{a}_{k-1}$ linear abhängig, denn $\underline{a}_k$ ist nicht mehr linear abhängig von $\underline{a}_1, \ldots, \underline{a}_{k-1}$. $\square$

Lemma 2.35 *Es gelten die Voraussetzungen aus Lemma 2.34. Nach endlich vielen Schritten ist* $\underline{a}_1 = \underline{0}$ *und die zu diesem Zeitpunkt vorliegenden Vektoren* $\underline{a}_2, \ldots, \underline{a}_{n+1}$ *bilden eine Basis von* L *mit* $\|\underline{a}_2\| = \lambda_1(L)$.

Beweis: Da nur ganzzahlige Vielfache erzeugender Vektoren von anderen erzeugenden Vektoren abgezogen werden, spannen während der Laufzeit des Algorithmus zu jedem Zeitpunkt $\underline{a}_1, \ldots, \underline{a}_n, \underline{a}_{n+1}$ das Gitter L auf. B_2 wird wegen Lemma 2.34 3. nach endlich vielen Schritten 0 und $\underline{a}_1$ dann wegen Lemma 2.34 1. mit $\underline{a}_2$ vertauscht. Da $\underline{a}_1$ wegen Lemma 2.34 1. vorher nie mit $\underline{a}_2$ vertauscht wurde, folgt die Behauptung. $\square$

Der Beweis zum folgenden Lemma zeigt uns, wie man einen kürzesten Gittervektor in einem Gitter durch Aufzählen aller Gitterpunkte in einer Kugel mit dem Radius $r \geq \lambda_1(L)$ berechnen kann.

Lemma 2.36 *Sei ein Gitter $L \subseteq \mathbb{Z}^d$ durch eine Basis $\underline{b}_1, \ldots, \underline{b}_n$ gegeben. Sei $r \geq \lambda_1(L)$, $r^2 \in \mathbb{N}$. Es gibt eine leicht berechenbare Menge $T \subset \mathbb{Z}^n$ mit $|T| \leq \prod_{j=1}^n \left(\lfloor 2r/\|\underline{b}_j^\star\| \rfloor + 1\right)$, so daß für beliebiges $\underline{v} = \sum_{i=1}^n \alpha_i \cdot \underline{b}_i \in L$ mit $\|\underline{v}\| \leq r$, $(\alpha_1, \ldots, \alpha_n) \in T$ ist.*

Beweis: Sei $\underline{v} = \sum_{j=1}^n \alpha_j \cdot \underline{b}_j$. Dann können wir auch schreiben

$$\underline{v} = \sum_{j=1}^n \beta_j \cdot \underline{b}_j^\star \text{ mit } \beta_j = \sum_{k=j}^n \alpha_k \mu_{kj} \text{ für } 1 \leq j \leq n,$$

denn

$$\beta_j = \langle \underline{v}, \underline{b}_j^\star \rangle / \langle \underline{b}_j^\star, \underline{b}_j^\star \rangle = \sum_{k=1}^n \alpha_k \langle \underline{b}_k, \underline{b}_j^\star \rangle / \langle \underline{b}_j^\star, \underline{b}_j^\star \rangle = \sum_{k=j}^n \alpha_k \mu_{kj}. \tag{2.46}$$

Für alle Gittervektoren $\underline{v} = \sum_{j=1}^n \alpha_j \cdot \underline{b}_j$, deren Länge r ist oder die kürzer sind als r, ist somit

$$\|\underline{v}\|^2 = \sum_{j=1}^n \left(\sum_{k=j}^n \alpha_k \mu_{kj} \right)^2 \|\underline{b}_j^\star\|^2 \leq r^2. \tag{2.47}$$

Daraus läßt sich folgern, daß alle $\alpha_n, \ldots, \alpha_1 \in \mathbb{Z}$ für die (2.47) gilt, den Ungleichungen

$$\left(\sum_{k=j}^n \alpha_k \mu_{kj} \right)^2 \cdot \|\underline{b}_j^\star\|^2 \leq r^2 - \sum_{\nu=j+1}^n \left(\sum_{k=\nu}^n \alpha_k \mu_{k\nu} \right)^2 \cdot \|\underline{b}_\nu^\star\|^2 \text{ für } j = n, \ldots, 1 \tag{2.48}$$

genügen. Erweitert man die Bandbreite, indem man sukzessive alle $\alpha_n, \ldots, \alpha_1 \in \mathbb{Z}$ berechnet für die gilt

$$\left(\sum_{k=j}^n \alpha_k \mu_{kj} \right)^2 \leq r^2 / \|\underline{b}_j^\star\|^2 \text{ für } j = n, \ldots, 1, \tag{2.49}$$

dann sind darunter mit (2.47) und (2.48) auch alle $\alpha_1, \ldots, \alpha_n \in \mathbb{Z}$, für die $\|\sum_{i=1}^n \alpha_i \cdot \underline{b}_i\| \leq r$ ist. Die Anzahl an möglichen Werten, die α_j mit (2.49) annehmen kann, wenn $\alpha_{j+1}, \ldots, \alpha_n$ fest gegeben sind, ist $\lfloor 2r/\|\underline{b}_j^\star\| \rfloor + 1$. □

Obwohl die Anzahl an Kandidaten für α_j über (2.49) größer ist als über (2.48) wird im folgenden Algorithmus α_j über (2.49) ermittelt. Der Aufzählschritt ist dann besonders einfach. Allerdings sollte bei einer Implementierung (2.48) verwendet werden.

Algorithmus 2.37 ENUMERATE(j)

EINGABE: Hilfsparameter $j = n$, Basis $\underline{b}_1, \ldots, \underline{b}_n$ eines Gittes L, Gram-Schmidt-Zerlegung $\underline{b}_1^\star, \ldots, \underline{b}_n^\star$ und die dazugehörigen μ_{ij} für $1 \leq j < i \leq n$, Radius $r \geq \lambda_1(L)$, Hilfsvektor $\underline{\alpha} = (\alpha_1, \ldots, \alpha_n) = \underline{0}$, $S = \{\}$.

AUSGABE: $S = \{\underline{b} \in L \mid \|\underline{b}\| \leq r\}$.

$t_j = \sum_{i=j+1}^{n} \alpha_i \mu_{ij}$;
if $(j > 1)$ **then**
    for $(-t_j - r/\|\underline{b}_j^\star\| \leq \alpha_j \leq -t_j + r/\|\underline{b}_j^\star\|)$ **do**
        ENUMERATE($j-1$);
    od
else
    for $(-t_1 - r/\|\underline{b}_1^\star\| \leq \alpha_1 \leq -t_1 + r/\|\underline{b}_1^\star\|)$ **do**
        $\underline{b} := \sum_{i=1}^{n} \alpha_i \cdot \underline{b}_i$;
        if $(\|\underline{b}\|^2 \leq r^2)$ **then**
            $S := S \cup \{\underline{b}\}$;
        fi
    od
fi

Satz 2.38 *Algorithmus 2.28 terminiert, und die zurückgegebenen Vektoren $\underline{b}_1$, …, $\underline{b}_n$ bilden eine Korkine-Zolotarev-reduzierte Basis von L.*

Beweis: Die Beweisführung ist induktiv. Wird KORKZOLO mit $n = 1$ und $\underline{b}_1$ als einzigem erzeugenden Basisvektor aufgerufen, dann liefert KORKZOLO das korrekte Ergebnis, denn $\underline{b}_1$ ist bereits Korkine-Zolotarev-reduziert.

Sei $n \geq 2$. Schritt 2 berechnet die LLL-Reduktion der Eingabebasis. Diese ist trivialerweise eine Basis von L. Aus Proposition 2.30 und der Induktionsvoraussetzung folgt, daß nach der Ausführung von Schritt 7 $\underline{b}_1$, …, $\underline{b}_n$ eine Basis von L bildet, für die (2.41) und (2.42), also

$$\|\underline{b}_i^\star\| = \lambda_1(L_i) \text{ für } 2 \leq i \leq n,$$

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

gilt. Daß der Sprung von 8 nach 3 nur endlich oft ausgeführt wird, wurde in Proposition 2.31 bewiesen. Folglich gilt ab Schritt 9 auch (2.43). Es folgt mit Lemma 2.33, daß es genügt, einen kürzesten Gittervektor von L in $L(\underline{b}_1, \ldots, \underline{b}_{j_0-1})$ zu suchen. Die Prozedur ENUMERATE berechnet auf Grund von Lemma 2.36 eine Menge S, die die kürzesten Gittervektoren enthält, denn es werden alle Gittervektoren $\underline{b}$ berechnet, mit $\|\underline{b}\| \leq r = \|\underline{b}_1\|$. In Schritt 11 wird ein kürzester Gittervektor $\underline{v}$ gefunden, der vom LLL-Algorithmus zu einer Basis von L erweitert wird. Dies folgt mit Lemma 2.35. Durch eine erneute Ausführung von Schritt 3 bis 7 wird eine Korkine-Zolotarev-reduzierte Basis berechnet, da $\underline{b}_1$ jetzt ein kürzester Gittervektor ist. □

Laufzeitanalyse

Wir werden nun folgenden Satz zeigen:

Satz 2.39 *Sei $L \subseteq \mathbb{Z}^d$ ein Gitter und $\underline{b}_1, \ldots, \underline{b}_n$ eine Basis von L, $\mathcal{B} \in \mathbb{R}$ mit $\mathcal{B} \geq \|\underline{b}_i\|^2$ für $1 \leq i \leq n$ und $\mathcal{B} \geq 2$. Für die Ausführung von Algorithmus 2.28 sind maximal $dn^{o(n)}(n^{n/2} + \log \mathcal{B})$ arithmetische Operationen auf ganzen Zahlen der binären Länge $O(n^2(\log n + \log \mathcal{B}))$ notwendig.*

Anzahl an ausgeführten arithmetischen Operationen:

Wir gehen bei der Laufzeitanalyse davon aus, daß KORKZOLO in jeder Rekursionstiefe mit Vektoren aufgerufen wird, deren Einträge nennergleich sind. Ähnlich wie im Beweis zu Satz 2.18 kann dann argumentiert werden, daß jede Operation auf rationalen Zahlen durch konstant viele arithmetische Operationen auf ganzen Zahlen ausgeführt werden kann. Mit den Voraussetzungen aus Satz 2.39 ist leicht einzusehen, daß nur rationale Zahlen auftreten, denn wir betrachten stets das Quadrat der euklidischen Norm, und in der Prozedur ENUMERATE genügt es, gerundete untere und obere Schranken zu betrachten. Die Voraussetzungen aus Satz 2.39 gelten für jede der folgenden Propositionen und jedes der folgenden Lemmas der Laufzeitanlyse.

Lemma 2.40 *Die Prozedur ENUMERATE in Schritt 10 zählt maximal $(18n)^{n/2}$ Elemente aus L auf, um S zu berechnen. Damit führt sie $dn^{n/2+o(n)}$ arithmetische Operationen auf ganzen Zahlen aus.*

Beweis: Aus Lemma 2.36 wissen wir, daß maximal $\prod_{j=1}^{j_0}(2\lfloor\|\underline{b}_1\|/\ |\underline{b}_j^\star\|\rfloor + 1)$ Elemente aus S aufgezählt werden. Wegen Schritt 9 gilt $\|\underline{b}_1\| \geq \|\underline{b}_j^\star\|$ für $1 \leq j \leq j_0 - 1$. Somit haben wir

$$\prod_{j=1}^{j_0-1}(2\,\|\underline{b}_1\|/\|\underline{b}_j^\star\| + 1) \leq 3^{j_0-1}\prod_{j=1}^{j_0-1}(\|\underline{b}_1\|/\|\underline{b}_j^\star\|).$$

$\prod_{j=1}^{j_0-1}\|\underline{b}_j^\star\|$ ist die Determinante des Gitters $L' := L(\underline{b}_1, \ldots, \underline{b}_{j_0-1})$ (vgl. dazu (1.7)). Nun ist $\|\underline{b}_2^\star\| = \lambda_1(L'_2)$, somit nach (1.13)

$$\|\underline{b}_2^\star\|^{j_0-2} \leq (j_0-2)^{(j_0-2)/2}\mathrm{d}(L'_2).$$

Aus $\|\underline{b}_1\| \leq 2/\sqrt{3}\,\|\underline{b}_2\|$ und $|\mu_{21}| \leq 1/2$ folgt, daß $\sqrt{2}\,\|\underline{b}_2^\star\| \geq \|\underline{b}_1\|$. Mit $\mathrm{d}(L') = \mathrm{d}(L'_2)\|\underline{b}_1\|$ (Lemma 1.13) und $\|\underline{b}_2^\star\| \leq \|\underline{b}_1\|$ haben wir

$$\|\underline{b}_1\|^{j_0-1} \leq (2(j_0-1))^{(j_0-1)/2}\mathrm{d}(L'),$$

womit die erste Behauptung folgt.

Damit weiß man, wie oft Schritt 8 in ENUMERATE getätigt wird. Bis zur erneuten Ausführung von Schritt 8 in ENUMERATE werden $O(n)$, in Schritt 8 selbst $O(d)$ arithmetische Operationen durchgeführt. Also sind insgesamt

$$O\left(nd(18n)^{n/2}\right) = dn^{n/2+o(n)}$$

arithmetische Operationen notwendig. $\square$

Proposition 2.41 *Es bezeichne $T_2(n,d)$ die maximale Anzahl an arithmetischen Operationen auf Zähler und Nenner aller auftretenden rationalen Zahlen, die in* jeder *Rekursionstiefe bei der Ausführung der Schritte 3, 5-7 und 9-11 von KORKZOLO benötigt werden. Dann ist*

$$T_2(n,d) = dn^{n/2+o(n)}.$$

Beweis: Wir stellen eine rekursive Formel für $T_2(n,d)$ auf. Für die Ausführung von Schritt 3 und für das Liften sind $dp(n)$ Operationen notwendig, wobei p ein Polynom in n ist (vgl. hierzu Abschnitt 2.1). Damit ist mit Proposition 2.31 und Lemma 2.40

$$T_2(n,d) \leq (\log n + 5)\,(T_2(n-1,d) + dp(n)) + dn^{n/2+o(n)}.$$

Aus $n \log\log n / \log n = o(n)$ folgt, daß $(\log n)^n = n^{o(n)}$. Der letzte Summand dominiert folglich und es ist

$$T_2(n,d) = dn^{n/2+o(n)}.$$

$\square$

Größe der auftretenden Zahlen:

Wir wollen Zähler und Nenner aller auftretenden rationalen Zahlen abschätzen. Deshalb leiten wir Schranken für die Nenner her und untersuchen, wie groß die rationalen Zahlen selbst werden können. Damit hätten wir auch Schranken für die Zähler gefunden.

Proposition 2.42 *Während der gesamten Laufzeit von KORKZOLO gibt es eine „aktuelle“ Basis des Ausgangsgitters L. Sie ist ab Schritt 1 nach jedem Schritt schwach reduziert.*

Beweis: Wir untersuchen KORKZOLO Schritt für Schritt und zeigen die Behauptung durch Induktion über n. Für $n = 1$ ist nichts zu zeigen. Sei $n \geq 2$. Während der Ausführung von Schritt 2 liegt stets explizit eine Basis von L vor, die nach Beendigung von Schritt 2 LLL-reduziert, also auch schwach reduziert ist. In den Schritten 3-5 wird eine Basis von L_2 berechnet. Sie ist schwach reduziert, denn die Gram-Schmidt-Zerlegung von $\underline{b}_2(2), \ldots, \underline{b}_n(2)$ ist $\underline{b}_2^\star, \ldots, \underline{b}_n^\star$.

Damit liegt nach Induktionsvoraussetzung während der Abarbeitung des rekursiven Aufrufes, also der Ausführung von Schritt 4, eine schwach reduzierte Basis von L_2 vor. Durch Liften der Basis von L_2 nach L ist eine Basis von L eindeutig bestimmt. Deshalb hat man während der Ausführung von Schritt 5 auch implizit eine Basis von L. Sie wird durch die Ausführung der Schritte 5 bis 7 explizit berechnet.

Man beachte, daß nach der Ausführung von Schritt 8 die neue (aktuelle) Basis von L schwach reduziert ist, denn der Gaußalgorithmus verändert nicht den zweidimensionalen Raum, der von den ersten beiden Basisvektoren aufgespannt wird. Da $\underline{b}_1, \ldots, \underline{b}_n$ nach Schritt 7 schwach reduziert waren, gilt damit auch für $3 \leq j < i \leq n$, daß nach Schritt 8 $|\mu_{ij}| \leq 1/2$ ist. Die Schritte 9 bis 11 haben keinen Einfluß auf die Basis des Gitters. Nach Schritt 12 ist die neue Basis schwach reduziert, denn $\underline{b}_1, \ldots, \underline{b}_\nu$ (Definition von ν siehe Schritt 12 im Algorithmus) sind LLL-reduziert und die orthogonalen Projektionen der $\underline{b}_{\nu+1}, \ldots, \underline{b}_n$ auf $\langle\langle \underline{b}_1, \ldots, \underline{b}_\nu \rangle\rangle$ bleiben vom LLL-Algorithmus unbeeinflußt, also auch μ_{ij} für $\nu \leq j < i \leq n$. $\square$

Proposition 2.43 *Sei $\underline{b}_1^\star, \ldots, \underline{b}_n^\star$ die Gram-Schmidt-Zerlegung der aktuellen Basis von L. Während der gesamten Laufzeit ist $\max\limits_{1 \leq i \leq n} \|\underline{b}_i^\star\|^2 \leq \mathcal{B}$.*

Beweis: Die Behauptung wird durch Induktion über n und durch eine schrittweise Untersuchung des Algorithmus gezeigt. Für $n = 1$ ist nichts zu zeigen. Sei $n \geq 2$. Der LLL-Algorithmus in Schritt 2 erhöht $\max_{1 \leq i \leq n} \|\underline{b}_i^\star\|^2$ nie ([LeLeLo82], S. 524). In Schritt 5 gilt die Behauptung nach Induktionsvoraussetzung, denn $\underline{b}_2^\star, \ldots, \underline{b}_n^\star$ ist die Gram-Schmidt-Zerlegung der Basis von L_2. Weil der Gaußalgorithmus nur $(\underline{b}_1, \underline{b}_2)$ unimodular transformiert, bleiben während der Ausführung von Schritt 8 die senkrechten Projektionen der $\underline{b}_i$, $3 \leq i \leq n$, auf $\langle\langle \underline{b}_1, \underline{b}_2 \rangle\rangle$, d.h. $\underline{b}_3^\star, \ldots, \underline{b}_n^\star$, unbeeinflußt. $\underline{b}_1^\star$ und $\underline{b}_2^\star$ ändern sich während der Ausführung des Gaußalgorithmus nur nach dem Vertauschen. Bezeichnen wir sie mit $\tilde{\underline{b}}_1$ und $\tilde{\underline{b}}_2^\star$ *vor* der Vertauschung und mit $\underline{b}_1$ und $\underline{b}_2^\star$ nach der Vertauschung, dann ist $\|\underline{b}_1\| < \|\tilde{\underline{b}}_1\|$, da $\|\tilde{\underline{b}}_1\| > \|\tilde{\underline{b}}_2\|$ und $\|\underline{b}_2^\star\| \leq \|\tilde{\underline{b}}_1\|$, da $\underline{b}_2^\star$ der Anteil von $\tilde{\underline{b}}_1$ ist, der senkrecht auf $\underline{b}_1$ steht. Insgesamt gilt damit die Aussage auch während der Ausführung von Schritt 8.

ENUMERATE hat keinen unmittelbaren Einfluß auf die $\underline{b}_i^\star$. Untersuchen wir nun Schritt 12. Es ist mit $\underline{v} = \underline{a}_1$

$$\|\underline{a}_1^\star\| \leq \|\underline{b}_1^\star\|.$$

Da $\underline{a}_2^\star, \ldots, \underline{a}_{n+1}^\star$ die Gram-Schmidt-Zerlegung von

$$\underline{b}_1 - \langle \underline{b}_1, v \rangle / \|\underline{v}\|^2 \cdot \underline{v}, \ldots, \underline{b}_n - \langle \underline{b}_n, \underline{v} \rangle / \|\underline{v}\|^2 \cdot \underline{v}$$

ist, gilt zusätzlich

$$\|\underline{a}_i^\star\| \leq \|\underline{b}_{i-1}^\star\|, \quad 2 \leq i \leq n+1.$$

Da sich $\max_{1 \leq i \leq n+1} \|\underline{a}_i^\star\|$ durch den LLL-Algorithmus nie vergrößert und nach dem Stoppen des LLL-Algorithmus $\underline{a}_2^\star, \ldots, \underline{a}_{n+1}^\star$ die Gram-Schmidt-Zerlegung der (aktuellen) Basis von L ist, gilt die Behauptung auch für Schritt 12. Für Schritt 13 wurde sie schon gezeigt, womit die Aussage bewiesen ist. □

Proposition 2.44 *KORKZOLO wird in jeder Rekursionstiefe mit Vektoren aufgerufen, die nur rationale Einträge haben. Seien analog zu (2.28) für $0 \leq j \leq n$ die d_j der aktuellen Basis von L definiert. Wird KORKZOLO in Rekursionstiefe $j-2$ aufgerufen, dann lassen sich alle Einträge der Vektoren mit ganzzahligem Zähler und Nenner d_{j-1} darstellen.*

Beweis: KORKZOLO wird in Rekursionstiefe $j-2$ mit $\underline{b}_j(j), \ldots, \underline{b}_n(j)$ der aktuellen Basis von L aufgerufen. Es ist $d_0 = 1$ und $d_j = \prod_{i=1}^{j} \|\underline{b}_i^\star\|^2$. Analog zum Beweis von (2.31) in Lemma 2.17 kann gezeigt werden, daß $d_{j-1}\underline{b}_i(j) \in \mathbb{Z}^d$ für $1 \leq j < i \leq n$. Da $\underline{b}_1, \ldots, \underline{b}_n \in \mathbb{Z}^d$ folgt damit die Aussage. □

Proposition 2.45 *Außer während der Ausführung des Gaußalgorithmus und des LLL-Algorithmus gilt, daß die Länge jedes Vektors der aktuellen Basis immer nach oben durch $(n\mathcal{B})^{1/2}$ beschränkt ist.*

Beweis: Aus Proposition 2.42 folgt, daß die aktuelle Basis ab Schritt 2 nach jedem Schritt schwach reduziert ist, d.h. $|\mu_{ij}| \leq 1/2$ für $1 \leq j < i \leq n$. Wegen Proposition 2.43 gilt $\|\underline{b}_i^\star\|^2 \leq \mathcal{B}$, $1 \leq i \leq n$ während der gesamten Laufzeit. Damit ist für $1 \leq i \leq n$

$$\|\underline{b}_i\|^2 \leq \|\underline{b}_i^\star\|^2 + 1/4 \sum_{j=1}^{i-1} \|\underline{b}_j^\star\|^2 \leq n\mathcal{B}.$$

□

Proposition 2.46 *In jeder Rekursionstiefe treten bei der Ausführung von Schritt 2, 8 und 12 nur rationale Zahlen auf, deren Zähler und Nenner jeweils durch $O(n^2(\log n + \log \mathcal{B}))$ Bits dargestellt werden können.*

Beweis: Wir zeigen die Aussage nur für Schritt 2. Für Schritt 8 und Schritt 12 läßt sich die Überlegung analog durchführen. Sei $\underline{b}_1, \ldots, \underline{b}_n$ die aktuelle Basis von L. Der LLL-Algorithmus wird in Schritt 2 und in Rekursionstiefe $j-2$, $2 \leq j \leq n$, mit $\underline{b}_j(j), \ldots, \underline{b}_n(j)$ aufgerufen. Es gelte im folgenden stets $j \leq i \leq n$. Die Einträge der $\underline{b}_i(j)$ lassen sich darstellen mit ganzzahligem Zähler und Nenner d_{j-1} (Proposition 2.44). Nach Proposition 2.43 ist $d_j \leq \mathcal{B}^j$, $0 \leq j \leq n$.

Wird der LLL-Algorithmus mit $d_{j-1}\underline{b}_i(j)$, $j \leq i \leq n$, die aus $\mathbb{Z}^d$ sind, aufgerufen, dann verhält er sich genauso, als wäre er mit $\underline{b}_i(j)$, $j \leq i \leq n$, aufgerufen worden, nur daß im ersten Fall die Ergebnisvektoren durch d_{j-1} dividiert werden müssen. Nun ist $\|\underline{b}_i(j)\| \leq \|\underline{b}_i\| \leq (n\mathcal{B})^{1/2}$ nach Proposition 2.45.

Mit den $d_{j-1}\underline{b}_i(j)$ für $j \leq i \leq n$ als Eingabebasis für den LLL-Algorithmus entstehen während der Ausführung des LLL-Algorithmus mit Satz 2.18 nur rationale Zahlen, deren Zähler und Nenner jeweils die binäre Länge $O(n \log(\mathcal{B}^{n-1}\ (n\mathcal{B})^{1/2}))$ $= O(n^2(\log\ n + \log \mathcal{B}))$ haben. □

Jetzt können wir den ersten Teil von Satz 2.39 beweisen.

Proposition 2.47 *Es bezeichne $T_1(n,d)$ die maximale Anzahl an arithmetischen Operationen, die KORKZOLO in jeder Rekursionstiefe bei allen Ausführungen des Gauß- bzw. des LLL-Algorithmus (Schritt 2, 8 und 12) insgesamt benötigt. Dann ist*

$$T_1(n,d) = n^{o(n)} d \log \mathcal{B}.$$

Beweis: Geben wir zuerst einen Hinweis zur Analyse von Schritt 12. Sei $L \subseteq \mathbb{Z}^d$ ein Gitter und $\underline{a}_1 \in L$ mit $\|\underline{a}_1\| = \lambda_1(L)$ und $\underline{a}_2, \ldots, \underline{a}_{n+1}$ eine Basis von L. Sei $\mathcal{A} \in \mathbb{R}$ derart, daß $\|\underline{a}_i\|^2 \leq \mathcal{A}$, $1 \leq i \leq n+1$ und $\mathcal{A} \geq 2$. Ähnlich wie Satz 2.18 bewiesen wurde, kann man zeigen: Wird der LLL-Algorithmus mit $\underline{a}_1, \ldots, \underline{a}_{n+1}$ aufgerufen und angehalten, sobald $\underline{a}_1 = \underline{0}$, dann wurden maximal $O(n^3 d \log \mathcal{A})$ arithmetische Operationen auf ganzen Zahlen der binären Länge $O(n \log \mathcal{A})$ ausgeführt. Ein detailierter Beweis ist in [Sch87] zu finden.

In Schritt 12 ist $\|\underline{a}_i\|^2 \leq \mathcal{B}$ für $1 \leq i \leq n+1$. Für Schritt 8 sind $O(d(n + \log \mathcal{B}))$ Operationen notwendig. Dann ist

$$T_1(n,d) = O\left(n^3 d \log \mathcal{B}\ +\ (\log n + 5)(T_1(n-1,d) + d(n + \log \mathcal{B}))\right).$$

Der LLL-Algorithmus in Schritt 2 und 12 und der Gaußalgorithmus in Schritt 8 lassen sich in Rekursionstiefe $j-1$ mit Vektoren aufrufen, deren Einträge ganzzahlige Zähler und Nenner d_{j-1} haben. Dann werden bei der Ausführung von Schritt 2 und 12 $O(dn^3(n \log \mathcal{B} + \log n))$ und bei der Ausführung von Schritt 8 $O(nd \log \mathcal{B})$ arithmetische Operationen auf ganzen Zahlen ausgeführt. Damit ist

$$\begin{aligned} T_1(n,d) &= O\left(n^3 d \log \mathcal{B} + (\log n)^n d n^4 (\log n + \log \mathcal{B})\right) \\ &= (\log n)^{n+o(n)} d \log \mathcal{B} \\ &= n^{o(n)} d \log \mathcal{B}, \end{aligned}$$

denn $n \log\log n / \log n = o(n)$. □

Lemma 2.48 *Während der gesamten Laufzeit treten nur rationale Zahlen auf, deren Zähler und Nenner jeweils die binäre Länge $O(n^2(\log n + \log \mathcal{B}))$ haben.*

Beweis: Für j gelte $2 \leq j \leq n$. Wir haben gezeigt, daß KORKZOLO in Rekursionstiefe $j-2$ mit $\underline{b}_j(j), \ldots, \underline{b}_n(j)$ der aktuellen Basis $\underline{b}_1, \ldots, \underline{b}_n$ von L aufgerufen wird und daß gilt

1. $\|\underline{b}_i(j)\| \leq \|\underline{b}_i\| \leq (n\mathcal{B})^{1/2}, \quad j \leq i \leq n,$
2. die Einträge der $\underline{b}_i(j)$ lassen sich darstellen mit ganzzahligem Zähler und ganzzahligem Nenner $d_{j-1} \leq \mathcal{B}^{j-1}$.

Dabei folgt 1. mit Proposition 2.45, 2. mit Proposition 2.43 und Proposition 2.44. Es bleibt die Aussage für die Längen aller auftretenden Zahlen bei der Abarbeitung eines Aufrufes von KORKZOLO in Rekursionstiefe $j-2$ zu zeigen. Der Beweis wird nicht über Induktion geführt, j sei beliebig mit $2 \leq j \leq n$.

Für Schritt 1 ist die Aussage trivial, für Schritt 2 folgt die Behauptung mit Proposition 2.46. Da 1. und 2. für $2 \leq j \leq n$ gilt, also auch für $j+1$, wenn $j \neq n$, entstehen während der Ausführung von Schritt 3 nur Zahlen der binären Länge $O(n(\log n + \log \mathcal{B}))$.

Die Berechnung der unimodularen Transformationsmatrix T in Schritt 5 kann durch das Lösen eines Gleichungssystems mit ganzzahligen Einträgen der binären Länge $O(n(\log n + \log \mathcal{B}))$ erfolgen, da sich die Einträge der von KORKZOLO zurückgegebenen Basis $\underline{b}_i(j)$, $j \leq i \leq n$, mit gemeinsamem Nenner d_{j-1} darstellen lassen.

Wird das Gleichungssystem über die Berechnung der Hermite-Normalform gelöst, dann treten ganze Zahlen der binären Länge $O(n^2(\log n + \log \mathcal{B}))$ auf (siehe dazu [DaKaTr87] oder [Mu94]). Damit ist klar, daß die binären Längen der in Schritt 6 auftretenden Zahlen ebenfalls von dieser Größenordnung sind. Da in Schritt 7 Vektoren entstehen, deren euklidische Längen nicht größer als $(n\mathcal{B})^{1/2}$ sind und deren Einträge sich mit Nenner $d_{j-1} \leq \mathcal{B}^{j-1}$ darstellen lassen, folgt die Behauptung auch für diesen Schritt.

Schritt 8 ist durch Proposition 2.46 abgedeckt. Die Analyse von Schritt 9 und 10 ist leicht, wenn man beachtet, daß die Gram-Schmidt-Zerlegung von $\underline{b}_j(j), \underline{b}_{j+1}(j), \ldots, \underline{b}_n(j)$ gerade $\underline{b}_j(j), \underline{b}_{j+1}(j+1), \ldots, \underline{b}_n(n)$, also $\underline{b}_j^\star, \ldots, \underline{b}_n^\star$ ist. Mit Proposition 2.43 ist $\|\underline{b}_i(i)\|^2 \leq \mathcal{B}$ für $j \leq i \leq n$, und auch die Zähler der μ_{ij} werden nicht größer als $\mathcal{B}^n$, denn $|\mu_{ij}| \leq 1/2$, $d_j \leq \mathcal{B}^j$ und $d_j \mu_{ij} \in \mathbb{Z}$ für $j < i \leq n$. Mit (2.49) aus dem Beweis zu Lemma 2.36 und $|\mu_{ij}| \leq 1/2$ für $j < i \leq n$ kann man induktiv zeigen, daß

$$|\alpha_i| \leq (3/2)^n \, \mathcal{B}^{n-1} \, (n\mathcal{B})^{1/2}$$

ist. Es treten damit in ENUMERATE nur Vektoren auf, deren Länge sich mit $O(n(\log n + \log \mathcal{B}))$ Bits darstellen laßt. Man beachte, daß die in ENUMERATE auftretenden unteren und oberen Schranken zwar echt reell sein können, davon aber nur der ganzzahlige Anteil interessant ist. Da S eine Teilmenge aller in ENUMERATE berechneten Vektoren ist, treten auch in Schritt 11 nur Vektoren dieser Größenordnung auf. Schritt 12 folgt aus Proposition 2.46. Schritt 13 wurde schon gezeigt, und Schritt 14 ist trivial. □

Damit ist Satz 2.39 vollständig bewiesen.

Bemerkung

Korollar 2.49 *Sei r mit $r^2 \in \mathbb{N}$ gegeben und $L \subseteq \mathbb{Z}^d$ ein Gitter, $\underline{b}_1, \ldots, \underline{b}_n$ eine Basis von L. Weiterhin sei $\mathcal{B} \in \mathbb{R}$ derart, daß $\|\underline{b}_i\|^2 \leq \mathcal{B}$ für $1 \leq i \leq n$ und $\mathcal{B} \geq 2$.*

1. *Nach $dn^{o(n)}(n^{n/2} + \log \mathcal{B})$ arithmetischen Operationen auf ganzen Zahlen der binären Länge $O(n^2(\log n + \log \mathcal{B}))$ und einem Vergleich ganzer Zahlen der binären Länge $O(\max\{\log \lambda_1(L), \log r\})$ läßt sich durch die Berechnung einer Korkine-Zolotarev-reduzierten Basis mit Algorithmus 2.28 die Frage beantworten, ob es ein $\underline{v} \in L$ mit $\|\underline{v}\| \leq r$ gibt.*

2. *Ist $\underline{b}_1, \ldots, \underline{b}_n$ Korkine-Zolotarev-reduziert und wird Algorithmus 2.37 (ENUMERATE) mit Radius $r = c\|\underline{b}_1\|$ aufgerufen, dann benötigt die Prozedur $d(cn)^{n/2+o(n)}$ arithmetische Operationen auf ganzen Zahlen der binären Länge $O(n(\log c + \log \mathcal{B})$, um $\{\underline{b} \in L \mid \|\underline{b}\| \leq r\}$ zu berechnen.*

Beweis: 1. folgt direkt aus Satz 2.39 und 2. folgt mit Lemma 2.40 und der Überlegung, daß $|\alpha_i| \leq (3/2)^n (c\mathcal{B})^n$ für $1 \leq i \leq n$ ist. □

2.7 Nahe und nächste Gitterpunkte

2.7.1 Beschreibung des Problems

Wir haben das SVP in Abschnitt 2.2 beschrieben als die Suche nach dem kürzesten Gittervektor in L. Genauso hätten wir sagen können, daß wir den nächsten Gitterpunkt zum Nullpunkt in L suchen. Bezeichnen wir das SVP als das homogene Problem, dann wäre das inhomogene Problem, einen nächsten Gitterpunkt in $L \subset \mathbb{R}^d$ zu einem beliebigen $\underline{b}_0 \in \mathbb{R}^d$ mit $\underline{b}_0 \neq \underline{0}$ zu finden, also ein $\underline{w} \in L$ mit $\|\underline{w} - \underline{b}_0\| \leq \|\underline{b} - \underline{b}_0\|$ für alle $\underline{b} \in L$. Wir wollen dieses Problem im folgenden stets mit CVP bezeichnen (engl.: „**C**losest **V**ector **P**roblem“). Wir bezeichnen den Abstand von $\underline{b}_0$ zu seinem nächsten Gitterpunkt in L mit

$$\mu(\underline{b}_0, L) := \min_{\underline{b} \in L} \|\underline{b} - \underline{b}_0\| \tag{2.50}$$

und definieren $\mu(L)$ als den maximalen Abstand von einem beliebigen $\underline{b}_0 \in V(L) = \langle\langle \underline{b}_1, \ldots, \underline{b}_n \rangle\rangle$ zu seinem nächsten Gitterpunkt in L, also formal

$$\mu(L) = \max_{\underline{b}_0 \in V(L)} \mu(\underline{b}_0, L). \tag{2.51}$$

Beispiel 2.50 Sei $L = \mathbb{Z} \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \mathbb{Z} \cdot \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \mathbb{Z} \times \mathbb{Z}$, $\underline{b}_0 = \begin{pmatrix} -9/8 \\ -1/2 \end{pmatrix}$, $\underline{c}_0 = \begin{pmatrix} 17/16 \\ 13/16 \end{pmatrix}$. Dann sind $\begin{pmatrix} -1 \\ -1 \end{pmatrix}$ und $\begin{pmatrix} 0 \\ -1 \end{pmatrix}$ die nächsten Gitterpunkte von $\underline{b}_0$ und $\begin{pmatrix} 1 \\ 1 \end{pmatrix}$ der einzige nächste Gitterpunkt von $\underline{c}_0$, außerdem ist $\mu(L) = 1/2\sqrt{2}$.

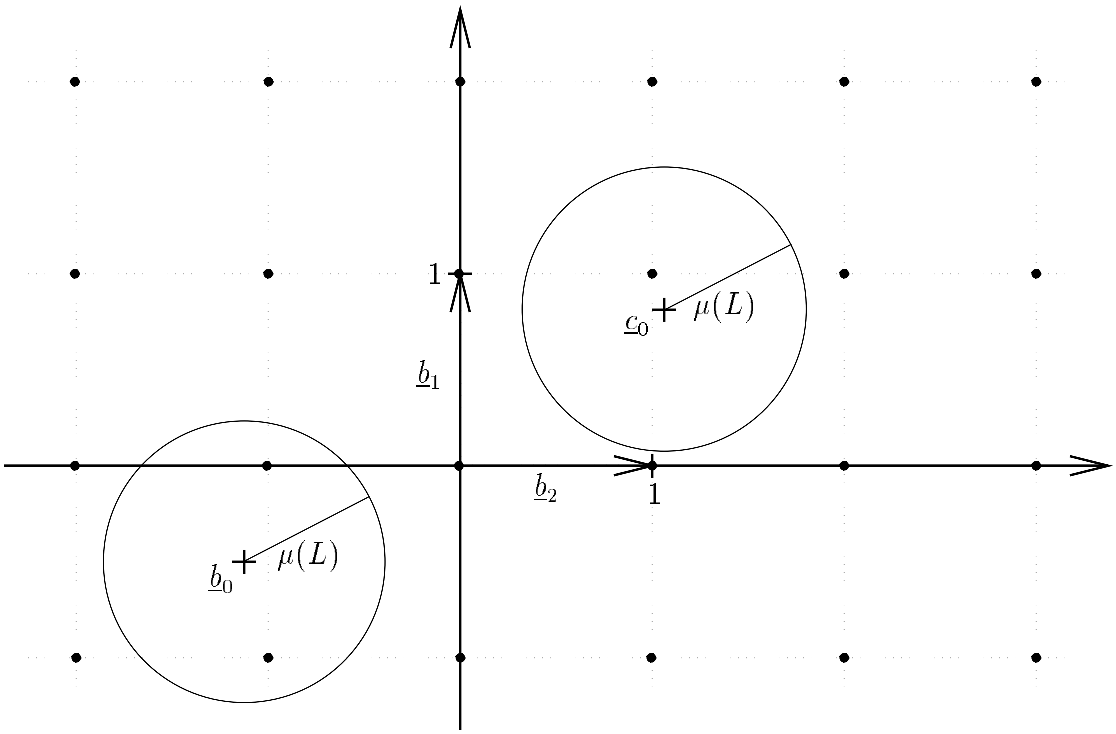

Abbildung 2.4: Nächste Gitterpunkte

Sei n die Dimension eines Gitters L. Es gibt obere und untere Schranken für $\mu(L)$, die nur von n und dem n-ten sukzessiven Minimum abhängen. Es gilt

$$1/2\,\lambda_n(L) \leq \mu(L) \leq 1/2\,n^{3/2}\,\lambda_n(L). \tag{2.52}$$

Einen Beweis für die untere Schranke und weitere Abschätzungen für $\mu(L)$ und $\mu(\underline{b}_0, L)$ findet man in [LaLeSch90]. Wir wollen uns jetzt damit beschäftigen, wie und in welcher Zeit wir das CVP lösen können, wenn $L \subseteq \mathbb{Z}^d$. Im Gegensatz zum SVP ist die Komplexität vom CVP bekannt. Van Emde Boas [Bo81] zeigte, daß das CVP $\mathcal{NP}$-hart ist. Babai [Ba86] bewies aber auch, daß man in Polynomzeit durch die LLL-Reduktion einer speziellen Basis einen Gitterpunkt $\underline{w}$ berechnen kann, für den gilt ($\hat{\underline{b}}_0$ ist die Projektion von $\underline{b}_0$ in $V(L)$)

$$\|\underline{w} - \underline{b}_0\| \leq 2^n \cdot \mu(\hat{\underline{b}}_0, L). \tag{2.53}$$

Er führte somit diese abgeschwächte Form vom CVP auf das Problem zurück, eine LLL-reduzierte Basis zu berechnen. Wir werden uns damit in Abschnitt 2.7.2 näher auseinandersetzen. Es gibt auch einen Algorithmus, der das CVP exakt löst [Ka87a]. Dessen Laufzeit ist zwar polynomiell in der maximalen Länge eines Vektors der gegebenen Vektoren, aber exponentiell in der Dimension des Gitters. Um die genannte Laufzeit zu erreichen, benötigt der Algorithmus eine Korkine-Zolotarev-reduzierte Basis von L. Er führt das n-dimensionale Ausgangsproblem auf exponentiell viele Probleme mit kleinerer Dimension zurück. Wir stellen den Algorithmus in Abschnitt 2.7.3 vor.

2.7.2 Rückführung auf LLL-Reduktion

Da CVP $\mathcal{NP}$-hart ist, wollen wir zunächst zeigen, wie man in *Polynomzeit* einen Gitterpunkt $\underline{w} \in L(\underline{y}_1, \ldots, \underline{y}_n) \subseteq \mathbb{Z}^d$ berechnen kann, der möglichst nahe bei einem vorgegebenem $\underline{b}_0 \in \mathbb{Z}^d$ liegt. Sei $\hat{\underline{b}}_0$ die Projektion von $\underline{b}_0$ in $\langle\langle \underline{y}_1, \ldots, \underline{y}_n \rangle\rangle$. Stellen wir uns eine Kugel um $\hat{\underline{b}}_0$ vor, dessen Radius um einen in der Dimension n *exponentiellen* Faktor größer sei als der Abstand von $\hat{\underline{b}}_0$ zu einem seiner nächsten Gitterpunkte. Ist $L \subseteq \mathbb{Z}^d$ und $\underline{b}_0 \in \mathbb{Z}^d$, dann können wir durch LLL-Reduktion einer speziellen Basis in Polynomzeit einen Gitterpunkt $\underline{w} \in L$ berechnen, der im Inneren dieser Kugel liegt.

Hinter dem Verfahren, das auf Babai [Ba86] zurückgeht, steht die Überlegung, $\underline{b}_0$ mit Hilfe der LLL-Reduktion $\underline{b}_1, \ldots, \underline{b}_n$ von $\underline{y}_1, \ldots, \underline{y}_n$ zu verkürzen. Der Gittervektor, um den $\underline{b}_0$ verkürzt wurde, ist dann ein Gitterpunkt, der nahe bei $\underline{b}_0$ liegt. Um den LLL-Algorithmus anwenden zu können, konstruieren wir ein neues Gitter L', das von

$$\check{\underline{y}}_i = \begin{pmatrix} 0 \\ \underline{y}_i \end{pmatrix} \text{ für } 1 \leq i \leq n,\ \check{\underline{y}}_{n+1} = \begin{pmatrix} C \\ \underline{b}_0 \end{pmatrix}, \tag{2.54}$$

C genügend groß (siehe unten), aufgespannt wird. L' ist ein Gitter, denn $\check{\underline{y}}_1, \ldots, \check{\underline{y}}_n, \check{\underline{y}}_{n+1}$ sind linear unabhängig.

Da $\check{\underline{y}}^{\star}_{n+1} = \begin{pmatrix} C \\ \underline{b}_0 - \hat{\underline{b}}_0 \end{pmatrix}$ und $\check{\underline{y}}_{n+1} - \check{\underline{y}}^{\star}_{n+1} = \begin{pmatrix} 0 \\ \hat{\underline{b}}_0 \end{pmatrix}$, kann $\check{\underline{y}}_{n+1}$ nur mit $\begin{pmatrix} 0 \\ \underline{w} \end{pmatrix}$, $\underline{w} \in L$, verkürzt werden.

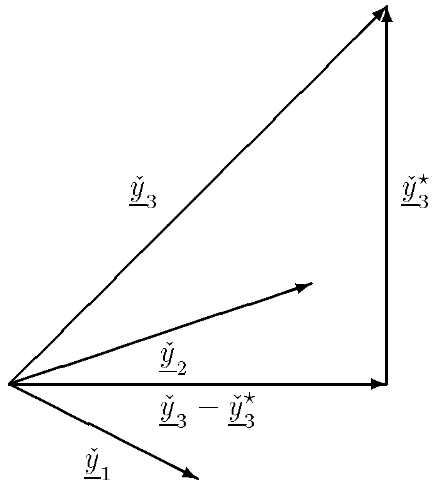

Abbildung 2.5: Ausgangslage für $n = 2$

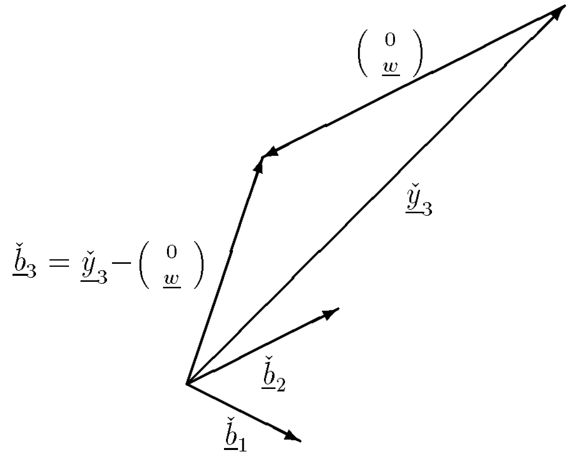

Abbildung 2.6: Nach der LLL-Reduktion

Wir berechnen die LLL-Reduktion $\check{\underline{b}}_1, \ldots, \check{\underline{b}}_n, \check{\underline{b}}_{n+1}$ von $\check{\underline{y}}_1, \ldots, \check{\underline{y}}_n, \check{\underline{y}}_{n+1}$. Um zu verhindern, daß nach der Reduktion von $\check{\underline{y}}_{n+1}$ die LLL-reduzierten $\check{\underline{b}}_1, \ldots, \check{\underline{b}}_n$ mit $\check{\underline{y}}_{n+1}$ verkürzt werden, was keinen Sinn machen würde, müssen wir C so wählen, daß der Algorithmus sofort abbricht.

Es genügt, $C := \lambda_n(Y)^2$ zu wählen, dann ist $\|\check{\underline{y}}^{\star}_{n+1}\|^2 = \lambda_n(Y)^2 + \|\underline{b}_0 - \hat{\underline{b}}_0\|^2$, und für $1 \leq i \leq n$ ist $\|\check{\underline{y}}^{\star}_i\|^2 \leq \|\check{\underline{y}}_i\|^2 \leq \lambda_n(Y)^2 \leq \|\check{\underline{y}}^{\star}_{n+1}\|^2 = \max_{1 \leq i \leq n+1} \|\check{\underline{y}}^{\star}_i\|^2$. Wie im Beweis zu Satz 2.18 erwähnt wurde, erhöht sich während der gesamten Laufzeit dieses Maximum niemals, womit der LLL-Algorithmus terminiert, nachdem die while-Schleife für $k = n + 1$ genau einmal durchlaufen wurde. Wir können nun folgendes zeigen ([GrLoSchr88], S. 149):

Satz 2.51 *Bezeichnen wir* $\check{\underline{b}}_i = \binom{0}{\underline{b}_i}$ *für* $1 \leq i \leq n$ *und* $\check{\underline{b}}_{n+1} = \binom{C}{\underline{b}_0 - \underline{w}}$ *die LLL-Reduktion von* $\check{\underline{y}}_1, \ldots, \check{\underline{y}}_n, \check{\underline{y}}_{n+1}$ *aus (2.54), dann gilt für alle* $\underline{v} \in L$:

$$\underline{w} \in L \textit{ und } \|\underline{w} - \hat{\underline{b}}_0\| \leq 2^n \, \|\underline{v} - \hat{\underline{b}}_0\|. \tag{2.55}$$

Haben wir $n = d$, *dann gilt sogar* $\|\underline{w} - \underline{b}_0\| \leq 2^n \, \|\underline{v} - \underline{b}_0\|$. *Ist* $L \subseteq \mathbb{Z}^d$ *und* $\underline{b}_0 \in \mathbb{Z}^d$, $\mathcal{B} \in \mathbb{R}$ *derart, daß* $\|\underline{b}_0\|^2 \leq \mathcal{B}$, $\|\underline{y}_i\|^2 \leq \mathcal{B}$ *für* $1 \leq i \leq n$, *dann läßt sich in* $O(n^3 d \log \mathcal{B})$ *arithmetischen Operationen auf ganzen Zahlen der Länge* $O(n \log \mathcal{B})$ *ein Gitterpunkt* $\underline{w} \in L$ *berechnen, für den (2.55) gilt.*

Beweis: Wir haben $\underline{w} \in L$, denn von $\check{\underline{y}}_{n+1}$ sind nur ganzzahlige Vielfache von $\underline{y}_1, \ldots, \underline{y}_n$ abgezogen worden. Da $\check{\underline{b}}_{n+1}$ LLL-reduziert und

$$\check{\underline{b}}_{n+1} - \check{\underline{b}}^\star_{n+1} = \sum_{i=1}^{n} \langle \check{\underline{b}}_{n+1}, \check{\underline{b}}^\star_i \rangle / \langle \check{\underline{b}}^\star_i, \check{\underline{b}}^\star_i \rangle \cdot \check{\underline{b}}^\star_i,$$

gilt auch mit $\alpha_i = \langle \check{\underline{b}}_{n+1}, \check{\underline{b}}^\star_i \rangle / \langle \check{\underline{b}}^\star_i, \check{\underline{b}}^\star_i \rangle$, $1 \leq i \leq n$,

$$\hat{\underline{b}}_0 - \underline{w} = \sum_{i=1}^{n} \alpha_i \cdot \underline{b}^\star_i$$

mit $|\alpha_i| \leq 1/2$ für $1 \leq i \leq n$. Sei $\underline{v}$ nun ein beliebiger Gitterpunkt in L, dann können wir schreiben

$$\hat{\underline{b}}_0 - \underline{v} = \sum_{i=1}^{n} \beta_i \cdot \underline{b}^\star_i.$$

Sei k der größte Index mit $\alpha_k \neq \beta_k$. Dann ist

$$\underline{v} - \underline{w} = \sum_{i=1}^{k} (\alpha_i - \beta_i) \cdot \underline{b}^\star_i \in L.$$

Daraus folgt, daß $\alpha_k - \beta_k \neq 0$ und ganzzahlig, also $|\alpha_k - \beta_k| \geq 1$ und damit $|\beta_k| \geq 1/2$. Wir können somit nach unten abschätzen

$$\|\underline{v} - \hat{\underline{b}}_0\|^2 \geq \sum_{i=k+1}^{n} \beta_i^2 \, \|\underline{b}^\star_i\|^2 + 1/4 \, \|\underline{b}^\star_k\|^2 = \sum_{i=k+1}^{n} \alpha_i^2 \, \|\underline{b}^\star_i\|^2 + 1/4 \, \|\underline{b}^\star_k\|^2.$$

Da $\underline{b}_1, \ldots, \underline{b}_n$ LLL-reduziert ist, gilt $\|\underline{b}^\star_j\| \leq 2^{i-j} \|\underline{b}^\star_i\|$ für $1 \leq j \leq i \leq n$ ((2.16) im Beweis zu Lemma 2.10). Damit folgt

$$\begin{aligned}
\|\underline{w} - \hat{\underline{b}}_0\|^2 &\leq \sum_{i=k+1}^{n} \alpha_i^2 \, \|\underline{b}^\star_i\|^2 + 1/4 \cdot \sum_{i=1}^{k} \|\underline{b}^\star_i\|^2 \\
&\leq \sum_{i=k+1}^{n} \alpha_i^2 \, \|\underline{b}^\star_i\|^2 + 1/4 \, \|\underline{b}^\star_k\|^2 \left(\sum_{i=1}^{k} 2^{2k-2i} \right) \\
&\leq 2^{2k} \, \|\underline{v} - \hat{\underline{b}}_0\|^2 \leq 2^{2n} \, \|\underline{v} - \hat{\underline{b}}_0\|^2.
\end{aligned}$$

Die Laufzeitaussage folgt aus Satz 2.18. $\square$

2.7.3 Der Algorithmus von Kannan

Beschreibung

Wir wollen nun einen Algorithmus angeben, der eine exakte Lösung für das CVP liefert [Ka87a]. Sei $\underline{b}_1, \ldots, \underline{b}_n$ eine Basis von L und $\hat{\underline{b}}_0$ die Projektion von $\underline{b}_0$ in $\langle\langle \underline{b}_1, \ldots, \underline{b}_n \rangle\rangle$. Wir stellen fest, daß ein nächster Gitterpunkt von $\underline{b}_0$ auch ein nächster Gitterpunkt von $\hat{\underline{b}}_0$ ist. Wie wir in Lemma 2.53 sehen werden, befinden sich die nächsten Gitterpunkte von $\hat{\underline{b}}_0$ in einer Kugel mit dem Radius $r = 1/2 \cdot (\sum_{j=1}^n \|\underline{b}_j^\star\|^2)^{1/2}$ um $\hat{\underline{b}}_0$. Mit Hilfe dieser Größe könnten wir, ähnlich wie in Algorithmus 2.37, alle Gitterpunkte aufzählen, die maximal den Abstand r von $\hat{\underline{b}}_0$ haben. Ein nächster Gitterpunkt wäre dann darunter. Wir könnten dann aber nur zeigen, daß deren Anzahl durch $r^n/\mathrm{d}(L)$ beschränkt ist. Für diesen Term ist uns keine obere Schranke kannt, die nur von der Dimension abhängt, d.h. die Anzahl an aufgezählten Gitterpunkten wäre nicht nur von der Dimension bestimmt. Wir gehen deshalb einen etwas anderen Weg.

Sei $\underline{b}_1, \ldots, \underline{b}_n$ Korkine-Zolotarev-reduziert und ein k derart gewählt, daß $\|\underline{b}_k^\star\| \geq \|\underline{b}_j^\star\|$ für $1 \leq j \leq n$. Dann können wir zeigen, daß die Anzahl aller Gitterpunkte $\underline{u} \in L(\underline{b}_k, \ldots, \underline{b}_n)$, für die es ein $\underline{v} \in L(\underline{b}_1, \ldots, \underline{b}_{k-1})$ gibt mit $\|\underline{u} + \underline{v} - \hat{\underline{b}}_0\| \leq \sqrt{n}/2 \cdot \|\underline{b}_k^\star\|$, nach oben durch $(n + \sqrt{n})^{n-k+1}$ beschränkt ist. Daraus ergibt sich die folgende Vorgehensweise: Wir berechnen mit der Prozedur ENUMERATECVP (Algorithmus 2.55) die Menge N, die unter anderem alle Gitterpunkte $\underline{u}$ mit der eben beschriebenen Eigenschaft enthält. Dann lösen wir rekursiv für jedes $\underline{u} \in N$ das $(k-1)$-dimensionale Problem, einen nächsten Gitterpunkt für $\underline{b}_0 - \underline{u}$ in $L(\underline{b}_1, \ldots, \underline{b}_{k-1})$ zu finden. Erhalten wir $\underline{v}$, dann ist $\underline{u} + \underline{v}$ ein möglicher nächster Gitterpunkt von $\underline{b}_0$ in L.

Algorithmus 2.52 CVP$(n;\ \underline{b}_0;\ \underline{b}_1, \ldots, \underline{b}_n)$

EINGABE: Dimension n und Korkine-Zolotarev-reduzierte Basis $\underline{b}_1, \ldots, \underline{b}_n$ des Gitters $L \subseteq \mathbb{Z}^d$, Gram-Schmidt-Zerlegung $\underline{b}_1^\star, \ldots, \underline{b}_n^\star$ und die dazugehörigen μ_{ij} für $1 \leq j < i \leq n$, Punkt $\underline{b}_0 \in \mathbb{Z}^d$.
AUSGABE: Ein nächster Gitterpunkt von $\underline{b}_0$ in L, also ein $\underline{w} \in L$ mit $\|\underline{w}\| = \min\{\|\underline{b} - \underline{b}_0\| \mid \underline{b} \in L\} = \mu(\underline{b}_0, L)$.

(1) Wähle k derart, daß $\|\underline{b}_k^\star\|^2 \geq \|\underline{b}_i^\star\|^2$ für $1 \leq i \leq n$.
(2) $C := \{\}$; /* In C stehen Kandidaten aus L, die als nächster Gitterpunkt in Frage kommen. */
(3) Berechne mit der Prozedur ENUMERATECVP eine Menge $N \subset L$, die unter anderem alle $\underline{u} \in L(\underline{b}_k, \ldots, \underline{b}_n)$ enthält, für die es ein $\underline{v} \in L(\underline{b}_1, \ldots, \underline{b}_{k-1})$ gibt, mit $\|\underline{u} + \underline{v} - \hat{\underline{b}}_0\|^2 \leq n/4 \cdot \|\underline{b}_k^\star\|^2$.

```
    if (k = 1) then
      C := N;
    else
      for (each u ∈ N) do
        v := CVP(k − 1; b_0 − u; b_1, ..., b_{k−1});
        C := C ∪ {u + v};
      od
    fi
    Suche w ∈ C mit ||w||^2 = min{||b − b_0||^2 | b ∈ C} = μ(b_0, L)^2.
    return w;
```

Korrektheitsbeweis

Lemma 2.53 *Sei $L \subset \mathbb{R}^d$ ein Gitter und $\underline{b}_1, \ldots, \underline{b}_n$ eine Basis von L. Sei $\underline{b}_0 \in \mathbb{R}^d$ und $\hat{\underline{b}}_0$ die Projektion von $\underline{b}_0$ in $\langle\langle \underline{b}_1, \ldots, \underline{b}_n \rangle\rangle$. Dann gibt es ein $\underline{b} \in L$ mit*

$$\|\underline{b} - \hat{\underline{b}}_0\| \leq 1/2 \left(\sum_{j=1}^{n} \|\underline{b}_j^\star\|^2 \right)^{1/2}. \tag{2.56}$$

Sei k derart, daß $\|\underline{b}_k^\star\| \geq \|\underline{b}_j^\star\|$ für $1 \leq j \leq n$, dann ist

$$\|\underline{b} - \hat{\underline{b}}_0\| \leq \sqrt{n}/2 \|\underline{b}_k^\star\|. \tag{2.57}$$

Beweis: Wir konstuieren ein $\underline{b} \in L$, das (2.56) und damit auch (2.57) erfüllt. Sei $\tau_j = \langle \hat{\underline{b}}_0, \underline{b}_j^\star \rangle / \langle \underline{b}_j^\star, \underline{b}_j^\star \rangle = \langle \underline{b}_0, \underline{b}_j^\star \rangle / \langle \underline{b}_j^\star, \underline{b}_j^\star \rangle$ für $1 \leq j \leq n$, womit $\hat{\underline{b}}_0 = \sum_{j=1}^{n} \tau_j \cdot \underline{b}_j^\star$ ist. Wählen wir sukzessive $\alpha_n, \alpha_{n-1}, \ldots, \alpha_1 \in \mathbb{Z}$ mit

$$|\alpha_j + \sum_{i=j+1}^{n} \alpha_i \mu_{ij} - \tau_j| \leq 1/2,$$

dann erfüllt

$$\underline{b} = \sum_{i=1}^{n} \alpha_i \cdot \underline{b}_i \in L$$

die obigen Bedingungen, denn wie wir in (2.48) gesehen haben, ist

$$\underline{b} = \sum_{j=1}^{n} \left(\sum_{i=j}^{n} \alpha_i \mu_{ij} \right) \cdot \underline{b}_j^\star$$

und damit haben wir

$$\|\underline{b} - \hat{\underline{b}}_0\| = \left(\sum_{j=1}^{n} \left(\sum_{i=j}^{n} \alpha_i \mu_{ij} - \tau_j \right)^2 \|\underline{b}_j^\star\|^2 \right)^{1/2} \leq 1/2 \left(\sum_{j=1}^{n} \|\underline{b}_j^\star\|^2 \right)^{1/2} \leq \sqrt{n}/2 \|\underline{b}_k^\star\|.$$

□

Lemma 2.54 *Sei $L \subseteq \mathbb{Z}^d$ ein Gitter und $\underline{b}_1, \ldots, \underline{b}_n$ eine Korkine-Zolotarev-reduzierte Basis von L. Wir können leicht eine Menge $N \subset L$ berechnen mit $|N| \leq (n+\sqrt{n})^{n-k+1}$ Elementen, so daß gilt: Ist $\underline{w} = \sum_{j=1}^{n} \alpha_j \cdot \underline{b}_j$ ein nächster Gitterpunkt von $\underline{b}_0$, dann ist $\underline{u} = \sum_{j=k}^{n} \alpha_j \cdot \underline{b}_j \in N$.*

Beweis: Seien $\hat{\underline{b}}_0$ und die τ_j für $1 \leq j \leq n$ wie in Lemma 2.53. Für alle Gitterpunkte $\underline{b} = \sum_{j=1}^{n} \alpha_j \cdot \underline{b}_j$, deren Abstand zu $\hat{\underline{b}}_0$ maximal $\sqrt{n}/2\|\underline{b}_k^\star\|$ ist, gilt mit (2.46) im Beweis zu Lemma 2.36

$$\|\underline{b} - \hat{\underline{b}}_0\|^2 = \sum_{j=1}^{n} \left(\sum_{i=j}^{n} \alpha_i \mu_{ij} - \tau_j \right)^2 \|\underline{b}_j^\star\|^2 \leq n/4\|\underline{b}_k^\star\|^2. \tag{2.58}$$

Berechnet man sukzessive alle $\alpha_n, \alpha_{n-1}, \ldots, \alpha_k \in \mathbb{Z}$, die den Ungleichungen

$$\left(\sum_{i=j}^{n} \alpha_i \mu_{ij} - \tau_j \right)^2 \|\underline{b}_j^\star\|^2 \leq n/4\|\underline{b}_k^\star\|^2 \text{ für } j = n, \ldots, k \tag{2.59}$$

genügen, dann sind darunter auch alle $\alpha_n, \ldots, \alpha_k \in \mathbb{Z}$, für die (2.58) gilt, denn es ist für $j = n, \ldots, k$

$$\left(\sum_{i=j}^{n} \alpha_i \mu_{ij} - \tau_j \right)^2 \|\underline{b}_j^\star\|^2 \leq n/4\|\underline{b}_k^\star\|^2 - \sum_{i=j+1}^{n} \left(\sum_{l=i}^{n} \alpha_l \mu_{li} - \tau_i \right)^2 \|\underline{b}_i^\star\|^2. \tag{2.60}$$

Damit können wir eine Menge $N \subset L(\underline{b}_k, \ldots, \underline{b}_n)$ berechnen, die alle $\underline{u}$ enthält, für die es ein $\underline{v} \in L(\underline{b}_1, \ldots, \underline{b}_{k-1})$ gibt, mit $\|\underline{u} + \underline{v} - \hat{\underline{b}}_0\| \leq \sqrt{n}/2\|\underline{b}_k^\star\|$. Da ein nächster Gitterpunkt $\underline{w}$ von $\underline{b}_0$ auch ein nächster Gitterpunkt von $\hat{\underline{b}}_0$ ist, ist der Teil von $\underline{w}$, der in $L(\underline{b}_{j_0}, \ldots, \underline{b}_n)$ liegt, auch in N. Wir wollen nun die Anzahl der Elemente von N abschätzen. Aus (2.59) folgt $|N| \leq \prod_{j=k}^{n}(1 + \sqrt{n})\|\underline{b}_k^\star\|/\|\underline{b}_j^\star\|$. Da $\underline{b}_1, \ldots, \underline{b}_n$ Korkine-Zolotarev-reduziert ist, ist $\underline{b}_k^\star$ ein kürzester Gittervektor von $L_k = L(\underline{b}_k(k), \underline{b}_{k+1}(k), \ldots, \underline{b}_n(k))$. Wegen (1.13) haben wir

$$\|\underline{b}_k^\star\| \leq \sqrt{n-k+1}\, \mathrm{d}(L_k)^{1/(n-k+1)}.$$

Mit $\mathrm{d}(L_k) = \|\underline{b}_k^\star\| \cdots \|\underline{b}_n^\star\|$ ist

$$|N| \leq (1 + \sqrt{n})^{n-k+1}(n-k+1)^{(n-k+1)/2} \leq (n + \sqrt{n})^{n-k+1}.$$ □

Im folgenden Algorithmus werden die Kandidaten für α_j über (2.59) aufgezählt. Bei einer Implementierung sollte jedoch (2.60) verwendet werden.

Algorithmus 2.55 ENUMERATECVP(j)

EINGABE: Hilfsparameter $j = n$. Weiterhin müssen bekannt sein: Basisvektoren $\underline{b}_k, \ldots, \underline{b}_n$, $\underline{b}_k^\star, \ldots, \underline{b}_n^\star$ und die dazugehörigen μ_{ij} für $k \leq j < i \leq n$, $\underline{b}_0$, Hilfsvektor $\underline{\alpha} = (\alpha_k, \ldots, \alpha_n) = \underline{0}$, $N = \{\}$.
AUSGABE: Eine Menge $N \subset L$, die unter anderem alle $\underline{u} \in L(\underline{b}_k, \ldots, \underline{b}_n)$ enthält, für die es ein $\underline{v} \in L(\underline{b}_1, \ldots, \underline{b}_{k-1})$ gibt mit $\|\underline{u}+\underline{v}-\hat{\underline{b}}_0\| \leq \sqrt{n}/2 \cdot \|\underline{b}_k^\star\|$.

(1) $t_j := \sum_{i=j+1}^n \alpha_i \mu_{ij} - \langle \underline{b}_0, \underline{b}_j^\star \rangle / \|\underline{b}_j^\star\|^2$;
(2) **if** $(j > k)$ **then**
(3) **for** $(-t_j - \sqrt{n}/2 \cdot \|\underline{b}_k^\star\| / \|\underline{b}_j^\star\| \leq \alpha_j \leq -t_j + \sqrt{n}/2 \cdot \|\underline{b}_k^\star\| / \|\underline{b}_j^\star\|)$ **do**
(4) ENUMERATECVP($j-1$);
(5) **od**
(6) **else**
(7) **for** $(-t_k - \sqrt{n}/2 \leq \alpha_k \leq -t_k + \sqrt{n}/2)$ **do**
(8) $N := N \cup \{\sum_{i=k}^n \alpha_i \cdot \underline{b}_i\}$;
(9) **od**
(10) **fi**

Lemma 2.56 *Algorithmus 2.52 berechnet einen nächsten Gitterpunkt $\underline{w}$ von $\underline{b}_0$ in L.*

Beweis: Wir beweisen die Aussage durch Induktion über die Gitterdimension n. Der Fall $n = 1$ folgt mit Lemma 2.54. Sei $n \geq 2$. Wir nehmen an, die Behauptung gelte für alle Gitter, deren Dimension kleiner als n ist. Ist $\underline{w} = \sum_{j=1}^n \alpha_j \cdot \underline{b}_j$ ein nächster Gitterpunkt von $\underline{b}_0$, dann ist mit Lemma 2.54 $\underline{u} = \sum_{j=k}^n \alpha_j \cdot \underline{b}_j \in N$. Ist $k = 1$, dann ist nichts zu zeigen. Sei $k > 1$. Weil $\underline{u} \in N$ ist, wird das CVP mit $\underline{b}_0 - \underline{u}$ aufgerufen und nach Induktionsvoraussetzung berechnet das CVP einen nächsten Gitterpunkt $\underline{v}$ von $\underline{b}_0 - \underline{u}$ in $L(\underline{b}_1, \ldots, \underline{b}_{k-1})$.

Wir zeigen: $\underline{u}+\underline{v}$ ist ein nächster Gitterpunkt von $\underline{b}_0$ in L. $\|\underline{u}+\underline{v}-\underline{b}_0\| > \|\underline{w}-\underline{b}_0\|$ ist falsch, denn $\|\underline{v}+\underline{u}-\underline{b}_0\| = \mu(\underline{b}_0-\underline{u}, L(\underline{b}_1, \ldots, \underline{b}_{k-1}))$ und $\underline{w}-\underline{u} \in L(\underline{b}_1, \ldots, \underline{b}_{k-1})$. $\|\underline{u} + \underline{v} - \underline{b}_0\| < \|\underline{w} - \underline{b}_0\|$ ist falsch, denn $\|\underline{w} - \underline{b}_0\| = \mu(\underline{b}_0, L)$ und $\underline{u} + \underline{v} \in L$. Also ist $\|\underline{u} + \underline{v} - \underline{b}_0\| = \|\underline{w} - \underline{b}_0\| = \mu(\underline{b}_0, L)$, folglich ist nach der Ausführung der Schritte 8-11 $\underline{u} + \underline{v} \in C$. □

Laufzeitanalyse

Satz 2.57 *Sei $L \subseteq \mathbb{Z}^d$ ein Gitter und $\underline{b}_1, \ldots, \underline{b}_n$ eine Korkine-Zolotarev-reduzierte Basis von L, $\underline{b}_0 \in \mathbb{Z}^d$, $\|\underline{b}_i\|^2 \leq \mathcal{B}$ für $0 \leq i \leq n$ und $\mathcal{B} \geq 2$. Dann läßt sich ein nächster Gitterpunkt von $\underline{b}_0$ in L mit Algorithmus 2.52 durch $n^{3/2n+O(1)}d$ arithmetische Operationen auf ganzen Zahlen berechnen. Es treten nur rationale Zahlen auf, und die binäre Länge von Zähler und Nenner ist jeweils $O(n \log \mathcal{B})$.*

Beweis:
Anzahl an arithmetischen Operationen:

Wir geben nur eine Skizze für den Laufzeitbeweis an. Sei $T(n,d)$ die Anzahl an arithmetischen Operationen, die Algorithmus 2.52 für die Berechnung eines nächsten Gitterpunktes von $\underline{b}_0$ in L benötigt. Da Algorithmus 2.52 mit Korkine-Zolotarev-reduzierter Basis aufgerufen wird und $\underline{b}_1, \dots, \underline{b}_{k-1}$ Korkine-Zolotarev-reduzierte Basis des Gitters $L(\underline{b}_1, \dots, \underline{b}_{k-1})$ für $1 < k \leq n+1$ ist, haben wir mit Lemma 2.54

$$T(n,d) \leq (n + \sqrt{n})^{(n-k+1)}(T(k-1,d) + dp(n)),$$

wobei $p(n)$ ein Polynom ist. Mit $(n+\sqrt{n})^{(n-k+1)} \leq n^{3/2(n-k+1)}$ ist

$$T(n,d) \leq n^{3/2n+O(1)}d.$$

Größe der auftretenden Zahlen:

Beachtet man, daß in Schritt 3 und Schritt 7 in der Prozedur ENUMERATECVP von der unteren und oberen Schranke nur die gerundeten Werte notwendig sind, dann sieht man leicht ein, daß alle während der Laufzeit auftretenden Zahlen rational sind. Wir wollen sie jetzt abschätzen. Es ist $|\mu_{ij}| \leq 1/2$ für $1 \leq j < i \leq n$ und $\|\underline{b}_i^\star\|^2 \leq \mathcal{B}$ für $1 \leq i \leq n$. Mit Lemma 2.17 sieht man, daß Zähler und Nenner der μ_{ij} bzw. der Einträge der $\underline{b}_i^\star$ durch $\mathcal{B}^{n+1}$ beschränkt sind. Wir schätzen jetzt alle α_i und die Länge aller Vektoren (mit ganzzahligen Einträgen) ab, die während der gesamten Laufzeit neu berechnet werden.

Sei $1 \leq i \leq n$. Bezeichnen wir mit k_i das in Rekursionstiefe $i-1$ berechnete k. Dann gilt in Rekursionstiefe 0 in ENUMERATECVP

$$|\alpha_{n-j}| \leq (3/2)^{k_1}\, n\mathcal{B} + (3/2)^{k_1}\,\mathcal{B}, \quad 0 \leq j \leq k_1.$$

Für alle $\underline{b} \in N^1$ (N^i sei die Menge N in Rekursionstiefe $i-1$) gilt

$$\begin{aligned} \|\underline{b}\| &\leq (k_1+1)\left((3/2)^{k_1}\, n\mathcal{B} + (3/2)^{k_1}\right)\mathcal{B} \\ &\leq 2^{k_1+1}\, 2\,(3/2)^{k_1}\, n\mathcal{B}^2. \end{aligned}$$

Algorithmus 2.52 wird rekursiv mit $\underline{b}_0 - \underline{u}$ aufgerufen, und es ist

$$\|\underline{b}_0 - \underline{u}\| \leq 2^{k_1+1}\, 2^2\,(3/2)^{k_1}\, n\mathcal{B}^2.$$

In Rekursionstiefe $i-1$ werde Algorithmus 2.52 mit $\underline{b}_0^{(i)} - \underline{u}^{(i)}$ aufgerufen. Wir zeigen durch Induktion über i, daß

$$\|\underline{b}_0^{(i)} - \underline{u}^{(i)}\| \leq 2^{2i}\, 2^{k_i+1}\,(3/2)^{k_i}\, n\mathcal{B}^{i+1}.$$

Den Induktionsanfang haben wir gezeigt. Sei $i \geq 2$. Für $k_i < j \leq k_{i+1}$ gilt in ENUMERATECVP mit Induktionsvoraussetzung

$$\begin{aligned} |\alpha_{n-j}| &\leq 2\,(3/2)^{k_{i+1}-k_i}\, 2^{2i}\, 2^{k_i+1}\, (3/2)^{k_i}\, n\mathcal{B}^{i+1} \\ &\leq (3/2)^{k_{i+1}}\, 2^{2i+1}\, 2^{k_i+1}\, n\mathcal{B}^{i+1}. \end{aligned}$$

Für alle $\underline{b} \in N^{(i+1)}$ ist

$$\begin{aligned} \|\underline{b}\| &\leq (k_{i+1} - k_i)\,(3/2)^{k_{i+1}}\, 2^{2i+1}\, 2^{k_i+1}\, n\mathcal{B}^{i+1}\,\mathcal{B} \\ &\leq 2^{k_{i+1}-k_i}\,(3/2)^{k_{i+1}}\, 2^{2i+1}\, 2^{k_i+1}\, n\mathcal{B}^{i+2} \\ &\leq 2^{k_{i+1}}\,(3/2)^{k_{i+1}}\, 2^{2i+1}\, n\mathcal{B}^{i+2} \end{aligned}$$

Damit ist

$$\|\underline{b}_0^{(i+1)} - \underline{u}^{(i+1)}\| \leq 2^{2i+2}\, 2^{k_{i+1}+1}\, (3/2)^{k_i+1}\, n\mathcal{B}^{i+2}.$$

Da $k_i \leq n$ und $i \leq n$, treten damit während der gesamten Laufzeit nur Zahlen der binären Länge $O(n \log \mathcal{B})$ auf. $\square$

Die nächste Aussage folgt mit Satz 2.39 und Satz 2.57:

Korollar 2.58 *Ist $\underline{b}_1, \ldots, \underline{b}_n$ eine beliebige Basis von $L \subseteq \mathbb{Z}^d$, dann läßt sich ein nächster Gitterpunkt von $\underline{b}_0 \in \mathbb{Z}^d$ in L durch*

$$d(n^{3/2n+O(1)} + n^{o(n)} \log \mathcal{B})$$

arithmetische Operationen auf ganzen Zahlen der binären Länge

$$O(n^2(\log n + \log \mathcal{B}))$$

berechnen.

Kapitel 3

Lösung der Gitterprobleme über $\mathbb{R}^d$

Da es keine Rechenmaschine mit einer exakten Arithmetik auf reellen Zahlen gibt, lassen sich mit den im vorherigen Kapitel vorgestellten Algorithmen reduzierte Gitterbasen, kurze und kürzeste Gittervektoren und nahe und nächste Gitterpunkte nur dann exakt berechnen, wenn vorausgesetzt wird, daß L nur Gitterpunkte mit rationalen Einträgen enthält. Wir wollen in diesem Kapitel aufzeigen, wie man durch Lösen entsprechender Probleme in einem rationalen Näherungsgitter genügend großer Genauigkeit mit den bisher vorgestellten Algorithmen eine Antwort für die Gitterprobleme im Reellen findet. Es wird untersucht werden, welche strukturellen Gittereigenschaften Einfluß auf die erforderliche Genauigkeit haben. Bevor wir darauf eingehen, wollen wir die Begriffe Näherung und Genauigkeit problemadäquat definieren.

3.1 Approximationen

Definition 3.1 (Approximation) *Sei $p \in \mathbb{N}$ und $b \in \mathbb{R}$. Eine rationale Zahl a heißt* **(rationale) Näherung** *oder* **(rationale) Approximation** *der* **Genauigkeit** *oder* **Präzision** *p für b, wenn*

$$|a - b| < 2^{-p}$$

und

$$a \in 2^{-(p+1)}\mathbb{Z}.$$

Bemerkung 3.2 Für alle $b \in \mathbb{R}$ existiert stets eine rationale Näherung der Genauigkeit p. Wähle

$$a = 2^{-(p+1)} \lceil 2^{p+1}\, b \rfloor .$$

Zu einer reellen Zahl kann es mehr als eine rationale Näherung der Genauigkeit p geben.

Beispiel 3.3 Die Zahlen 5/4 und 3/2 sind rationale Näherungen der Genauigkeit 1 für 11/8.

Definition 3.4 (Approximationen für Vektoren und Matrizen) *Ein Vektor* $\underline{a} = (a_1, \ldots, a_d)$ *heißt (rationale) Näherung der Genauigkeit* p *für* $\underline{b} = (b_1, \ldots, b_d) \in \mathbb{R}^d$*, wenn*

$$|a_j - b_j| < 2^{-p} \text{ und } a_j \in 2^{-(p+1)}\mathbb{Z} \text{ für } 1 \leq j \leq d.$$

Eine Matrix $A = (a_{ij})_{1\leq i\leq n,\, 1\leq j\leq d}$ *heißt (rationale) Näherung der Genauigkeit* p *für* $B = (b_{ij})_{1\leq i\leq n,\, 1\leq j\leq d} \in \mathbb{R}^{d\times n}$*, wenn*

$$|a_{ij} - b_{ij}| < 2^{-p} \text{ und } a_{ij} \in 2^{-(p+1)}\mathbb{Z} \text{ für } 1 \leq i \leq n,\ 1 \leq j \leq d.$$

Wir wollen noch einige Aussagen über die Qualität der Näherungen machen. Ist $\underline{a}$ eine rationale Näherung der Genauigkeit p für $\underline{b} \in \mathbb{R}^d$, dann gilt

$$\|\underline{a} - \underline{b}\| < 2^{-p}\sqrt{d}. \tag{3.1}$$

Für eine Matrix $C \in \mathbb{R}^{d\times n}$ sei mit $\|\cdot\|$ die Frobeniusnorm definiert als

$$\|C\| = \left(\sum_{i=1}^{d}\sum_{j=1}^{n} c_{ij}^2\right)^{1/2}.$$

Ist A eine rationale Näherung der Genauigkeit p für $B \in \mathbb{R}^{d\times n}$, dann gilt

$$\|A - B\| < \sqrt{n}\,\sqrt{d}\,2^{-p}. \tag{3.2}$$

Sei $q \in \mathbb{N}$ und $p(q) = \lceil 1/2 \log n + 1/2 \log d + q\rceil$. Im folgenden bezeichne stets A eine rationale Näherung der Genauigkeit $p(q)$ für $B \in \mathbb{R}^{d\times n}$. Damit gilt

$$\|A - B\| < 2^{-q}.$$

3.2 Gitterbasisreduktion über $\mathbb{R}^d$

Vorgehensweise

Sei $L \subset \mathbb{R}^d$ ein Gitter und $B = (\underline{b}_1, \ldots, \underline{b}_n) \in \mathbb{R}^{d\times n}$ eine Basis von L, A wie oben erklärt. Berechnen wir mit den Reduktionsalgorithmen aus Kapitel 2 eine reduzierte Basis $\tilde{A}$ des Näherungsgitters $L(A)$, dann macht $\tilde{A}$ im allgemeinen keine Aussage über das Aussehen einer reduzierten Basis des Ausgangsgitters. Durch eine geringfügige Änderung der Reduktionsalgorithmen läßt sich aber zusätzlich die Transformationsmatrix $T \in \mathbb{Z}^{n\times n}$ mit $AT = \tilde{A}$ gewinnen. T ist unimodular und folglich BT Basis des Ausgangsgitters.

Es ist nun aber nicht von vornherein klar, ob auch bei einer guten Näherung A für B die Basis BT ähnlichen Bedingungen genügt, wie die reduzierte Basis $\tilde{A}$. (Wir beziehen uns hier auf die Eigenschaften (2.14) in Satz 2.9 und (2.38) in Satz 2.24, denn es ist auf Grund der räumlichen Projektionsaussagen in den Definitionen im Allgemeinen nicht zu erwarten, daß sich über die Reduktion der Basis eines Näherungsgitters eine Transformationsmatrix berechnen läßt, so daß BT LLL- oder gar Korkine-Zolotarev-reduziert ist.) Insbesondere könnte der Eingabefehler den Reduktionseffekt zerstören, wenn die Transformationsmatrix große Einträge hat.

Wir werden jedoch im folgenden zeigen, daß bei genügend großer Genauigkeit die Matrix T die Ausgangsbasis B nahezu genauso gut reduziert wie die Approximationsbasis A. Vorher wird untersucht werden, bei welcher Genauigkeit A vom Rang n ist, eine notwendige Eigenschaft, um überhaupt ein Näherungsgitter zu erhalten. Außerdem wird untersucht, wie nahe die sukzessiven Minima von $L(A)$ und L bei welcher Präzision beieinander liegen. Im anschließenden Abschnitt wird als Anwendung der bewiesenen Aussagen gezeigt, daß eine Basis mit echt reellwertigen Einträgen nahezu genauso schnell reduziert werden kann wie eine Basis mit nur ganzzahligen Einträgen.

Hinreichende Genauigkeit

Wir werden nun folgende zwei Sätze zeigen:

Satz 3.5 *Sei* $0 < c \leq 1$ *und*

$$q \geq q_1(B,c) = \lceil n + \log(\lambda_n(L)/\lambda_1(L)) + \log(\mathrm{dft}(B)/\lambda_1(B)) + \log(1/c)\rceil,$$

dann ist A *vom Rang* n *und*

$$\lambda_i(L(A)) \leq (1+c)\lambda_i(L), \quad 1 \leq i \leq n.$$

Um den nächsten Satz formulieren zu können, nehmen wir an, A sei vom Rang n und $T \in \mathbb{Z}^{n\times n}$ regulär mit

$$\|A\underline{t}_i\| \leq s_i\,\lambda_i(L(A)), \quad 1 \leq i \leq n,$$

wobei $s_1, \ldots, s_n$ eine monoton steigende Folge von reellen Zahlen größer oder gleich 1 ist. Weiterhin sei $s = s_n$.

Satz 3.6 *Sei* $0 < c \leq 1$ *und*

$$q \geq q_2(B,c,s) = \lceil n + \log(\lambda_n(L)/\lambda_1(L)) + \log(\mathrm{dft}(B)/\lambda_1(B)) + \log(3s/c)\rceil.$$

Dann sind

$$\lambda_i(T) \le (3/2)^{n+1}\sqrt{n}s_i\,\mathrm{dft}(B)\lambda_i(L)/\lambda_1(B), \quad 1 \le i \le n$$

und

$$\|B\underline{t}_i\| \le (1+c)s_i\lambda_i(L), \quad 1 \le i \le n.$$

Wir wollen nun die beiden Sätze nach einigen Bemerkungen und dem Beweis einiger Hilfsaussagen zeigen. Für beliebiges $\underline{t} \in \mathbb{R}^d$ und $C \in \mathbb{R}^{d\times n}$ ist $\|C\underline{t}\| \le \|C\|\,\|\underline{t}\|$. Es gilt damit

$$\|B\underline{t} - A\underline{t}\| < 2^{-q}\|\underline{t}\|,$$

denn A ist eine rationale Näherung der Genauigkeit $p(q) = \lceil 1/2 \log n + 1/2 \log d + q\rceil$, d.h. $\|A - B\| < 2^{-q}$. Daraus läßt sich das folgende Lemma ableiten:

Lemma 3.7 *Wenn* $\underline{t} \in \mathbb{R}^d\backslash\{\underline{0}\}$, $0 \le c \le 1$ *und*

$$q \ge q_3(B, \underline{t}, c) = \lceil \log\|\underline{t}\| - \log(c\,\|B\underline{t}\|)\rceil,$$

dann ist

$$\|A\underline{t} - B\underline{t}\| < c\,\|B\underline{t}\|.$$

Beweis:

$$\begin{array}{rcl} q & \ge & \log\|\underline{t}\| - \log(c\,\|B\underline{t}\|) \\ 2^{-q} & \le & c\,\|B\underline{t}\|/\|\underline{t}\| \\ 2^{-q}\,\|\underline{t}\| & \le & c\,\|B\underline{t}\|. \end{array}$$

Mit $\|B\underline{t} - A\underline{t}\| < 2^{-q}\,\|\underline{t}\|$ folgt die Behauptung. □

Wir wollen mit Hilfe der Determinante zeigen, von welcher Genauigkeit A sein muß, damit der Rang von A gerade n ist. Da in der Regel $n < d$ ist, d.h. die Determinante von A nicht definiert ist, machen wir folgende Konstruktion: Seien $\underline{b}_1, \ldots, \underline{b}_n \in \mathbb{R}^d$ linear unabhängig, $B = (\underline{b}_1, \ldots, \underline{b}_n)$. Ist $n < d$, dann gibt es $\underline{b}_{n+1}, \ldots, \underline{b}_d \in \mathbb{R}^d$, so daß $\underline{b}_1, \ldots, \underline{b}_d$ linear unabhängig sind. Seien nun $\underline{y}_1, \ldots, \underline{y}_d \in \mathbb{R}^d$ derart, daß

$$\underline{y}_i = \underline{b}_i \text{ für } 1 \le i \le n$$

und

$$\underline{y}_i = 1/\|\underline{b}_i^\star\| \cdot \underline{b}_i^\star \text{ für } n < i \le d.$$

Für $n < i \le d$ ist dann $\|\underline{y}_i\| = 1$ und $\underline{y}_i^\star = \underline{y}_i$, denn $\underline{y}_i^\star = \underline{y}_i - \sum_{j=1}^{i-1} \underbrace{\langle \underline{y}_i, \underline{y}_j^\star\rangle/\|\underline{y}_j^\star\|^2}_{=0} \cdot \underline{y}_j^\star$.

Es gilt mit (1.5) in Definition 1.11

$$|\det Y| = \prod_{i=1}^{d} \|\underline{y}_i^\star\| = \prod_{i=1}^{n} \|\underline{y}_i^\star\| \cdot \prod_{i=n+1}^{d} 1 = \prod_{i=1}^{n} \|\underline{b}_i^\star\| = \sqrt{\det(B^\top B)} = \mathrm{d}(L(B)). \quad (3.3)$$

Sei weiterhin

$$X = (\underline{a}_1, \ldots, \underline{a}_n, \underline{y}_{n+1}, \ldots, \underline{y}_d).$$

Bemerkung 3.8 Ist $\det X \neq 0$, dann sind $\underline{a}_1, \ldots, \underline{a}_n$ linear unabhängig.

Lemma 3.9 *Wenn* $q \geq \lceil -\log \lambda_1(B) \rceil$, *dann ist*

$$|\det X - \det Y| < 2^{-q+n}\, \mathrm{dft}(B)\, \mathrm{d}(L)/\lambda_1(B).$$

Beweis: Wähle $c = 1$ und für $\underline{t}$ einen Einheitsvektor in Lemma 3.7. Dann ist $q_3(B, \underline{t}, c) \leq \lceil -\log \lambda_1(B) \rceil$. Hieraus folgt $\|\underline{a}_i\| \leq 2\|\underline{b}_i\|$ für $1 \leq i \leq n$. Dies und die Ungleichung von Hadamard (Lemma 1.12) führt zu

$$\begin{aligned}
|\det Y - \det X| &= |\sum_{j=1}^{d} \det(\underline{y}_1, \ldots, \underline{y}_{j-1}, \underline{x}_j, \ldots, \underline{x}_d) \\
&\quad - \det(\underline{y}_1, \ldots, \underline{y}_j, \underline{x}_{j+1}, \ldots, \underline{x}_d)| \\
&\leq \sum_{j=1}^{d} |\det(\underline{y}_1, \ldots, \underline{y}_{j-1}, \underline{x}_j - \underline{y}_j, \underline{x}_{j+1}, \ldots, \underline{x}_d)| \\
&= \sum_{j=1}^{\mathbf{n}} |\det(\underline{y}_1, \ldots, \underline{y}_{j-1}, \underline{x}_j - \underline{y}_j, \underline{x}_{j+1}, \ldots, \underline{x}_d)|, \\
&\quad \text{denn } \underline{x}_j - \underline{y}_j = \underline{0} \text{ für } n < j \leq d, \\
&\leq \sum_{j=1}^{n} \|\underline{y}_1\| \cdots \|\underline{y}_{j-1}\| \|\underline{x}_j - \underline{y}_j\| \|\underline{x}_{j+1}\| \cdots \|\underline{x}_{\mathbf{n}}\|, \\
&\quad \text{denn } \|\underline{x}_i\| = 1 \text{ für } n < i \leq d, \\
&< 2^{-q} \sum_{j=1}^{n} 2^{n-j} \prod_{\substack{i=1 \\ i \neq j}}^{n} \|\underline{b}_i\| \\
&\leq 2^{-q}\, \mathrm{dft}(B)\, \mathrm{d}(L)/\lambda_1(B) \sum_{j=0}^{n-1} 2^j \\
&= 2^{-q}\, (2^n - 1)\, \mathrm{dft}(B)\, \mathrm{d}(L)/\lambda_1(B) \\
&< 2^{-q+n}\, \mathrm{dft}(B)\, \mathrm{d}(L)/\lambda_1(B).
\end{aligned}$$

□

Korollar 3.10 (Rang von A) *Sei* $0 < c \leq 1$. *Wenn*

$$q \geq q_4(B, c) = \lceil n + \log(\mathrm{dft}(B)/\lambda_1(B)) - \log c \rceil,$$

dann

$$|\det X - \det Y| < c|\det Y|.$$

Insbesondere hat dann A *den Rang* n.

Beweis:

$$\begin{aligned}
q &\geq \log \mathrm{dft}(B) - \log \lambda_1(B) + n - \log c \\
2^{-q} &\leq \lambda_1(B)/\, \mathrm{dft}(B)\, 2^{-n}\, c \\
c &\geq 2^{-q+n}\, \mathrm{dft}(B)/\lambda_1(B) \\
c\, \mathrm{d}(L) &\geq 2^{-q+n}\, \mathrm{dft}(B)\mathrm{d}(L)/\lambda_1(B).
\end{aligned}$$

Da wegen (3.3) $\mathrm{d}(L) = |\det Y|$ und $q_4(B, c) \geq \lceil -\log \lambda_1(B) \rceil$, folgt mit Lemma 3.9 und Bemerkung 3.8 die Behauptung. □

Bevor wir zum Beweis von Satz 3.5 kommen, zeigen wir noch eine nützliche Folgerung aus der Cramerschen Regel. Für $\underline{b} = (b_1, \ldots, b_n) \in \mathbb{R}^d$ sei $|\underline{b}| := \max\{|b_i| \mid 1 \leq i \leq d\}$. Wir beachten, daß $\|\underline{b}\| \leq \sqrt{d}\,|\underline{b}|$ ist.

Korollar 3.11 (Folgerung aus der Cramerschen Regel) *Sei* $B = (\underline{b}_1, \ldots, \underline{b}_n) \in \mathbb{R}^{d \times n}$, $\underline{b} \in \langle\langle \underline{b}_1, \ldots, \underline{b}_n \rangle\rangle$, $\underline{t} = (t_1, \ldots, t_n) \in \mathbb{R}^n$ *mit* $B\underline{t} = \underline{b}$. *Für* $1 \leq i \leq n$ *gilt*

$$|\underline{t}| \leq \|\underline{b}\| \,\mathrm{dft}(B)/\lambda_1(B).$$

Beweis: Sei $n = d$. Dann gilt mit der Cramerschen Regel für $1 \leq i \leq n$

$$|t_i| = |\det(\underline{b}_1, \ldots, \underline{b}_{i-1}, \underline{b}, \underline{b}_{i+1}, \ldots, \underline{b}_n)/\mathrm{d}(L)|,$$

woraus sich mit einer leichten Folgerung aus der Hadamardschen Ungleichung (siehe Lemma 1.12) die Behauptung ergibt.
Sei $n < d$, $\underline{y}_1, \ldots, \underline{y}_d$ wie in der Konstruktion weiter oben definiert. Dann gilt mit der Cramerschen Regel für $1 \leq i \leq n$

$$|t_i| = |\det(\underline{b}_1, \ldots, \underline{b}_{i-1}, \underline{b}, \underline{b}_{i+1}, \ldots, \underline{b}_n, \underline{y}_{n+1}, \ldots, \underline{y}_d)/\det Y|$$

Wegen der speziellen Wahl der $\underline{y}_i$ für $n < i \leq d$ gilt auch hier mit der obigen Folgerung aus der Hadamardschen Ungleichung

$$\begin{aligned} & |\det(\underline{b}_1, \ldots, \underline{b}_{i-1}, \underline{b}, \underline{b}_{i+1}, \ldots, \underline{b}_n, \underline{y}_{n+1}, \ldots, \underline{y}_d)| \\ \leq\ & \|\underline{b}_1\| \cdots \|\underline{b}_{i-1}\|\, \|\underline{b}\|\, \|\underline{b}_{i+1}\| \cdots \|\underline{b}_n\|. \end{aligned}$$

Mit $|\det Y| = \mathrm{d}(L)$ in (3.3) folgt damit auch für $n < d$ die Behauptung. □

Beweis: (von Satz 3.5) Damit A regulär ist, genügt es nach Korollar 3.10, daß

$$q \geq q_4(B, 1) = \lceil n + \log(\mathrm{dft}(B)/\lambda_1(B)) \rceil.$$

Sei $U = (\underline{u}_1, \ldots, \underline{u}_n) \in \mathbb{Z}^{n \times n}$ regulär mit

$$\|B\underline{u}_i\| = \lambda_i(L), \quad 1 \leq i \leq n.$$

Ein solchen U existiert ([Ca59], VIII.1). Wenden wir Lemma 3.7 an, dann sehen wir, daß für

$$q \geq \lceil \log \lambda_n(U) - \log(c\lambda_1(L)) \rceil \tag{3.4}$$

gilt

$$\|A\underline{u}_i\| \leq (1 + c)\,\|B\underline{u}_i\| = (1 + c)\,\lambda_i(L), \quad 1 \leq i \leq n$$

und deshalb auf Grund von

$$\max\{\|A\underline{u}_1\|,\ldots,\|A\underline{u}_i\|\} \geq \lambda_i(L(A)), \quad 1 \leq i \leq n$$

auch

$$\lambda_i(L(A)) \leq (1+c)\lambda_i(L), \quad 1 \leq i \leq n.$$

Es folgt aus Korollar 3.11, daß

$$\lambda_n(U) \leq \sqrt{n}\,\lambda_n(L)\,\mathrm{dft}(B)/\lambda_1(B).$$

Deshalb schließt $q \geq q_1(B,c)$ auch (3.4) ein. Damit ist der Beweis von Satz 3.5 abgeschlossen. □

Das folgende Lemma benötigen wir für den anschließenden Beweis von Satz 3.6.

Lemma 3.12 (Defekt von A) *Ist $q \geq q_1(B,1/3)$, dann ist*

$$\mathrm{dft}(A)/\lambda_1(A) \leq (3/2)^n\ \mathrm{dft}(B)/\lambda_1(B).$$

Beweis: Aus Korollar 3.10 folgt

$$1/|\det Y| > 2/3 \cdot 1/|\det X|.$$

Außerdem ist $|\det X| \leq \mathrm{d}(L(A))$, denn

$$\begin{aligned} |\det X| &= \|\underline{x}_1^\star\|\cdots\|\underline{x}_n^\star\|\,\|\underline{x}_{n+1}^\star\|\cdots\|\underline{x}_d^\star\| \\ &\leq \|\underline{x}_1^\star\|\cdots\|\underline{x}_n^\star\|\,\underbrace{\|\underline{x}_{n+1}\|\cdots\|\underline{x}_d\|}_{=1} = \mathrm{d}(L(A)). \end{aligned}$$

Somit haben wir

$$3/2 \cdot 1/\mathrm{d}(L) = 3/2 \cdot 1/|\det Y| \geq 1/|\det X| \geq 1/\mathrm{d}(L(A)). \tag{3.5}$$

Durch die Wahl von $c = 1/3$ und $\underline{t}$ als Einheitsvektor folgt aus Lemma 3.7

$$\|\underline{a}_i\| \leq (4/3)\|\underline{b}_i\|, \quad 1 \leq i \leq n.$$

Mit $\|\underline{a}_{i_0}\| = \lambda_1(A)$ ist

$$\begin{aligned} \mathrm{dft}(A)\,\mathrm{d}(L(A))/\lambda_1(A) &= \prod_{\substack{i=1\\ i\neq i_0}}^{n} \|\underline{a}_i\| \\ &\leq (4/3)^{n-1} \prod_{\substack{i=1\\ i\neq i_0}}^{n} \|\underline{b}_i\| \\ &\leq (4/3)^{n-1} \left(\prod_{i=1}^{n} \|\underline{b}_i\|\right)/\lambda_1(B) \\ &= (4/3)^{n-1}\ \mathrm{dft}(B)\,\mathrm{d}(L)/\lambda_1(B). \end{aligned}$$

Mit (3.5) folgt hieraus die Behauptung. □

Beweis: (von Satz 3.6) Zuerst bemerken wir, daß

$$q_2(B,c,s) \geq q_1(B,c).$$

Somit ist A regulär nach Satz 3.5. Durch Anwendung von Lemma 3.7 mit

$$q \geq \log \lambda_n(T) - \log((c/3)\lambda_1(L)) \tag{3.6}$$

und wegen der speziellen Wahl von T haben wir für $1 \leq i \leq n$

$$\|B\underline{t}_i\| < (1/(1-c/3))\|A\underline{t}_i\| \leq (s_i/(1-c/3))\lambda_i(L(A)).$$

Aus Satz 3.5 und (3.6) folgt, daß

$$\lambda_i(L(A)) \leq (1+c/3)\lambda_i(L), \quad 1 \leq i \leq n. \tag{3.7}$$

Da

$$(1+c/3)/(1-c/3) \leq 1+c$$

(dies sieht man durch Umformung zu $0 < c/3 - c^2/3$) ist

$$\|B\underline{t}_i\| < (1+c)\, s_i\, \lambda_i(L)$$

für

$$q \geq \max\{\log \lambda_n(T) - \log((c/3)\lambda_1(L)),\ q_2(B,c,s)\}.$$

Es bleibt $\|\underline{t}_i\|$ abzuschätzen. Es folgt aus Korollar 3.11, daß für $1 \leq i \leq n$

$$\|\underline{t}_i\| \leq \sqrt{n}\, s_i\, \lambda_i(L(A))\ \mathrm{dft}(A)/\lambda_i(A). \tag{3.8}$$

Beachtet man, daß für alle $n \in \mathbb{N}$

$$n \geq n\log(3/2) + 1/2\log n + \log(4/3)$$

ist, so folgt durch Abschätzung von (3.8) mit Lemma 3.12 und (3.7), daß $q_2(B,c,s) \geq \log \lambda_n(T) - \log((c/3)\lambda_1(L))$. Damit ist die Behauptung bewiesen. □

Anwendungen

In diesem Abschnitt wenden wir Satz 3.6 auf den LLL-Algorithmus und den Algorithmus zur Berechnung Korkine-Zolotarev-reduzierter Basen an:

Satz 3.13 *Sei $L \subset \mathbb{R}^d$ ein Gitter und $B = (\underline{b}_1, \ldots, \underline{b}_n) \in \mathbb{R}^{d\times n}$ eine Basis von L. Es gibt einen Algorithmus, der für $0 < c \leq 1$ bei gegebener rationaler Näherung der Basis B der Genauigkeit*

$$p \geq q_0(B,c,2^{(n-1)/2}) = \lceil q_2(B,c,2^{(n-1)/2}) + 1/2\log(dn)\rceil$$

eine unimodulare Matrix $T \in \mathbb{Z}^{n\times n}$ berechnet, so daß

$$\lambda_i(BT) \leq (1+c)\,2^{(n-1)/2}\,\lambda_i(L), \quad 1 \leq i \leq n$$

und

$$\lambda_i(T) \leq (3/2)^{n+1} \sqrt{n}\, 2^{(n-1)/2} \, \mathrm{dft}(B)\, \lambda_i(L)/\lambda_1(B), \quad 1 \leq i \leq n.$$

Mit

$$\beta = \log \lambda_n(B) + \log(1/\lambda_1(L))$$

führt der Algorithmus

$$O\left(n^3 d(n(\log n + \beta) + \log d + \log(1/c))\right)$$

arithmetische Operationen auf ganzen Zahlen der binären Länge

$$O\left(n(n(\log n + \beta) + \log d + \log(1/c))\right)$$

aus.

Satz 3.14 *Sei $L \subset \mathbb{R}^d$ ein Gitter und $B = (\underline{b}_1, \ldots, \underline{b}_n) \in \mathbb{R}^{d\times n}$ eine Basis von L. Es gibt einen Algorithmus, der für $0 < c \leq 1$ bei gegebener rationaler Näherung der Basis B der Genauigkeit*

$$p \geq q_0(B, c, \sqrt{n+3}/2) = \lceil q_2(B, c, \sqrt{n+3}/2) + 1/2 \log(dn) \rceil$$

eine unimodulare Matrix $T \in \mathbb{Z}^{n\times n}$ berechnet, so daß

$$\lambda_i(BT) \leq (1+c)\sqrt{(i+3)/2}\, \lambda_i(L), \quad 1 \leq i \leq n$$

und

$$\lambda_i(T) \leq (3/2)^{n+1} \sqrt{n(i+3)/2}\; \mathrm{dft}(B)\, \lambda_i(L)/\lambda_1(B), \quad 1 \leq i \leq n.$$

Mit

$$\beta = \log \lambda_n(B) + \log(1/\lambda_1(L))$$

führt der Algorithmus

$$dn^{o(n)}(n^{n/2} + \log d + \beta + \log(1/c))$$

arithmetische Operationen auf ganzen Zahlen der binären Länge

$$O(n^2(n(\log n + \beta) + \log(1/c) + \log d))$$

aus.

Beweis: (von Satz 3.13) Sei eine Basis $X = (\underline{x}_1, \ldots, \underline{x}_n) \in \mathbb{Z}^{d\times n}$ eines n-dimensionalen Gitters $L(X)$ gegeben. Mit dem LLL-Algorithmus läßt sich eine Basis $Y \in \mathbb{Z}^{d\times n}$ berechnen derart, daß

$$\|\underline{y}_i\| \leq 2^{(n-1)/2}\, \lambda_i(L(X)), \quad 1 \leq i \leq n.$$

Der Algorithmus führt dabei $O(n^3 d \log \max\{\lambda_n(X), 2\})$ arithmetische Operationen auf ganzen Zahlen der binären Länge $O(n \log \max\{\lambda_n(X), 2\})$ aus. Wählen wir für $0 < c \leq 1$ eine rationale Approximation A von B der Genauigkeit

$p \geq q_0(B, c, 2^{(n-1)/2}) = \lceil q_2(B, c, 2^{(n-1)/2}) + 1/2 \log(dn) \rceil$, dann ist $\|A - B\| < 2^{-q_2(B,c,2^{(n-1)/2})}$. Wenden wir den LLL-Algorithmus auf $X = 2^{p+1}A$ an und führen zusätzlich jede Reduktionsoperation nicht nur an A sondern auch an der $n \times n$-Einheitsmatrix I aus, dann ist nach vollständiger Abarbeitung des Algorithmus das transformierte I die Matrix T mit $XT = Y$ und es gilt mit Satz 3.6

$$\|B\underline{t}_i\| \leq (1 + c)\, 2^{(n-1)/2}\, \lambda_i(L), \quad 1 \leq i \leq n.$$

Wir analysieren nun die Laufzeit. Wir schätzen zunächst $\lambda_n(X)$ ab. Es gilt $\lambda_n(A) < 2\lambda_n(B)$ mit Lemma 3.7. Wegen $X = 2^{p+1}A$ und $p \geq q_0(B, c, 2^{(n-1)/2})$, ist

$$\begin{aligned}\log \lambda_n(X) &= O(n + \log n + \log d + \log(\lambda_n(L)/\lambda_1(L)) \\ &\quad + \log(\lambda_n(B)/\lambda_1(B)) + \log(\mathrm{dft}(B)) + \log(1/c)).\end{aligned}$$

Da

$$\lambda_i(L) \leq \lambda_i(B), \quad 1 \leq i \leq n,$$

und weil mit $\mathrm{d}(L) \geq (\lambda_1(L)/\sqrt{n})^n$ gilt $\mathrm{dft}(B) \leq (\sqrt{n}\, \lambda_n(B)/\lambda_1(L))^n$, folgt

$$\log \lambda_n(X) = O(n(\log n + \beta) + \log d + \log(1/c)),$$

β wie im Satz definiert. Die Anzahl an arithmetischen Operationen, die notwendig sind, um T zu berechnen, ist beschränkt durch die Anzahl an arithmetischen Operationen, um die reduzierte Basis zu finden, nämlich

$$O\left(n^3 d(n(\log n + \beta) + \log d + \log(1/c))\right).$$

Wir müssen jetzt noch zeigen, daß durch die Berechnung von T keine größeren Zahlen auftreten, als wenn wir nur die reduzierte Basis berechnen würden. Sei X' eine Basis von $L(X)$, die während der Reduktion berechnet wird. Es gibt ein unimodulares $T' \in \mathbb{Z}^{n \times n}$ mit $XT' = X'$ und mit Korollar 3.11 folgt, daß

$$\begin{aligned}\log |T'| &\leq \log \lambda_n(X') + \log \mathrm{dft}(2^{p+1}A) - \log \lambda_1(2^{p+1}A) \\ &\leq \log \lambda_n(X') + \log \mathrm{dft}(A),\end{aligned}$$

denn $\mathrm{dft}(2^{p+1}A) = \mathrm{dft}(A)$ und $\lambda_1(2^{p+1}A) \geq 1$. Nun ist

$$\lambda_n(X')^2 \leq d \cdot |X'|^2,$$

also

$$\log \lambda_n(X') \leq 1/2(\log d + \log |X'|^2)$$

und aus der Laufzeitanalyse des LLL-Algorithmus folgt

$$\log |X'| = O(n \log \max\{\lambda_n(X), 2\}),$$

damit ist

$$\log \lambda_n(X') = O(n(n(\log n + \beta) + \log d + \log(1/c))).$$

Da $p \geq q_1(B, 1/3)$, ist mit $\mathrm{d}(L) \geq (\lambda_1(L)/\sqrt{n})^n$ und Lemma 3.12

$$\begin{aligned}\mathrm{dft}(A) &\leq (3/2)^{n+1}\, \mathrm{dft}(B) \\ &\leq (3/2)^{n+1}(\sqrt{n}\lambda_n(B)/\lambda_1(L))^n \\ &= O(n(\log n + \beta)).\end{aligned}$$

Insgesamt ist damit

$$\log|T'| = O(n(n(\log n + \beta) + \log d + \log(1/c))),$$

folglich treten nur Zahlen dieser Größenordnung auf. $\square$

Die Beweisführung für Satz 3.14 ist nahezu identisch der zu Satz 3.13:

Beweis: (von Satz 3.14) Analog zum Anfang vom Beweis von Satz 3.13 lassen sich mit Algorithmus 2.28 berechnen

$$\|\underline{y}_i\| \leq \sqrt{(i+3)/4}\, \lambda_i(L(X)), \quad 1 \leq i \leq n.$$

Es genügt jetzt, $p \geq q_0(B, c, \sqrt{(n+3)/4})$ zu wählen, womit auf Grund von Satz 3.6

$$\|B\underline{t}_i\| \leq (1+c)\sqrt{(i+3)/4}\, \lambda_i(L(X)), \quad 1 \leq i \leq n,$$

ist. Algorithmus 2.28 führt $d\, n^{o(n)}(n^{n/2} + \log\max\{\lambda_n(X), 2\})$ arithmetische Operationen auf ganzen Zahlen der binären Länge $O(n^2(\log n + \log\max\{\lambda_n(X), 2\}))$ aus (Satz 2.39). Durch analoge Argumentation wie im Beweis zu Satz 3.13 haben wir

$$\log\lambda_n(X) = O(n(\log n + \beta) + \log d + \log(1/c)).$$

Daraus ergibt sich für die Anzahl an ausgeführten arithmetischen Operationen

$$d\, n^{o(n)}(n^{n/2} + \log d + \beta + \log(1/c)).$$

Man kann auf dieselbe Weise wie im Beweis zu Satz 3.13 zeigen, daß die zusätzliche Berechnung von T zu keinen größeren Zahlen führt. $\square$

Bemerkung 3.15 Die bewiesenen Aussagen haben nur dann Sinn, wenn die Genauigkeit berechenbar ist. Die Dimension n und $\lambda_1(B)$ stellen kein Problem dar; $\lambda_n(L)$ läßt sich durch $\lambda_n(B)$ nach oben abschätzen. Da

$$\mathrm{dft}(B) = \|\underline{b}_1\| \cdots \|\underline{b}_n\| / \mathrm{d}(L) \leq \|\underline{b}_1\| \cdots \|\underline{b}_n\|\, n^{n/2}/\lambda_1(L)^n,$$

ist nur $\lambda_1(L)$ problematisch. Ist $\lambda_1(L)$ unbekannt, was in der Regel der Fall ist, benötigen wir eine untere Schranke für das erste sukzessive Minimum. In den meisten Anwendungen der Reduktionsalgorithmen ist eine solche Schranke bekannt, so z.B. bei der Reduktion von $\mathbb{Z}$-Basen von Ordnungen, wo $\lambda_1(L) \geq \sqrt{n/2}$ ist (siehe dazu [Ba93], S.43-45). Mit Lemma 2.11 gilt aber auch für jedes Gitter $L \subset \mathbb{R}^d$ mit beliebiger Basis $\underline{b}_1, \ldots, \underline{b}_n$

$$\lambda_1(L) \geq \min\{\|\underline{b}_1^\star\|, \ldots, \|\underline{b}_n^\star\|\}.$$

Da $\underline{b}_1, \ldots, \underline{b}_n$ echt reellwertige Einträge haben können, ist die exakte Berechnung von $\|\underline{b}_1^\star\|, \ldots, \|\underline{b}_n^\star\|$ kritisch. Man könnte aber z.B. $\|\underline{b}_1^\star\|, \ldots, \|\underline{b}_n^\star\|$ mit Gleitkommaarithmetik berechnen und daraus mittels Fehleranalyse untere Schranken ermitteln.

3.3 Kürzester Gittervektor über $\mathbb{R}^d$

Wir wollen versuchen, mit den im letzten Abschnitt gezeigten Aussagen eine Lösung für das SVP in $\mathbb{R}^d$ zu finden. Betrachten wir dazu Satz 3.14. Es gibt sicher ein von L abhängendes c_0 mit $0 < c_0 \leq 1$, so daß bei Genauigkeit $p \geq q_0(B, c_0, \sqrt{n+3}/2)$ nicht nur

$$\|B\underline{t}_1\| \leq (1 + c_0)\lambda_1(L),$$

sondern sogar

$$\|B\underline{t}_1\| = \lambda_1(L)$$

folgt. Dies gilt auf Grund der Diskretheitseigenschaft von Gittern, die besagt, daß es ein $\delta > 0$ gibt, so daß $\|\underline{b}_1 - \underline{b}_2\| > \delta$ für beliebige $\underline{b}_1, \underline{b}_2 \in L$ gilt. Man stellt sich sofort die Frage, ob diejenigen Gittereigenschaften zur Ermittlung einer unteren Schranke für c_0 ausreichen, die bisher in die Genauigkeit eingegangen sind. Genügt auf Grund von Bemerkung 3.15 namentlich die Kenntnis einer unteren Schranke für $\lambda_1(L)$ und einer oberen Schranke für $\lambda_n(L)$, die durch die maximale Länge eines Vektors der Basis gegeben ist? Dies ist nicht der Fall, was das folgende Beispiel verdeutlichen soll.

Beispiel 3.16 Für $0 < \epsilon \leq 1$ sei $L_\epsilon \subset \mathbb{R}^5$ ein Gitter und B eine Basis von L_ϵ. Weiterhin gäbe es eine unimodulare Transformationsmatrix $T \in \mathbb{Z}^{n\times n}$ mit $L_\epsilon = \sum_{i=1}^5 \mathbb{Z}\, B\underline{t}_i$

$$= \mathbb{Z}\begin{pmatrix} 2+2\epsilon \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} + \mathbb{Z}\begin{pmatrix} 0 \\ 2+2\epsilon \\ 0 \\ 0 \\ 0 \end{pmatrix} + \mathbb{Z}\begin{pmatrix} 0 \\ 0 \\ 2+2\epsilon \\ 0 \\ 0 \end{pmatrix} + \mathbb{Z}\begin{pmatrix} 0 \\ 0 \\ 0 \\ 2+2\epsilon \\ 0 \end{pmatrix} + \mathbb{Z}\begin{pmatrix} 1+\epsilon \\ 1+\epsilon \\ 1+\epsilon \\ 1+\epsilon \\ 1 \end{pmatrix}.$$

Ähnlich wie im Beispiel 2.4 sieht man, daß $\lambda_1(L) = 2$ und $\lambda_2(L) = \ldots = \lambda_5(L) = 2 + 2\epsilon$ ist. ϵ hat keinen Einfluß auf $\lambda_1(L) = 2$, und für $0 < \epsilon \leq 1$ ist $\lambda_5(L) \leq 3$. Für beliebig kleines c gibt es $\epsilon = c$, so daß

$$\begin{array}{rcccccl}
\|B\underline{t}_1\| & \leq & 2+2\epsilon & \leq & (1+c)\,2 & = & (1+c)\,\lambda_1(L), \\
\|B\underline{t}_2\| & \leq & 2+2\epsilon & \leq & (1+c)\,\sqrt{5}/2\,(2+2\epsilon) & = & (1+c)\,\sqrt{5}/2\,\lambda_2(L), \\
\|B\underline{t}_3\| & \leq & 2+2\epsilon & \leq & (1+c)\,\sqrt{6}/2\,(2+2\epsilon) & = & (1+c)\,\sqrt{6}/2\,\lambda_3(L), \\
\|B\underline{t}_4\| & \leq & 2+2\epsilon & \leq & (1+c)\,\sqrt{7}/2\,(2+2\epsilon) & = & (1+c)\,\sqrt{7}/2\,\lambda_4(L), \\
\|B\underline{t}_5\| & \leq & (1+\epsilon)\,\sqrt{5} & \leq & (1+c)\,\sqrt{2}\,(2+2\epsilon) & = & (1+c)\,\sqrt{8}/2\,\lambda_5(L),
\end{array}$$

und insbesondere ist

$$\|B\underline{t}_1\| \neq \lambda_1(L) = 2,$$

$$\|B\underline{t}_1\| \leq \|B\underline{t}_2\| \leq \ldots \leq \|B\underline{t}_5\|.$$

Oben genanntes c_0 kann also nicht nur von einer unteren Schranke für $\lambda_1(L)$ und einer oberen Schranke für $\lambda_n(L)$ abhängen. Da B beliebige Basis von L sein kann, läßt sich c_0 also nur über eine weitere Gittereigenschaft bestimmen.

Für $1 \leq i \leq n$ sei

$$\xi_i(L) = \min\{r > 0 \mid \text{Es gibt } i \text{ linear unabhängige Gittervektoren in } L, \text{ deren euklidische Längen paarweise verschieden und kleiner oder gleich } r \text{ sind.}\}$$

Vergleichen wir diese Definition mit Definition 1.16 (sukzessive Minima), dann stellen wir fest, daß $\xi_i(L)$ sich von $\lambda_i(L)$ nur durch den Zusatz "paarweise verschieden" unterscheidet. Während die sukzessiven Minima von gleicher Größe sein können, gilt $\xi_1(L) < \ldots < \xi_n(L)$. Wie man leicht sieht, gilt nun folgender Zusammenhang: Betrachten wir Satz 3.14. Wählen wir $0 < c \leq 1$ in $q_0(B, c, \sqrt{n+3}/2)$, so klein, daß

$$(1 + c)\lambda_1(L) < \xi_2(L),$$

also

$$c < (\xi_2(L) - \lambda_1(L))/\lambda_1(L),$$

dann muß

$$\|B\underline{t}_1\| = \lambda_1(L)$$

sein. $\|B\underline{t}_1\|$ muß das erste sukzessive Minimum annehmen, denn gäbe es $m \in \mathbb{Z}\backslash\{-1, 0, 1\}$ mit $B\underline{t}_1 = m\underline{v}$, $\|\underline{v}\| = \lambda_1(L)$, dann ließe sich $\underline{v}$ nicht erzeugen auf Grund der linearen Unabhängigkeit der $B\underline{t}_2, \ldots, B\underline{t}_n$ von $B\underline{t}_1$. Man mag sich fragen, warum sich eine untere Schranke für c_0 nicht mit Hilfe der $\lambda_i(L)$ beschreiben läßt. Dies ist deshalb nicht möglich, weil die sukzessiven Minima von gleicher Größe sein können, insbesondere $\lambda_i(L) = \lambda_1(L)$ für $1 < i \leq n$ gelten kann.

Wir können c direkt wählen, nähmlich $c = 1/2(\xi_2(L) - \lambda_1(L))/\xi_2(L)$. Dann brauchen wir außerdem nicht zusätzlich zu fordern, daß $c \leq 1$ sein muß. Es ist

$$1/2(\xi_2(L) - \lambda_1(L))/\xi_2(L) < (\xi_2(L) - \lambda_1(L))/\lambda_1(L)$$

und

$$1/2(\xi_2(L) - \lambda_1(L))/\xi_2(L) \leq 1.$$

Wir haben somit folgendes gezeigt:

Korollar 3.17 (Kürzester Gittervektor über $\mathbb{R}^d$) *Sei $L \subset \mathbb{R}^d$ ein Gitter und $B = (\underline{b}_1, \ldots, \underline{b}_n) \in \mathbb{R}^{d \times n}$ eine Basis von L. Es gibt einen Algorithmus, der bei gegebener rationaler Näherung der Basis B der Genauigkeit $p \geq q_0(B, 1/2\,(\xi_2(L) - \lambda_1(L))/\xi_2(L), \sqrt{n+3}/2)$ eine unimodulare Matrix $T \in \mathbb{Z}^{n \times n}$ berechnet, so daß*

$$\|B\underline{t}_1\| = \lambda_1(L).$$

Mit $\beta = \log \lambda_n(B) + \log(1/\lambda_1(L))$ führt der Algorithmus

$$dn^{o(n)}\left(n^{n/2} + \log d + \beta + \log(1/(\xi_2(L) - \lambda_1(L)))\right)$$

arithmetische Operationen auf Zahlen der binären Länge

$$O\left(n^2(n(\log n + \beta) + \log d + \log(1/(\xi_2(L) - \lambda_1(L)))\right)$$

aus.

Bemerkung 3.18 Dieses Ergebnis ist nur von theoretischem Interesse, denn eine untere Schranke für $\xi_2(L) - \lambda_1(L)$ ist in der Regel nicht bekannt!

3.4 Nahe und nächste Gitterpunkte über $\mathbb{R}^d$

Vorgehensweise

Sei $L \subset \mathbb{R}^d$ ein Gitter und $B = (\underline{b}_1, \ldots, \underline{b}_n)$ eine Basis von L, $\underline{b}_0 \in \mathbb{R}^d \backslash \{\underline{0}\}$. Sei $A = (\underline{a}_1, \ldots, \underline{a}_n)$ eine Näherung der Genauigkeit $p(q) = \lceil 1/2 \log(nd) + q \rceil$, $q \in \mathbb{N}$ und $\underline{a}_0$ eine Näherung der gleichen Genauigkeit für $\underline{b}_0$.

Zunächst wollen wir uns fragen, ob unter der Voraussetzung, daß die Einträge von A und $\underline{a}_0$ rational sind und A ein n-dimensionales Gitter $L(A)$ erzeugt, sich analog zu Abschnitt 3.2 ein $\underline{t} \in \mathbb{Z}^n$ berechnen läßt mit $\|A\underline{t} - \underline{a}_0\| = \mu(\underline{a}_0, L(A))$. Die Frage läßt sich bejahen. Ein nächster Gitterpunkt von $\underline{a}_0$ in $L(A)$ bei gegebener Korkine-Zolotarev-reduzierter Basis $\tilde{A}$ von $L(A)$ wird mit Algorithmus 2.52 über die Berechnung eines Transformationsvektors $\underline{u} \in \mathbb{Z}^n$ mit $\|\tilde{A}\underline{u} - \underline{a}_0\| = \mu(\underline{a}_0, L(A))$ ermittelt. In Abschnitt 3.2 haben wir gesehen, daß sich $U \in \mathbb{Z}^{n \times n}$ mit $AU = \tilde{A}$ leicht bestimmen läßt, damit kann auch $\underline{t} = U\underline{u}$ berechnet werden, so daß $A\underline{t}$ ein nächster Gitterpunkt von $\underline{a}_0$ ist. Mit dem LLL-Algorithmus läßt sich, wenn $n = d$ ist, durch die Reduktion der aus A und $\underline{a}_0$ konstruierten Basis schon aus der dazugehörigen Transformationsmatrix ein $\underline{t} \in \mathbb{Z}^n$ mit $\|A\underline{t} - \underline{a}_0\| \leq 2^n \mu(\underline{a}_0, L(A))$ gewinnen, wie wir noch sehen werden.

Deshalb liegt es nicht fern, mit der gleichen Methode wie in Abschnitt 3.2 herzuleiten, von welcher Genauigkeit A und $\underline{a}_0$ sein müssen, damit $B\underline{t}$ ein möglichst naher Gitterpunkt von $\underline{b}_0$ ist. Wir werden sehen, daß der Abstand von $\underline{b}_0$ zu einem seiner nächsten Gitterpunkte als Kehrwert in die Präzision eingeht. Daher müssen wir voraussetzen, daß $\underline{b}_0 \notin L$ ist. Zu Beginn wird untersucht werden, wie groß die Genauigkeit von A und $\underline{a}_0$ sein muß, damit A den Rang n hat und $\mu(\underline{a}_0, L(A))$ sich nur geringfügig von $\mu(\underline{b}_0, L)$ unterscheidet; beides sind notwendige Aussagen um beweisen zu können, bei welcher Präzision $B\underline{t}$ ein Gitterpunkt ausreichender Güte ist, wenn $A\underline{t}$ ein nächster bzw. naher Gitterpunkt von $\underline{a}_0$ ist.

Die bewiesenen Ergebnisse finden schließlich Anwendung im LLL-Algorithmus und dem Algorithmus von Kannan. Es wird hierbei insbesondere untersucht werden, in welcher Zeit sich die Transformationsvektoren berechnen lassen.

Hinreichende Genauigkeit

Seien A bzw. $\underline{a}_0$ Näherungen der Genauigkeit $p(q) = \lceil 1/2 \log(nd) + q \rceil$, $q \in \mathbb{N}$ für B bzw. $\underline{b}_0$. Es gilt dann $\|A - B\| < 2^{-q}$. Wir setzen $\check{B} := (\underline{b}_1, \ldots, \underline{b}_n, \underline{b}_0)$.

Satz 3.19 *Sei $\underline{b}_0 \notin L$, $0 < c \leq 1$ und*

$$\begin{aligned} q \;\geq\; q_5(B, \underline{b}_0, c) \;&=\; \lceil n + \log(1 + \sqrt{n}\,(\|\underline{b}_0\| + \lambda_{n+1}(\check{B}))\,\mathrm{dft}(B)/\lambda_1(B)) \\ &\quad - \log(c \cdot \mu(\underline{b}_0, L)/\lambda_{n+1}(\check{B}) \cdot 1/2 \cdot \lambda_1(L))\rceil. \end{aligned}$$

Dann hat A den Rang n und

$$\mu(\underline{a}_0, L(A)) \leq (1+c)\mu(\underline{b}_0, L). \tag{3.9}$$

Um den nächsten Satz formulieren zu können, nehmen wir an, A habe den Rang n und $\underline{t} \in \mathbb{Z}^n$ sei derart, daß $\|\underline{a}_0 - A\underline{t}\| \leq \sigma\mu(\underline{a}_0, L(A))$, wobei $\sigma \geq 1$.

Satz 3.20 *Sei* $\underline{b}_0 \notin L$, $0 < c \leq 1$ *und*

$$\begin{aligned} q \geq q_6(B, \underline{b}_0, c, \sigma) &= \lceil n + 1 + \log(1 + \sqrt{n}\,\mathrm{dft}(B)/\lambda_1(B)(\|\underline{b}_0\| + \sigma\lambda_{n+1}(\check{B}))) \\ &\quad - \log(c/3 \cdot \mu(\underline{b}_0, L)/\lambda_{n+1}(\check{B}) \cdot 1/2 \cdot \lambda_1(L))\rceil, \end{aligned}$$

dann ist

$$\|\underline{b}_0 - B\underline{t}\| \leq (1+c)\,\sigma\,\mu(\underline{b}_0, L).$$

Wir zeigen jetzt eine Hilfsaussage, bevor wird Satz 3.19 beweisen.

Lemma 3.21 *Sei* $0 < c \leq 1$*, dann ist*

$$q_5(B, \underline{b}_0, c) \geq q_1(B, c) = \lceil n + \log(\lambda_n(L)\,\mathrm{dft}(B)/\lambda_1(B)) - \log(c \cdot \lambda_1(L))\rceil.$$

Beweis: Es ist

$$\log(1 + \sqrt{n}\,(\|\underline{b}_0\| + \lambda_{n+1}(\check{B}))\,\mathrm{dft}(B)/\lambda_1(B)) \geq \log(\lambda_n(L)\,\mathrm{dft}(B)/\lambda_1(B)).$$

Außerdem haben wir

$$-\log(c \cdot \mu(\underline{b}_0, L)/\lambda_{n+1}(\check{B}) \cdot 1/2 \cdot \lambda_1(L)) \geq -\log(c \cdot \lambda_1(L)),$$

denn $\mu(\underline{b}_0, L) \leq \|\underline{b}_0\| \leq \lambda_{n+1}(\check{B})$, weil $\underline{0} \in L$ und damit $\mu(\underline{b}_0, L) \leq \|\underline{b}_0 - \underline{0}\|$. □

Beweis: (von Satz 3.19) Mit Lemma 3.21 haben wir $q_5(B, \underline{b}_0, c) \geq q_1(B, c)$ und somit sind $\underline{a}_1, \ldots, \underline{a}_n$ mit Satz 3.5 linear unabhängig. Wir wollen jetzt (3.9) zeigen. Sei $\underline{s} \in \mathbb{Z}^n$ derart, daß $\mu(\underline{b}_0, L) = \|\underline{b}_0 - B\underline{s}\|$, dann

$$\begin{aligned} \|\underline{a}_0 - A\underline{s}\| - \|\underline{b}_0 - B\underline{s}\| &\leq \|\underline{b}_0 - B\underline{s} - \underline{a}_0 + A\underline{s}\| \\ &\leq \|\underline{b}_0 - \underline{a}_0\| + \|B\underline{s} - A\underline{s}\| \\ &\leq 2^{-q} + 2^{-q}\|\underline{s}\| \\ &= 2^{-q}\,(1 + \|\underline{s}\|). \end{aligned}$$

Wir wollen haben $2^{-q}\,(1 + \|\underline{s}\|) \leq c\,\|\underline{b}_0 - B\underline{s}\| = c \cdot \mu(\underline{b}_0, L)$, was mit $q \geq \log(1 + \|\underline{s}\|) - \log(c\,\mu(\underline{b}_0, L))$ gilt. Nun ist $\|\underline{s}\| \leq \|B\underline{s}\|\sqrt{n}\,\mathrm{dft}(B)/\lambda_1(B)$ (Korollar 3.11) und $\|B\underline{s}\| \leq \|\underline{b}_0\| + \mu(\underline{b}_0, L)$, folglich

$$\log(1 + \|\underline{s}\|) \leq \log(1 + (\|\underline{b}_0\| + \mu(\underline{b}_0, L))\,\sqrt{n}\;\mathrm{dft}(B)/\lambda_1(B)).$$

Es gilt $\mu(\underline{b}_0, L) \leq \|\underline{b}_0\| \leq \lambda_{n+1}(\check{B})$ und damit ist

$$\log(1 + \|\underline{s}\|) \leq \log(1 + \sqrt{n}\,(\|\underline{b}_0\| + \lambda_{n+1}(\check{B}))\,\mathrm{dft}(B)/\lambda_1(B)).$$

Mit $\lambda_1(L) \leq \lambda_{n+1}(\check{B})$ gilt

$$-\log \mu(\underline{b}_0, L) \leq -\log(\mu(\underline{b}_0, L)/\lambda_{n+1}(\check{B}) \cdot 1/2 \cdot \lambda_1(L)).$$

Insgesamt ist somit

$$q_5(B, \underline{b}_0, c) \geq \log(1 + \|\underline{s}\|) - \log(c \cdot \mu(\underline{b}_0, L)),$$

woraus (3.9) folgt. □

Lemma 3.22 *Sei $0 < c \leq 1$. Ist $q \geq q_5(B, \underline{b}_0, c/3)$, dann gilt*

$$\mathrm{dft}(A)/\lambda_1(A) \leq (3/2)^n\ \mathrm{dft}(B)/\lambda_1(B), \tag{3.10}$$

$$\mu(\underline{a}_0, L(A)) \leq (1 + c/3)\mu(\underline{b}_0, L), \tag{3.11}$$

$$\|\underline{a}_0\| \leq (1 + c/3)\|\underline{b}_0\|. \tag{3.12}$$

Beweis: Aus Lemma 3.12 folgt mit $q_5(B, \underline{b}_0, c/3) \geq q_1(B, 1/3)$ (Lemma 3.21) Ungleichung (3.10). Ungleichung (3.11) folgt aus Satz 3.19. Da $\mu(\underline{b}_0, L) \leq \|\underline{b}_0\|$ und $1/2\,\lambda_1(L) \leq \lambda_{n+1}(\check{B})$, ist $q_5(B, \underline{b}_0, c/3) \geq -\log\|\underline{b}_0\|$, damit gilt $\|\underline{a}_0 - \underline{b}_0\| \leq c/3\|\underline{b}_0\|$, folglich auch (3.12). □

Beweis: (von Satz 3.20) Mit $q \geq \lceil\log(1 + \|\underline{t}\|) - \log(c/3 \cdot \mu(\underline{b}_0, L))\rceil$ folgt die erste, durch die spezielle Wahl von $\underline{t}$ die zweite der folgenden Abschätzungen

$$\begin{aligned} \|\underline{b}_0 - B\underline{t}\| &\leq 1/(1 - c/3) \cdot \|\underline{a}_0 - A\underline{t}\| \\ &\leq \sigma \cdot 1/(1 - c/3) \cdot \mu(\underline{a}_0, L(A)). \end{aligned}$$

Ist $q \geq q_5(B, \underline{b}_0, c/3)$, dann ist

$$\mu(\underline{a}_0, L(A)) \leq (1 + c/3)\mu(\underline{b}_0, L).$$

Da $(1 + c/3)/(1 - c/3) \leq (1 + c)$, haben wir

$$\|\underline{b}_0 - B\underline{t}\| \leq (1 + c)\sigma\mu(\underline{b}_0, L)$$

für

$$q \geq \max\{\log(1 + \|\underline{t}\|) - \log(c/3 \cdot \mu(\underline{b}_0, L)), q_5(B, \underline{b}_0, c/3)\}.$$

Mit Korollar 3.11 gilt

$$\log(1 + \|\underline{t}\|) \leq \log(1 + \sqrt{n}\|A\underline{t}\|\,\mathrm{dft}(A)/\lambda_1(A)).$$

Weil $\|A\underline{t}\| - \|\underline{a}_0\| \leq \|\underline{a}_0 - A\underline{t}\| \leq \sigma\mu(\underline{a}_0, L(A))$, ist

$$\|A\underline{t}\| \leq \|\underline{a}_0\| + \sigma\mu(\underline{a}_0, L(A)).$$

Somit gilt

$$\log(1+\|\underline{t}\|) \leq \log\left(1+\sqrt{n}\,\mathrm{dft}(A)/\lambda_1(A)\,(\|\underline{a}_0\| + \sigma\mu(\underline{a}_0, L(A)))\right).$$

Mit $q \geq q_5(B, \underline{b}_0, c/3)$ folgt durch Lemma 3.22, daß $\log(1+\|\underline{t}\|)$

$$\begin{aligned} &\leq \log(1+\sqrt{n}(3/2)^n\ \mathrm{dft}(B)/\lambda_1(B)\cdot((1+c/3)\|\underline{b}_0\| + \sigma(1+c/3)\mu(\underline{b}_0, L))) \\ &\leq (n+1)\log(3/2) + \log\left(1+\sqrt{n}\,\mathrm{dft}(B)/\lambda_1(B)\,(\|\underline{b}_0\| + \sigma\mu(\underline{b}_0, L))\right). \end{aligned}$$

Mit $\mu(\underline{b}_0, L) \leq \|\underline{b}_0\| \leq \lambda_{n+1}(\check{B})$ und $(n+1)\log(3/2) \leq n+1$ folgt, daß

$$q_6(B, \underline{b}_0, c, \sigma) \geq \log(1+\|\underline{t}\|) - \log(c/3\cdot\mu(\underline{b}_0, L)).$$

Da außerdem

$$q_6(B, \underline{b}_0, c, \sigma) \geq q_5(B, \underline{b}_0, c/3),$$

folgt die Behauptung sofort. □

Anwendungen

Satz 3.23 *Sei $L \subset \mathbb{R}^d$ ein Gitter und $B = (\underline{b}_1, \ldots, \underline{b}_n) \in \mathbb{R}^{d\times n}$ eine Basis von L, $\underline{b}_0 \in \mathbb{R}^d$ und $\underline{b}_0 \notin L$, $\check{B} := (\underline{b}_1, \ldots, \underline{b}_n, \underline{b}_0)$. Es gibt einen Algorithmus, der für $0 < c \leq 1$ bei gegebenen rationalen Näherungen der Basis B und des Punktes $\underline{b}_0$ der Genauigkeit*

$$q_7(B, \underline{b}_0, c, 1) = \lceil q_6(B, \underline{b}_0, c, 1) + 1/2\ \log(nd)\rceil$$

ein $\underline{t} \in \mathbb{Z}^n$ berechnet mit

$$\|\underline{b}_0 - B\underline{t}\| \leq (1+c)\mu(\underline{b}_0, L).$$

Mit $\delta = n\left(\log(n\lambda_{n+1}(\check{B})/\lambda_1(L)) + \log(\lambda_{n+1}(\check{B})/\mu(\underline{b}_0, L)) + \log d\right)$ werden dabei insgesamt

$$dn^{3/2n+O(1)} + n^{o(n)}(\delta + \log(1/c))$$

arithmetische Operationen auf ganzen Zahlen der binären Länge

$$O\left(n^2(\delta + \log(1/c))\right)$$

ausgeführt.

Weil wir nicht zeigen können, von welcher Genauigkeit $p(q)$ die Näherungen A bzw. $\underline{a}_0$ von B bzw. $\underline{b}_0$ sein müssen, damit im Falle daß $n \neq d$ ist für die Projektionen $\hat{\underline{a}}_0$ von $\underline{a}_0$ in $\langle\langle \underline{a}_1, \ldots, \underline{a}_n\rangle\rangle$ und $\hat{\underline{b}}_0$ von $\underline{b}_0$ in $\langle\langle \underline{b}_1, \ldots, \underline{b}_n\rangle\rangle$ gilt $\|\hat{\underline{a}}_0 - \hat{\underline{b}}_0\| < 2^{-q}$, müssen wir im nächsten Satz $n = d$ voraussetzen.

Satz 3.24 *Sei $L \subset \mathbb{R}^n$ ein Gitter und $B = (\underline{b}_1, \ldots, \underline{b}_n) \in \mathbb{R}^{n\times n}$ eine Basis von L, $\underline{b}_0 \in \mathbb{R}^n$ und $\underline{b}_0 \notin L, \check{B} := (\underline{b}_1, \ldots, \underline{b}_n, \underline{b}_0)$. Es gibt einen Algorithmus, der für $0 < c \leq 1$ bei gegebenen rationalen Näherungen der Basis B und des Punktes $\underline{b}_0$ der Genauigkeit*

$$q_7(B, \underline{b}_0, c, 2^n) = \lceil q_6(B, \underline{b}_0, c, 2^n) + \log n \rceil$$

ein $\underline{t} \in \mathbb{Z}^n$ berechnet mit

$$\|\underline{b}_0 - B\underline{t}\| \leq (1+c)\, 2^n \mu(\underline{b}_0, L).$$

Mit $\theta = n(\log(n\lambda_{n+1}(\check{B})/\lambda_1(L)) + \log(\lambda_{n+1}(\check{B})/\mu(\underline{b}_0, L)))$ werden dabei insgesamt

$$O\left(n^4(\theta + \log(1/c))\right)$$

arithmetische Operationen auf ganzen Zahlen der binären Länge

$$O\left(n(\theta + \log(1/c))\right)$$

ausgeführt.

Beweis: (von Satz 3.23) Es ist $q_7(B, \underline{b}_0, c, 1) = \lceil q_6(B, \underline{b}_0, c, 1) + 1/2 \log(nd) \rceil$. Wähle c mit $0 < c \leq 1$ und eine rationale Näherung A bzw. $\underline{a}_0$ der Genauigkeit $p \geq q_7(B, \underline{b}_0, c, 1)$ für B bzw. $\underline{b}_0$. Dann ist $\|A - B\| < 2^{-q_6(B,\underline{b}_0,c,1)}$ und $\|\underline{a}_0 - \underline{b}_0\| < 2^{-q_6(B,\underline{b}_0,c,1)}$.

Sei nun $X = 2^{p+1}A$ und $\underline{x}_0 = 2^{p+1}\underline{a}_0$, dann ist $X = (\underline{x}_1, \ldots, \underline{x}_n)$ Basis eines Gitters $L(X) \subseteq \mathbb{Z}^d$ und $\underline{x}_0 \in \mathbb{Z}^d$. Berechnet man mit Algorithmus 2.28 eine Korkine-Zolotarev-reduzierte Basis $Y = XU$ von $L(X)$ und führt man alle Transformationen an X auch an der Einheitsmatrix $I \in \mathbb{Z}^{n\times n}$ aus, dann hat man am Ende auch die Matrix U. Anschließend läßt sich ein nächster Gitterpunkt von $\underline{x}_0$ in $L(X)$ mit Algorithmus 2.52 berechnen, wobei man als Eingabebasis das neue Y nimmt. Um $\underline{u} \in \mathbb{Z}^n$, den speziell zu einem nächsten Gitterpunkt gehörenden Transformationsvektor, zu erhalten, speichere man nicht nur N bzw. C, sondern auch die Menge aller berechneten Transformationsvektoren aller Elemente aus N bzw. C ab. Dann gilt mit Satz 3.20 für $U\underline{u} = \underline{t}$

$$\|B\underline{t} - \underline{b}_0\| \leq (1+c)\mu(\underline{b}_0, L).$$

Wir bestimmen nun die Größenordnung der Laufzeit, die notwendig ist, um $\underline{t}$ zu berechnen. Zunächst leiten wir eine obere Schranke für die maximale Anzahl an durchgeführten arithmetischen Operationen her.

Sei $\check{X} := (\check{\underline{x}}_1, \ldots, \check{\underline{x}}_n, \check{\underline{x}}_{n+1}) := (\underline{x}_1, \ldots, \underline{x}_n, \underline{x}_0)$ und $\check{A} := (\check{\underline{a}}_1, \ldots, \check{\underline{a}}_n, \check{\underline{a}}_{n+1}) := (\underline{a}_1, \ldots, \underline{a}_n, \underline{a}_0)$. Für die Berechnung von $Y = XU$ und eines nächsten Gitterpunktes $\underline{y} = XU\underline{u}$ sind mit Korollar 2.58 $d(n^{3/2+O(1)} + n^{o(n)} \log \max\{\lambda_{n+1}(\check{X}), 2\})$ arithmetische Operationen auf ganzen Zahlen der binären Länge $O(n^2(\log n + \log \max\{\lambda_{n+1}(\check{X}), 2\}))$ notwendig. Die Anzahl an arithmetischen Operationen zur

Berechnung von U ist beschränkt durch die Anzahl an arithmetischen Operationen zur Berechnung von Y. Die Berechnung von $\underline{u}$ erfordert keine zusätzliche Handhabung. Die Multiplikation von U mit $\underline{u}$ erfordert $O(n^2)$ arithmetische Operationen.

Wir wollen jetzt $\lambda_{n+1}(\check{X})$ abschätzen. Es ist $\log \lambda_{n+1}(\check{X}) \leq \log(2^{p+1}\lambda_{n+1}(\check{A}))$. Da $p \geq q_7(B, \underline{b}_0, c, 1) \geq q_5(B, \underline{b}_0, c/3) \geq q_1(B, 1/3)$ sind $\|\underline{a}_i\| \leq 4/3\,\|\underline{b}_i\|$ für $1 \leq i \leq n$, wie wir im Beweis zu Lemma 3.12 gesehen haben. Weil $p \geq \log(1/c) + \log(1/\mu(\underline{b}_0, L)) \geq \log(1/c) + \log(1/\|\underline{b}_0\|)$, ist auch $\|\underline{a}_0 - \underline{b}_0\| < c\,\|\underline{b}_0\|$ und folglich $\|\underline{a}_0\| \leq 4/3\,\|\underline{b}_0\|$. Für $\lambda_{n+1}(\check{A})$ folgt damit, daß

$$\lambda_{n+1}(\check{A}) \leq 4/3\,\lambda_{n+1}(\check{B}).$$

Wir haben dadurch

$$\log \lambda_{n+1}(\check{X}) \leq \log(2^{p+1}\,4/3\,\lambda_{n+1}(\check{B}))$$

und daraus auf Grund von $p \geq q_7(B, \underline{b}_0, c, 1)$

$$\begin{aligned}\log \lambda_{n+1}(\check{X}) &= O(\log(1 + \sqrt{n}\,\mathrm{dft}(B)/\lambda_1(B)\,(\|\underline{b}_0\| + \lambda_{n+1}(\check{B})))\\ &\quad + \log(1/c \cdot \lambda_{n+1}(\check{B})/\mu(\underline{b}_0, L) \cdot 1/\lambda_1(L))\\ &\quad + n + \log n + \log d + \log \lambda_{n+1}(\check{B})).\end{aligned}$$

Wir vereinfachen jetzt diese Abschätzung. Aus (1.13), folgt $\mathrm{d}(L) \geq (\lambda_1(L)\,/\sqrt{n})^n$, damit ist

$$\mathrm{dft}(B) \leq \left(\sqrt{n}\,\lambda_n(B)/\lambda_1(L)\right)^n.$$

Außerdem ist

$$(\|\underline{b}_0\| + \lambda_{n+1}(\check{B}))/\lambda_1(B) \leq 2\lambda_{n+1}(\check{B})/\lambda_1(L).$$

Insgesamt folgt, daß $\log \lambda_{n+1}(\check{X})$

$$= O(n \log(n\lambda_{n+1}(\check{B})/\lambda_1(L)) + \log(1/c) + \log(\lambda_{n+1}(\check{B})/\mu(\underline{b}_0, L)) + \log d).$$

Damit ist die Aussage über die Anzahl an ausgeführten arithmetischen Operationen gezeigt.

Sei X' eine Basis von $L(X)$, die während der Reduktion berechnet wird. Dann gibt es eine unimodulare Matrix $U' \in \mathbb{Z}^{n \times n}$ mit $XU' = X'$. Um die maximale Länge aller auftretenden Zahlen zu erhalten, genügt es nicht, $\lambda_{n+1}(\check{X})$ abzuschätzen. Wir müssen zusätzlich $|U'|$ untersuchen, denn wir führen jede Transformation auch an der Einheitsmatrix aus, und dadurch könnten größere Zahlen auftreten, als wenn nur die Korkine-Zolotarev-reduzierte Basis von X berechnet würde. Weil die auftretenden Zahlen eine andere Größenordnung haben als in Satz 3.14, muß dies hier erneut gezeigt werden. Aus Korollar 3.11 folgt, daß

$$|U'| \leq \lambda_n(X')\,\mathrm{dft}(X)/\lambda_1(X)$$

und daraus

$$\log |U'| \leq \log \lambda_n(X') + \log \mathrm{dft}(A),$$

denn $\mathrm{dft}(2^{p+1}A) = \mathrm{dft}(A)$ und $\lambda_1(2^{p+1}A) \geq 1$. Wir wollen jetzt die einzelnen Summanden weiter abschätzen. Es ist

$$\log \lambda_n(X') \leq 1/2(\log d + \log |X'|^2),$$

$$\log |X'| = O(n^2(\log n + \log \max\{2, \lambda_n(X)\}),$$

wie aus der Laufzeitanalyse des Algorithmus zur Berechnung einer Korkine-Zolotarev-reduzierten Basis folgt. Weil $p \geq q_1(B, 1/3)$, ist weiterhin mit $\mathrm{d}(L(A)) \geq (\lambda_1(L(A))/\sqrt{n})^n$

$$\mathrm{dft}(A) \leq (3/2)^{n+1}(\sqrt{n}\,\lambda_n(B)/\lambda_1(L))^n.$$

Insgesamt ist also $\log |U'|$

$$\begin{aligned} &= O\left(n^2\left(n \log(n\lambda_{n+1}(\check{B})/\lambda_1(L)) + \log(\lambda_{n+1}(\check{B})/\mu(\underline{b}_0, L)) + \log d + \log c\right)\right), \\ &= O\left(n^2(\log n + \log \max\{2, \lambda_{n+1}(\check{X})\}\right), \end{aligned}$$

womit die Berechnung von U zu keinen größeren Zahlen führt. Auch die Berechnung von $\underline{t} = U\underline{u}$ ändert nichts an dieser Größenordnung, da dabei höchstens doppelt so große Zahlen auftreten. □

Beweis: (von Satz 3.24) Sei $q_7(B, \underline{b}_0, c, 2^n) = \lceil q_6(B, \underline{b}_0, c, 2^n) + \log n \rceil$. Es ist $\|A - B\| < 2^{-q_6(B,\underline{b}_0,c,2^n)}$ und $\|\underline{a}_0 - \underline{b}_0\| < 2^{-q_6(B,\underline{b}_0,c,2^n)}$, wenn c mit $0 < c \leq 1$ ist und $A = (\underline{a}_1, \ldots, \underline{a}_n)$ bzw. $\underline{a}_0$ rationale Näherungen der Genauigkeit $p \geq q_7(B, \underline{b}_0, c, 2^n)$ für B bzw. $\underline{b}_0$ sind. Sei

$$\check{A} = \left(\begin{pmatrix} 0 \\ \underline{a}_1 \end{pmatrix}, \ldots, \begin{pmatrix} 0 \\ \underline{a}_n \end{pmatrix}, \begin{pmatrix} \lambda_n(A)^2 \\ \underline{a}_0 \end{pmatrix}\right),$$

außerdem $X = (\underline{x}_1, \ldots, \underline{x}_n) = 2^{p+1}A$ und $\underline{x}_0 = 2^{p+1}\underline{a}_0$. Weiterhin sei $\check{X} = (\check{\underline{x}}_1, \ldots, \check{\underline{x}}_{n+1}) = 2^{p+1}\check{A}$. $\check{\underline{x}}_1, \ldots, \check{\underline{x}}_{n+1}$ sind linear unabhängig und $\check{\underline{x}}_i \in \mathbb{Z}^{n+1}$ für $1 \leq i \leq n+1$.

Berechnen wir die LLL-Reduktion $\check{Y} = \check{X}U$ von $\check{X}$ und führen wir alle Reduktionsoperationen nicht nur an der Basis, sondern auch an der Einheitsmatrix $I \in \mathbb{Z}^{n+1 \times n+1}$ aus, dann ist am Ende der Berechnung das transformierte I gerade U. Auf Grund von Satz 2.51 ist

$$\begin{pmatrix} \lambda_n(X)^2 \\ \underline{x}_0 - \underline{w} \end{pmatrix} = \left(\begin{pmatrix} 0 \\ \underline{x}_1 \end{pmatrix}, \ldots, \begin{pmatrix} 0 \\ \underline{x}_n \end{pmatrix}, \begin{pmatrix} \lambda_n(X)^2 \\ \underline{x}_0 \end{pmatrix}\right) \cdot \underline{u}_{n+1}$$

und $\|\underline{w} - \underline{x}_0\| \leq 2^n \mu(\underline{x}_0, L(X))$, $\underline{w} \in L(X)$. Es ist $u_{n+1\,n+1} = 1$ und mit $\underline{t} = (-u_{n+1\,1}, \ldots, -u_{n+1\,n})^\top$ gilt $\underline{w} = X\underline{t}$. Mit Satz 3.20 haben wir damit

$$\|B\underline{t} - \underline{b}_0\| \leq (1 + c)\, 2^n\, \mu(\underline{b}_0, L).$$

Wir wollen nun die Laufzeit analysieren. Die Anzahl an arithmetischen Operationen, die notwendig sind, um U zu berechnen, ist $O(n^4 \log\{\lambda_{n+1}(\check{X}), 2\})$, denn sie ist beschränkt durch die Anzahl an arithmetischen Operationen zur Berechnung der reduzierten Basis $\check{Y}$. Genauso wie im Beweis zu Satz 3.23 kann man beweisen, daß $\log \lambda_{n+1}(\check{X})$

$$= O\left(n \log(n\lambda_{n+1}(\check{B})/\lambda_1(L)) + \log(1/c) + \log(\lambda_{n+1}(\check{B})/\mu(\underline{b}_0, L)\right),$$

womit die Behauptung über die maximale Anzahl an ausgeführten arithmetischen Operationen folgt. Analog zum Beweis von Satz 3.14 kann gezeigt werden, daß trotz der Berechnung von U nur Zahlen der binären Länge $O(n(\theta + \log(1/c)))$ auftreten. Hierbei muß nur berücksichtigt werden, daß

$$\mathrm{dft}(\check{A}) = \prod_{i=1}^{n+1}(\|\underline{\check{a}}_i\|/\|\underline{\check{a}}_i^\star\|) = \left(\prod_{i=1}^{n+1}\|\underline{\check{a}}_i\|\right)/(\mathrm{d}(L(A))\lambda_n(A)) = \mathrm{dft}(A),$$

denn $\|\underline{\check{a}}_{n+1}^\star\| = \lambda_n(A)$, außerdem $\lambda_1(\check{A}) \geq 1$. Damit gilt auch mit $p \geq q_1(B, 1/3)$

$$\mathrm{dft}(\check{A}) = \mathrm{dft}(A) \leq (3/2)^n(\sqrt{n}\lambda_{n+1}(\check{B})/\lambda_1(L))^n. \qquad \square$$

Bemerkung 3.25 Die zuletzt gezeigten Sätze sind nur dann praktisch umsetzbar, wenn durch die Problemstellung eine untere Schranke für $\mu(\underline{b}_0, L)$ bekannt ist. Inwieweit die anderen Gittereigenschaften, die in die Genauigkeit eingehen, bekannt sind, wenn das Gitter durch eine Basis gegeben ist, wurde schon in Bemerkung 3.15 untersucht.

Nächste Gitterpunkte über $\mathbb{R}^d$

Wir wollen uns fragen, wie die Genauigkeit von A und $\underline{a}_0$ zu wählen ist, damit ein $\underline{t} \in \mathbb{Z}^n$ derart berechnet werden kann, daß $B\underline{t}$ ein nächster Gitterpunkt von $\underline{b}_0$ in L ist. Ganz analog zu Kapitel 3.3 gibt es in Satz 3.23 ein c_0 mit $0 < c_0 \leq 1$, so daß bei Genauigkeit $p \geq q_7(B, \underline{b}_0, c_0, 1)$ sogar $\|B\underline{t} - \underline{b}_0\| = \mu(\underline{b}_0, L)$ gilt. Die Existenz eines solchen c_0 ist auch hier darauf zurückzuführen, daß es eine Kugel um $\underline{b}_0$ gibt, die allein die nächsten Gitterpunkte von $\underline{b}_0$ in L enthält. Wie in Kapitel 3.3 hängt auch hier c_0 von einer neuen Gittereigenschaft ab, was das folgende Beispiel verdeutlichen soll (siehe auch Beispiel 3.16).

Beispiel 3.26 Für $0 < \epsilon \leq 1$ sei $L_\epsilon = \mathbb{Z}\binom{2}{0} + \mathbb{Z}\binom{0}{2+\epsilon} \subset \mathbb{R}^2$ ein Gitter. Mit $\underline{b}_0 = (1,1)$ ist $\mu(\underline{b}_0, L) = \sqrt{2}$. Nehmen wir an, L_ϵ sei durch eine von $\left(\binom{2}{0}, \binom{0}{2+\epsilon}\right)$ verschiedenen Basis gegeben, d.h. hauptsächlich sei nicht bekannt, daß $\lambda_2(L) = 2 + \epsilon$ ist. Dann ließe sich c_0 nicht durch die Kenntnis von $\lambda_1(L)$ und $\mu(\underline{b}_0, L)$ bestimmen. Denn für beliebig kleines c gibt es $\epsilon = c$ (ϵ hat keinen Einfluß auf $\lambda_1(L)$ und $\mu(\underline{b}_0, L)$) mit

$$\left\|\binom{0}{2+\epsilon} - \binom{1}{1}\right\| = \sqrt{1 + (1+\epsilon)^2} \leq (1+c)\sqrt{2},$$

also insbesondere ist $\left\|\binom{0}{2+\epsilon} - \underline{b}_0\right\| \neq \mu(\underline{b}_0, L)$.

Sei
$$\nu(\underline{b}_0, L) = \min\{\|\underline{b} - \underline{b}_0\| \mid \underline{b} \in L \text{ und } \|\underline{b} - \underline{b}_0\| > \mu(\underline{b}_0, L)\}.$$
Wählen wir in Satz 3.23
$$c < (\nu(\underline{b}_0, L) - \mu(\underline{b}_0, L))/\mu(\underline{b}_0, L) \text{ und } 0 < c \leq 1,$$
dann muß $\|B\underline{t} - \underline{b}_0\| = \mu(\underline{b}_0, L)$ sein. Dies folgt aus $(1+c)\mu(\underline{b}_0, L) < \nu(\underline{b}_0, L)$ und der Definition von $\nu(\underline{b}_0, L)$. Wir können c direkt wählen, nämlich
$$c = 1/2\,(\nu(\underline{b}_0, L) - \mu(\underline{b}_0, L))/\nu(\underline{b}_0, L),$$
dann ist auch garantiert, daß $c \leq 1$ ist. Das folgende Korollar ist nur dann in der Praxis brauchbar, wenn nicht nur für $\mu(\underline{b}_0, L)$, sondern auch für $\nu(\underline{b}_0, L) - \mu(\underline{b}_0, L)$ eine untere Schranke bekannt ist. Eine obere Schranke für $\nu(\underline{b}_0, L)$ ist nicht notwendig, wenn wir berücksichtigen, daß
$$\begin{aligned}\nu(\underline{b}_0, L)/\mu(\underline{b}_0, L) &\leq (\mu(\underline{b}_0, L) + \lambda_1(L))/\mu(\underline{b}_0, L)\\ &\leq (\mu(\underline{b}_0, L) + \lambda_{n+1}(\check{B}))/\mu(\underline{b}_0, L)\\ &\leq 2\lambda_{n+1}(\check{B})/\mu(\underline{b}_0, L).\end{aligned}$$

Korollar 3.27 (Nächster Gitterpunkt über $\mathbb{R}^d$) *Sei $L \subset \mathbb{R}^d$ ein Gitter und $B = (\underline{b}_1, \ldots, \underline{b}_n) \in \mathbb{R}^{d\times n}$ eine Basis von L, $\underline{b}_0 \in \mathbb{R}^d$ und $\underline{b}_0 \notin L, \check{B} := (\underline{b}_1, \ldots, \underline{b}_n, \underline{b}_0)$. Es gibt einen Algorithmus, der bei gegebenen rationalen Näherungen der Basis B und des Punktes $\underline{b}_0$ der Genauigkeit $p(q) = \lceil \log(nd) + q \rceil$ mit*
$$\begin{aligned} q \;\geq\; q_8(B, \underline{b}_0, c) = \lceil & n + 1 + \log(1 + \sqrt{n}(\|\underline{b}_0\| + \lambda_{n+1}(\check{B}))\,\mathrm{dft}(B)/\lambda_1(B))\\ & +2\log\lambda_{n+1}(\check{B}) - \log(1/6\cdot(\nu(\underline{b}_0, L) - \mu(\underline{b}_0, L))\cdot\mu(\underline{b}_0, L)\cdot 1/2\cdot\lambda_1(L))\rceil\end{aligned}$$
ein $\underline{t} \in \mathbb{Z}^n$ berechnet mit
$$\|\underline{b}_0 - B\underline{t}\| = \mu(\underline{b}_0, L).$$
Mit $\delta = n(\log(n\lambda_{n+1}(\check{B})/\lambda_1(L)) + \log(\lambda_{n+1}(\check{B})/\mu(\underline{b}_0, L)) + \log d)$ werden dabei insgesamt
$$dn^{3/2n+O(1)} + n^{o(n)}(\,\delta + \log(1/(\nu(\underline{b}_0, L) - \mu(\underline{b}_0, L)))\,)$$
arithmetische Operationen auf ganzen Zahlen der binären Länge
$$O\left(n^2(\delta + \log(1/(\nu(\underline{b}_0, L) - \mu(\underline{b}_0, L))))\right)$$
ausgeführt.

Literaturverzeichnis

[Ba86] Babai L.: *On Lovász' Lattice Reduction and the Nearest Lattice Point Problem*, Combinatorica **6**, Seite 1-13, 1986.

[Ba93] Bartholomes J.: *Gitterbasisreduktion im Reellen*, Diplomarbeit an der Universität des Saarlandes, 1993.

[Bo81] P. van Emde Boas: *Another $\mathcal{NP}$-complete Partition Problem and the Complexity of Computing Short Vectors in a Lattice*, Rep. MI/UVA 81-04, Amsterdam, 1981.

[Bu92] Buchmann, J: *Einführung in die Computeralgebra*, Skript zur Vorlesung im WS 91/92 an der Universität des Saarlandes.

[Bu94] Buchmann, J: *Reducing Lattice Bases by Means of Approximations*, First International Symposium ANTS, Springer Lecture Notes in Computer Science **877**, Seite 160-168, 1994.

[Ca59] Cassels, J.W.S.: *An Introduction to the Geometry of Numbers*, Springer, Berlin/Heidelberg, 1959.

[DaKaTr87] Domich, P.D., Kannan, R., Trotter, L.E.Jr.: *Hermite Normal Form Computation Using Modular Determinant Arithmetic*, Mathematics of Operations Research **12**, No. 1, Seite 50-59, 1987.

[Duden] Claus, V., Schwill, A.: *Schülerduden, Die Informatik*, Dudenverlag, 1986.

[Ga1889] Gauß, C.F.: *Disquisitiones Arithmeticae*, Springer, Berlin, 1889.

[GaSi76] Von zur Gathen, J., Sieveking, M.: *A Bound on the Solutions of Linear Integer Equalities and Inequalities*, Proc. Am. Math. Soc. **72**, Seite 155-58, 1976.

[GrLoSchr88] Grötschel, M., Lovász, L., Schrijver, A.: *Geometric Algorithms and Combinatorical Optimization*, Springer, Berlin/Heidelberg, 1988.

[He85] Helfrich, B.: *Algorithms to Construct Minkowsi Reduced and Hermite Reduced Lattice Bases*, Theoretical Computer Science, Seite 125-39, North-Holland, 1985.

[He05] Hermite, C.: *Zweiter Brief an Jacobi*, Oevres, I. J. Math. **40**, Seite 122-35, 1905.

[Ka83] Kannan, R.: *Improved Algorithms for Integer Programming and Related Lattice Problems*, 15th ACM Symposium on Theory of Computing, Seite 193-206, 1983.

[Ka87a] Kannan, R.: *Minkowski's Convex Body Theorem and Integer Programming*, Mathematics of Operations Research **12**, Seite 415-40, 1987.

[Ka87b] Kannan, R.: *Algorithmic Geometry of Numbers*, Annual Review of Computter Science **3**, Seite 231-67, Annual Reviews, Inc., New York, 1987.

[KaBa79] Kannan, R., Bachem, A.: *Polynomial Time Algorithms for Computing the Smith and Hermite Normal Forms of an Integer Matrix*, SIAM J. Comput. **8**, Seite 499-507, 1979.

[KoZo1873] Korkine, A., Zolotareff, G.: *Sur les formes quadratiques*, Mathematische Annalen **6**, Seite 366-89, 1873.

[La78] Lamprecht, E.: *Einführung in die Algebra*, UTB Birkhäuser, 1978.

[La80] Lamprecht, E.: *Lineare Algebra 1*, UTB Birkhäuser, 1980.

[LaLeSch90] Lagarias, J.C., Lenstra, H.W.Jr., Schnorr, C.P.: *Korkine-Zolotarev Bases and Successive Minima of a Lattice and its Reciprocal Lattice*, Combinatorica **10**, no. 4, Seite 333-48, 1990.

[Le69] Lekkerkerker C.G.: *Geometry of Numbers*, North-Holland, Amsterdam, 1969.

[LeLeLo82] Lenstra, A.K., Lenstra, H.W.Jr., Lovász, L.: *Factoring Polynomials with Rational Coefficients*, Mathematische Annalen **261**, Seite 515-34, 1982.

[LoMeWi86] J. Loeckx, K. Mehlhorn, R. Wilhelm: *Grundlagen der Programmiersprachen*, B.G. Teubner-Verlag, 1986.

[Me84] Mehlhorn, K.: *Data Structures and Algorithms* **1***: Sorting and Searching*, Springer, 1984.

[Mi10] Minkowski, H.: *Geometrie der Zahlen*, Teubner, Leipzig, 1910.

[Mi67] Minkowski, H.: *Diskontinuitätsbereich für arithmetische Äquivalenz*, Collected Works **2**, Seite 53-100, Chelsea, NY, 1967.

[Mu94] Müller, A.: *Effiziente Algorithmen für Probleme in der linearen Algebra über* $\mathbb{Z}$, Diplomarbeit an der Universität des Saarlandes, 1994.

[Sch84] Schnorr, C.P.: *A Hierarchy Of Polynomial Time Basis Reduction Algorithms*, Coll. Math. Soc. Janos Bolyai, Theory of Algorithms **44**, Seite 375-86, Amsterdam, North-Holland, 1984.

[Sch87] Schnorr, C.P.: *A Hierarchy Of Polynomial Time Lattice Basis Reduction Algorithms*, Theoretical Computer Science **53**, Seite 201-24, 1987.

[VoFr76] Voyatov, A.A., Frumkin, M.A.: *An Algorithm for Finding the General Integral Solution of a System of Linear Equations*, Issledovaniya po Diskretnoi Optimalizatsic, Seite 128-40, Moskau, Nauka, auf russisch, 1976.