Teil IV
Anhänge

A Algebraische Strukturen

Wir stellen einige Grundbegriffe der Algebra zusammen, die in jedem einführenden Lehrbuch zu finden sind, zum Beispiel in den Büchern von Bosch [17] oder Wüstholz [90]. Dies dient auch zur Festlegung unserer Notation.

A.1 Gruppen, Ringe, Körper

Definition A.1
Eine nichtleere Menge G mit einer binären Verknüpfung $\circ$ heißt *Gruppe*, wenn folgende Bedingungen erfüllt sind.
a. Assoziativität: $(a \circ b) \circ c = a \circ (b \circ c)$ für alle $a, b, c \in G$;
b. es existiert ein *neutrales Element* e, das heißt $e \circ a = a \circ e = a$ für alle $a \in G$;
c. jedes Element a besitzt ein *Inverses*, das heißt, es existiert ein Element $b \in G$ mit $a \circ b = b \circ a = e$.

Gilt über die Gruppenaxiome hinaus auch Kommutativität (das heißt $a \circ b = b \circ a$ für alle $a, b \in G$), dann heißt G *abelsch*. Eine *Halbgruppe* $(G, \circ)$ erfüllt die Eigenschaften a. und b.

Definition A.2
Eine nichtleere Menge R zusammen mit zwei binären Operationen $+$ und $\cdot$ („Addition“ und „Multiplikation“) heißt *Ring*, wenn gilt:
a. $(R, +)$ ist eine abelsche Gruppe, mit neutralem Element 0;
b. $(R, \cdot)$ ist eine Halbgruppe;
c. es gelten die Distributivgesetze $a(b + c) = ab + ac$, $(a + b)c = ab + ac$.
Ein Ring heißt *kommutativ*, wenn die Multiplikation kommutativ ist.

Ein *Einselement* $1 \in R \setminus \{0\}$ in einem Ring ist ein neutrales Element bezüglich der Multiplikation. Sofern nicht ausdrücklich auf das Gegenteil hingewiesen wird, besitzen alle Ringe, denen wir begegnen, ein Einselement. Die Menge

$$R^\times := \{a \in \mathbb{R} : \text{es existiert ein } b \in R \text{ mit } ab = 1\}$$

bildet bezüglich der Multiplikation eine Gruppe, die *Einheitengruppe* von R. Ist $(R \setminus \{0\}, \cdot)$ eine abelsche Gruppe, dann ist $(R, +, \cdot)$ ein *Körper*.

Es gibt Ringe, in denen $ab = 0$ mit $a, b \neq 0$ gelten kann. a und b heißen dann *Nullteiler*. In Ringen ohne Nullteiler darf man kürzen, das heißt aus $ac = bc$ folgt

$(a-b)c = 0$ und damit $a = b$. Ein kommutativer Ring ohne Nullteiler ist ein *Integritätsbereich*.

Sei R ein Integritätsbereich (mit Einselement). Ein Element $p \in R \setminus \{0\}$ mit $p \notin R^\times$ heißt *irreduzibel*, falls für jede Zerlegung $p = ab$ mit $a, b \in R$ gilt, dass $a \in R^\times$ oder $b \in R^\times$. Ein Element $p \in R \setminus \{0\}$ mit $p \notin R^\times$ heißt *prim*, falls für alle $a, b \in R$ mit $p|ab$ folgt, dass $p|a$ oder $p|b$. R heißt *faktoriell*, falls jedes von Null verschiedene Element, welches keine Einheit ist, Primelement oder Produkt von endlich vielen Primelementen ist.

In faktoriellen Ringen stimmt die Menge der primen Elemente mit der Menge der irreduziblen Elemente überein. Darüber hinaus ist eine Zerlegung eines Elements in Primfaktoren eindeutig, bis auf Einheiten und die Reihenfolge. Genauer: Hat $a \in R \setminus (\{0\} \cup R^\times)$ die Primfaktorzerlegungen $a = p_1 \cdot \ldots \cdot p_r = q_1 \cdot \ldots \cdot q_r$, so folgt $r = s$, und nach einer geeigneten Permutation der q_i gilt $p_i = e_i q_i$ mit Einheiten e_i für $i \in \{1, \ldots, r\}$.

Beispiel A.3
Der Ring

$$\mathbb{Z}[\sqrt{-5}] = \{a + b\sqrt{-5} : a, b \in \mathbb{Z}\}$$

ist nicht faktoriell. Die Zahl 6 hat beispielsweise die Zerlegungen

$$6 = 2 \cdot 3 = (1 + \sqrt{-5})(1 - \sqrt{-5}),$$

und man kann zeigen, dass die auftretenden Faktoren 2, 3, $1 + \sqrt{-5}$, $1 - \sqrt{-5}$ irreduzible Elemente in $\mathbb{Z}[\sqrt{-5}]$ sind.

Für einen Integritätsbereich R lässt sich in Analogie zu dem Übergang von den ganzen Zahlen zu den rationalen Zahlen der *Quotientenkörper* Q von R definieren. Die Elemente von Q sind die „Brüche“ p/q mit $p \in R$ und $q \in R \setminus \{0\}$. Man addiert und multipliziert in Q so wie man mit rationalen Zahlen rechnet:

$$\frac{p}{q} + \frac{s}{t} = \frac{pt + qs}{qt} \quad \text{und} \quad \frac{p}{q} \cdot \frac{s}{t} = \frac{ps}{qt}.$$

Zwei Elemente $\frac{p}{q}$ und $\frac{p'}{q'}$ stellen in Q genau dann das gleiche Element dar, wenn $pq' = p'q$ gilt.

A.2 Polynomringe

Sei R ein kommutativer Ring mit Einselement. Dann definiert auch die Menge aller (formalen) Polynome $a_n x^n + \cdots + a_1 x + a_0$ mit $a_i \in R$ in der Unbestimmten x einen Ring. Die Addition und Multiplikation zweier Polynome $f = \sum_{i=0}^{n} a_i x_i$

und $g = \sum_{j=0}^{m} b_j x^j$ sind definiert durch

$$f + g := \sum_{i=0}^{\max(m,n)} (a_i + b_i) x^i ,$$
$$f \cdot g := \sum_{i=0}^{m+n} c_i x^i \quad \text{mit } c_i := \sum_{j+k=i} a_j b_k .$$

Hierbei vereinbaren wir $a_i = b_j = 0$ für alle $i > n$ und alle $j > m$. Der Koeffizientenring R ist in $R[x]$ durch die konstanten Polynome eingebettet. Ein Einselement in R ist auch ein Einselement in $R[x]$, und für Integritätsbereiche R gilt $R[x]^\times = R^\times$. Man sagt, $R[x]$ entsteht aus R durch Adjunktion einer Unbestimmten x.

Über einem endlichen Körper K gibt es verschiedene Polynome, deren Einsetzungsabbildungen

$$K \to K : x \mapsto f(x)$$

gleich sind; im Falle eines Körpers mit unendlich vielen Elementen ist hingegen die Abbildung eines Polynoms auf die Einsetzungsabbildung stets injektiv. Siehe hierzu auch Aufgabe 10.29.

Bei der Betrachtung von Polynomringen ist die folgende Aussage essentiell:

Satz A.4
Ist R ein faktorieller Ring, dann ist auch $R[x]$ faktoriell.

Induktiv folgt, dass für jeden faktoriellen Ring R auch der Ring der Polynome $R[x_1, \ldots, x_n]$ in den Unbestimmten $x_1, \ldots, x_n$ faktoriell ist.

Für einen Körper K ist der Quotientenkörper des Polynomrings $K[x_1, \ldots, x_n]$ der *Körper der rationalen Funktionen* über K; dieser wird üblicherweise mit $K(x_1, \ldots, x_n)$ bezeichnet.

Ein Körper K heißt *algebraisch abgeschlossen*, wenn jedes nichtkonstante Polynom f aus $K[x]$ eine Nullstelle in K besitzt, das heißt ein $a \in K$ mit $f(a) = 0$. Es gilt:

Satz A.5
Jeder algebraisch abgeschlossene Körper besitzt unendlich viele Elemente.

Beweisidee. Enthält ein Körper K nur endlich viele Elemente $a_1, \ldots, a_k$, dann kann mittels eines Lagrange-Interpolationspolynoms ein Polynom f vom Grad $k - 1$ mit $f(a_i) = 1$ für alle i angegeben werden. □

Jeder Körper K besitzt einen *algebraischen Abschluss*, das heißt einen bezüglich der Inklusion minimalen algebraisch abgeschlossenen Körper, der K enthält. Bis auf Isomorphie ist der algebraische Abschluss eines Körpers eindeutig.

B Trennungssätze

Aus dem Wechselspiel zwischen Analysis und Konvexität ergibt sich eine reichhaltige Theorie, von der wir in diesem Buch nur einen ersten Anfangspunkt benötigen. Für eine umfassende Darstellung sei auf die Monographie von Gruber [52] verwiesen. Ein Einstieg findet sich auch bei Grünbaum [53, §2].

Zwei Mengen $A, B \subseteq \mathbb{R}^n$ liegen *(strikt) getrennt*, wenn es eine affine Hyperebene H gibt mit $A \subseteq H_\circ^+$ und $B \subseteq H_\circ^-$ (vergleiche (2.3) und (2.4)). Wenn A und B nur jeweils in den beiden *abgeschlossenen* affinen Teilräumen von H liegen, spricht man von *schwacher Trennung*.

Eine Teilmenge von $\mathbb{R}^n$ heißt *kompakt*, wenn sie abgeschlossen und beschränkt ist. Polytope sind kompakt.

Satz B.1
Sei C eine abgeschlossene, konvexe Menge im $\mathbb{R}^n$ und $p \in \mathbb{R}^n \setminus C$. Dann existiert eine Hyperebene $H \subseteq \mathbb{R}^n$ mit $p \in H$ und $H \cap C = \emptyset$.

Weil jede konvexe Menge zusammenhängend ist, aber $\mathbb{R}^n \setminus H$ nicht zusammenhängend, ist p also schwach von C getrennt.

Beweis. Ohne Einschränkung können wir $p = 0$ und $C \neq \emptyset$ annehmen. Sei nun c ein beliebiger Punkt in C und $\bar{B} := \bar{B}(0, \|c\|)$ die abgeschlossene Kugel um 0 mit Radius $\|c\|$, wobei $\|\cdot\|$ die euklidische Norm bezeichnet.

Da die Menge $C \cap \bar{B}$ nicht leer und kompakt ist, wird das Minimum der euklidischen Norm auf der Menge $C \cap \bar{B}$ an einem Punkt b angenommen. Sei $H := \{x \in \mathbb{R}^n : \sum_{i=1}^n b_i x_i = 0\}$. Wegen der Annahme $p \notin C$ ist $b \neq 0$. Da $0 \in H$ ist, genügt es nun zu zeigen, dass gilt

$$\langle b, c\rangle \;=\; \sum_{i=1}^{n} b_i c_i \;\geq\; \|b\|^2 \;>\; 0 \tag{B.1}$$

für alle $c \in C$.

Angenommen, es existiert ein $c \in C$ mit $\sum_{i=1}^n b_i c_i < \|b\|^2$. Da C konvex ist, enthält C die Strecke $[b, c]$, und die Punkte dieser Strecke haben die Form

$$x(\lambda) \;:=\; b + \lambda(c - b)\,, \quad 0 \leq \lambda \leq 1\,.$$

Wir zeigen nun, dass es ein $\lambda \in (0,1)$ mit $||x(\lambda)|| < ||b||$ gibt, im Widerspruch zu Wahl von b. Betrachte hierzu die durch

$$\varphi : \mathbb{R} \to \mathbb{R}\,, \quad \varphi(\lambda) \;:=\; \|b\|^2 - \|x(\lambda)\|^2 \;=\; -\lambda^2\|c - b\|^2 - 2\lambda\langle b, c - b\rangle$$

definierte differenzierbare Funktion in λ. Die Ableitung an der Stelle $\lambda = 0$ ist $2(\|b\|^2 - \langle b, c\rangle) > 0$. Folglich existiert ein $\varepsilon > 0$ mit $\|x(\lambda)\| = \|b + \lambda(c - b)\| < \|b\|$ für $0 < \lambda < \varepsilon$. □

Durch Inspektion des Beweises erhalten wir sofort die schärfere Aussage, dass p und C strikt getrennt liegen.

Korollar B.2
Sei C eine abgeschlossene, konvexe Menge im $\mathbb{R}^n$ und $p \in \mathbb{R}^n \setminus C$. Dann existiert eine Hyperebene $H \subseteq \mathbb{R}^n$ mit $p \in H_\circ^-$ und $C \subseteq H_\circ^+$.

Beweis. Dadurch, dass die Ungleichung $\langle b, c\rangle > 0$ in (B.1) strikt ist, lässt sich die im Beweis von Satz B.1 konstruierte Hyperebene H um ein wenig auf C zu verschieben, ohne C zu berühren. Die explizite Rechnung ist analog zu der im Beweis von Satz 3.8, siehe auch Abbildung 3.3. □

Eine affine Hyperebene H heißt *Stützhyperebene* an eine konvexe Menge $C \subseteq \mathbb{R}^n$, falls $H \cap C \neq \emptyset$ gilt und C vollständig in einem der beiden durch H definierten abgeschlossenen affinen Halbräume H^+ oder H^- liegt. Mindestens einer der beiden offenen Halbräume $H_\circ^+$ oder $H_\circ^-$ hat dann also einen leeren Durchschnitt mit C.

Im Fall $\dim C < n$ ist es möglich, dass beide offenen Halbräume einen leeren Durchschnitt mit C haben; für $C \neq 0$ ist jede C enthaltende Hyperebene dann bereits eine Stützhyperebene.

Korollar B.3
Sei C eine abgeschlossene, konvexe Teilmenge des $\mathbb{R}^n$. Dann ist jeder Punkt des Randes von C in einer Stützhyperebene enthalten.

Beweis. Ohne Einschränkung sei $p = 0$ ein Randpunkt von C. Da p ein Randpunkt von C ist, existiert eine Folge $(p^{(k)})_{k\in\mathbb{N}}$ außerhalb von C, die gegen den Nullpunkt konvergiert. Nach Satz B.1 existiert für jedes Folgenelement $p^{(k)}$ eine Hyperebene

$$H^{(k)} = \left\{ x \in \mathbb{R}^n : b^{(k)} + \sum_{i=1}^{n} a_i^{(k)} x_i = 0 \right\},$$

mit $a^{(k)} \in \mathbb{R}^n \setminus \{0\}$ und $b^{(k)} \in \mathbb{R}$, so dass C im Halbraum

$$(H^{(k)})^+ = \left\{ x \in \mathbb{R}^n : b^{(k)} + \sum_{i=1}^{n} a_i^{(k)} x_i \geq 0 \right\}$$

enthalten ist. Wir können weiter annehmen, dass $\|a^{(k)}\| = 1$ gilt. Dann ist $|b^{(k)}|$ der euklidische Abstand von $H^{(k)}$ zum Nullpunkt. Da $p^{(k)}$ gegen den Nullpunkt konvergiert, ist die Folge $(a^{(k)}, b^{(k)})$ im $\mathbb{R}^{n+1}$ beschränkt. Nach dem Satz von Bolzano-Weierstraß existiert daher eine konvergente Teilfolge (siehe etwa [65]).

Sei (a, b) der Grenzwert dieser Teilfolge, und $H = \{x \in \mathbb{R}^n : b + \sum_{i=1}^n a_i x_i = 0\}$ die hierdurch definierte Hyperebene. Aus Stetigkeitsgründen folgt $b = 0$ und dass C im Halbraum

$$H^+ = \left\{ x \in \mathbb{R}^n : \sum_{i=1}^n a_i x_i \geq 0 \right\}$$

enthalten ist. Wegen $0 \in H$ ist H eine Stützhyperebene an C. □

Von den hier bewiesenen Sätzen gibt es eine Reihe weiterer Verschärfungen und Varianten, die in der Literatur ebenfalls unter der Bezeichnung *Trennungssätze* subsumiert werden. Gelegentlich wird auch das Farkas-Lemma aus Aufgabe 4.25 hierzu gezählt.

C Algorithmen und Komplexität

An dieser Stelle sollen einige Begriffe zu Algorithmen und Komplexität skizziert werden. Systematische Einführungen finden sich beispielsweise in den Büchern von Cormen, Leiserson, Rivest und Stein [29], Wegener [89], Schöning [79] oder Garey und Johnson [43].

C.1 Komplexität von Algorithmen

Algorithmen werden in der Regel danach beurteilt, wieviel Rechenzeit und wieviel Speicherplatz sie benötigen. Dieser Ressourcenbedarf wird in Abhängigkeit von der Eingabegröße gemessen.

Die *Codierungslänge* (oder *Größe*) sizeof(x) eines Datenobjekts x ist die Anzahl der Bits, die notwendig sind, um dieses Objekt im Rechner zu speichern. Das zugrunde gelegte Rechnermodell ist hier das der *Turingmaschine* beziehungsweise des *von-Neumann-Rechners*. Eine natürliche Zahl $n > 0$ etwa hat eine Binärdarstellung mit $\lfloor \log_2 n \rfloor + 1$ Ziffern, also gilt sizeof$(n) = \lfloor \log_2 n \rfloor + 1$. Rationale Zahlen können als Paare natürlicher Zahlen mit einem zusätzlichen Vorzeichenbit codiert werden, Matrizen oder Polynome werden gespeichert als Folgen ihrer Koeffizienten (etwa von rationalen Zahlen) und so weiter.

Die *Zeitkomplexität* $t_A(n)$ eines Algorithmus A bezeichnet die maximale Zahl von Schritten, die A zur Lösung einer Instanz des Problems der Codierungslänge n benötigt. Analog beschreibt die *Speicherplatzkomplexität* $s_A(n)$ die maximale Anzahl an Speicherzellen, die zur Lösung einer Probleminstanz der Größe n benötigt werden. Der Schwerpunkt unserer Darstellung liegt auf der Zeitkomplexität von Algorithmen.

Oft ist es unmöglich, die genaue Komplexität eines Algorithmus A zu bestimmen. Man ist jedoch zumindest daran interessiert, das Wachstum der Funktionen $t_A(n)$ und $s_A(n)$ in Abhängigkeit der Größe n der Eingabeinstanz möglichst gut zu kennen. Abschätzungen für dieses Wachstum dienen als Maßstab zur Bewertung der Güte eines Algorithmus.

Um von technischen Aspekten wie der verwendeten Hardware (innerhalb unseres Maschinenmodells) oder der verwendeten Programmiersprache abstrahieren zu können, ist es beispielsweise nützlich, konstante Faktoren zu vernachlässigen. Darüber hinaus erscheint es ebenso sinnvoll, nicht nur konstante Faktoren zu vernachlässigen, sondern sich bei der Komplexitätsanalyse allein auf die do-

minanten Terme der auftretenden Komplexitätsfunktionen zu beschränken. Man spricht hierbei von der *asymptotischen Analyse*.

Zur asymptotischen Charakterisierung der oberen Schranke einer Komplexitätsfunktion $f : \mathbb{N} \to \mathbb{R}_{\geq 0}$ benutzt man die Bezeichung

$$f \in O(g),$$

falls zwei Konstanten $c, n_0 \in \mathbb{N}$ existieren, so dass für alle $n \geq n_0$ gilt

$$f(n) \leq c \cdot g(n).$$

Man sagt „f ist höchstens von der Ordnung g“. Üblich sind auch arithmetische Ausdrücke, in denen $O(n)$ als Term auftritt.

Beispiel C.1
Die Klasse $O(1)$ ist die Klasse der konstant beschränkten Funktionen. Mit $f \in n^{O(1)}$ ist gemeint, dass f durch ein Polynom in n beschränkt ist.

Ist man an unteren Schranken für eine Komplexitätsfunktion f interessiert, dann wird die folgende Bezeichnung verwendet. Wir sagen

$$f \in \Omega(g),$$

gelesen „f ist mindestens von der Ordnung g“, falls zwei Konstanten $c, n_0 \in \mathbb{N}$ existieren, so dass für alle $n \geq n_0$ gilt

$$f \geq c \cdot g.$$

Wir schreiben

$$f \in \Theta(g),$$

falls $f \in O(g)$ und $g \in O(f)$, das heißt, falls die Wachstumsordnungen von f und g gleich sind.

Beispiel C.2 (Binäre Suche)
Gegeben sei eine aufsteigend sortierte Folge $(a_1, \ldots, a_n)$ paarweise verschiedener natürlicher Zahlen. Für eine Zahl $x \in \mathbb{N}$ soll nun algorithmisch getestet werden, ob x in der gegebenen Folge bereits enthalten ist. Ein naives Verfahren würde x nacheinander mit jedem der Elemente $a_1, \ldots, a_n$ vergleichen und daraufhin die passende Antwort ausgeben. Dieses Verfahren benötigt im ungünstigsten Fall, der zum Beispiel eintritt, wenn das gesuchte Element nicht in der Folge enthalten ist, $\Theta(n)$ viele Schritte.

Das Prinzip „teile und herrsche“ („divide and conquer“) führt wegen der Sortierung der Folge zu einer Verbesserung. Durch einen Vergleich von x mit $a_{\lfloor n/2 \rfloor}$ kann festgestellt werden, ob x in der ersten Hälfte oder der zweiten Hälfte der Folge enthalten sein müsste. Durch rekursive Wiederholung dieses Schrittes kann in $O(\log n)$ vielen Schritten festgestellt werden, ob x in der Folge enthalten ist.

Das Prinzip der binären Suche wird beispielsweise bei der Bestimmung des nächsten Nachbarn in Abschnitt 6.5 angewandt.

Auf dem Prinzip „teile und herrsche“ beruht auch der in Abschnitt 5.3 vorgestellte Algorithmus 5.4 zur Berechnung der konvexen Hülle in der Ebene.

Ein einfaches Problem, anhand dessen viele Paradigmen aus der Theorie effizienter Algorithmen studiert werden können, ist das Sortieren von Zahlen. Sortieren spielt auch für geometrische Algorithmen (beispielsweise für ebene Konvexe-Hülle-Algorithmen) eine entscheidende Rolle. Es gilt:

Satz C.3
Das Sortieren von n Zahlen ist in $O(n \log n)$ Schritten möglich.

Beweisskizze. Wir betrachten ohne Einschränkung eine Folge $A = (a_1, \ldots, a_n)$ paarweise verschiedener Zahlen mit einer Zweierpotenz n. Mit dem auf dem „teile und herrsche“-Prinzip beruhenden *Sortieren durch Mischen* (*merge sort*) können wir ein Verfahren angeben, das die behauptete Laufzeitschranke nicht überschreitet. Der Algorithmus C.1 besteht aus den angegebenen drei Schritten.

1 **Aufteilen.** Die Folge A wird in zwei Teilfolgen $A_1 = (a_1, \ldots, a_{n/2})$, $A_2 = (a_{n/2+1}, \ldots, a_n)$ zerlegt.
2 **Rekursion.** Rekursiv wird nun jede der beiden Teilfolgen mit der gleichen Methode sortiert. Seien B_1 und B_2 die beiden daraus resultierenden sortierten Teilfolgen.
3 **Mischen.** Füge die beiden sortierten Folgen B_1 und B_2 zu einer sortierten Gesamtfolge für die Folge A zusammen.

Algorithmus C.1. `MergeSort`: Sortieren durch Mischen

Für die Laufzeit $t(n)$ von merge-sort ergibt sich daher die rekursive Beziehung

$$t(n) \leq 2t\left(\frac{n}{2}\right) + dn$$

mit einer Konstanten $d > 0$. Durch Lösen dieser Rekursionsbeziehung erhalten wir die obere Schranke für das Sortieren. □

Eine grundsätzliche Aussage der Komplexitätstheorie besagt, dass kein Algorithmus, der lediglich auf dem Vergleich von Zahlen als Elementarschritt beruht, eine asymptotisch bessere Laufzeit haben kann. Dies lässt sich durch ein Entscheidungsbaummodell beweisen [29]. Wir erhalten auf diese Weise eine asymptotisch exakte Abschätzung für die Laufzeitkomplexität des Sortierproblems.

Satz C.4
Das vergleichsbasierte Sortieren von n Zahlen hat die Komplexität $\Theta(n \log n)$.

C.2 Die Komplexitätsklassen P und NP

Ein *Entscheidungsproblem* ist ein Problem, das nur zwei mögliche Lösungen hat: „Ja“ oder „Nein“. Ein *Optimierungsproblem* erfordert das Auffinden einer *optimalen* Lösung aus einer (möglicherweise großen) Menge zulässiger Lösungen. Hierbei wird die Güte einer Lösung über den Wert einer Kostenfunktion gemessen. Jedes Optimierungsproblem induziert eine Filtrierung von Entscheidungsproblemen: Das Optimierungsproblem $\max\{c(x) : x \in X\}$ und die Schranke k führen auf die Frage, ob eine Lösung $x \in X$ existiert mit der Gütegarantie $c(x) \geq k$.

Es hängt vom konkreten Anwendungsgebiet ab, welche Klasse von Algorithmen als *effizient* betrachtet wird. Im Optimierungskontext etwa gelten für praktische Anwendungen nur Algorithmen als praktikabel, deren Laufzeit von oben durch ein Polynom in der Codierungslänge der Eingabe beschränkt ist. Algorithmen mit exponentiellem Aufwand versucht man soweit wie möglich zu vermeiden. Dagegen existieren für die Berechnung einer Gröbnerbasis wie in Kapitel 9 bislang nur Algorithmen, deren Zeitkomplexität *doppelt exponentiell* in der Eingabelänge beschränkt ist. Dennoch beruhen zahlreiche moderne Anwendungen auf solchen Methoden.

Definition C.5
Ein Algorithmus A heißt *Polynomialzeit-Algorithmus*, wenn es ein univariates Polynom p gibt, so dass A zu jeder Eingabe x in $O(p(\text{sizeof}(x)))$ Schritten terminiert.

Eine wichtige Frage der Komplexitätstheorie beschäftigt sich damit, für welche Probleme solche Algorithmen existieren. Im Folgenden konzentrieren wir unsere Ausführungen vor allem auf Entscheidungsprobleme. Als Maßstab für *Effizienz* gilt hier die Definition C.5.

Die Komplexitätsklasse P. Die Klasse P (Polynomialzeit) bezeichnet die Menge aller Entscheidungsprobleme, für deren Lösung ein Polynomialzeit-Algorithmus existiert.

Die Klasse der Algorithmen, die mit polynomial beschränktem Speicheraufwand auskommen heißt PSPACE. Offensichtlich ist P in PSPACE enthalten.

Die Komplexitätsklasse NP. Die nachfolgend definierte Klasse NP (nichtdeterministisch Polynomialzeit) enthält Probleme, die zumindest nichtdeterministisch effizient gelöst werden können. Im Gegensatz zum deterministischen Fall, bei dem es in einer Situation genau eine Handlungsmöglichkeit gibt, wird bei nichtdeterministischen Betrachtungen eine Vielzahl möglicher Aktivitäten zugelassen.

Betrachtet man beispielsweise die Suche nach einem Beweis für einen mathematischen Satz, dann gibt es im Falle einer falschen Behauptung überhaupt keinen Beweis. Ist die Behauptung jedoch wahr, dann lassen sich im Allgemeinen

verschiedene Beweise führen. Wichtig für den Nachweis der Richtigkeit des Satzes ist lediglich, dass *wenigstens ein* Beweis angegebenen werden kann. Natürlich kann das Finden eines Beweises beliebig schwierig sein. Wird jedoch ein Beweis vorgelegt, dann ist es im Allgemeinen nicht mehr schwer, ihn nachzuvollziehen und die Behauptung zu akzeptieren. In der Komplexitätstheorie werden solche Beweise auch als *Zertifikate* (oder *Zeugen*) bezeichnet.

Definition C.6
Ein Entscheidungsproblem $\mathcal{A}$ liegt in NP, falls ein Polynom p und ein polynomialer Algorithmus A existieren, der für jede Eingabe x und jedes mögliche Zertifikat y der Länge höchstens $p(\text{sizeof}(x))$ einen Wert $t(x,y)$ berechnet, so dass gilt:

a. Lautet die Antwort zur Eingabe x „Nein“, dann gilt $t(x,y) = 0$ für alle möglichen Zertifikate.
b. Lautet die Antwort zur Eingabe x „Ja“, dann gilt $t(x,y) = 1$ für wenigstens ein Zertifikat.

Die Frage „P = NP?“. Offenbar ist die Klasse P in der Klasse NP enthalten. Ein wichtiges offenes Problem der Komplexitätstheorie ist die Frage

$$\text{„P} \stackrel{?}{=} \text{NP“}.$$

Ihre Bedeutung erklärt sich daraus, dass es viele wichtige Aufgabenstellungen gibt, für die keine polynomialen Algorithmen bekannt sind, aber für die die Mitgliedschaft zur Klasse NP nachgewiesen werden kann. Ohne eine Klärung der Frage „$\text{P} \stackrel{?}{=} \text{NP}$“ ist es nicht möglich zu entscheiden, ob es sich bei diesen Aufgabenstellungen um Probleme handelt, die überhaupt nicht in Polynomialzeit lösbar sind, oder ob bisher nur noch keine solchen Algorithmen gefunden werden konnten.

Ein Entscheidungsproblem $\mathcal{A}$ heißt NP-*schwer*, falls sich jedes Problem in NP in Polynomialzeit auf $\mathcal{A}$ reduzieren lässt. NP-*vollständig* heißt ein Entscheidungsproblem $\mathcal{A}$ in NP, wenn jedes Problem aus NP mittels einer geeigneten Polynomialzeitreduktion auf $\mathcal{A}$ zurückgeführt werden kann. Exakte Definitionen dieser Begriffe stehen beispielsweise bei Garey und Johnson [43].

NP-vollständige Probleme verkörpern die „schwersten“ Probleme in der Klasse NP. Es gilt: Falls irgendein NP-vollständiges Problem in Polynomialzeit gelöst werden kann, so ist das für alle anderen Probleme in NP auch möglich, und es gilt P = NP.

Ein Beispiel für ein NP-vollständiges Entscheidungsproblem ist die Frage nach der Existenz eines *Hamilton-Kreises* in einem endlichen Graphen:

Beispiel C.7
Gegeben sei ein (ungerichteter) endlicher Graph. Existiert ein geschlossener Weg durch den Graphen, der jeden Knoten des Graphen genau einmal besucht?

Fast alle Experten auf dem Gebiet der Komplexitätstheorie vermuten, dass die Klassen P und NP verschieden sind.

Die Komplexitätsklasse #P. Analog zu Entscheidungsproblemen kann man auch Zählprobleme untersuchen. Hierbei ist die Ausgabe jeweils eine natürliche Zahl. Wie bei Optimierungsproblemen gibt es einen direkten Zusammenhang mit den Entscheidungsproblemen.

Definition C.8
Ein Zählproblem $\mathcal{A}$ liegt in #P, falls ein Entscheidungsproblem $\mathcal{B} \in$ NP existiert, so dass die Aufgabe für $\mathcal{A}$ darin besteht, die Anzahl der $\mathcal{B}$ validierenden Lösungen zu bestimmen.

Ähnlich zu den Begriffen „NP-schwer“ und „NP-vollständig“ lassen sich auch entsprechende Klassen von Zählproblemen definieren. Ein Zählproblem ist #P-*schwer*, wenn sich jedes Problem in #P in Polynomialzeit darauf reduzieren lässt, und es heißt #P-*vollständig*, falls es zusätzlich selbst in #P liegt.

Beispiel C.9
Die Frage, wie viele verschiedene Hamiltonkreise es in einem gegebenen endlichen Graphen gibt, ist #P-vollständig.

Weitere Komplexitätsklassen. Die Anzahl der Komplexitätsklassen, die in der Literatur betrachtet werden, scheint ständig zuzunehmen. Mittlerweile spricht man auch von einem „Zoo“ von Komplexitätsklassen.

In den Anmerkungen zu Kapitel 9 tritt bei uns ferner noch EXPSPACE auf, die Klasse der Algorithmen, deren Speicherbedarf durch $\exp^{O(1)}$ beschränkt ist.

D Software

Es gibt sehr viel Software zum Thema *Algorithmische Geometrie*. Die Palette reicht von der Implementierung einzelner Algorithmen bis hin zu großen Systemen mit einem weiten Anwendungsspektrum. Dieser Abschnitt soll für vier Softwarepakete kurz auflisten, wofür sie sich im Hinblick auf die algorithmische Geometrie einsetzen lassen.

D.1 `polymake`

Das System `polymake` ist spezialisiert auf Algorithmen zum Studium der Geometrie und Kombinatorik von Polytopen und Polyedern in beliebiger Dimension [45, 44]. Mehrere Konvexe-Hülle-Verfahren stehen zur Verfügung, und es können Voronoi-Diagramme sowie Delone-Zerlegungen berechnet werden. Über die Behandlung von Polytopen hinaus bietet die aktuelle Version 2.3 unter anderem Methoden zur Untersuchung algebraischer Invarianten endlicher Simplizialkomplexe sowie Algorithmen für polyedrische Flächen und zur Untersuchung von Starrheit.

`polymake` ist ein Open-Source-System, das in Perl und C++ geschrieben ist und in beiden Sprachen erweitert werden kann. Zusätzlich bietet es eine umfangreiche C++-Klassenbibliothek zur linearen Algebra und algorithmischen Geometrie, die auch unabhängig vom Rest des Systems genutzt werden kann.

Im WWW ist `polymake` unter `www.polymake.de` vertreten.

D.2 `Maple`

`Maple` ist ein kommerzielles mathematisches Softwaresystem mit breiter Funktionalität. Hinsichtlich der algorithmischen Geometrie bietet die aktuelle Version 11 nur wenige der Algorithmen, die im ersten Teil des Buches vorgestellt wurde, darunter einen Konvexe-Hülle-Algorithmus in der Ebene und eine Bibliothek zur Lösung linearer Programme. Hingegen kann `Maple` Gröbnerbasen berechnen und verfügt über die Eliminationstechniken aus dem zweiten Teil. Zusätzlich gibt es einfache Visualisierungsmöglichkeiten.

Für `Maple` gibt es zahlreiche Erweiterungen und Anwendungsbeispiele, über die man sich auf `www.maplesoft.com` informieren kann. `Maple` besitzt sowohl eine eigene Programmiersprache als auch C- und Java-Schnittstellen.

Zwar ist Maple jedem der hier genannten spezialisierten Programme in dessen Domäne in puncto Methodenreichtum und Geschwindigkeit weit unterlegen, aber es bietet andererseits die Möglichkeit, Verfahren aus allen Bereichen zu kombinieren.

Beim Ausprobieren unserer Code-Beispiele ist zu berücksichtigen, dass die Syntax zwischen verschiedenen Maple-Versionen teilweise differiert.

D.3 Singular

Singular ist ein Open-Source-Softwareprojekt, das der algorithmischen kommutativen Algebra und algebraischen Geometrie gewidmet ist [49, 48]. Die aktuelle Version trägt die Nummer 3.0.3. Zahlreiche Verfahren für die Berechnung von Gröbnerbasen sind implementiert. Elimination und viele Verfeinerungen, wie etwa das Conti-Traverso-Verfahren aus Abschnitt 10.6, stehen zur Verfügung. Zusätzlich bietet das System unter anderem Algorithmen zur Invarianten- und Codierungstheorie sowie numerische Verfahren zur Lösung polynomialer Gleichungssysteme.

Die Web-Site ist www.singular.uni-kl.de. Singular kann in einer eigenen Sprache programmiert werden.

D.4 CGAL

Die „Computational Geometry Algorithms Library“ (CGAL) ist ein umfassendes Open-Source-Softwaresystem vor allem für die niedrigdimensionale algorithmische Geometrie [20]. Voronoi-Diagramme und Delone-Triangulierungen sind in vielen Varianten und Verfeinerungen verfügbar, darunter auch die in Abschnitt 13.1 behandelten Voronoi-Diagramme von Geradensegmenten. Es existiert ein Konvexe-Hülle-Algorithmus in beliebiger Dimension.

Das Anwendungsspektrum reicht von Arrangements von Geraden und Kurven, Gittererzeugung, geometrische Datenverarbeitung, Suchstrukturen bis zur Bewegungsplanung.

CGAL ist eine C++-Bibliothek, die in ihrer derzeit aktuellen Version 3.3 mit vielen Beispielprogrammen erhältlich ist von der Web-Site www.cgal.org.

E Notation

Die Elemente eines Vektorraums schreiben wir üblicherweise als Spaltenvektoren. Während wir dies im ersten Teil des Buches konsequent durchzuhalten versuchen, sind wir im zweiten und dritten Teil diesbezüglich etwas großzügiger, um die Notation zu entlasten.

Die Tabelle unten führt die wichtigsten verwendeten Symbole auf, zumeist mit einem Verweis auf die Seite des ersten Auftretens.

$\lvert M\rvert$	Anzahl der Elemente der Menge M	
$\mathbb{N} = \{0, 1, 2, \dots\}$	natürliche Zahlen	
$\mathbb{Z}$	ganze Zahlen	
$\mathbb{Q}$	rationale Zahlen	
$\mathbb{R}$	reelle Zahlen	
$\mathbb{C}$	komplexe Zahlen	
Id	Einheitsmatrix (passender Dimension)	
$\mathrm{Sym}(M)$	Menge der Permutationen der Menge M, symmetrische Gruppe auf M	
$\mathrm{sgn}(\sigma)$	Signum der Permutation $\sigma \in \mathrm{Sym}(M)$	
$\operatorname{int} M$	Inneres einer Menge $M \subseteq \mathbb{R}^n$	18
$\overline{M}$	Abschluss von M	18
∂M	Rand von M	18
$(K^n)^*$	Dualraum des Vektorraums K^n	
$\mathbb{P}^n_K$	n-dimensionaler projektiver Raum über K	11
$\mathrm{G}_{k,n} K$	k-te Grassmannsche von K^n	210
$\operatorname{lin} M$	lineare Hülle der Teilmenge M eines Vektorraums	
$\operatorname{aff} M$	affine Hülle	16
$\operatorname{conv} M$	konvexe Hülle	16
$[x, y] = \operatorname{conv}\{x, y\}$	Strecke zwischen zwei Punkten $x, y \in \mathbb{R}^n$	
$\operatorname{pos} M$	positive Hülle	36
$(x_0 : x_1 : \cdots : x_n)^T$	homogene Koordinaten eines Punktes im projektiven Raum	12
$[a_0 : a_1 : \cdots : a_n]$	(orientierte) homogene Koordinaten einer Hyperebene	13, 17

$\langle \cdot, \cdot \rangle$	inneres Produkt bzw. euklidisches Skalarprodukt	13, 17
$\| \cdot \|$	euklidische Norm	
$\operatorname{vol} M$	n-dimensionales Volumen von $M \subseteq \mathbb{R}^n$	
M°	zu M polare Menge	31
$\mathcal{F}(P)$	Seitenverband eines Polytops P	37
$I(V, \mathcal{H})$	Inzidenzmatrix der doppelten Beschreibung $(V, \mathcal{H})$	72
$[\mathcal{C}]$	von einer Familie $\mathcal{C}$ von Polyedern (mit Schnittbedingung) erzeugter polyedrischer Komplex	85
$\mathrm{VR}_S(p)$	Voronoi-Region des Punktes p bezüglich $S \subseteq \mathbb{R}^n$	83
$\mathrm{VD}(S)$	Voronoi-Diagramm von $S \subseteq \mathbb{R}^n$	86
$\mathcal{P}(S)$	Polyeder, durch das $\mathrm{VD}(S)$ als vertikale Projektion entsteht	88
$\mathcal{P}^*(S)$	Delone-Polytop	105
$\mathrm{DZ}(S)$	Delone-Zerlegung	107
$\mathrm{ggT}(f, g)$	größter gemeinsamer Teiler von f und g	
$\mathrm{kgV}(f, g)$	kleinstes gemeinsames Vielfaches	
$\deg_x f$	Grad des Polynoms f in der Unbestimmten x	
$\operatorname{tdeg} f$	Totalgrad von f	135
$\mathrm{Res}_x(f, g)$	Resultante von f und g bezüglich der Unbestimmten x	130
$\langle f_1, \ldots, f_t \rangle$	von den Polynomen $f_1, \ldots, f_t$ erzeugtes Ideal	145
$\mathrm{V}(I)$	durch das Ideal I definierte affine oder projektive algebraische Varietät	145
I_k	k-tes Eliminationsideal von I	145, 169
$\mathrm{rem}_\prec(f; g_1, \ldots, g_t)$	Rest der multivariaten Division	147, 151
$\prec_{\mathrm{lex}}$	lexikographische Monomordnung	150
$\prec_{\mathrm{revlex}}$	umgekehrt lexikographische Monomordnung	152
$\prec_{\mathrm{grevlex}}$	graduierte umgekehrt lexikographische Monomordnung	152
M_C	mediale Achse der Kurve C	193
$\lambda_C(p)$	lokale Detailgröße der Kurve C im Punkt p	194
$\bigwedge^k V$	k-te äußere Potenz des Vektorraums V	207
$\bigwedge V$	äußere Algebra des Vektorraums V	207
$x \wedge y$	äußeres Produkt von x und y	208
P, NP, #P	Komplexitätsklassen	248

Literaturverzeichnis

[1] William W. Adams und Philippe Loustaunau. *An introduction to Gröbner bases*, Band 3 der *Graduate Studies in Mathematics*. American Mathematical Society, Providence, RI, 1994.

[2] Martin Aigner. *Diskrete Mathematik*. Vieweg Studium: Aufbaukurs Mathematik. Vieweg, Wiesbaden, 5. Auflage, 2004.

[3] Martin Aigner und Günter M. Ziegler. *Das BUCH der Beweise*. Springer-Verlag, Berlin, 2. Auflage, 2004.

[4] Ernst Althaus und Kurt Mehlhorn. Traveling salesman-based curve reconstruction in polynomial time. *SIAM J. Comput.*, 31(1):27–66, 2001.

[5] Nina Amenta, Marshall Bern und David Eppstein. The crust and the β-skeleton: Combinatorial curve reconstruction. *Graphical Models and Image Processing*, 60:125–136, 1998.

[6] Enrique Arrondo. Another elementary proof of the Nullstellensatz. *Amer. Math. Monthly*, 113(2):169–171, 2006.

[7] David Avis. lrslib 4.2. `http://cgm.cs.mcgill.ca/~avis/C/lrs.html`.

[8] David Avis, David Bremner und Raimund Seidel. How good are convex hull algorithms? *Comput. Geom.*, 7(5-6):265–301, 1997.

[9] David Avis und Komei Fukuda. A pivoting algorithm for convex hulls and vertex enumeration of arrangements and polyhedra. *Discrete Comput. Geom.*, 8(3):295–313, 1992.

[10] Joseph L. Awange und Erik W. Grafarend. *Solving algebraic computational problems in geodesy and geoinformatics*. Springer-Verlag, Berlin, 2005.

[11] Saugata Basu, Richard Pollack und Marie-Françoise Roy. *Algorithms in real algebraic geometry*, Band 10 der *Algorithms and Computation in Mathematics*. Springer-Verlag, Berlin, 2. Auflage, 2006.

[12] Thomas Becker und Volker Weispfenning. *Gröbner bases*, Band 141 der *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1993.

[13] Dimitris Bertsimas und Robert Weismantel. *Optimization over integers*. Dynamic Ideas, Belmont, MA, 2005.

[14] Albrecht Beutelspacher und Ute Rosenbaum. *Projektive Geometrie.* Vieweg, Wiesbaden, 2. Auflage, 2004.

[15] Harry Blum. A transformation for extracting new descriptors of shape. In Weiant Whaten-Dunn, Hg., *Proc. Symposium on Models for the Perception of Speech and Visual Form*, 362–380. MIT Press, Cambridge, MA, 1967.

[16] Jean-Daniel Boissonnat und Mariette Yvinec. *Algorithmic geometry*. Cambridge University Press, Cambridge, 1998.

[17] Siegfried Bosch. *Algebra*. Springer-Verlag, Berlin, 6. Auflage. Auflage, 2006.

[18] Arne Brøndsted. *An introduction to convex polytopes*, Band 90 der *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1983.

[19] Bruno Buchberger. *Ein Algorithmus zum Auffinden der Basiselemente des Restklassenrings nach einem nulldimensionalen Polynomideal*. Dissertation, Universität Innsbruck, 1965.

[20] `CGAL`, Computational Geometry Algorithms Library. `www.cgal.org`.

[21] Timothy M. Chan. Optimal output-sensitive convex hull algorithms in two and three dimensions. *Discrete Comput. Geom.*, 16(4):361–368, 1996.

[22] Timothy M. Chan, Jack Snoeyink und Chee-Keng Yap. Primal dividing and dual pruning: output-sensitive construction of four-dimensional polytopes and three-dimensional Voronoi diagrams. *Discrete Comput. Geom.*, 18(4):433–454, 1997.

[23] Bernard Chazelle. An optimal convex hull algorithm in any fixed dimension. *Discrete Comput. Geom.*, 10(4):377–409, 1993.

[24] Vašek Chvátal. *Linear programming*. W. H. Freeman and Company, New York, 1983.

[25] Kenneth L. Clarkson und Peter W. Shor. Algorithms for diametral pairs and convex hulls that are optimal, randomized, and incremental. In *Proc. Fourth Annual Symposium on Computational Geometry (Urbana, IL, 1988)*, 12–17, New York, 1988. ACM.

[26] CoCoA-Team. `CoCoA`: a system for doing Computations in Commutative Algebra. `cocoa.dima.unige.it`.

[27] George E. Collins. Quantifier elimination for real closed fields by cylindrical algebraic decomposition. In *Automata theory and formal languages (Second GI Conf., Kaiserslautern, 1975)*, 134–183. Lecture Notes in Comput. Sci., Vol. 33. Springer, Berlin, 1975.

[28] Pasqualina Conti und Carlo Traverso. Buchberger algorithm and integer programming. In *Applied algebra, algebraic algorithms and error-correcting codes (New Orleans, LA, 1991)*, Band 539 der *Lecture Notes in Comput. Sci.*, 130–139. Springer, Berlin, 1991.

[29] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest und Cliff Stein. *Algorithmen – Eine Einführung*. Oldenbourg, München, 2. Auflage, 2007.

[30] David Cox, John Little und Donal O'Shea. *Ideals, varieties, and algorithms*. Undergraduate Texts in Mathematics. Springer, New York, 3. Auflage, 2007.

[31] David A. Cox, John Little und Donal O'Shea. *Using algebraic geometry*, Band 185 der *Graduate Texts in Mathematics*. Springer, New York, 2. Auflage, 2005.

[32] Mark de Berg, Marc van Kreveld, Mark Overmars und Otfried Schwarzkopf. *Computational geometry*. Springer-Verlag, Berlin, 2. Auflage, 2000.

[33] Tamal K. Dey und Piyush Kumar. A simple provable algorithm for curve reconstruction. In *Proc. Symposium on Discrete Algorithms (Baltimore, MD)*, 893–894, 1999.

[34] Leonard E. Dickson. Finiteness of the odd perfect and primitive abundant numbers with n distinct prime factors. *Amer. J. Math.*, 35:413–422, 1913.

[35] Peter Dietmaier. The Stewart-Gough platform of general geometry can have 40 real postures. In J. Lenarcic und M.L. Husty, Hg., *Advances in Robot Kinematics: Analysis and Control*, 7–16. Kluwer Academic Publishers, Dordrecht, 1998.

[36] Martin E. Dyer und Alan M. Frieze. On the complexity of computing the volume of a polyhedron. *SIAM J. Comput.*, 17(5):967–974, 1988.

[37] Herbert Edelsbrunner. *Algorithms in combinatorial geometry*, Band 10 der *EATCS Monographs on Theoretical Computer Science*. Springer-Verlag, Berlin, 1987.

[38] Gerd Fischer. *Ebene algebraische Kurven*. Vieweg, Braunschweig, 1994.

[39] Gerd Fischer und Jens Piontkowski. *Ruled varieties*. Vieweg, Braunschweig, 2001.

[40] Steven Fortune. A sweepline algorithm for Voronoĭ diagrams. *Algorithmica*, 2(2):153–174, 1987.

[41] Komei Fukuda. cddlib 0.94b. `http://www.ifor.math.ethz.ch/~fukuda/cdd_home/cdd.html`.

[42] Komei Fukuda und Alain Prodon. Double description method revisited. In *Combinatorics and computer science (Brest, 1995)*, Band 1120 der *Lecture Notes in Comput. Sci.*, 91–111. Springer-Verlag, Berlin, 1996.

[43] Michael R. Garey und David S. Johnson. *Computers and intractability: A guide to the theory of NP-completeness*. W. H. Freeman and Co., San Francisco, CA, 1979.

[44] Ewgenij Gawrilow und Michael Joswig. `polymake`: a framework for analyzing convex polytopes. In *Polytopes – combinatorics and computation (Oberwolfach, 1997)*, Band 29 der *DMV Sem.*, 43–73. Birkhäuser, Basel, 2000.

[45] Ewgenij Gawrilow und Michael Joswig. `polymake` 2.3. Technical report, Technische Universität Berlin und Technische Universität Darmstadt, 2007. Mit Beiträgen von Thilo Rörig und Niko Witte, `www.polymake.de`.

[46] Jacob E. Goodman und Joseph O'Rourke, Hg. *Handbook of discrete and computational geometry*. Chapman & Hall/CRC, Boca Raton, FL, 2. Auflage, 2004.

[47] Daniel R. Grayson und Michael E. Stillman. `Macaulay 2`, a software system for research in algebraic geometry. `http://www.math.uiuc.edu/Macaulay2/`.

[48] Gert-Martin Greuel und Gerhard Pfister. *A* `Singular` *introduction to commutative algebra*. Springer-Verlag, Berlin, 2002.

[49] Gert-Martin Greuel, Gerhard Pfister und Hans Schönemann. `Singular` 3.0.3. A computer algebra system for polynomial computations, Centre for Computer Algebra, Universität Kaiserslautern, 2007. `www.singular.uni-kl.de`.

[50] Peter Gritzmann. *Optimierung*. Vieweg, Wiesbaden. In Vorbereitung.

[51] Martin Grötschel, Lászlo Lovász und Alexander Schrijver. *Geometric algorithms and combinatorial optimization*, Band 2 der *Algorithms and Combinatorics*. Springer, 2. Auflage, 1993.

[52] Peter Gruber. *Convex and discrete geometry*, Band 336 der *Grundlehren der Mathematischen Wissenschaften*. Springer, Berlin, 2007.

[53] Branko Grünbaum. *Convex polytopes*, Band 221 der *Graduate Texts in Mathematics*. Springer-Verlag, New York, 2. Auflage, 2003.

[54] Dan Halperin, Lydia Kavraki und Jean-Claude Latombe. Robotics. In *Handbook of discrete and computational geometry*, CRC Press Ser. Discrete Math. Appl., 1065–1094. CRC, Boca Raton, FL, 2. Auflage, 2004.

[55] Heisuke Hironaka. Resolution of singularities of an algebraic variety over a field of characteristic zero. I, II. *Ann. of Math. (2) 79 (1964), 109–203; ibid. (2)*, 79:205–326, 1964.

[56] John Hobby. METAPOST. `http://cm.bell-labs.com/who/hobby/MetaPost.html`.

[57] William V. D. Hodge und Dan Pedoe. *Methods of algebraic geometry. Vol. I, II*. Cambridge University Press, Cambridge, 1947.

[58] Stephan Holzer und Oliver Labs. `surfex` 0.89. Technical report, Universität Mainz und Universität Saarbrücken, 2007. `www.surfex.AlgebraicSurface.net`.

[59] Hoon Hong, Christopher W. Brown et al. `QEPCAD` b 1.46. Technical report, RISC Linz und U.S. Naval Academy, Annapolis, 2007. `http://www.cs.usna.edu/~qepcad/B/QEPCAD.html`.

[60] Michael Joswig. Beneath-and-beyond revisited. In *Algebra, geometry, and software systems*, 1–21. Springer-Verlag, Berlin, 2003.

[61] David E. Joyce. Euclid's elements. `http://aleph0.clarku.edu/~djoyce/java/elements/elements.html`, 1998.

[62] Leonid Khachiyan, Endre Boros, Konrad Borys, Khaled Elbassioni und Vladimir Gurvic. Generating all vertices of a polyhedron is hard. In *Proc. Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithms*, 758–765, 2006.

[63] Frances Kirwan. *Complex algebraic curves*, Band 23 der *London Mathematical Society Student Texts*. Cambridge University Press, Cambridge, 1992.

[64] Rolf Klein. *Algorithmische Geometrie*. Springer, Berlin, 2. Auflage, 2005.

[65] Konrad Königsberger. *Analysis 1*. Springer-Verlag, Berlin, 6. Auflage, 2004.

[66] Konrad Königsberger. *Analysis 2*. Springer-Verlag, Berlin, 5. Auflage, 2004.

[67] Bernhard Korte und Jens Vygen. *Combinatorial optimization*, Band 21 der *Algorithms and Combinatorics*. Springer-Verlag, Berlin, 3. Auflage, 2006.

[68] Wolfgang Kühnel. *Differentialgeometrie*. Vieweg Studium: Aufbaukurs Mathematik. Vieweg, Braunschweig, 2. Auflage, 2003.

[69] Jean-Pierre Lazard, Daniel Merlet. The (true) Stewart platform has 12 configurations. In *Proc. IEEE International Conference on Robotics and Automation (San Diego, CA)*, 2160–2165, 1994.

[70] Ernst W. Mayr und Albert R. Meyer. The complexity of the word problems for commutative semigroups and polynomial ideals. *Adv. in Math.*, 46(3):305–329, 1982.

[71] J. Michael McCarthy. *Geometric design of linkages*, Band 11 der *Interdisciplinary Applied Mathematics*. Springer-Verlag, New York, 2000.

[72] Peter McMullen. The maximum numbers of faces of a convex polytope. *Mathematika*, 17:179–184, 1970.

[73] Richard Morris. `SingSurf`: A program for calculating singular algebraic curves and surfaces. `www.singsurf.org`, 2005.

[74] Ketan Mulmuley. *Computational geometry: An introduction through randomized algorithms*. Prentice Hall, Englewood Cliffs, NJ, 1993.

[75] Konrad Polthier, Eike Preuss, Klaus Hildebrandt und Ulrich Reitebuch. `JavaView`, version 3.95. `www.javaview.de`, 2005.

[76] Helmut Pottmann und Johannes Wallner. *Computational line geometry*. Springer-Verlag, Berlin, 2001.

[77] Franco P. Preparata und Se June Hong. Convex hulls of finite sets of points in two and three dimensions. *Comm. ACM*, 20(2):87–93, 1977.

[78] J.L. Rabinowitsch. Zum Hilbertschen Nullstellensatz. *Math. Ann.*, 102:520, 1929.

[79] Uwe Schöning. *Algorithmik.* Spektrum Akademischer Verlag, Heidelberg, 2001.

[80] Alexander Schrijver. *Theory of linear and integer programming*. John Wiley & Sons Ltd., Chichester, 1986.

[81] Frank Sottile und Thorsten Theobald. Line problems in nonlinear computational geometry. *Discrete and computational geometry – Twenty years later*, Contemporary Mathematics, American Mathematical Society, Providence, RI, 2007.

[82] Ralph Stöcker und Heiner Zieschang. *Algebraische Topologie*. B. G. Teubner, Stuttgart, 1988.

[83] Josef Stoer und Roland Bulirsch. *Numerische Mathematik 2*. Springer-Verlag, Berlin, 5. Auflage, 2005.

[84] Bernd Sturmfels. *Gröbner bases and convex polytopes*, Band 8 der *University Lecture Series*. American Mathematical Society, Providence, RI, 1996.

[85] Bernd Sturmfels. *Solving systems of polynomial equations*, Band 97 der *CBMS Regional Conference Series in Mathematics*. American Mathematical Society, Providence, RI, 2002.

[86] Santosh Vempala. Geometric random walks: a survey. In *Combinatorial and computational geometry*, Band 52 der *Math. Sci. Res. Inst. Publ.*, 577–616. Cambridge Univ. Press, Cambridge, 2005.

[87] Joachim von zur Gathen und Jürgen Gerhard. *Modern computer algebra*. Cambridge University Press, Cambridge, 2. Auflage, 2003.

[88] Roger Webster. *Convexity*. The Clarendon Press Oxford University Press, New York, 1994.

[89] Ingo Wegener. *Theoretische Informatik – eine algorithmenorientierte Einführung*. Teubner, Wiesbaden, 3. Auflage, 2005.

[90] Gisbert Wüstholz. *Algebra*. Vieweg, Wiesbaden, 2004.

[91] Günter M. Ziegler. *Lectures on polytopes*, Band 152 der *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1995.

Index