Einführung in die Lineare und Kombinatorische Optimierung

(Algorithmische Diskrete Mathematik I, kurz ADM I)

Skriptum zur Vorlesung im WS 2012/2013

Prof. Dr. Martin Grötschel
Institut für Mathematik
Technische Universität Berlin

Version vom 24. Oktober 2012

Vorwort

Bei dem vorliegenden Skript handelt es sich um die Ausarbeitung der vierstündigen Vorlesung „Einführung in die Lineare und Kombinatorische Optimierung“ (mit zugehörigen Übungen und Tutorien), die die grundlegende Vorlesung des dreisemestrigen Zyklus „Algorithmische Diskrete Mathematik“ bildet. Diese Vorlesung wurde von mir im Wintersemester 20012/13 zusammen mit Benjamin Hiller an der TU Berlin gehalten, der auch an der Ausarbeitung des vorliegenden Vorlesungsskripts beteiligt war.

Das erste Ziel dieser Vorlesung ist, das Verständnis für Fragestellung der mathematischen Optimierung und deren Anwendungen zu wecken und das Vorgehen bei der mathematischen Modellierung von Optimierungsproblemen aus der Praxis kennenzulernen. Das zweite und wichtigere Ziel ist die Einführung in die Methoden zur Lösung derartiger Probleme. Die erste Vorlesung des Zyklus vermittelt Grundlagen der Theorie der Graphen und Netzwerke sowie der linearen, kombinatorischen und ganzzahligen Optimierung. Hierbei wird auf algorithmische Aspekte besonderer Wert gelegt.

Grundkenntnisse der linearen Algebra werden vorausgesetzt, die aus der diskreten Mathematik (vornehmlich Graphentheorie) benötigten Begriffe und Grundlagen werden in der Vorlesung vorgestellt.

In der Vorlesung werden insbesondere kombinatorische Optimierungsprobleme behandelt, die sich graphentheoretisch formulieren lassen, wobei vornehmlich Probleme untersucht werden, die mit Hilfe polynomialer Algorithmen gelöst werden können. Hierzu gehört natürlich eine Einführung in die Komplexitätstheorie (die Klassen P und NP, NP-Vollständigkeit). Es werden gleichfalls einige Heuristiken und Approximationsverfahren für „schwere“ Probleme vorgestellt sowie Methoden zu deren Analyse. Mit der Darstellung des Simplexalgorithmus und einiger seiner theoretischen Konsequenzen (Dualitätssatz, Sätze vom komplementären Schlupf) beginnt die Einführung in die lineare Optimierung, wobei auch bereits einige Aspekte der ganzzahligen Optimierung behandelt werden.

Es gibt kein einzelnes Buch, das den gesamten, in dieser Vorlesung abgehandelten Themenkreis abdeckt. Daher sind in die einzelnen Kapitel Literaturhinweise eingearbeitet worden. Hinweise auf aktuelle Lehrbücher, die als Begleittexte zur Vorlesung geeignet sind finden sich auf der zur Vorlesung gehörigen Webseite:

`http://www.zib.de/groetschel/teaching/WS1213/Lecture-WS1213-deutsch.html`

Die vorliegende Ausarbeitung ist ein Vorlesungsskript und kein Buch. Obwohl mit der gebotenen Sorgfalt geschrieben, war nicht genügend Zeit für das bei Lehrbüchern notwendige intensive Korrekturlesen und das Einarbeiten umfassender Literaturhinweise. Die daher vermutlich vorhandenen Fehler bitte ich zu entschuldigen (und mir wenn möglich mitzuteilen). Das Thema wird nicht erschöpfend behandelt. Das Manuskript enthält nur die wesentlichen Teile der Vorlesung. Insbesondere sind die Schilderungen komplexer Anwendungsfälle, der Schwierigkeiten bei der Modellierung praktischer Probleme, der Probleme bei der praktischen Umsetzung und die Darstellung der Erfolge, die in den letzten Jahren beim Einsatz der hier vorgestellten Methodik in der Industrie erzielt wurden, nicht in das Skript aufgenommen worden.

Martin Grötschel

Inhaltsverzeichnis

1 Einführung

1.1 Einführendes Beispiel

Ein Unternehmen produziert am Standort A ein Gut, das es mit der Bahn zu den Städten B, C und D transportieren möchte. Genauer, es werden wöchentlich 6 Waggons benötigt, von denen 2 nach B, einer nach C sowie 3 nach D befördert werden müssen. Auf Anfrage des Unternehmens nennt die Bahn die maximalen wöchentlichen Beförderungskapazitäten (in Waggons) sowie die Kosten pro Waggon, siehe Tabelle 1.1.

Um sich die Situation zu veranschaulichen, macht der verantwortliche Planer eine Zeichnung der im Netz möglichen Verbindungen, ihrer Kapazitäten, sowie ihrer Kosten, siehe Abbildung 1.1. Diese Abbildung repräsentiert einen *gerichteten Graphen* (auch *Digraph* genannt) $D = (V, A)$, der aus *Knoten* V und *Bögen* (auch *gerichtete Kanten* genannt) $A \subseteq V \times V$ besteht, wobei die Bögen die Knoten miteinander verbinden. Die Knoten entsprechen dabei den Städten A, B, C und D, die Bögen den möglichen Verbindungen zwischen diesen Städten. Zusätzlich zu dieser Information enthält der Graph in Abbildung 1.1 weitere Daten, die abstrakt als *Bogengewichte* bezeichnet werden und jedem Bogen gewisse von der jeweiligen Fragestellung abhängige Werte zuordnen. Konkret haben wir die Bogengewichte $l\colon A \to \mathbb{R}_+$, $u\colon A \to \mathbb{R}_+$ und $c\colon A \to \mathbb{R}_+$, die jeweils die Mindestzahl der Waggons pro Woche (hier immer 0), maximale Anzahl der Waggons pro Woche sowie Kosten pro Waggon auf jeder Verbindung angeben. (Die „Benamung“der Variablen und Daten erfolgt entsprechend dem üblichen Vorgehen in der englischsprachigen Literatur; „V“ steht für „vertices“, „A“ für „arcs“, „l“ und „u“ für „lower bound“ und „upper bound“, „c“ für „cost“ usw.)

Der Planer möchte nun die kostengünstigste Möglichkeit bestimmen, die 6 Waggons zu ihren Zielorten zu transportieren. Dies ist ein typisches Problem der *kombinatorischen Optimierung*, welche sich (ganz allgemein formuliert) damit beschäftigt, aus einer endlichen Menge, deren Elemente „Lösungen“ genannt werden und die bewertet sind, eine

Verbindung	**maximale Anzahl Waggons pro Woche**	**Kosten pro Waggon**
A nach B	4	5
A nach C	3	1
B nach C	2	2
B nach D	3	2
C nach D	4	1

Tabelle 1.1: Kapazitäten und Kosten der Verbindungen im Bahnnetz.

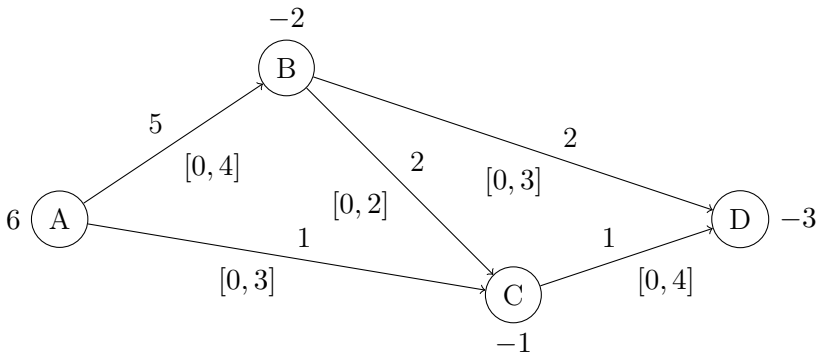


Abbildung 1.1: Beispielproblem als gerichteter Graph. Die Zahl an einem Knoten gibt an, wieviele Waggons dort bereitgestellt werden (positiv) bzw. wieviele Waggons angeliefert werden (negativ). Zahlen oberhalb eines Bogens geben die Transportkosten pro Waggon, Intervalle unterhalb eines Bogens die Kapazitäten der Verbindung an.

beste Lösung auszusuchen. Die in der kombinatorischen Optimierung auftretenden Probleme sind in der Regel „strukturiert“. In unserem Fall ist eine Grundstruktur durch einen Digraphen und die Funktionen l, u und c gegeben. Die Menge der Lösungen, über die optimiert werden soll, besteht aus der Menge aller möglichen Transporte von A zu den Zielen B, C und D. Kombinatorische Optimierungsprobleme können durch Enumeration aller Lösungen gelöst werden. Dies ist vielleicht in diesem Beispiel, aber im Allgemeinen keine gute Idee. In der Vorlesung geht es u. a. darum, mathematische Methoden zu entwickeln, um deartige Probleme (einigermaßen) effizient zu lösen.

In unserem Beispiel überlegt sich nun der Planer, dass er lediglich entscheiden muss, wieviele Waggons auf jeder der 5 Verbindungen befördert werden sollen. Dazu führt er die *Variablen* $f_{AB}, f_{AC}, f_{BC}, f_{BD}$ und f_{CD} ein. „f“ steht für „flow“ oder „Fluss“. Er überlegt sich, dass jede der Variablen mindestens den Wert 0 haben muss und nicht größer als die von der Bahn vorgegebene Kapazitätsgrenze sein darf. Außerdem notiert er sich, dass jede Variable nur ganzzahlige Werte annehmen kann. Insgesamt gelangt er so zu den Bedingungen

$$\begin{aligned} f_{AB} &\in \mathbb{Z}, \quad 0 \leq f_{AB} \leq 4, \\ f_{AC} &\in \mathbb{Z}, \quad 0 \leq f_{AC} \leq 3, \\ f_{BC} &\in \mathbb{Z}, \quad 0 \leq f_{BC} \leq 2, \\ f_{BD} &\in \mathbb{Z}, \quad 0 \leq f_{BD} \leq 3, \\ f_{CD} &\in \mathbb{Z}, \quad 0 \leq f_{CD} \leq 4. \end{aligned}$$

Mit diesen Variablen können die Gesamtkosten eines Transportes leicht als

$$5f_{AB} + f_{AC} + 2f_{BC} + 2f_{BD} + f_{CD}$$

dargestellt werden, und das Ziel ist, diese Gesamtkosten so gering wie möglich zu halten. Deswegen spricht man hier auch von der Zielfunktion (englisch: objective function). Nun

muss der Planer noch die Bedingungen festlegen, die den gewünschten Transport von A nach B, C und D beschreiben. Zunächst müssen 6 Waggons A verlassen, was sich als Gleichung

$$f_{AB} + f_{AC} = 6$$

schreiben lässt. Von den Waggons, die in B ankommen, sollen 2 dort verbleiben und der Rest weiter nach C und D fahren:

$$f_{AB} = 2 + f_{BC} + f_{BD}.$$

Analog können auch die Bedingungen in C und D formuliert werden.

Insgesamt hat der Planer nun folgende mathematische Formulierung vorliegen:

$$\begin{array}{lr}
\min 5f_{AB} + f_{AC} + 2f_{BC} + 2f_{BD} + f_{CD} & \text{(1.1a)} \\
f_{AB} + f_{AC} = 6 & \text{(1.1b)} \\
-f_{AB} + f_{BC} + f_{BD} = -2 & \text{(1.1c)} \\
-f_{AC} - f_{BC} + f_{CD} = -1 & \text{(1.1d)} \\
-f_{BD} - f_{CD} = -3 & \text{(1.1e)} \\
0 \leq f_{AB} \leq 4 & \text{(1.1f)} \\
0 \leq f_{AC} \leq 3 & \text{(1.1g)} \\
0 \leq f_{BC} \leq 2 & \text{(1.1h)} \\
0 \leq f_{BD} \leq 3 & \text{(1.1i)} \\
0 \leq f_{CD} \leq 4 & \text{(1.1j)} \\
f_{AB},\ f_{AC},\ f_{BC},\ f_{BD},\ f_{CD} \in \mathbb{Z} & \text{(1.1k)}
\end{array}$$

Ein Optimierungsproblem dieser Art, in dem alle Variablen ganzzahlig alle Nebenbedingungen lineare Gleichungen oder Ungleichungen sind, und dessen Zielfunktion ebenfalls linear ist, heißt *Ganzzahliges Lineares Programm* oder als englische Abkürzung kurz *ILP* (oft auch nur *IP*, wenn aus dem Kontext klar ist, dass Nebenbedingungen und Zielfunktion linear sind). Sind alle Variablen kontinuierlich, so spricht man von einem *Linearen Programm* oder kurz *LP*. Zum Beispiel ist das Optimierungsproblem (1.1a)–(1.1j) ein LP.

Um nun eine optimale Transportvariante zu bestimmen, beschließt der Planer, die Ganzzahligkeitsbedingungen (1.1k) zunächst zu ignorieren, da dann nur ein lineares Gleichungssystem mit Variablenschranken übrig bleibt. Dem Planer fällt auf, dass die vier Gleichungen (1.1b) bis (1.1e) linear abhängig sind, weil sie sich zu 0 summieren. Man kann sich leicht überlegen, dass eine beliebige Gleichung gestrichen werden kann und die verbleibenden drei Gleichungen linear unabhängig sind. Wie wir aus der Linearen Algebra wissen, ist dann der Lösungsraum des Gleichungssystems ein 2-dimensionaler affiner Unterraum des $\mathbb{R}^5$. Mithilfe des Gauss-Algorithmus berechnet der Planer folgende Para-

metrisierung dieses Unterraums:

$$\begin{pmatrix} f_{AB} \\ f_{AC} \\ f_{BC} \\ f_{BD} \\ f_{CD} \end{pmatrix} = \begin{pmatrix} 6 \\ 1 \\ 3 \\ 0 \\ 0 \end{pmatrix} + s \begin{pmatrix} -1 \\ -1 \\ 0 \\ 1 \\ 0 \end{pmatrix} + t \begin{pmatrix} 0 \\ 1 \\ -1 \\ 0 \\ 1 \end{pmatrix}. \tag{1.2}$$

Aus der Parametrisierung (1.2) und den Schranken (1.1f) bis (1.1j) leitet der Planer das folgende LP her:

$$\begin{aligned} \min\ & -6s + t && \text{(1.3a)} \\ & -s + t \geq -1 && \text{(1.3b)} \\ & -s + t \leq 1 && \text{(1.3c)} \\ & 2 \leq s \leq 3 && \text{(1.3d)} \\ & 0 \leq t \leq 3. && \text{(1.3e)} \end{aligned}$$

Dieses LP ist „äquivalent“ zu LP (1.1a)–(1.1j) in folgendem Sinn: Es gibt eine bijektive Abbildung zwischen den Lösungsräumen der beiden LPs, die mit den durch die Zielfunktionen gegebenen Ordnungen kompatibel ist. Mit anderen Worten: Jede Optimallösung von LP (1.3) entspricht genau einer Optimallösung von LP (1.1a)–(1.1j) und umgekehrt. Daher genügt es, das LP (1.3) zu lösen.

Da in LP (1.3) nur zwei Variablen vorkommen, kann man die zulässige Menge aller Paare (s, t), die alle Bedingungen erfüllen, graphisch darstellen (siehe Abbildung 1.2). Aus dieser Abbildung kann man direkt die optimale Lösung ablesen: Die optimale Lösung ist derjenige Punkt der grauen Fläche, der am weitesten in der dem Zielfunktionsvektor entgegengesetzten Richtung liegt (weil minimiert wird). Im Beispiel ist dies der Punkt $(3, 2)$, der der Lösung $f_{AB} = 3$, $f_{AC} = 3$, $f_{BC} = 0$, $f_{BD} = 1$, $f_{CD} = 2$ entspricht. Da alle Werte ganzzahlig sind, ist der Planer zufrieden und kommuniziert den enstprechenden Plan dem Bahnunternehmen.

Ist es immer so, dass bei ganzzahligen Problemdaten ein lineares Programm eine Optimallösung besitzt, die ganzzahlig ist? Natürlich nicht! Und deswegen geht unsere Story weiter: Nach einigen Tagen bekommt der Planer von der Bahn den Bescheid, dass die Waggons so leider nicht befördert werden können, weil die Kapazität des Rangierbahnhofs in C nicht ausreicht. Die Bahn beschreibt die Kapazitätsbeschränkung des Rangierbahnhofs genauer und der Planer bekommt die zusätzliche Bedingung

$$2f_{AC} + f_{BC} \leq 5,$$

die er via (1.2) in die äquivalente Bedingung

$$s + t \leq 4 \tag{1.4}$$

übersetzt. Das resultierende LP (1.3) mit der zusätzlichen Ungleichung (1.4) hat nun keine ganzzahlige Optimallösung mehr (siehe Abbildung 1.2); die Optimallösung ist $(2.5, 1.5)$ bzw. $f_{AB} = 3.5$, $f_{AC} = 2.5$, $f_{BC} = 0$, $f_{BD} = 1.5$, $f_{CD} = 1.5$ im Originalproblem.

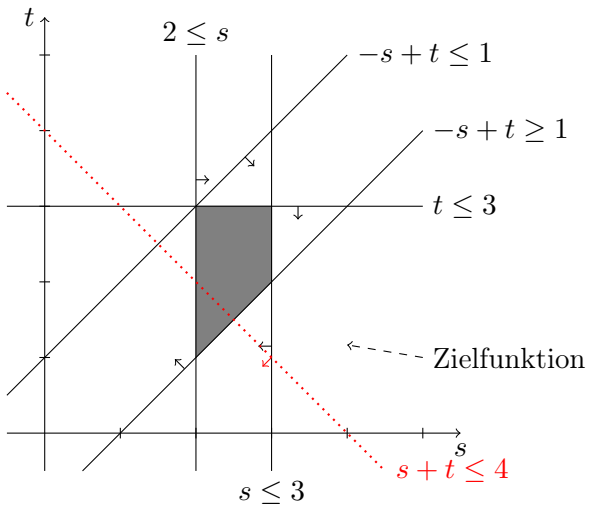


Abbildung 1.2: Graphische Darstellung des Optimierungsproblems (1.3). Die graue Menge ist die Menge der zulässigen Lösungen. Der gestrichelte Pfeil zeigt die Richtung, in der die Zielfunktion ansteigt. Die gepunktete Ungleichung entspricht der zusätzlichen Kapazitätsbedingung der Bahn.

Schluss, Variante 1: Der Planer ist nun mit seinem Schulwissen am Ende und fragt seinen Freund, einen Mathematiker, um Rat. Dieser empfiehlt ihm, sein Problem mit der Software SCIP (siehe `http://scip.zib.de`) zur Lösung ganzzahliger Optimierungsproblems zu lösen. Damit findet der Planer die neue Lösung $f_{AB} = 4$, $f_{AC} = 2$, $f_{BC} = 0$, $f_{BD} = 2$, $f_{CD} = 1$, mit der nun auch die Bahn einverstanden ist.

Schluss, Variante 2: Der Planer schaut sich die zulässige Menge in Abbildung 1.2 genau an und stellt fest, dass nur die ganzzahligen Punkte $(2, 1)$ und $(2, 2)$ zulässig sind. Er wählt den billigeren der beiden und gelangt zu der neuen Lösung $f_{AB} = 4$, $f_{AC} = 2$, $f_{BC} = 0$, $f_{BD} = 2$, $f_{CD} = 1$, mit der nun auch die Bahn einverstanden ist.

Die vorliegende Einführung ist natürlich „nur“ ein didaktisches Beispiel. Wir haben gezeigt, wie man aus einer praktischen Fragestellung (die wir so vereinfacht haben, dass ihre Lösung graphisch gefunden werden kann), ein mathematisches Problem erstellt. Man nennt solch ein Vorgehen *mathematische Modellierung*. Es ist keineswegs so, dass jedem praktischen Problem ein eindeutiges mathematisches Modell entspricht. Es gibt viele unterschiedliche Modellierungsmethoden. Für welche man sich entscheidet, ist eine Frage des Geschmacks, der Vorbildung, oder der Algorithmen, die zur Lösung der mathematischen Modelle verfügbar sind. Ein einfaches Beispiel ist die Modellierung einer ja/nein-Entscheidung: Wir möchten ausdrücken, dass eine reelle Variable x nur die Werte 0 oder 1 annehmen darf ($x = 0$ entspricht „nein“, $x = 1$ entspricht „ja“). Das können wir z. B. auf die folgenden Weisen erreichen:

- $x \in \{0, 1\}$,
- $0 \leq x \leq 1$, $x \in \mathbb{Z}$,

- $x = x^2$.

Welche Modellierung sinnvoll ist, kann man nicht grundsätzlich entscheiden. Die Wahl des Modells hängt vom gewählten Gesamtmodell und der geplanten algorithmischen Vorgehensweise ab.

1.2 Optimierungsprobleme

Wir wollen zunächst ganz informell, ohne auf technische Spitzfindigkeiten einzugehen, mathematische Optimierungsprobleme einführen. Sehr viele Probleme lassen sich wie folgt formulieren.

Gegeben seien eine Menge S und eine geordnete Menge $(T, \leq)$, d. h. zwischen je zwei Elementen $s, t \in T$ gilt genau eine der folgenden Beziehungen $s < t$, $s > t$ oder $s = t$. Ferner sei eine Abbildung $f : S \to T$ gegeben. Gesucht ist ein Element $x^* \in S$ mit der Eigenschaft $f(x^*) \geq f(x)$ für alle $x \in S$ (Maximierungsproblem) oder $f(x^*) \leq f(x)$ für alle $x \in S$ (Minimierungsproblem). Es ist üblich, hierfür eine der folgenden Schreibweisen zu benutzen:

$$\begin{array}{lll} \max_{x \in S} f(x) & \text{oder} & \max\{f(x) \mid x \in S\}, \\ \min_{x \in S} f(x) & \text{oder} & \min\{f(x) \mid x \in S\}. \end{array} \tag{1.5}$$

In der Praxis treten als geordnete Mengen $(T, \leq)$ meistens die reellen Zahlen $\mathbb{R}$, die rationalen Zahlen $\mathbb{Q}$ oder die ganzen Zahlen $\mathbb{Z}$ auf, alle mit der natürlichen Ordnung versehen (die wir deswegen auch gar nicht erst notieren). Die Aufgabe (1.5) ist viel zu allgemein, um darüber etwas Interessantes sagen zu können. Wenn S durch die Auflistung aller Elemente gegeben ist, ist das Problem entweder sinnlos oder trivial (man rechnet ganz einfach $f(x)$ für alle $x \in S$ aus). Das heißt, S muss irgendwie (explizit oder implizit) strukturiert sein, so dass vernünftige Aussagen über S möglich sind, ohne dass man alle Elemente in S einzeln kennt. Das gleiche gilt für die Funktion $f : S \to T$. Ist sie nur punktweise durch $x \mapsto f(x)$ gegeben, lohnt sich das Studium von (1.5) nicht. Erst wenn f durch hinreichend strukturierte "Formeln" bzw. "Eigenschaften" bestimmt ist, werden tieferliegende mathematische Einsichten möglich.

Die Optimierungsprobleme, die in der Praxis auftreten, haben fast alle irgendeine "vernünftige" Struktur. Das muss nicht unbedingt heißen, dass die Probleme dadurch auf einfache Weise lösbar sind, aber immerhin ist es meistens möglich, sie in das zur Zeit bekannte und untersuchte Universum der verschiedenen Typen von Optimierungsproblemen einzureihen und zu klassifizieren.

Im Laufe des Studiums werden Ihnen noch sehr unterschiedliche Optimierungsaufgaben begegnen. Viele werden von einem der folgenden Typen sein.

(1.6) Definition (Kontrollproblem) Gegeben sei ein Steuerungsprozess (z. B. die Bewegungsgleichung eines Autos), etwa der Form

$$\dot{x}(t) = f(t, x(t), u(t)),$$

wobei u eine Steuerung ist (Benzinzufuhr). Ferner seien eine Anfangsbedingung

$$x(0) = x_0,$$

(z. B.: das Auto steht) sowie eine Endbedingung

$$x(T) = x_1$$

(z. B. das Auto hat eine Geschwindigkeit von 50 km/h) gegeben. Gesucht ist eine Steuerung u für den Zeitraum $[0, T]$, so dass z. B.

$$\int_0^T |u|^2 \, \mathrm{d}t$$

minimal ist (etwa minimaler Benzinverbrauch). △

(1.7) Definition (Approximationsproblem) Gegeben sei eine (numerisch schwierig auszuwertende) Funktion f, finde eine Polynom p vom Grad n, so dass

$$\|f - p\| \quad \text{oder} \quad \|f - p\|_\infty$$

minimal ist. △

(1.8) Definition (Nichtlineares Optimierungsproblem) Es seien f, g_i $(i = 1, \ldots, m)$, h_j $(j = 1, \ldots, p)$ differenzierbare Funktionen von $\mathbb{R}^n \to \mathbb{R}$, dann heißt

$$\begin{aligned} &\min f(x) \\ &g_i(x) \leq 0 \qquad i = 1, \ldots, m \\ &h_i(x) = 0 \qquad j = 1, \ldots, p \\ &\quad x \in \mathbb{R}^n \end{aligned}$$

ein nichtlineares Optimierungsproblem. Ist eine der Funktionen nicht differenzierbar, so spricht man von einem nichtdifferenzierbaren Optimierungsproblem. Im Allgemeinen wird davon ausgegangen, daß alle betrachteten Funktionen zumindest stetig sind. △

(1.9) Definition (Konvexes Optimierungsproblem) Eine Menge $S \subseteq \mathbb{R}^n$ heißt **konvex**, falls gilt: Sind $x, y \in S$ und ist $\lambda \in \mathbb{R}$, $0 \leq \lambda \leq 1$, dann gilt $\lambda x + (1 - \lambda)y \in S$. Eine Funktion $f : \mathbb{R}^n \to \mathbb{R}$ heißt **konvex**, falls für alle $\lambda \in \mathbb{R}$, $0 \leq \lambda \leq 1$ und alle $x, y \in \mathbb{R}^n$ gilt

$$\lambda f(x) + (1 - \lambda) f(y) \geq f(\lambda x + (1 - \lambda) y).$$

Ist $S \subseteq \mathbb{R}^n$ konvex (z. B. kann S wie folgt gegeben sein $S = \{x \in \mathbb{R}^n \mid g_i(x) \leq 0,\ i = 1, \ldots, m\}$ wobei die g_i konvexe Funktionen sind), und ist $f : \mathbb{R}^n \to \mathbb{R}$ eine konvexe Funktion, dann ist

$$\min_{x \in S} f(x)$$

ein konvexes Minimierungsproblem. △

(1.10) Definition (Lineares Optimierungsproblem (Lineares Programm))
Gegeben seien $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{(m,n)}$, $b \in \mathbb{R}^m$, dann heißt

$$\begin{aligned} &\max c^T x \\ &Ax \leq b \\ &x \in \mathbb{R}^n \end{aligned}$$

lineares Optimierungsproblem. △

(1.11) Definition (Lineares ganzzahliges Optimierungsproblem) Gegeben seien $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{(m,n)}$, $b \in \mathbb{R}^m$, dann heißt

$$\begin{aligned} &\max c^T x \\ &Ax \leq b \\ &x \in \mathbb{Z}^n \end{aligned}$$

lineares ganzzahliges (oder kurz: ganzzahliges) Optimierungsproblem. △

Selbst bei Optimierungsproblemen wie (1.6), . . . , (1.11), die nicht sonderlich allgemein erscheinen mögen, kann es sein, dass (bei spezieller Wahl der Zielfunktion f und der Nebenbedingungen) die Aufgabenstellung (finde ein $x^* \in S$, so dass $f(x^*)$ so groß (oder klein) wie möglich ist) keine vernünftige Antwort besitzt. Es mag sein, dass f über S unbeschränkt ist; f kann beschränkt sein über S, aber ein Maximum kann innerhalb S nicht erreicht werden, d. h., das „max“ müßte eigentlich durch „sup“ ersetzt werden. S kann leer sein, ohne dass dies a priori klar ist, etc. Der Leser möge sich Beispiele mit derartigen Eigenschaften überlegen! Bei der Formulierung von Problemen dieser Art muss man sich also Gedanken darüber machen, ob die betrachtete Fragestellung überhaupt eine sinnvolle Antwort erlaubt.

In unserer Vorlesung werden wir uns lediglich mit den Problemen (1.10) und (1.11) beschäftigen. Das lineare Optimierungsproblem (1.10) ist sicherlich das derzeit für die Praxis bedeutendste Problem, da sich außerordentlich viele und sehr unterschiedliche reale Probleme als lineare Programme formulieren lassen, bzw. durch die Lösung einer endlichen Anzahl von LPs gelöst werden können. Außerdem liegt eine sehr ausgefeilte Theorie vor. Mit den modernen Verfahren der linearen Optimierung können derartige Probleme mit Hunderttausenden (und manchmal sogar mehr) von Variablen und Ungleichungen fast „mühelos“ gelöst werden. Dagegen ist Problem (1.11) viel schwieriger. Die Einschränkung der Lösungsmenge auf die zulässigen ganzzahligen Lösungen führt direkt zu einem Sprung im Schwierigkeitsgrad des Problems. Verschiedene spezielle lineare ganzzahlige Programme können in beliebiger Größenordnung gelöst werden. Bei wenig strukturierten allgemeinen Problemen des Typs (1.11) versagen dagegen auch die besten Lösungsverfahren manchmal bereits bei weniger als 100 Variablen und Nebenbedingungen.

Über Kontrolltheorie (Probleme des Typs (1.6)), Approximationstheorie (Probleme des Typs (1.7)), Nichtlineare Optimierung (Probleme des Typs (1.8)) und (1.9) werden an der TU Berlin Spezialvorlesungen angeboten. Es ist anzumerken, dass sich sowohl die

Theorie als auch die Algorithmen zur Lösung von Problemen des Typs (1.6) bis (1.9) ganz erheblich von denen zur Lösung von Problemen des Typs (1.10) und (1.11) unterscheiden.

Ziel dieser Vorlesung ist zunächst, das Verständnis für Fragestellung der Optimierung und deren Anwendungen zu wecken. Die Studierenden sollen darüber hinaus natürlich in die Optimierungstheorie eingeführt werden und einige Werkzeuge theoretischer und algorithmischer Natur zur Lösung von linearen, kombinatorischen und ganzzahligen Optimierungsproblemen kennenlernen. Damit wird grundlegendes Rüstzeug zur Behandlung (Modellierung, numerischen Lösung) von brennenden Fragen der heutigen Zeit bereitgestellt. Die wirklich wichtigen Fragen benötigen zur ihrer Lösung jedoch erheblich mehr Mathematik sowie weitergehendes algorithmisches und informationstechnisches Know-how. Und ohne enge Zusammenarbeit mit Fachleuten aus Anwendungsdisziplinen wird man die meisten Fragen nicht anwendungsadäquat beantworten können. Einige Themen von derzeit großer Bedeutung, bei denen der Einsatz von mathematischer Optimierung (in Kombination mit vielfältigen Analysetechniken und praktischen Erfahrungen aus anderen Disziplinen) wichtig ist, hat der Präsident von acatech (Deutsche Akademie der Technikwissenschaften), Reinhard F. Hüttl, zu Beginn seiner Rede zur Eröffnung der Festveranstaltung 2012 am 16. Oktober 2012 im Konzerthaus am Gendarmenmarkt in Berlin beschrieben:

> *„Als wir im letzten Jahr an diesem Ort zur acatech-Festveranstaltung zusammenkamen, war die Energiewende bereits das, was man bürokratisch als „Beschlusslage“ bezeichnen würde. Gut ein Jahr nach diesem Beschluss bestimmen mehr Fragen als Antworten die Diskussion um unsere zukünftige Energieversorgung: Was wollen wir erreichen? - Den schnellstmöglichen Ausstieg aus der Kernenergie? Eine Verlangsamung des Klimawandels? Den raschen Ausbau erneuerbarer Energien? Möglichst hohe Energieeffizienz? Und wer sollte für welches Ziel und welche Maßnahme die Umsetzung verantworten? Und vor allem: Welche Ziele haben Priorität, und, um die dominierende Frage der letzten Tage und Wochen aufzugreifen, zu welchem Preis – im wörtlichen wie im übertragenen Sinne – wollen und können wir diese Ziele erreichen? Die Debatte der vergangenen Zeit hat gezeigt, dass es auf diese Fragen viele Antworten gibt. – In etwa so viele, wie es Interessensgruppen, Verbände, Unternehmen, Parteien und Parlamente gibt. Vielleicht lässt genau deshalb die Umsetzung der Energiewende auf sich warten. Auch die Wissenschaft hat keine endgültigen Antworten auf diese zentrale gesellschaftliche Frage. Aber: Wissenschaft kann helfen, Daten und Fakten zu sichten, ans Licht zu befördern und zu gewichten. Wissenschaft kann Handlungsoptionen darstellen und auch bewerten. Dies tun wir in unseren aktuellen Publikationen zu Fragen der Energie, der Mobilität, aber auch zu vielen anderen Themen. Das Entscheidungsprimat aber – und dies ist mir gerade in diesem Kontext sehr wichtig – lag und liegt bei der Politik.“*

Die mathematische Optimierung ist eine der wissenschaftlichen Disziplinen, die bei der Behandlung der skizierten Problemfelder und Fragen eingesetzt werden muss. Aber sie

ist hier nur eine Hilfswissenschaft, denn die Formulierung von Gewichtungen und Zielen und die Bewertung von Ergebnissen unterliegen komplexen gesellschaftlichen Debatten. Wir sollten jedoch die Bedeutung der Mathematik hierbei nicht unterschätzen, sie hilft komplexe politische Entscheidungen (ein wenig) rationaler zu machen.

2 Grundlagen und Notation

2.1 Graphen und Digraphen: Wichtige Definitionen und Bezeichnungen

Bei der nachfolgenden Zusammenstellung von Begriffen und Bezeichnungen aus der Graphentheorie handelt es sich nicht um eine didaktische Einführung in das Gebiet der diskreten Mathematik. Dieses Kapitel ist lediglich als Nachschlagewerk gedacht, in dem die wichtigsten Begriffe und Bezeichnungen zusammengefasst und definiert sind.

2.1.1 Grundbegriffe der Graphentheorie

Die Terminologie und Notation in der Graphentheorie ist leider sehr uneinheitlich. Wir wollen daher hier einen kleinen Katalog wichtiger graphentheoretischer Begriffe und Bezeichnungen zusammenstellen und zwar in der Form, wie sie (in der Regel) in meinen Vorlesungen benutzt werden. Definitionen werden durch Kursivdruck hervorgehoben. Nach einer Definition folgen gelegentlich (in Klammern) weitere Bezeichnungen, um auf alternative Namensgebungen in der Literatur hinzuweisen.

Es gibt sehr viele Bücher über Graphentheorie. Wenn man zum Beispiel in der Datenbank MATH des Zentralblattes für Mathematik nach Büchern sucht, die den Begriff „graph theory“ im Titel enthalten, erhält man beinahe 400 Verweise. Bei über 50 Büchern taucht das Wort „Graphentheorie“ im Titel auf. Ich kenne natürlich nicht alle dieser Bücher. Zur Einführung in die mathematische Theorie empfehle ich u. a. Aigner (1984), Bollobás (1998), Diestel (2006)[1], Bondy and Murty (2008) und West (2005). Stärker algorithmisch orientiert und anwendungsbezogen sind z. B. Ebert (1981), Golombic (New York), Jungnickel (1994), und G. Nägler (1987) sowie Krumke and Noltemeier (2005).

Übersichtsartikel zu verschiedenen Themen der Graphentheorie sind in den Handbüchern Graham et al. (1995) und Gross and Yellen (2004) zu finden.

2.1.2 Graphen

Ein *Graph* G ist ein Tripel (V, E, Ψ) bestehend aus einer nicht-leeren Menge V, einer Menge E und einer *Inzidenzfunktion* $\Psi : E \to V^{(2)}$. Hierbei bezeichnet $V^{(2)}$ die Menge der ungeordneten Paare von (nicht notwendigerweise verschiedenen) Elementen von V. Ein Element aus V heißt *Knoten* (oder *Ecke* oder *Punkt* oder *Knotenpunkt*; englisch: *vertex* oder *node* oder *point*), ein Element aus E heißt *Kante* (englisch: *edge* oder *line*).

[1] http://www.math.uni-hamburg.de/home/diestel/books/graphentheorie/GraphentheorieIII.pdf

Zu jeder Kante $e \in E$ gibt es also Knoten $u, v \in V$ mit $\Psi(e) = uv = vu$. (In der Literatur werden auch die Symbole $[u, v]$ oder $\{u, v\}$ zur Bezeichnung des ungeordneten Paares uv benutzt. Wir lassen zur Bezeichnungsvereinfachung die Klammern weg, es sei denn, dies führt zu unklarer Notation. Zum Beispiel bezeichnen wir die Kante zwischen Knoten 1 und Knoten 23 nicht mit 123, wir schreiben dann $\{1, 23\}$.)

Die Anzahl der Knoten eines Graphen heißt *Ordnung* des Graphen. Ein Graph heißt *endlich*, wenn V und E endliche Mengen sind, andernfalls heißt G *unendlich*. Wir werden uns nur mit endlichen Graphen beschäftigen und daher ab jetzt statt "endlicher Graph" einfach "Graph" schreiben. Wie werden versuchen, die natürliche Zahl n für die Knotenzahl und die natürliche Zahl m für die Kantenzahl eines Graphen zu reservieren. (Das gelingt wegen der geringen Anzahl der Buchstaben unseres Alphabets nicht immer.)

Gilt $\Psi(e) = uv$ für eine Kante $e \in E$, dann heißen die Knoten $u, v \in V$ *Endknoten* von e, und wir sagen, dass u und v mit e *inzidieren* oder *auf* e *liegen*, dass e die Knoten u und v *verbindet*, und dass u und v *Nachbarn* bzw. *adjazent* sind. Wir sagen auch, dass zwei Kanten *inzident* sind, wenn sie einen gemeinsamen Endknoten haben. Eine Kante e mit $\Psi(e) = uu$ heißt *Schlinge*; Kanten e, f mit $\Psi(e) = uv = \Psi(f)$ heißen *parallel*, man sagt in diesem Falle auch, dass die Knoten u und v durch eine *Mehrfachkante* verbunden sind. Graphen, die weder Mehrfachkanten noch Schlingen enthalten, heißen *einfach*. Der einfache Graph, der zu jedem in G adjazenten Knotenpaar u, v mit $u \neq v$ genau eine u und v verbindende Kante enthält, heißt der G *unterliegende einfache Graph*. Mit $\Gamma(v)$ bezeichnen wir die Menge der Nachbarn eines Knotens v. Falls v in einer Schlinge enthalten ist, ist v natürlich mit sich selbst benachbart. $\Gamma(W) := \bigcup_{v \in W} \Gamma(v)$ ist die Menge der Nachbarn von $W \subseteq V$. Ein Knoten ohne Nachbarn heißt *isoliert*.

Die Benutzung der Inzidenzfunktion Ψ führt zu einem relativ aufwendigen Formalimus. Wir wollen daher die Notation etwas vereinfachen. Dabei entstehen zwar im Falle von nicht-einfachen Graphen gelegentlich Mehrdeutigkeiten, die aber i. a. auf offensichtliche Weise interpretiert werden können. Statt $\Psi(e) = uv$ schreiben wir von nun an einfach $e = uv$ (oder äquivalent $e = vu$) und meinen damit die Kante e mit den Endknoten u und v. Das ist korrekt, solange es nur eine Kante zwischen u und v gibt. Gibt es mehrere Kanten mit den Endknoten u und v, und sprechen wir von der Kante uv, so soll das heißen, dass wir einfach eine der parallelen Kanten auswählen. Von jetzt an vergessen wir also die Inzidenzfunktion Ψ und benutzen die Abkürzung $G = (V, E)$, um einen Graphen zu bezeichnen. Manchmal schreiben wir auch, wenn erhöhte Präzision erforderlich ist, E_G oder $E(G)$ bzw. V_G oder $V(G)$ zur Bezeichnung der Kanten- bzw. Knotenmenge eines Graphen G.

Zwei Graphen $G = (V, E)$ und $H = (W, F)$ heißen *isomorph*, wenn es eine bijektive Abbildung $\varphi : V \to W$ gibt, so dass $uv \in E$ genau dann gilt, wenn $\varphi(u)\varphi(v) \in F$ gilt. Isomorphe Graphen sind also — bis auf die Benamung der Knoten und Kanten — identisch.

Eine Menge F von Kanten heißt *Schnitt*, wenn es eine Knotenmenge $W \subseteq V$ gibt, so dass $F = \delta(W) := \{uv \in E \mid u \in W, v \in V \setminus W\}$ gilt; manchmal wird $\delta(W)$ der *durch* W *induzierte Schnitt* genannt. Statt $\delta(\{v\})$ schreiben wir kurz $\delta(v)$. Ein Schnitt, der keinen anderen nicht-leeren Schnitt als echte Teilmenge enthält, heißt *Cokreis* (oder *minimaler Schnitt*). Wollen wir betonen, dass ein Schnitt $\delta(W)$ bezüglich zweier Knoten $s, t \in V$

die Eigenschaft $s \in W$ und $t \in V \setminus W$ hat, so sagen wir, $\delta(W)$ ist ein *s und t trennender Schnitt* oder kurz ein *$[s,t]$-Schnitt*.

Generell benutzen wir die eckigen Klammern [. , .], um anzudeuten, dass die Reihenfolge der Objekte in der Klammer ohne Bedeutung ist. Z. B. ist ein $[s,t]$-Schnitt natürlich auch ein $[t,s]$-Schnitt, da ja $\delta(W) = \delta(V \setminus W)$ gilt.

Wir haben oben Bezeichnungen wie $\Gamma(v)$ oder $\delta(W)$ eingeführt unter der stillschweigenden Voraussetzung, dass man weiß, in Bezug auf welchen Graphen diese Mengen definiert sind. Sollten mehrere Graphen involviert sein, so werden wir, wenn Zweideutigkeiten auftreten können, die Graphennamen als Indizes verwenden, also z. B. $\Gamma_G(v)$ oder $\delta_G(V)$ schreiben. Analog wird bei allen anderen Symbolen verfahren.

Der *Grad* (oder die *Valenz*) eines Knotens v (Bezeichnung: $\deg(v)$) ist die Anzahl der Kanten, mit denen er inzidiert, wobei Schlingen doppelt gezählt werden. Hat ein Graph keine Schlingen, so ist der Grad von v gleich $|\delta(v)|$. Ein Graph heißt *k-regulär*, wenn jeder Knoten den Grad k hat, oder kurz *regulär*, wenn der Grad k nicht hervorgehoben werden soll.

Sind W eine Knotenmenge und F eine Kantenmenge in $G = (V,E)$, dann bezeichnen wir mit $E(W)$ die Menge aller Kanten von G mit beiden Endknoten in W und mit $V(F)$ die Menge aller Knoten, die Endknoten mindestens einer Kante aus F sind.

Sind $G = (V,E)$ und $H = (W,F)$ zwei Graphen, so heißt der Graph $(V \cup W, E \cup F)$ die *Vereinigung* von G und H, und $(V \cap W, E \cap F)$ heißt der *Durchschnitt* von G und H. G und H heißen *disjunkt*, falls $V \cap W = \emptyset$, *kantendisjunkt*, falls $E \cap F = \emptyset$. Wir sprechen von einer *disjunkten* bzw. *kantendisjunkten Vereinigung* von zwei Graphen, wenn sie disjunkt bzw. kantendisjunkt sind.

Sind $G = (V,E)$ und $H = (W,F)$ Graphen, so dass $W \subseteq V$ und $F \subseteq E$ gilt, so heißt H *Untergraph* (oder *Teilgraph*) von G. Falls $W \subseteq V$, so bezeichnet $G - W$ den Graphen, den man durch *Entfernen* (oder *Subtrahieren*) aller Knoten in W und aller Kanten mit mindestens einem Endknoten in W gewinnt. $G[W] := G - (V \setminus W)$ heißt der *von W induzierte Untergraph* von G. Es gilt also $G[W] = (W, E(W))$. Für $F \subseteq E$ ist $G - F := (V, E \setminus F)$ der Graph, den man durch *Entfernen* (oder Subtrahieren) der Kantenmenge F enthält. Statt $G - \{f\}$ schreiben wir $G - f$, analog schreiben wir $G - w$ statt $G - \{w\}$ für $w \in V$. Ein Untergraph $H = (W,F)$ von $G = (V,E)$ heißt *aufspannend*, falls $V = W$ gilt.

Ist $G = (V,E)$ ein Graph und $W \subseteq V$ eine Knotenmenge, so bezeichnen wir mit $G \cdot W$ den Graphen, der durch *Kontraktion der Knotenmenge W* entsteht. Das heißt, die Knotenmenge von $G \cdot W$ besteht aus den Knoten $V \setminus W$ und einem neuen Knoten w, der die Knotenmenge W ersetzt. Die Kantenmenge von $G \cdot W$ enthält alle Kanten von G, die mit keinem Knoten aus W inzidieren, und alle Kanten, die genau einen Endknoten in W haben, aber dieser Endknoten wird durch w ersetzt (also können viele parallele Kanten entstehen). Keine der Kanten von G, die in $E(W)$ liegen, gehört zu $G \cdot W$. Falls $e = uv \in E$ und falls G keine zu e parallele Kante enthält, dann ist der Graph, der durch *Kontraktion der Kante e* entsteht (Bezeichnung $G \cdot e$), der Graph $G \cdot \{u,v\}$. Falls G zu e parallele Kanten enthält, so erhält man $G \cdot e$ aus $G \cdot \{u,v\}$ durch Addition von so vielen Schlingen, die den neuen Knoten w enthalten, wie es Kanten in G parallel zu e gibt. Der Graph $G \cdot F$, den man durch *Kontraktion einer Kantenmenge $F \subseteq E$* erhält, ist der

Graph, der durch sukzessive Kontraktion (in beliebiger Reihenfolge) der Kanten aus F gewonnen wird. Ist e eine Schlinge von G, so sind $G \cdot e$ und $G - e$ identisch.

Ein einfacher Graph heißt *vollständig*, wenn jedes Paar seiner Knoten durch eine Kante verbunden ist. Offenbar gibt es — bis auf Isomorphie — nur einen vollständigen Graphen mit n Knoten. Dieser wird mit K_n bezeichnet. Ein Graph G, dessen Knotenmenge V in zwei disjunkte nicht-leere Teilmengen V_1, V_2 mit $V_1 \cup V_2 = V$ zerlegt werden kann, so dass keine zwei Knoten in V_1 und keine zwei Knoten in V_2 benachbart sind, heißt *bipartit* (oder *paar*). Die Knotenmengen V_1, V_2 nennt man eine *Bipartition* (oder *2-Färbung*) von G. Falls G zu je zwei Knoten $u \in V_1$ und $v \in V_2$ genau eine Kante uv enthält, so nennt man G *vollständig bipartit*. Den — bis auf Isomorphie eindeutig bestimmten — vollständig bipartiten Graphen mit $|V_1| = m, |V_2| = n$ bezeichnen wir mit $K_{m,n}$.

Ist G ein Graph, dann ist das *Komplement* von G, bezeichnet mit $\overline{G}$, der einfache Graph, der dieselbe Knotenmenge wie G hat und bei dem zwei Knoten genau dann durch eine Kante verbunden sind, wenn sie in G nicht benachbart sind. Ist G einfach, so gilt $\overline{\overline{G}} = G$. Der *Kantengraph* (englisch: *line graph*) $L(G)$ eines Graphen G ist der einfache Graph, dessen Knotenmenge die Kantenmenge von G ist und bei dem zwei Knoten genau dann adjazent sind, wenn die zugehörigen Kanten in G einen gemeinsamen Endknoten haben.

Eine *Clique* in einem Graphen G ist eine Knotenmenge Q, so dass je zwei Knoten aus Q in G benachbart sind. Eine *stabile Menge* in einem Graphen G ist eine Knotenmenge S, so dass je zwei Knoten aus S in G nicht benachbart sind. Für stabile Mengen werden auch die Begriffe *unabhängige Knotenmenge* oder *Coclique* verwendet. Eine Knotenmenge K in G heißt *Knotenüberdeckung* (oder *Überdeckung von Kanten durch Knoten*), wenn jede Kante aus G mit mindestens einem Knoten in K inzidiert. Die größte Kardinalität (= Anzahl der Elemente) einer stabilen Menge (bzw. Clique) in einem Graphen bezeichnet man mit $\alpha(G)$ (bzw. $\omega(G)$); die kleinste Kardinalität einer Knotenüberdeckung mit $\tau(G)$.

Eine Kantenmenge M in G heißt *Matching* (oder *Paarung* oder *Korrespondenz* oder *unabhängige Kantenmenge*), wenn M keine Schlingen enthält und je zwei Kanten in M keinen gemeinsamen Endknoten besitzen. M heißt *perfekt*, wenn jeder Knoten von G Endknoten einer Kante des Matchings M ist. Ein perfektes Matching wird auch *1-Faktor* genannt. Eine Kantenmenge F in G heißt *k-Faktor* (oder *perfektes k-Matching*), wenn jeder Knoten von G in genau k Kanten aus F enthalten ist. Eine *Kantenüberdeckung* (oder *Überdeckung von Knoten durch Kanten*) ist eine Kantenmenge, so dass jeder Knoten aus G mit mindestens einer Kante dieser Menge inzidiert. Die größte Kardinalität eines Matchings in G bezeichnet man mit $\nu(G)$, die kleinste Kardinalität einer Kantenüberdeckung mit $\rho(G)$.

Eine Zerlegung der Knotenmenge eines Graphen in stabile Mengen, die so genannten *Farbklassen*, heißt *Knotenfärbung*; d. h. die Knoten werden so gefärbt, dass je zwei benachbarte Knoten eine unterschiedliche Farbe haben. Eine Zerlegung der Kantenmenge in Matchings heißt *Kantenfärbung*; die Kanten werden also so gefärbt, dass je zwei inzidente Kanten verschieden gefärbt sind. Eine Zerlegung der Knotenmenge von G in Cliquen heißt *Cliquenüberdeckung* von G. Die minimale Anzahl von stabilen Mengen (bzw. Cliquen) in einer Knotenfärbung (bzw. Cliquenüberdeckung) bezeichnet man mit

$\chi(G)$ (bzw. $\overline{\chi}(G)$), die minimale Anzahl von Matchings in einer Kantenfärbung mit $\gamma(G)$. Die Zahl $\gamma(G)$ heißt *chromatischer Index* (oder *Kantenfärbungszahl*), $\chi(G)$ *Färbungszahl* (oder *Knotenfärbungszahl* oder *chromatische Zahl*).

Ein Graph $G = (V, E)$ kann in die Ebene gezeichnet werden, indem man jeden Knoten durch einen Punkt repräsentiert und jede Kante durch eine Kurve (oder Linie oder Streckenstück), die die beiden Punkte verbindet, die die Endknoten der Kante repräsentieren. Ein Graph heißt *planar* (oder plättbar), falls er in die Ebene gezeichnet werden kann, so dass sich keine zwei Kanten (d. h. die sie repräsentierenden Kurven) schneiden — außer möglicherweise in ihren Endknoten. Eine solche Darstellung eines planaren Graphen G in der Ebene nennt man auch *Einbettung* von G in die Ebene.

2.1.3 Digraphen

Die Kanten eines Graphen haben keine Orientierung. In vielen Anwendungen spielen aber Richtungen eine Rolle. Zur Modellierung solcher Probleme führen wir gerichtete Graphen ein. Ein *Digraph* (oder *gerichteter Graph*) $D = (V, A)$ besteht aus einer (endlichen) nichtleeren *Knotenmenge* V und einer (endlichen) Menge A von *Bögen* (oder *gerichteten Kanten*; englisch: *arc*). Ein *Bogen* a ist ein geordnetes Paar von Knoten, also $a = (u, v)$, u ist der *Anfangs-* oder *Startknoten*, v der *End-* oder *Zielknoten* von a; u heißt *Vorgänger* von v, v *Nachfolger* von u, a *inzidiert mit* u und v. (Um exakt zu sein, müssten wir hier ebenfalls eine Inzidenzfunktion $\Psi = (t, h) : A \to V \times V$ einführen. Für einen Bogen $a \in A$ ist dann $t(a)$ der Anfangsknoten (englisch: tail) und $h(a)$ der Endknoten (englisch: head) von a. Aus den bereits oben genannten Gründen wollen wir jedoch die Inzidenzfunktion nur in Ausnahmefällen benutzen.) Wie bei Graphen gibt es auch hier *parallele Bögen* und *Schlingen*. Die Bögen (u, v) und (v, u) heißen *antiparallel*.

In manchen Anwendungsfällen treten auch "Graphen" auf, die sowohl gerichtete als auch ungerichtete Kanten enthalten. Wir nennen solche Objekte *gemischte Graphen* und bezeichnen einen gemischten Graphen mit $G = (V, E, A)$, wobei V die Knotenmenge, E die Kantenmenge und A die Bogenmenge von G bezeichnet.

Falls $D = (V, A)$ ein Digraph ist und $W \subseteq V, B \subseteq A$, dann bezeichnen wir mit $A(W)$ die Menge der Bögen, deren Anfangs- und Endknoten in W liegen, und mit $V(B)$ die Menge der Knoten, die als Anfangs- oder Endknoten mindestens eines Bogens in B auftreten. Unterdigraphen, induzierte Unterdigraphen, aufspannende Unterdigraphen, Vereinigung und Durchschnitt von Digraphen, das Entfernen von Bogen- oder Knotenmengen und die Kontraktion von Bogen- oder Knotenmengen sind genau wie bei Graphen definiert.

Ist $D = (V, A)$ ein Digraph, dann heißt der Graph $G = (V, E)$, der für jeden Bogen $(i, j) \in A$ eine Kante ij enthält, der D *unterliegende Graph*. Analog werden der D *unterliegende einfache Graph* und der *unterliegende einfache Digraph* definiert. Wir sagen, dass ein Digraph eine "ungerichtete" Eigenschaft hat, wenn der ihm unterliegende Graph diese Eigenschaft hat (z. B., D ist bipartit oder planar, wenn der D unterliegende Graph G bipartit oder planar ist). Geben wir jeder Kante ij eines Graphen G eine Orientierung, d. h., ersetzen wir ij durch einen der Bögen (i, j) oder (j, i), so nennen wir den so entstehenden Digraphen D *Orientierung* von G.

Ein einfacher Digraph heißt *vollständig*, wenn je zwei Knoten $u \neq v$ durch die beiden

Bögen $(u,v),(v,u)$ verbunden sind. Ein *Turnier* ist ein Digraph, der für je zwei Knoten $u \neq v$ genau einen der Bögen (u,v) oder (v,u) enthält. (Der einem Turnier unterliegende Graph ist also ein vollständiger Graph; jedes Turnier ist die Orientierung eines vollständigen Graphen.)

Für $W \subseteq V$ sei $\delta^+(W) := \{(i,j) \in A |\ i \in W, j \notin W\}, \delta^-(W) := \delta^+(V \setminus W)$ und $\delta(W) := \delta^+(W) \cup \delta^-(W)$. Die Bogenmenge $\delta^+(W)$ (bzw. $\delta^-(W)$) heißt *Schnitt*. Ist $s \in W$ und $t \notin W$, so heißt $\delta^+(W)$ auch (s,t)-Schnitt. (Achtung: in einem Digraphen ist ein (s,t)-Schnitt kein (t,s)-Schnitt!)

Statt $\delta^+(\{v\}), \delta^-(\{v\}), \delta(\{v\})$ schreiben wir $\delta^+(v), \delta^-(v), \delta(v)$. Der *Außengrad* (*Innengrad*) von v ist die Anzahl der Bögen mit Anfangsknoten (Endknoten) v. Die Summe von Außengrad und Innengrad ist der *Grad* von v. Ein Schnitt $\delta^+(W), \emptyset \neq W \neq V$, heißt *gerichteter Schnitt*, falls $\delta^-(W) = \emptyset$, d. h. falls $\delta(W) = \delta^+(W)$. Ist $r \in W$, so sagen wir auch, dass $\delta^+(W)$ ein *Schnitt mit Wurzel* r ist.

2.1.4 Ketten, Wege, Kreise, Bäume

Das größte Durcheinander in der graphentheoretischen Terminologie herrscht bei den Begriffen Kette, Weg, Kreis und bei den damit zusammenhängenden Namen. Wir haben uns für folgende Bezeichnungen entschieden.

In einem Graphen oder Digraphen heißt eine endliche Folge $W = (v_0, e_1, v_1, e_2, v_2, \cdots, e_k, v_k)$, $k \geq 0$, die mit einem Knoten beginnt und endet und in der Knoten und Kanten (Bögen) alternierend auftreten, so dass jede Kante (jeder Bogen) e_i mit den beiden Knoten v_{i-1} und v_i inzidiert, eine *Kette*. Der Knoten v_0 heißt *Anfangsknoten*, v_k *Endknoten* der Kette; die Knoten $v_1, \ldots, v_{k-1}$ heißen *innere Knoten*; W wird auch $[v_0, v_k]$-*Kette* genannt. Die Zahl k heißt *Länge* der Kette (= Anzahl der Kanten bzw. Bögen in W, wobei einige Kanten/Bögen mehrfach auftreten können und somit mehrfach gezählt werden). Abbildung 1 (b) zeigt eine Kette der Länge 13 im Graphen G aus Abbildung 1 (a). Aus einem solchen Bild kann man in der Regel nicht entnehmen, in welcher Reihenfolge die Kanten durchlaufen werden.

Falls (in einem Digraphen) alle Bögen e_i der Kette W der Form (v_{i-1}, v_i) (also gleichgerichtet) sind, so nennt man W *gerichtete Kette* bzw. (v_0, v_k)-*Kette*. Ist $W = (v_0, e_1, v_1, \ldots, e_k, v_k)$ eine Kette, und sind i, j Indizes mit $0 \leq i < j \leq k$, dann heißt die Kette $(v_i, e_{i+1}, v_{i+1}, \ldots, e_j, v_j)$ das $[v_i, v_j]$-*Segment* (bzw. (v_i, v_j)-*Segment*, wenn W gerichtet ist) von W. Jede (gerichtete) Kante, die zwei Knoten der Kette W miteinander verbindet, die aber nicht Element von W ist, heißt *Diagonale* (oder *Sehne*) von W.

Gibt es in einem Graphen keine parallelen Kanten, so ist eine Kette W bereits durch die Folge $(v_0, \ldots, v_k)$ ihrer Knoten eindeutig festgelegt. Desgleichen ist in einem Digraphen ohne parallele Bögen eine gerichtete Kette durch die Knotenfolge $(v_0, \ldots, v_k)$ bestimmt. Zur Bezeichnungsvereinfachung werden wir daher häufig von der Kette $(v_0, \ldots, v_k)$ in einem Graphen bzw. der gerichteten Kette $(v_0, \ldots, v_k)$ in einem Digraphen sprechen, obgleich bei parallelen Kanten (Bögen) die benutzten Kanten (Bögen) hiermit nicht eindeutig festgelegt sind. Diese geringfügige Ungenauigkeit sollte aber keine Schwierigkeiten bereiten. Gelegentlich interessiert man sich mehr für die Kanten (Bögen) einer Kette, insbesondere wenn diese ein Weg oder ein Kreis (siehe unten) ist. In solchen Fällen ist es

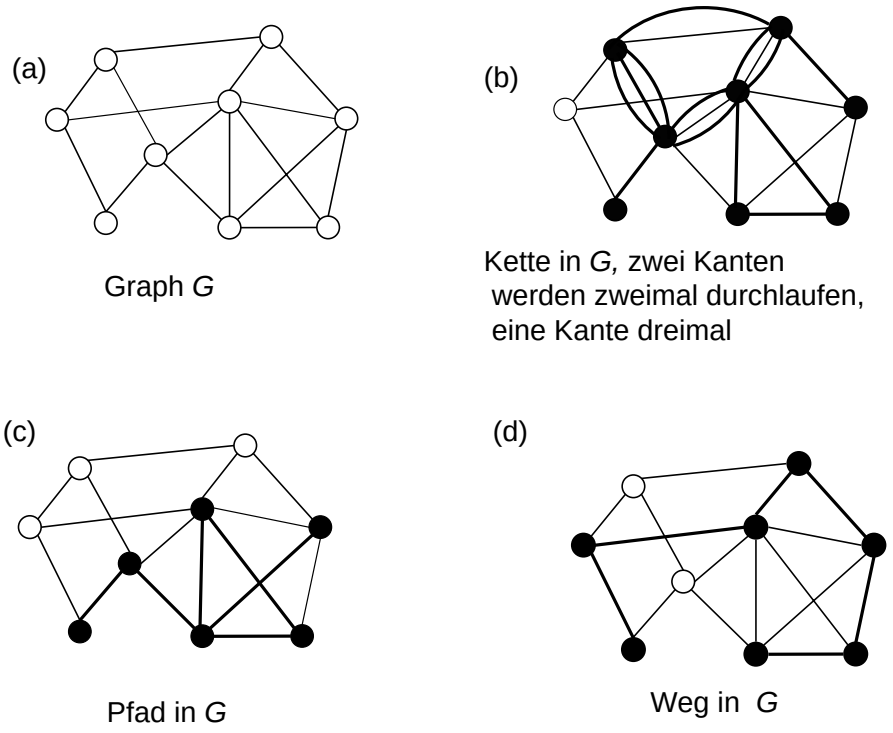


Abbildung 2.1: Kette, Pfad und Weg in einem Graph G.

zweckmäßiger, eine Kette als Kantenfolge $(e_1, e_2, \ldots, e_k)$ zu betrachten. Ist C die Menge der Kanten (Bögen) eines Kreises oder eines Weges, so spricht man dann einfach vom Kreis oder Weg C, während $V(C)$ die Menge der Knoten des Kreises oder Weges bezeichnet. Je nach behandeltem Themenkreis wird hier die am besten geeignete Notation benutzt.

Eine Kette, in der alle Knoten voneinander verschieden sind, heißt *Weg* (siehe Abbildung 1 (d)). Eine Kette, in der alle Kanten oder Bögen verschieden sind, heißt *Pfad*. Ein Beispiel ist in Abb. 1 (c) dargestellt. Ein Weg ist also ein Pfad, aber nicht jeder Pfad ist ein Weg. Ein Weg oder Pfad in einem Digraphen, der eine gerichtete Kette ist, heißt *gerichteter Weg* oder *gerichteter Pfad*. Wie bei Ketten sprechen wir von $[u, v]$-*Wegen*, (u, v)-*Wegen* etc.

Im Englischen heißt Kette *walk* oder *chain*. Im Deutschen benutzen z. B. Domschke (1982), Hässig (1979) und und Ghouila-Houri (1969) ebenfalls das Wort Kette, dagegen schreiben Aigner (1984), Diestel (2006) und Wagner (1970) hierfür "Kantenzug", während König (1936), Halin (1989) und Sachs (1970) "Kantenfolge" benutzen; Ebert (1981) schließlich nennt unsere Ketten "ungerichtete Pfade". Dieses Wirrwarr setzt sich bezüglich der Begriffe Pfad und Weg auf ähnliche Weise fort.

Eine Kette heißt *geschlossen*, falls ihre Länge nicht Null ist und falls ihr Anfangsknoten mit ihrem Endknoten übereinstimmt. Ein geschlossener (gerichteter) Pfad, bei dem

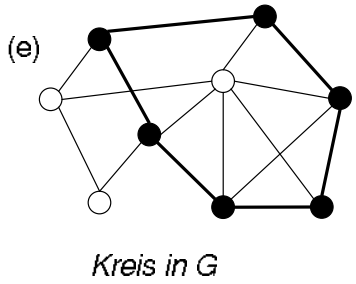


Abbildung 2.2: Ein Kreis in einem Graph G.

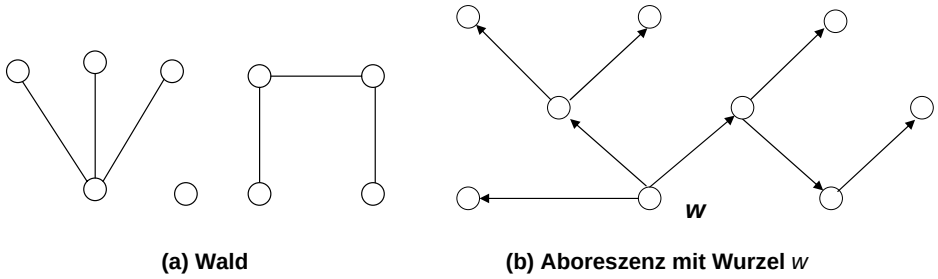


Abbildung 2.3: Ein Wald und eine Arboreszenz.

der Anfangsknoten und alle inneren Knoten voneinander verschieden sind, heißt *Kreis*, (*gerichteter Kreis*). Offensichtlich enthält jeder geschlossene Pfad einen Kreis, siehe Abb. 2.2.

Ein (gerichteter) Pfad, der jede Kante (jeden Bogen) eines Graphen (Digraphen) genau einmal enthält, heißt (gerichteter) *Eulerpfad*. Ein geschlossener Eulerpfad heißt *Eulertour*. Ein *Eulergraph* (*Eulerdigraph*) ist ein Graph (Digraph), der eine (gerichtete) Eulertour enthält.

Ein (gerichteter) Kreis (Weg) der Länge $|V|$ (bzw. $|V| - 1$) heißt (gerichteter) *Hamiltonkreis* (*Hamiltonweg*). Ein Graph (Digraph), der einen (gerichteten) Hamiltonkreis enthält, heißt *hamiltonsch*. Manchmal sagen wir statt Hamiltonkreis einfach *Tour*.

Ein *Wald* ist ein Graph, der keinen Kreis enthält, siehe Abb. 2.3(a). Ein zusammenhängender Wald heißt *Baum*. Ein Baum in einem Graphen heißt *aufspannend*, wenn er alle Knoten des Graphen enthält. Ein *Branching* B ist ein Digraph, der ein Wald ist, so dass jeder Knoten aus B Zielknoten von höchstens einem Bogen von B ist. Ein zusammenhängendes Branching heißt *Arboreszenz*, siehe Abb. 2.3(b). Eine *aufspannende Arboreszenz* ist eine Arboreszenz in einem Digraphen D, die alle Knoten von D enthält. Eine Arboreszenz enthält einen besonderen Knoten, genannt *Wurzel*, von dem aus jeder andere Knoten auf genau einem gerichteten Weg erreicht werden kann. Arboreszenzen werden auch *Wurzelbäume* genannt. Ein Digraph, der keinen gerichteten Kreis enthält, heißt *azyklisch*.

Ein Graph heißt *zusammenhängend*, falls es zu jedem Paar von Knoten s, t einen $[s,t]$-Weg in G gibt. Ein Digraph D heißt *stark zusammenhängend*, falls es zu je zwei Knoten s, t von D sowohl einen gerichteten (s,t)-Weg als auch einen gerichteten (t,s)-Weg in D gibt. Die *Komponenten* (*starken Komponenten*) eines Graphen (Digraphen) sind die bezüglich Kanteninklusion (Bogeninklusion) maximalen zusammenhängenden Untergraphen von G (maximalen stark zusammenhängenden Unterdigraphen von D). Eine Komponente heißt *ungerade Komponente*, falls ihre Knotenzahl ungerade ist, andernfalls heißt sie *gerade Komponente*.

Sei $G = (V, E)$ ein Graph. Eine Knotenmenge $W \subseteq V$ heißt *trennend*, falls $G - W$ unzusammenhängend ist. Für Graphen $G = (V, E)$, die keinen vollständigen Graphen der Ordnung $|V|$ enthalten, setzen wir $\kappa(G) := \min\{|W| \mid W \subseteq V \text{ ist trennend}\}$. Die Zahl $\kappa(G)$ heißt *Zusammenhangszahl* (oder *Knotenzusammenhangszahl*) von G. Für jeden Graphen $G = (V, E)$, der einen vollständigen Graphen der Ordnung $|V|$ enthält, setzen wir $\kappa(G) := |V| - 1$. Falls $\kappa(G) \geq k$, so nennen wir G *k-fach knotenzusammenhängend* (kurz: *k-zusammenhängend*). Ein wichtiger Satz der Graphentheorie (Satz von Menger) besagt, dass G k-fach zusammenhängend genau dann ist, wenn jedes Paar $s, t, s \neq t$, von Knoten durch mindestens k knotendisjunkte $[s,t]$-Wege miteinander verbunden ist. (Eine Menge von $[s,t]$-Wegen heißt *knotendisjunkt*, falls keine zwei Wege einen gemeinsamen inneren Knoten besitzen und die Menge der in den $[s,t]$-Wegen enthaltenen Kanten keine parallelen Kanten enthält.)

Eine Kantenmenge F eines Graphen $G = (V, E)$ heißt *trennend*, falls $G - F$ unzusammenhängend ist. Für Graphen G, die mehr als einen Knoten enthalten, setzen wir $\lambda(G) := \min\{|F| \mid F \subseteq E \text{ trennend}\}$. Die Zahl $\lambda(G)$ heißt *Kantenzusammenhangszahl*. Für Graphen G mit nur einem Knoten setzen wir $\lambda(G) = 0$. Falls $\lambda(G) \geq k$, so nennen wir G *k-fach kantenzusammenhängend* (kurz: *k-kantenzusammenhängend*). Eine Version des Menger'schen Satzes besagt, dass G k-kantenzusammenhängend genau dann ist, wenn jedes Paar $s, t, s \neq t$, von Knoten durch mindestens k kantendisjunkte $[s,t]$-Wege verbunden ist. Für Graphen G mit mindestens einem Knoten sind die Eigenschaften "G ist zusammenhängend", "G ist 1-kantenzusammenhängend" äquivalent.

Analoge Konzepte kann man in Digraphen definieren. Man benutzt hierbei den Zusatz "stark", um den "gerichteten Zusammenhang" zu kennzeichnen. Wir sagen, dass ein Digraph $D = (V, A)$ *stark k-zusammenhängend* (bzw. *stark k-bogenzusammenhängend*) ist, falls jedes Knotenpaar $s, t, s \neq t$ durch mindestens k knotendisjunkte (bzw. bogendisjunkte) (s,t)-Wege verbunden ist.

Wir setzen $\vec{\kappa}(D) := \max\{k \mid D \text{ stark } k\text{-zusammenhängend}\}$ und $\vec{\lambda}(D) := \max\{k \mid D \text{ stark } k\text{-bogenzusammenhängend}\}$; $\vec{\lambda}(D)$ heißt die *starke Zusammenhangszahl* von D, $\vec{\lambda}(D)$ die *starke Bogenzusammenhangszahl* von D.

Ein Kante e von G heißt *Brücke* (oder *Isthmus*), falls $G - e$ mehr Komponenten als G hat. Ein Knoten v von G heißt *Trennungsknoten* (oder *Artikulation*), falls die Kantenmenge E von G so in zwei nicht-leere Teilmengen E_1 und E_2 zerlegt werden kann, dass $V(E_1) \cap V(E_2) = \{v\}$ gilt. Ist G schlingenlos mit $|V| \geq 2$, dann ist v ein Trennungsknoten genau dann, wenn $\{v\}$ eine trennende Knotenmenge ist, d. h. wenn $G - v$ mehr Komponenten als G besitzt. Ein zusammenhängender Graph ohne Trennungsknoten wird *Block* genannt. Blöcke sind entweder isolierte Knoten, Schlingen oder Graphen mit 2 Knoten,

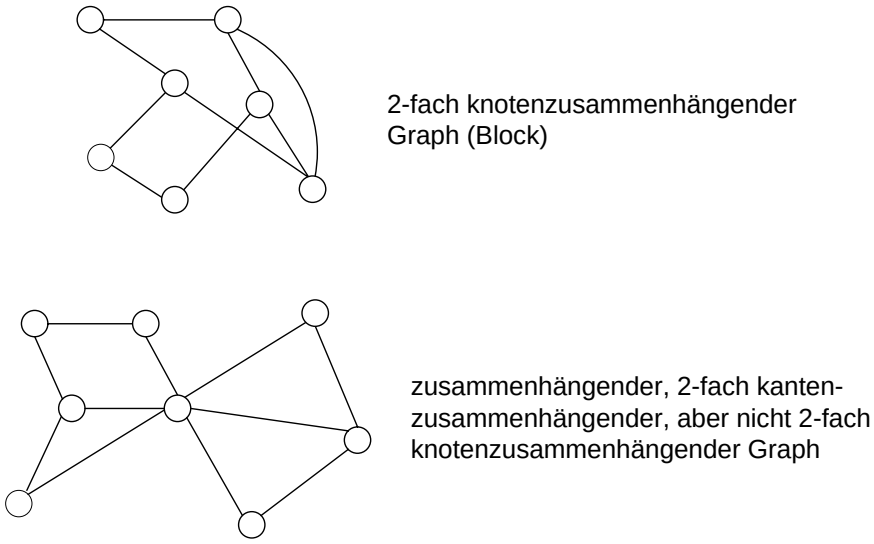


Abbildung 2.4: Ein Block und ein 2-fach kantenzusammenhängender Graph, der kein Block ist.

die durch eine Kante oder mehrere parallele Kanten verbunden sind oder, falls $|V| \geq 3$, 2-zusammenhängende Graphen. Ein *Block eines Graphen* ist ein Untergraph, der ein Block und maximal bezüglich dieser Eigenschaft ist. Jeder Graph ist offenbar die Vereinigung seiner Blöcke.

Abbildung 2.4 zeigt einen 2-fach knotenzusammenhängenden Graphen (Block) sowie einen zusammenhängenden, 2-fach kantenzusammenhängenden, aber nicht 2-fach knotenzusammenhängenden Graphen.

2.2 Lineare Algebra

2.2.1 Grundmengen

Wir benutzen folgende Bezeichnungen:

$$\begin{aligned}
\mathbb{N} &= \{1, 2, 3, \ldots\} = \text{ Menge der } \textit{natürlichen Zahlen,} \\
\mathbb{Z} &= \text{ Menge der } \textit{ganzen Zahlen,} \\
\mathbb{Q} &= \text{ Menge der } \textit{rationalen Zahlen,} \\
\mathbb{R} &= \text{ Menge der } \textit{reellen Zahlen,}
\end{aligned}$$

Mit M_+ bezeichnen wir die Menge der nichtnegativen Zahlen in M für $M \in \{\mathbb{Z}, \mathbb{Q}, \mathbb{R}\}$. Wir betrachten $\mathbb{Q}$ und $\mathbb{R}$ als Körper mit der üblichen Addition und Multiplikation und der kanonischen Ordnung "$\leq$". Desgleichen betrachten wir $\mathbb{N}$ und $\mathbb{Z}$ als mit den üblichen Rechenarten versehen. Wir werden uns fast immer in diesen Zahlenuniversen bewegen, da diese die für die Praxis relevanten sind. Manche Sätze gelten jedoch nur, wenn wir

uns auf $\mathbb{Q}$ oder $\mathbb{R}$ beschränken. Um hier eine saubere Trennung zu haben, treffen wir die folgende Konvention. Wenn wir das Symbol

$$\mathbb{K}$$

benutzen, so heißt dies immer das $\mathbb{K}$ einer der angeordneten Körper $\mathbb{R}$ oder $\mathbb{Q}$ ist. Sollte ein Satz nur für $\mathbb{R}$ oder nur für $\mathbb{Q}$ gelten, so treffen wir die jeweils notwendige Einschränkung.

Für diejenigen, die sich für möglichst allgemeine Sätze interessieren, sei an dieser Stelle folgendes vermerkt. Jeder der nachfolgend angegebenen Sätze bleibt ein wahrer Satz, wenn wir als Grundkörper $\mathbb{K}$ einen archimedisch angeordneten Körper wählen. Ein bekannter Satz besagt, dass jeder archimedisch angeordnete Körper isomorph zu einem Unterkörper von $\mathbb{R}$ ist, der $\mathbb{Q}$ enthält. Unsere Sätze bleiben also richtig, wenn wir statt $\mathbb{K} \in \{\mathbb{Q}, \mathbb{R}\}$ irgendeinen archimedisch angeordneten Körper $\mathbb{K}$ mit $\mathbb{Q} \subseteq \mathbb{K} \subseteq \mathbb{R}$ wählen.

Wir können in fast allen Sätzen (insbesondere bei denen, die keine Ganzzahligkeitsbedingungen haben) auch die Voraussetzung "archimedisch" fallen lassen, d. h. fast alle Sätze gelten auch für angeordnete Körper. Vieles, was wir im $\mathbb{K}^n$ beweisen, ist auch in beliebigen metrischen Räumen oder Räumen mit anderen als euklidischen Skalarprodukten richtig. Diejenigen, die Spaß an derartigen Verallgemeinerungen haben, sind eingeladen, die entsprechenden Beweise in die allgemeinere Sprache zu übertragen.

In dieser Vorlesung interessieren wir uns für so allgemeine Strukturen nicht. Wir verbleiben in den (für die Praxis besonders wichtigen) Räumen, die über den reellen oder rationalen Zahlen errichtet werden. Also, nochmals, wenn immer wir das Symbol $\mathbb{K}$ im weiteren gebrauchen, gilt

$$\mathbb{K} \in \{\mathbb{R}, \mathbb{Q}\},$$

und $\mathbb{K}$ ist ein Körper mit den üblichen Rechenoperationen und Strukturen. Natürlich ist $\mathbb{K}_+ = \{x \in \mathbb{K} \mid x \geq 0\}$

Die Teilmengenbeziehung zwischen zwei Mengen M und N bezeichnen wir wie üblich mit $M \subseteq N$. Gilt $M \subseteq N$ und $M \neq N$, so schreiben wir $M \subset N$. $M \setminus N$ bezeichnet die mengentheoretische Differenz $\{x \in M \mid x \notin N\}$.

2.2.2 Vektoren und Matrizen

Ist R eine beliebige Menge, $n \in \mathbb{N}$, so bezeichnen wir mit

$$R^n$$

die Menge aller *n-Tupel* oder Vektoren der Länge n mit Komponenten aus R. (Aus technischen Gründen ist es gelegentlich nützlich, Vektoren $x \in R^0$, also Vektoren ohne Komponenten, zu benutzen. Wenn wir dies tun, werden wir es explizit erwähnen, andernfalls setzen wir immer $n \geq 1$ voraus.) Wir betrachten Vektoren $x = (x_i)_{i=1,\ldots,n} \in R^n$ immer als *Spaltenvektoren* d. h.

$$x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}.$$

Wollen wir mit Zeilenvektoren rechnen, so schreiben wir x^T (lies: x *transponiert*). Die Menge $\mathbb{K}^n$ ist bekanntlich ein n-dimensionaler Vektorraum über $\mathbb{K}$. Mit

$$y^T x := \sum_{i=1}^{n} x_i y_i$$

bezeichnen wir das *innere Produkt* zweier Vektoren $x, y \in \mathbb{K}^n$. Wir nennen x und y *senkrecht (orthogonal)*, falls $x^T y = 0$ gilt. Der $\mathbb{K}^n$ ist für uns immer (wenn nichts anderes gesagt wird) mit der *euklidischen Norm*

$$\|x\| := \sqrt{x^T x}$$

ausgestattet.

Für Mengen $S, T \subseteq \mathbb{K}^n$ und $\alpha \in \mathbb{K}$ benutzen wir die folgenden Standardbezeichnungen für Mengenoperationen

$$\begin{aligned} S + T &:= \{x + y \in \mathbb{K}^n \mid x \in S, y \in T\}, \\ S - T &:= \{x - y \in \mathbb{K}^n \mid x \in S, y \in T\}, \\ \alpha S &:= \{\alpha x \in \mathbb{K}^n \mid x \in S, \}. \end{aligned}$$

Einige Vektoren aus $\mathbb{K}^n$ werden häufig auftreten, weswegen wir sie mit besonderen Symbolen bezeichnen. Mit e_j bezeichnen wir den Vektor aus $\mathbb{K}^n$, dessen j-te Komponente 1 und dessen übrige Komponenten 0 sind. Mit 0 bezeichnen wir den Nullvektor, mit $\mathbb{1}$ den Vektor, dessen Komponenten alle 1 sind. Also

$$e_j = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \quad 0 = \begin{pmatrix} 0 \\ \vdots \\ \vdots \\ \vdots \\ 0 \end{pmatrix}, \quad \mathbb{1} = \begin{pmatrix} 1 \\ \vdots \\ \vdots \\ \vdots \\ 1 \end{pmatrix}.$$

Welche Dimension die Vektoren e_j, 0, $\mathbb{1}$ haben, ergibt sich jeweils aus dem Zusammenhang.

Für eine Menge R und $m, n \in \mathbb{N}$ bezeichnet

$$R^{(m,n)} \text{ oder } R^{m \times n}$$

die Menge der (m, n)-*Matrizen* (m Zeilen, n Spalten) mit Einträgen aus R. (Aus technischen Gründen werden wir gelegentlich auch $n = 0$ oder $m = 0$ zulassen, d. h. wir werden auch Matrizen mit m Zeilen und ohne Spalten bzw. n Spalten und ohne Zeilen betrachten. Dieser Fall wird jedoch immer explizit erwähnt, somit ist in der Regel $n \geq 1$ und $m \geq 1$ vorausgesetzt.) Ist $A \in R^{(m,n)}$, so schreiben wir

$$A = (a_{ij})_{\substack{i=1,\ldots,m \\ j=1,\ldots,n}}$$

und meinen damit, dass A die folgende Form hat

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix},$$

Wenn nichts anderes gesagt wird, hat A die Zeilenindexmenge $M = \{1, \dots, m\}$ und die Spaltenindexmenge $N = \{1, \dots, n\}$. Die *j-te Spalte* von A ist ein m-Vektor, den wir mit $A_{\cdot j}$ bezeichnen,

$$A_{\cdot j} = \begin{pmatrix} a_{1j} \\ \vdots \\ a_{mj} \end{pmatrix}.$$

Die *i-te Zeile* von A ist ein Zeilenvektor der Länge n, den wir mit $A_{i\cdot}$ bezeichnen, d. h.

$$A_{i\cdot} = (a_{i1}, a_{i2}, \dots, a_{in}).$$

Wir werden in dieser Vorlesung sehr häufig Untermatrizen von Matrizen konstruieren, sie umsortieren und mit ihnen rechnen müssen. Um dies ohne Zweideutigkeiten durchführen zu können, führen wir zusätzlich eine etwas exaktere als die oben eingeführte Standardbezeichnungsweise ein.

Wir werden — wann immer es aus technischen Gründen notwendig erscheint — die Zeilen- und Spaltenindexmengen einer (m, n)-Matrix A nicht als Mengen sondern als Vektoren auffassen. Wir sprechen dann vom

$$\begin{aligned} \text{vollen Zeilenindexvektor} \quad & M = (1, 2, \dots, m) \\ \text{vollen Spaltenindexvektor} \quad & N = (1, 2, \dots, n) \end{aligned}$$

von A. Ein *Zeilenindexvektor* von A ist ein Vektor mit höchstens m Komponenten, der aus M durch Weglassen einiger Komponenten von M und Permutation der übrigen Komponenten entsteht. Analog entsteht ein *Spaltenindexvektor* durch Weglassen von Komponenten von N und Permutation der übrigen Komponenten. Sind also

$$\begin{aligned} I = (i_1, i_2, \dots, i_p) \quad & \text{ein Zeilenindexvektor von } A \text{ und} \\ J = (j_1, j_2, \dots, j_q) \quad & \text{ein Spaltenindexvektor von } A, \end{aligned}$$

so gilt immer $i_s, i_t \in \{1, \dots, m\}$ und $i_s \neq i_t$ für $1 \leq s < t \leq m$, und analog gilt $j_s, j_t \in \{1, \dots, n\}$ und $j_s \neq j_t$ für $1 \leq s < t \leq n$. Wir setzen

$$A_{IJ} := \begin{pmatrix} a_{i_1 j_1} & a_{i_1 j_2} & \dots & a_{i_1 j_q} \\ a_{i_2 j_1} & a_{i_2 j_2} & \dots & \\ \vdots & \vdots & \ddots & \vdots \\ a_{i_p j_1} & a_{i_p j_2} & \dots & a_{i_p j_q} \end{pmatrix}$$

und nennen A_{IJ} *Untermatrix von* A. A_{IJ} ist also eine (p,q)-Matrix, die aus A dadurch entsteht, dass man die Zeilen, die zu Indizes gehören, die nicht in I enthalten sind, und die Spalten, die zu Indizes gehören, die nicht in J enthalten sind, streicht und dann die so entstehende Matrix umsortiert.

Ist $I = (i)$ und $J = (j)$, so erhalten wir zwei Darstellungsweisen für Zeilen bzw. Spalten von A:

$$\begin{aligned} A_{IN} &= A_{i\cdot}\ , \\ A_{MJ} &= A_{\cdot j}\ . \end{aligned}$$

Aus Gründen der Notationsvereinfachung werden wir auch die folgende (etwas unsaubere) Schreibweise benutzen. Ist M der volle Zeilenindexvektor von A und I ein Zeilenindexvektor, so schreiben wir auch

$$I \subseteq M,$$

obwohl I und M keine Mengen sind, und für $i \in \{1,\ldots,m\}$ benutzen wir

$$i \in I \quad \text{oder } i \notin I,$$

um festzustellen, dass i als Komponente von I auftritt oder nicht. Analog verfahren wir bezüglich der Spaltenindizes.

Gelegentlich spielt die tatsächliche Anordnung der Zeilen und Spalten keine Rolle. Wenn also z. B. $I \subseteq \{1,\ldots,n\}$ und $J \subseteq \{1,\ldots,m\}$ gilt, dann werden wir auch einfach schreiben

$$A_{IJ},$$

obwohl diese Matrix dann nur bis auf Zeilen- und Spaltenpermutationen definiert ist. Wir werden versuchen, diese Bezeichnungen immer so zu verwenden, dass keine Zweideutigkeiten auftreten. Deshalb treffen wir ab jetzt die Verabredung, dass wir — falls wir Mengen (und nicht Indexvektoren) I und J benutzen — die Elemente von $I = \{i_1,\ldots,i_p\}$ und von $J = \{j_1,\ldots,j_q\}$ kanonisch anordnen, d. h. die Indizierung der Elemente von I und J sei so gewählt, dass $i_1 < i_2 < \ldots < i_p$ und $j_1 < j_2 < \ldots < j_q$ gilt. Bei dieser Verabredung ist dann A_{IJ} die Matrix die aus A durch Streichen der Zeilen i, $i \notin I$, und der Spalten j, $j \notin J$, entsteht.

Ist $I \subseteq \{1,\ldots,m\}$ und $J \subseteq \{1,\ldots,n\}$ oder sind I und J Zeilen- bzw. Splaltenindexvektoren, dann schreiben wir auch

$$A_{I\cdot} \quad \text{statt} \quad A_{IN}$$

$$A_{\cdot J} \quad \text{statt} \quad A_{MJ}$$

$A_{I\cdot}$ entsteht also aus A durch Streichen der Zeilen i, $i \notin I$, $A_{\cdot J}$ durch Streichen der Spalten j, $j \notin J$.

Sind $A, B \in \mathbb{K}^{(m,n)}$, $C \in \mathbb{K}^{(n,s)}$, $\alpha \in \mathbb{K}$, so sind

$$\begin{aligned} \text{die } \textit{Summe} &\quad A + B\ , \\ \text{das } \textit{Produkt} &\quad \alpha A, \\ \text{das } \textit{Matrixprodukt} &\quad \mathrm{AC} \end{aligned}$$

wie in der linearen Algebra üblich definiert.

Für einige häufig auftretende Matrizen haben wir spezielle Symbole reserviert. Mit 0 bezeichnen wir die Nullmatrix (alle Matrixelemente sind Null), wobei sich die Dimension der Nullmatrix jeweils aus dem Zusammenhang ergibt. (Das Symbol 0 kann also sowohl eine Zahl, einen Vektor als auch eine Matrix bezeichnen). Mit I bezeichnen wir die Einheitsmatrix. Diese Matrix ist quadratisch, die Hauptdiagonalelemente von I sind Eins, alle übrigen Null. Wollen wir die Dimension von I betonen, so schreiben wir auch I_n und meinen damit die (n,n)-Einheitsmatrix. Diejenige (m,n)-Matrix, bei der alle Elemente Eins sind, bezeichnen wir mit E. Wir schreiben auch $E_{m,n}$ bzw. E_n, um die Dimension zu spezifizieren (E_n ist eine (n,n)-Matrix). Ist x ein n-Vektor, so bezeichnet $\mathrm{diag}(x)$ diejenige (n,n)-Matrix $A = (a_{ij})$ mit $a_{ii} = x_i$ $(i = 1,\ldots,n)$ und $a_{ij} = 0$ $(i \neq j)$.

Wir halten an dieser Stelle noch einmal Folgendes fest: Wenn wir von einer Matrix A sprechen, ohne anzugeben, welche Dimension sie hat und aus welchem Bereich sie ist, dann nehmen wir implizit an, dass $A \in \mathbb{K}^{(m,n)}$ gilt. Analog gilt immer $x \in \mathbb{K}^n$, wenn sich nicht aus dem Zusammenhang anderes ergibt.

2.2.3 Kombinationen von Vektoren, Hüllen, Unabhängigkeit

Ein Vektor $x \in \mathbb{K}^n$ heißt *Linearkombination* der Vektoren $x_1,\ldots,x_k \in \mathbb{K}^n$, falls es einen Vektor $\lambda = (\lambda_1,\ldots,\lambda_k)^T \in \mathbb{K}^k$ gibt mit

$$x = \sum_{i=1}^{k} \lambda_i x_i \ .$$

Gilt zusätzlich:

$$\left.\begin{array}{l} \lambda \geq 0 \\ \lambda^T \mathbb{1} = 1 \\ \lambda \geq 0 \quad \text{und} \quad \lambda^T \mathbb{1} = 1 \end{array}\right\} \quad \text{so heißt } x \quad \left\{\begin{array}{l} \textit{konische} \\ \textit{affine} \\ \textit{konvexe} \end{array}\right\} \textit{Kombination}$$

der Vektoren $x_1,\ldots,x_k$. Diese Kombinationen heißen *echt*, falls weder $\lambda = 0$ noch $\lambda = e_j$ für ein $j \in \{1,\ldots,k\}$ gilt.

Für eine nichtleere Teilmenge $S \subseteq \mathbb{K}^n$ heißt

$$\left.\begin{array}{l} \mathrm{lin}(S) \\ \mathrm{cone}(S) \\ \mathrm{aff}(S) \\ \mathrm{conv}(S) \end{array}\right\} \quad \text{die} \quad \left\{\begin{array}{l} \textit{lineare} \\ \textit{konische} \\ \textit{affine} \\ \textit{konvexe} \end{array}\right\} \quad \textit{Hülle} \text{ von } S, \text{ d. h.}$$

die Menge aller Vektoren, die als lineare (konische, affine oder konvexe) Kombination von endlich vielen Vektoren aus S dargestellt werden können. Wir setzen außerdem

$$\begin{array}{ll} \mathrm{lin}(\emptyset) & := \mathrm{cone}(\emptyset) := \{0\}, \\ \mathrm{aff}(\emptyset) & := \mathrm{conv}(\emptyset) := \emptyset. \end{array}$$

Ist A eine (m,n)-Matrix, so schreiben wir auch

$$\mathrm{lin}(A),\ \mathrm{cone}(A),\ \mathrm{aff}(A),\ \mathrm{conv}(A)$$

und meinen damit die lineare, konische, affine bzw. konvexe Hülle der Spaltenvektoren $A_{\cdot 1}, A_{\cdot 2}, \ldots, A_{\cdot n}$ von A. Eine Teilmenge $S \subseteq \mathbb{K}^n$ heißt

$$\left\{ \begin{array}{l} \textit{linearer Raum} \\ \textit{Kegel} \\ \textit{affiner Raum} \\ \textit{konvexe Menge} \end{array} \right\} \quad \text{falls} \quad \left\{ \begin{array}{l} S = \operatorname{lin}(S) \\ S = \operatorname{cone}(S) \\ S = \operatorname{aff}(S) \\ S = \operatorname{conv}(S) \end{array} \right\}.$$

Die hier benutzten Begriffe sind üblicher Standard, wobei für „linearen Raum“ in der lineraren Algebra in der Regel *Vektorraum* oder *Untervektorraum* benutzt wird. Der Begriff *Kegel* wird jedoch – u. a. in verschiedenen Zweigen der Geometrie – allgemeiner verwendet. „Unser Kegel“ ist in der Geometrie ein „abgeschlossener konvexer Kegel“.

Eine nichtleere endliche Teilmenge $S \subseteq \mathbb{K}^n$ heißt *linear* (bzw. *affin*) *unabhängig*, falls kein Element von S als echte Linearkombination (bzw. Affinkombination) von Elementen von S dargestellt werden kann. Die leere Menge ist affin, jedoch nicht linear unabhängig. Jede Menge $S \subseteq \mathbb{K}^n$, die nicht linear bzw. affin unabhängig ist, heißt *linear* bzw. *affin abhängig*. Aus der linearen Algebra wissen wir, dass eine linear (bzw. affin) unabhängige Teilmenge des $\mathbb{K}^n$ höchstens n (bzw. $n+1$) Elemente enthält. Für eine Teilmenge $S \subseteq \mathbb{K}^n$ heißt die Kardinalität einer größten linear (bzw. affin) unabhängigen Teilmenge von S der *Rang* (bzw. *affine Rang*) von S. Wir schreiben dafür $\operatorname{rang}(S)$ bzw. $\operatorname{arang}(S)$. Die *Dimension* einer Teilmenge $S \subseteq \mathbb{K}^n$, Bezeichnung: $\dim(S)$, ist die Kardinalität einer größten affin unabhängigen Teilmenge von S minus 1, d. h. $\dim(S) = \operatorname{arang}(S) - 1$.

Der *Rang einer Matrix* A, bezeichnet mit $\operatorname{rang}(A)$, ist der Rang ihrer Spaltenvektoren. Aus der linearen Algebra wissen wir, dass $\operatorname{rang}(A)$ mit dem Rang der Zeilenvektoren von A übereinstimmt. Gilt für eine (m,n)-Matrix A, $\operatorname{rang}(A) = \min\{m,n\}$, so sagen wir, dass A *vollen Rang* hat. Eine (n,n)-Matrix mit vollem Rang ist *regulär*, d. h. sie besitzt eine (eindeutig bestimmte) inverse Matrix (geschrieben A^{-1}) mit der Eigenschaft $AA^{-1} = I$.

2.3 Polyeder und lineare Programme

Ein wichtiger Aspekt der Vorlesung ist die Behandlung von Problemen der linearen Programmierung bzw. die Modellierung von kombinatorischen Optimierungsproblemen als (ganzzahlige) lineare Programme. In diesem Abschnitt stellen wir einige grundlegende Begriffe der zugehörigen Theorie bereit, stellen dar, in welchen Formen lineare Programme vorkommen, und wie man sie ineinander transformiert. Schließlich beweisen wir als Einführung den schwachen Dualitätssatz.

(2.1) Definition

a) Eine Teilmenge $G \subseteq \mathbb{K}^n$ heißt *Hyperebene*, falls es einen Vektor $a \in \mathbb{K}^n \setminus \{0\}$ und $\alpha \in \mathbb{K}$ gibt mit

$$G = \{x \in \mathbb{K}^n \mid a^T x = \alpha\}.$$

Der Vektor a heißt *Normalenvektor* zu G.

b) Eine Teilmenge $H \subseteq \mathbb{K}^n$ heißt *Halbraum*, falls es einen Vektor $a \in \mathbb{K}^n \setminus \{0\}$ und $\alpha \in \mathbb{K}$ gibt mit

$$H = \{x \in \mathbb{K}^n \mid a^T x \leq \alpha\}.$$

Wir nennen a den *Normalenvektor* zu H. Die Hyperebene $G = \{x \in \mathbb{K}^n \mid a^T x = \alpha\}$ heißt die zum Halbraum H gehörende Hyperebene oder die H *berandende Hyperebene*, und H heißt der zu G gehörende Halbraum.

c) Eine Teilmenge $P \subseteq \mathbb{K}^n$ heißt *Polyeder*, falls es ein $m \in \mathbb{Z}_+$, eine Matrix $A \in \mathbb{K}^{(m,n)}$ und einen Vektor $b \in \mathbb{K}^m$ gibt mit

$$P = \{x \in \mathbb{K}^n \mid Ax \leq b\}.$$

Um zu betonen, dass P durch A und b definiert ist, schreiben wir auch

$$P = P(A, b) := \{x \in \mathbb{K}^n \mid Ax \leq b\}.$$

d) Ein Polyeder P heißt *Polytop*, wenn es beschränkt ist, d. h. wenn es ein $B \in \mathbb{K}$, $B > 0$ gibt mit $P \subseteq \{x \in \mathbb{K}^n \mid \|x\| \leq B\}$. △

Polyeder können wir natürlich auch in folgender Form schreiben

$$P(A, b) = \bigcap_{i=1}^{m} \{x \in \mathbb{K}^n \mid A_{i.} x \leq b_i\}.$$

Halbräume sind offensichtlich Polyeder. Aber auch die leere Menge ist ein Polyeder, denn $\emptyset = \{x \mid 0^T x \leq -1\}$, und der gesamte Raum ist ein Polyeder, denn $\mathbb{K}^n = \{x \mid 0^T x \leq 0\}$. Sind alle Zeilenvektoren $A_{i.}$ von A vom Nullvektor verschieden, so sind die bei der obigen Durchschnittsbildung beteiligten Mengen Halbräume. Ist ein Zeilenvektor von A der Nullvektor, sagen wir $A_{1.} = 0^T$, so ist $\{x \in \mathbb{K}^n \mid A_{1.} x \leq b_1\}$ entweder leer (falls $b_1 < 0$) oder der gesamte Raum $\mathbb{K}^n$ (falls $b_1 \geq 0$). Das heißt, entweder ist $P(A, b)$ leer oder die Mengen $\{x \mid A_{i.} \leq b_i\}$ mit $A_{i.} = 0$ können bei der obigen Durchschnittsbildung weggelassen werden. Daraus folgt

Jedes Polyeder $P \neq \mathbb{K}^n$ ist Durchschnitt von endlich vielen Halbräumen.

Gilt $P = P(A, b)$, so nennen wir das Ungleichungssystem $Ax \leq b$ ein P *definierendes System* (von linearen Ungleichungen). Sind $\alpha > 0$ und $1 \leq i < j \leq m$, so gilt offensichtlich

$$P(A, b) = P(A, b) \cap \{x \mid \alpha A_{i.} x \leq \alpha b_i\} \cap \{x \mid (A_{i.} + A_{j.}) x \leq b_i + b_j\}.$$

Daraus folgt, dass A und b zwar $P = P(A, b)$ eindeutig bestimmen, dass aber P unendlich viele Darstellungen der Form $P(D, d)$ hat.

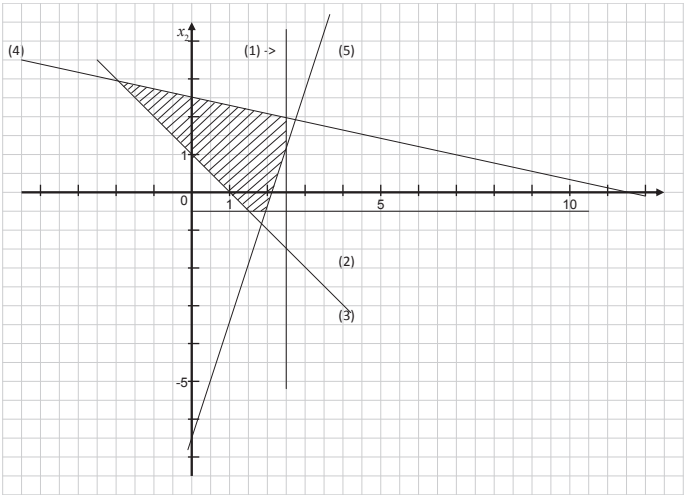


Abbildung 2.5: Darstellung des Polyeders aus Beispiel (2.2).

(2.2) Beispiel Wir betrachten das Ungleichungssystem

$$\begin{aligned} 2x_1 &\leq 5 && (1)\\ -2x_2 &\leq 1 && (2)\\ -x_1 - x_2 &\leq -1 && (3)\\ 2x_1 + 9x_2 &\leq 23 && (4)\\ 6x_1 - 2x_2 &\leq 13 && (5) \end{aligned}$$

Hieraus erhalten wir die folgende Matrix A und den Vektor b:

$$A = \begin{pmatrix} 2 & 0 \\ 0 & -2 \\ -1 & -1 \\ 2 & 9 \\ 6 & -2 \end{pmatrix}, \qquad b = \begin{pmatrix} 5 \\ 1 \\ -1 \\ 23 \\ 13 \end{pmatrix}$$

Das Polyeder $P = P(A, b)$ ist die Lösungsmenge des obigen Ungleichungssystems (1)–(5) und ist in Abbildung 2.5 graphisch dargestellt. △

Die Mengen zulässiger Lösungen linearer Programme treten nicht immer in der Form $Ax \leq b$ auf. Häufig gibt es auch Gleichungen und vorzeichenbeschränkte Variable. Vorzeichenbeschränkungen sind natürlich auch lineare Ungleichungssysteme, und ein Gleichungssytem $Dx = d$ kann in der Form von zwei Ungleichungssystemen $Dx \leq d$ und $-Dx \leq -d$ geschrieben werden. Allgemeiner gilt

(2.4) Bemerkung Die Lösungsmenge des Systems

$$\begin{array}{ll} Bx + Cy & = c \\ Dx + Ey & \leq d \\ x & \geq 0 \\ x \in \mathbb{K}^p, & y \in \mathbb{K}^q \end{array}$$

ist ein Polyeder. △

Beweis *Setze* $n := p + q$ *und*

$$A := \begin{pmatrix} B & C \\ -B & -C \\ D & E \\ -I & 0 \end{pmatrix}, \quad b := \begin{pmatrix} c \\ -c \\ d \\ 0 \end{pmatrix}.$$

Dann ist $P(A, b)$ *die Lösungsmenge des vorgegebenen Gleichungs- und Ungleichungssystems.* □

Ein spezieller Polyedertyp wird uns häufig begegnen, weswegen wir für ihn eine besondere Bezeichnung wählen wollen. Für $A \in \mathbb{K}^{(m,n)}$, $b \in \mathbb{K}^m$ setzen wir

$$P^{=}(A, b) := \{x \in \mathbb{K}^n \mid Ax = b, x \geq 0\}.$$

Nicht alle Polyeder können in der Form $P^{=}(A, b)$ dargestellt werden, z. B. nicht $P = \{x \in \mathbb{K} \mid x \leq 1\}$.

Wir werden später viele Sätze über Polyeder P beweisen, deren Aussagen *darstellungsabhängig* sind, d. h. die Art und Weise, wie P gegeben ist, geht explizit in die Satzaussage ein. So werden sich z. B. die Charakterisierungen gewisser Polyedereigenschaften von $P(A, b)$ (zumindest formal) von den entsprechenden Charakterisierungen von $P^{=}(A, b)$ unterscheiden. Darstellungsabhängige Sätze wollen wir jedoch nur einmal beweisen (normalerweise für Darstellungen, bei denen die Resultate besonders einprägsam oder einfach sind), deshalb werden wir uns nun Transformationsregeln überlegen, die angeben, wie man von einer Darstellungsweise zu einer anderen und wieder zurück kommt.

(2.5) Transformationen

Regel I: *Einführung von Schlupfvariablen*

Gegeben seien $a \in \mathbb{K}^n$, $\alpha \in \mathbb{K}$. Wir schreiben die Ungleichung

$$a^T x \leq \alpha \tag{2.6}$$

in der Form einer Gleichung und einer Vorzeichenbeschränkung

$$a^T x + y = \alpha, \quad y \geq 0. \tag{2.7}$$

y ist eine neue Variable, genannt *Schlupfvariable*. Es gilt:

$$x \text{ erfüllt (2.6)} \quad \Longrightarrow \quad \begin{pmatrix} x \\ y \end{pmatrix} \text{erfüllt (2.7) für } y = \alpha - a^T x,$$

$$\begin{pmatrix} x \\ y \end{pmatrix} \text{erfüllt (2.7)} \quad \Longrightarrow \quad x \text{ erfüllt (2.7)}.$$

Allgemein: Ein Ungleichungssystem $Ax \leq b$ kann durch Einführung eines Schlupfvariablenvektors y transformiert werden in ein Gleichungssystem mit Vorzeichenbedingung $Ax + y = b$, $y \geq 0$. Zwischen den Lösungsmengen dieser beiden Systeme bestehen die oben angegebenen Beziehungen. $P(A, b)$ und $\{\binom{x}{y} \in \mathbb{K}^{n+m} \mid Ax + Iy = b, y \geq 0\}$ sind jedoch zwei durchaus verschiedene Polyeder in verschiedenen Vektorräumen.

Regel II: *Einführung von vorzeichenbeschränkten Variablen*

Ist x eine (eindimensionale) nicht vorzeichenbeschränkte Variable, so können wir zwei vorzeichenbeschränkte Variablen x^+ und x^- einführen, um x darzustellen. Wir setzen

$$x := x^+ - x^- \quad \text{mit} \quad x^+ \geq 0,\ x^- \geq 0. \qquad \triangle$$

Mit den Regeln I und II aus (2.5) können wir z. B. jedem Polyeder $P(A, b) \subseteq \mathbb{K}^n$ ein Polyeder $P^=(D, d) \subseteq \mathbb{K}^{2n+m}$ wie folgt zuordnen. Wir setzen

$$D := (A, -A, I_m),\ d := b,$$

d. h. es gilt

$$P^=(D, d) = \left\{(x, y, z) \in \mathbb{K}^{2n+m} \mid Ax - Ay + z = b, x, y, z \geq 0\right\}.$$

Es ist üblich, die oben definierten Polyeder $P(A, b)$ und $P^=(D, d)$ *äquivalent* zu nennen. Hierbei hat "Äquivalenz" folgende Bedeutung. Für $x \in \mathbb{K}^n$ sei

$$x^+ := (x_1^+, \ldots, x_n^+)^T \quad \text{mit} \quad x_i^+ = \max\{0, x_i\},$$
$$x^- := (x_1^-, \ldots, x_n^-)^T \quad \text{mit} \quad x_i^- = \max\{0, -x_i\}.$$

Dann gilt:

$$x \in P(A, b) \quad \Longrightarrow \quad \begin{pmatrix} x^+ \\ x^- \\ z \end{pmatrix} \in P^=(D, d) \text{ für } z = b - Ax$$

$$\begin{pmatrix} u \\ v \\ w \end{pmatrix} \in P^=(D, d) \quad \Longrightarrow \quad x := u - v \in P(A, b).$$

Besonders einfache Polyeder, auf die sich jedoch fast alle Aussagen der Polyedertheorie zurückführen lassen, sind polyedrische Kegel.

(2.8) Definition Ein Kegel $C \subseteq \mathbb{K}^n$ heißt *polyedrisch* genau dann, wenn C ein Polyeder ist. △

(2.9) Bemerkung Ein Kegel $C \subseteq \mathbb{K}^n$ ist genau dann polyedrisch, wenn es eine Matrix $A \in \mathbb{K}^{(m,n)}$ gibt mit

$$C = P(A, 0).$$ △

Beweis *Gilt $C = P(A, 0)$, so ist C ein Polyeder und offensichtlich ein Kegel.*

Sei C ein polyedrischer Kegel, dann existieren nach Definition (2.1) c) eine Matrix A und ein Vektor b mit $C = P(A, b)$. Da jeder Kegel den Nullvektor enthält, gilt $0 = A0 \leq b$. Angenommen, es existiert ein $\overline{x} \in C$ mit $A\overline{x} \not\leq 0$, d. h. es existiert eine Zeile von A, sagen wir $A_{i\cdot}$, mit $t := A_{i\cdot}\overline{x} > 0$. Da C ein Kegel ist, gilt $\lambda\overline{x} \in C$ für alle $\lambda \in \mathbb{K}_+$. Jedoch für $\overline{\lambda} := \frac{b_i}{t} + 1$ gilt einerseits $\overline{\lambda}\overline{x} \in C$ und andererseits $A_{i\cdot}(\overline{\lambda}\overline{x}) = \overline{\lambda}t > b_i$, ein Widerspruch. Daraus folgt, für alle $x \in C$ gilt $Ax \leq 0$. Hieraus ergibt sich $C = P(A, 0)$. □

In der folgenden Tabelle 2.1 haben wir alle möglichen Transformationen aufgelistet. Sie soll als "Nachschlagewerk" dienen.

Eine wichtige Eigenschaft von linearen Programmen ist die Tatsache, dass man effizient erkennen kann, ob ein gegebener Punkt tatsächlich der gesuchte Optimalpunkt ist oder nicht. Dahinter steckt die sogenannte *Dualitätstheorie*, die wir anhand unseres Einführungsbeispiels aus Kapitel 1.1 erläutern wollen. Durch Anwenden der Transformationsregeln können wir das LP (1.3) in der Form

$$\begin{aligned}
\min\ & -6s + t && \text{(2.10a)}\\
& -s + t \geq -1 && \text{(2.10b)}\\
& s - t \geq -1 && \text{(2.10c)}\\
& s \geq 2 && \text{(2.10d)}\\
& -s \geq -3 && \text{(2.10e)}\\
& t \geq 0 && \text{(2.10f)}\\
& -t \geq -3. && \text{(2.10g)}
\end{aligned}$$

schreiben. Wir wollen nun zeigen, dass der Punkt $x^* = (3, 2)$ mit Zielfunktionswert -16 minimal ist.

Ein Weg, dies zu beweisen, besteht darin, untere Schranken für das Minimum zu finden. Wenn man sogar in der Lage ist, eine untere Schranke zu bestimmen, die mit dem Zielfunktionswert einer Lösung des Ungleichungssystems übereinstimmt, ist man fertig: Der Zielfunktionswert irgendeiner Lösung ist immer eine obere Schranke, und wenn untere und obere Schranke gleich sind, ist der optimale Wert gefunden.

Um eine untere Schranke für die Zielfunktion $-6s + t$ zu finden, können wir uns zunächst die Schranken der Variablen anschauen. Wegen Ungleichung (2.10e) haben wir $-6s \geq -18$. Zusammen mit Ungleichung (2.10f) erhalten wir $-6s + t \geq -18$. Um zu dieser unteren Schranke zu gelangen, haben wir positive Vielfache der Variablenschranken

Tabelle 2.1: Transformationen zwischen Polyedern.

Transformation nach von	$By \leq d$	$By = d$ $y \geq 0$	$By \leq d$ $y \geq 0$
$Ax = b$ hierbei ist $x_i^+ = \max\{0, x_i\}$ $x_i^- = \max\{0, -x_i\}$	$\begin{pmatrix} A \\ -A \end{pmatrix} y \leq \begin{pmatrix} b \\ -b \end{pmatrix}$ $y = x$	$(A, -A)\ y = b$ $y \geq 0$ $y = \begin{pmatrix} x^+ \\ x^- \end{pmatrix}$	$\begin{pmatrix} A & -A \\ -A & A \end{pmatrix} y \leq \begin{pmatrix} b \\ -b \end{pmatrix}$ $y \geq 0$ $y = \begin{pmatrix} x^+ \\ x^- \end{pmatrix}$
$Ax \leq b$	$Ay \leq b$ $y = x$	$(A, -A, I)\ y = b$ $y \geq 0$ $y = \begin{pmatrix} x^+ \\ x^- \\ b - Ax \end{pmatrix}$	$(A, -A)\ y \leq b$ $y \geq 0$ $y = \begin{pmatrix} x^+ \\ x^- \end{pmatrix}$
$Ax = b$ $x \geq 0$	$\begin{pmatrix} A \\ -A \\ -I \end{pmatrix} y \leq \begin{pmatrix} b \\ -b \\ 0 \end{pmatrix}$ $y = x$	$Ay = b$ $y \geq 0$ $y = x$	$\begin{pmatrix} A \\ -A \end{pmatrix} y \leq \begin{pmatrix} b \\ -b \end{pmatrix}$ $y \geq 0$ $y = x$
$Ax \leq b$ $x \geq 0$	$\begin{pmatrix} A \\ -I \end{pmatrix} y \leq \begin{pmatrix} b \\ 0 \end{pmatrix}$ $y = x$	$(A, I)\ y = b$ $y \geq 0$ $y = \begin{pmatrix} x \\ b - Ax \end{pmatrix}$	$Ay \leq b$ $y \geq 0$ $y = x$

so addiert, das wir auf der linken Seite die Zielfunktion erhalten. (Da unsere Ungleichungen alle in der Form "$\geq$" geschrieben sind, dürfen wir nur positiv skalieren, denn zwei Ungleichungen $a_1x_1 + a_2x_2 \leq \alpha$ und $b_1x_1 + b_2x_2 \geq \beta$ kann man nicht addieren.) Natürlich können wir dies mit allen Nebenbedingungsungleichungen machen, um so potenziell bessere untere Schranken zu bekommen. Gibt jede beliebige Skalierung und Addition eine untere Schranke? Machen wir noch einen Versuch. Addieren wir die Ungleichungen (2.10b) und (2.10e), so erhalten wir $-2s + t \geq -4$. Das hilft uns nicht weiter, denn die linke Seite der Ungleichung kann nicht zum Abschätzen der Zielfunktion benutzt werden: Damit die rechte Seite der neuen Ungleichung eine untere Schranke für die Zielfunktion liefert, muss auf der linken Seite jeder Koeffizient höchstens so groß sein wie der entsprechende Koeffizient der Zielfunktion.

Diese Erkenntnisse liefern uns ein neues mathematisches Problem. Wir suchen nichtnegative Multiplikatoren (Skalierungsfaktoren) der Ungleichungen (2.10b)–(2.10g) mit gewissen Eigenschaften. Multiplizieren wir die Ungleichungen mit $y_1, \ldots, y_6$, so darf die Summe $-y_1 + y_2 + y_3 - y_4$ nicht den Wert des ersten Koeffizienten der Zielfunktion (also -6) überschreiten. Analog darf die Summe $y_1 - y_2 + y_5 - y_6$ nicht den Wert 1 überschreiten. Und die y_i sollen nicht negativ sein. Ferner soll die rechte Seite der Summenungleichung, also $-y_1 - y_2 + 2y_3 - 3y_4 - 3y_6$ so groß wie möglich werden. Daraus folgt, dass wir die folgende Aufgabe lösen müssen:

$$\begin{aligned}
\max\ & -y_1 - y_2 + 2y_3 - 3y_4 - 3y_6 && \text{(2.11a)}\\
& -y_1 + y_2 + y_3 - y_4 \leq -6 && \text{(2.11b)}\\
& y_1 - y_2 + y_5 - y_6 \leq 1 && \text{(2.11c)}\\
& y_1,\ y_2,\ y_3,\ y_4,\ y_5,\ y_6 \geq 0. && \text{(2.11d)}
\end{aligned}$$

Auch (2.11) ist ein lineares Programm. Aus unseren Überlegungen folgt, dass der Maximalwert von (2.11) höchstens so groß ist wie der Minimalwert von (1.3), da jede Lösung von (2.11) eine untere Schranke für (1.3) liefert. Betrachten wir z. B. den Punkt

$$y^* = (1, 0, 0, 5, 0, 0).$$

Durch Einsetzen in die Ungleichungen von (2.11) sieht man dass y^* alle Ungleichungen erfüllt. Addieren wir also zu Ungleichung (2.10b) das 5-fache der Ungleichung (2.10f), so erhalten wir

$$-6s + t \geq -16.$$

Damit wissen wir, dass der Minimalwert von (1.3) mindestens -16 ist. Der Punkt $x^* = (3, 2)$ liefert gerade diesen Wert, er ist also optimal. Und außerdem ist y^* eine Optimallösung von (2.11).

Die hier dargestellten Ideen bilden die Grundgedanken der Dualitätstheorie der linearen Programmierung, und sie sind außerordentlich nützlich bei der Entwicklung von Algorithmen und in Beweisen. In der Tat haben wir bereits ein kleines Resultat erzielt, das wir wie folgt formal zusammenfassen wollen.

(2.12) Satz Es seien $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, und A sei eine reelle (m, n)-Matrix. Betrachten wir die Aufgaben

(P) $\min\{c^T x \mid Ax \geq b, x \geq 0\}$ und

(D) $\max\{y^T b \mid y^T A \leq c^T, y \geq 0\}$,

dann gilt Folgendes: Seien $x_0 \in \mathbb{R}^n$ und $y_0 \in \mathbb{R}^m$ Punkte mit $Ax_0 \geq b$, $x_0 \geq 0$ und $y_0^T A \leq c^T$, $y_0 \geq 0$, dann gilt

$$y_0^T b \leq c^T x_0. \qquad \triangle$$

Beweis *Durch einfaches Einsetzen:*

$$y_0^T b \leq y_0^T (Ax_0) = (y_0^T A) x_0 \leq c^T x_0. \qquad \square$$

Satz (2.12) wird *schwacher Dualitätssatz* genannt, (D) heißt das *zu (P) duale LP* und (P) wird in diesem Zusammenhang als *primales LP* bezeichnet. Für Optimallösungen x^* und y^* von (P) bzw. (D) gilt nach (2.12) $y^{*T} b \leq c^T x^*$, das duale Problem liefert also stets eine untere Schranke für das primale Problem. Wir werden später zeigen, dass in diesem Falle sogar immer $y^{*T} b = c^T x^*$ gilt. Diese *starke Dualität*, also das Übereinstimmen der Optimalwerte von (P) und (D) ist allerdings nicht ganz so einfach zu beweisen wie Satz (2.12).

Die schwache Dualität überträgt sich auch auf ganzzahlige lineare Programme, für die jedoch im Allgemeinen kein starker Dualitätssatz gilt. Genauer gilt folgender Zusammenhang:

$$\begin{aligned}
&\min\{c^T x \mid Ax \geq b, x \geq 0, x \in \mathbb{Z}\} \\
&\geq \min\{c^T x \mid Ax \geq b, x \geq 0\} \\
&= \max\{y^T b \mid y^T A \leq c^T, y \geq 0\} \\
&\geq \max\{y^T b \mid y^T A \leq c^T, y \geq 0, y \in \mathbb{Z}\}.
\end{aligned}$$

Literaturverzeichnis

Zur linearen Algebra existieren unzählige Bücher. Deswegen geben wir hierzu keine Literaturliste an, sondern listen hier ausschließlich Literatur zur Graphentheorie und zur linearen Programmierung.

M. Aigner. *Graphentheorie: Eine Entwicklung aus dem 4-Farben-Problem.* Teubner Verlag, Studienbücher : Mathematik, Stuttgart, 1984. ISBN 3-519-02068-8.

B. Bollobás. *Modern Graph Theory.* Springer Verlag, New York, 1998. ISBN 0-387-98488-7.

J. A. Bondy and U. S. R. Murty. *Graph Theory.* Springer, Berlin, 2008.

V. Chvátal. *Linear Programming.* Freeman, 1983.

G. B. Dantzig. *Linear Programming and Extensions.* Princeton University Press, 1998.

R. Diestel. *Graphentheorie.* Springer-Verlag, Heidelberg, 3. auflage edition, 2006. ISBN 3-540-21391-0.

W. Domschke. *Logistik: Rundreisen und Touren.* Oldenbourg-Verlag, München - Wien, (4., erweiterte Aufl. 1997, 1982.

J. Ebert. Effiziente Graphenalgorithmen. *Studientexte: Informatik*, 1981.

M. C. Golombic. *Algorithmic Graph Theory and Perfect Graphs.* Academic Press, New York. ISBN 1980.

R. L. Graham, M. Grötschel, and L. Lovász, editors. *Handbook of Combinatorics, Volume I.* Elsevier (North-Holland); The MIT Press, Cambridge, Massachusetts, 1995. ISBN ISBN 0-444-82346-8/v.1 (Elsevier); ISBN 0-262-07170-3/v.1 (MIT).

J. L. Gross and J. Yellen. *Handbook of Graph Theory.* CRC Press, Boca Raton, 2004. ISBN 1-58488-090-2.

R. Halin. *Graphentheorie.* Akademie-Verlag Berlin, 2. edition, 1989.

K. Hässig. *Graphentheoretische Methoden des Operations Research.* Teubner-Verlag, Stuttgart, 1979.

D. Jungnickel. *Graphen, Netzwerke und Algorithmen.* BI Wissenschaftsverlag, Mannheim, 3. auflage edition, 1994.

D. König. *Theorie der endlichen und unendlichen Graphen.* Akademische Verlagsgesellschaft, Leipzig, 1936. mehrfach auf deutsch und in englischer Übersetzung nachgedruckt.

S. O. Krumke and H. Noltemeier. *Graphentheoretische Konzepte und Algorithmen.* Teubner, Wiesbaden, 2005. ISBN 3-519-00526-3.

J. Matousek and B. Gärtner. *Using and Understanding Linear Programming.* Springer, 2006.

M. Padberg. *Linear Optimization and Extensions.* Springer, 2001.

H. Sachs. *Einführung in die Theorie der endlichen Graphen.* Teubner, Leipzig, 1970, und Hanser, München, 1971, 1970.

A. Schrijver. *Theory of Linear and Integer Programming.* Wiley, 1998.

H. W. und G. Nägler. *Graphen, Algorithmen, Programme.* VEB Fachbuchverlag, Leipzig, 1987.

B. und Ghouila-Houri. *Programme, Spiele, Transportnetze.* Teubner Verlag, Leipzig, 1969.

R. J. Vanderbei. *Linear Programming – Foundations and Extensions.* Springer, 2008.

K. Wagner. *Graphentheorie*. BI Wissenschaftsverlag, Mannheim, 1970.

D. B. West. *Introduction to Graph Theory*. Prentice Hall, Upper Saddle River, third edition, 2005. ISBN 0-13-014400-2.

3 Diskrete Optimierungsprobleme

Dieses Kapitel enthält eine Liste von algorithmischen Fragestellungen der Graphentheorie. Wir werden — neben historisch interessanten Aufgaben — insbesondere Optimierungsprobleme aufführen, die ein weites Anwendungsspektrum besitzen.

3.1 Kombinatorische Optimierungsprobleme

Bevor wir auf graphentheoretische Optimierungsprobleme eingehen, führen wir kombinatorische Optimierungsprobleme in allgemeiner Form ein.

(3.1) Definition (Allgemeines kombinatorisches Optimierungsproblem)
Gegeben seien eine endliche Menge $\mathcal{I}$ und eine Funktion $f : \mathcal{I} \to \mathbb{R}$, die jedem Element von $\mathcal{I}$ einen "Wert" zuordnet. Gesucht ist ein Element $I^* \in \mathcal{I}$, so daß $f(I^*)$ so groß (oder klein) wie möglich ist. △

Eine Problemformulierung dieser Art ist relativ sinnlos, da über ein Problem, das wie oben gegeben ist, kaum vernünftige mathematische Aussagen gemacht werden können. Algorithmisch ist (3.1) auf triviale Weise lösbar: man durchlaufe alle Elemente I von $\mathcal{I}$, werte die Funktion $f(I)$ aus und wähle das Element I^* mit dem größten (oder kleinsten) Wert $f(I^*)$ aus. Falls die Elemente $I \in \mathcal{I}$ algorithmisch bestimmbar und $f(I)$ auswertbar ist, hat der eben beschriebene Enumerationsalgorithmus eine sogenannte lineare Laufzeit, da jedes Element von $\mathcal{I}$ nur einmal betrachtet wird.

Die üblicherweise auftretenden kombinatorischen Optimierungsprobleme sind jedoch auf andere, wesentlich strukturiertere Weise gegeben. Die Menge $\mathcal{I}$ ist nicht durch explizite Angabe aller Elemente spezifiziert sondern implizit durch die Angabe von Eigenschaften, die die Elemente von $\mathcal{I}$ haben sollen. Ebenso ist die Funktion f nicht punktweise sondern durch "Formeln" definiert.

In dieser Vorlesung wollen wir uns hauptsächlich auf den folgenden Problemtyp konzentrieren.

(3.2) Definition (Komb. Optimierungsproblem mit linearer Zielfunktion)
Gegeben seien eine endliche Menge E (genannt *Grundmenge*), eine Teilmenge $\mathcal{I}$ der Potenzmenge 2^E von E (die Elemente von $\mathcal{I}$ heißen *zulässige Mengen* oder *zulässige Lösungen*) und eine Funktion $c : E \to \mathbb{R}$. Für jede Menge $F \subseteq E$ definieren wir ihren "Wert" durch

$$c(F) := \sum_{e \in F} c(e),$$

und wir suchen eine Menge $I^* \in \mathcal{I}$, so dass $c(I^*)$ so groß (oder klein) wie möglich ist. △

Zur Notationsvereinfachung werden wir in Zukunft einfach *kombinatorisches Optimierungsproblem* sagen, wenn wir ein Problem des Typs (3.2) meinen. Da ein derartiges Problem durch die Grundmenge E, die zulässigen Lösungen $\mathcal{I}$ und die Zielfunktion c definiert ist, werden wir kurz von einem kombinatorischen Optimierungsproblem $(E, \mathcal{I}, c)$ sprechen.

Die Zielfunktion haben wir durch Formulierung (3.2) bereits sehr speziell strukturiert. Aber Problem (3.2) ist algorithmisch immer noch irrelevant, falls wir eine explizite Angabe von $\mathcal{I}$ unterstellen. Wir werden nachfolgend (und im Verlaufe der Vorlesung noch sehr viel mehr) Beispiele des Typs (3.2) kennenlernen. Fast alle der dort auftretenden zulässigen Mengen lassen sich auf folgende Weise charakterisieren:

$$\mathcal{I} = \{I \subseteq E \mid I \text{ hat Eigenschaft } \Pi\}.$$

Wir werden uns damit beschäftigen, welche Charakteristika die Eigenschaft Π haben muss, damit die zugehörigen Probleme $(E, \mathcal{I}, c)$ auf einfache Weise gelöst werden können. Nehmen wir an, dass E insgesamt n Elemente enthält, dann führt natürlich jede Eigenschaft Π, die impliziert, dass $\mathcal{I}$ (relativ zu n) nur sehr wenige Elemente enthält, dazu, dass $(E, \mathcal{I}, c)$ einfach lösbar ist, falls man die Elemente von $\mathcal{I}$ explizit angeben kann. Typischerweise haben jedoch die interessanten kombinatorischen Optimierungsprobleme eine Anzahl von Lösungen, die exponentiell in n ist, etwa $n!$ oder 2^n. Eine vollständige Enumeration der Elemente solcher Mengen ist offenbar auch auf den größten Rechnern (für z. B. $n \geq 40$) nicht in „vernünftiger Zeit“ durchführbar. Das Ziel der kombinatorischen Optimierung besteht — kurz und vereinfachend gesagt — darin, Algorithmen zu entwerfen, die (erheblich) schneller als die Enumeration aller Lösungen sind.

3.2 Klassische Fragestellungen der Graphentheorie

Nachfolgend werden eine Reihe von graphentheoretischen Problemen skizziert, die die Entwicklung der Graphentheorie nachhaltig beeinflusst haben.

(3.3) Euler und das Königsberger Brückenproblem Fast jedes Buch über Graphentheorie (Geben Sie einfach einmal “Königsberg bridges” in Google ein.) enthält einen Stadtplan von Königsberg und erläutert, wie Euler die Königsberger Karte zu dem Graphen aus Abbildung 3.1 “abstrahiert” hat.

Euler hat die Frage untersucht, ob es in diesem “Königsberger Brückengraphen” einen geschlossenen Pfad gibt, der alle Kanten genau einmal enthält. Heute nennen wir einen solchen Pfad *Eulertour*. Er hat das Problem nicht nur für den Graphen aus Abbildung 3.1 gelöst, sondern für alle Graphen: Ein Graph enthält eine Eulertour genau dann, wenn er zusammenhängend ist und jeder Knoten einen geraden Grad hat. Diesen Satz hat Euler 1736 bewiesen und damit die Graphentheorie begründet. △

(3.4) Das Haus vom Nikolaus Jeder kennt die Aufgabe aus dem Kindergarten: Zeichne das Haus des Nikolaus, siehe Abbildung 3.2, in einem Zug!

Was hat diese Fragestellung mit dem Königsberger Brückenproblem zu tun? △

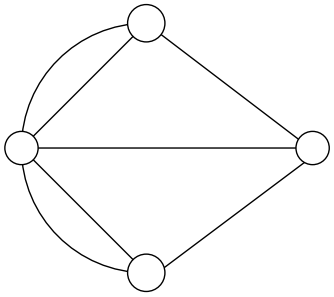


Abbildung 3.1: Das Königsberger Brücken Problem.

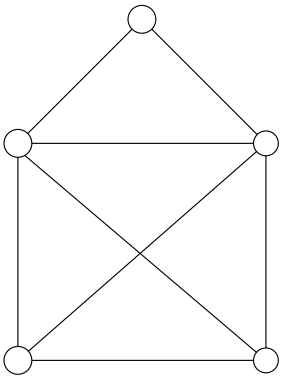


Abbildung 3.2: Das Haus vom Nikolaus.

(3.5) Hamiltonsche Kreise Der irische Mathematiker Sir William Hamilton (z.B. durch die "Erfindung" der Quaternionen bekannt) hat sich Ende der 50er Jahre des 19. Jahrhunderts mit Wege-Problemen beschäftigt und sich besonders dafür interessiert, wie man auf dem Dodekaedergraphen, siehe Abbildung 3.3, Kreise findet, die alle Knoten durchlaufen (heute *hamiltonsche Kreise* genannt) und die noch gewissen Zusatzanforderungen genügen. Er fand diese Aufgabe so spannend, dass er sie als Spiel vermarktet hat (offenbar nicht sonderlich erfolgreich). Ein Exemplar dieses Spiels mit dem Namen "The Icosian Game" befindet sich noch in der Bibliothek des Trinity College in Dublin, Irland, siehe Abbildung 3.4.

Die Aufgabe, in einem Graphen, einen Hamiltonkreis zu finden, sieht so ähnlich aus wie das Problem, eine Eulertour zu bestimmen. Sie ist aber viel schwieriger. Das hamiltonische Graphen-Problem hat sich später zum Travelling-Salesman-Problem "entwickelt". Historische Bemerkungen hierzu findet man zum Beispiel in Applegate et al. (2006) und Cook (2012). △

(3.6) Färbung von Landkarten Nach Aigner (1984), der die Entwicklung der Graphentheorie anhand der vielfältigen Versuche, das 4-Farben-Problem zu lösen, darstellt, begann die mathematische Beschäftigung mit dem Färbungsproblem im Jahre 1852 mit einem Brief von Augustus de Morgan an William Hamilton:

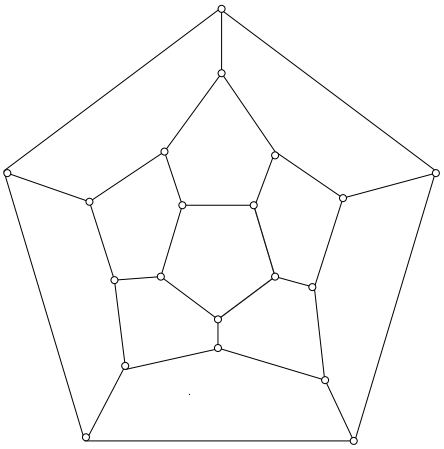


Abbildung 3.3: Graph mit einem Hamilton Kreis.

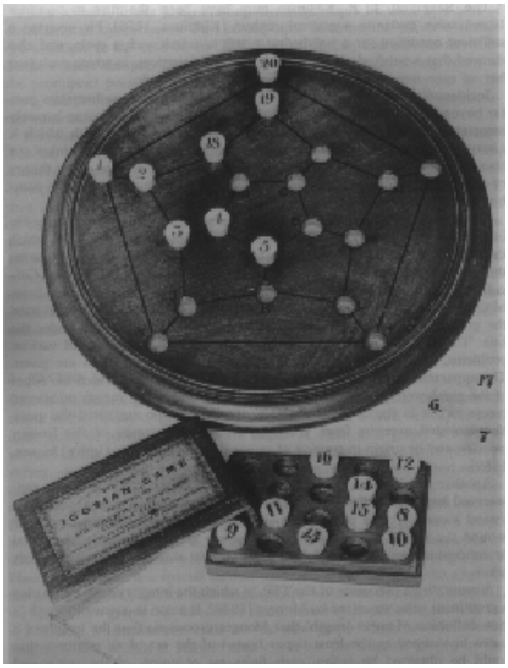
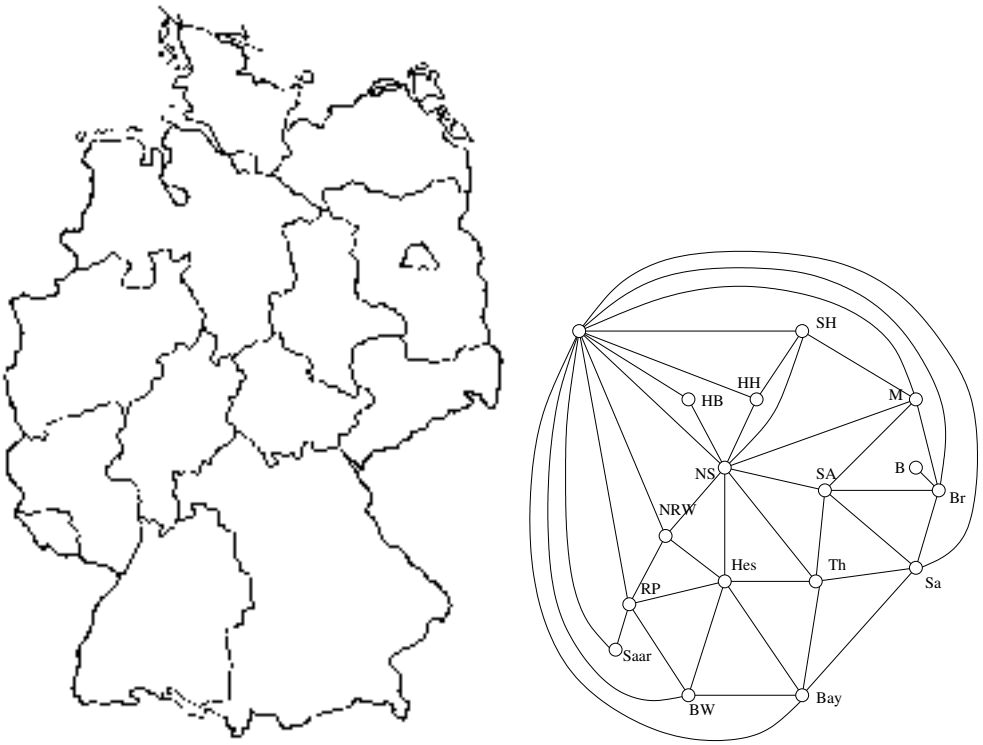


Abbildung 3.4: The Icosian Game.



(a) (b)

Abbildung 3.5: Abbildung (a) ist eine Karte von Deutschland mit seinen Bundesländern. In (b) repräsentiert jede Landeshauptstadt ihr zugehöriges Bundesland. Die Nachbarschaft eines jeden Knoten besteht aus den entsprechenden Nachbarländern.

"Ein Student fragte mich heute, ob es stimmt, dass die Länder jeder Karte stets mit höchstens 4 Farben gefärbt werden können, unter der Maßgabe, dass angrenzende Länder verschiedene Farben erhalten." Der Urheber der Frage war *Francis Guthrie*.

Aus einer Landkarte kann man einen Graphen machen, indem jedes Land durch einen Knoten repräsentiert wird und je zwei Knoten genau dann durch eine Kante verbunden werden, wenn die zugehörigen Länder benachbart sind. Abbildung 3.5 (a) zeigt die Karte der deutschen Bundesländer. Der "Bundesländergraph" in Abbildung 3.5 (b) hat daher je einen Knoten für die 16 Länder und einen weiteren Knoten für die "Außenwelt". Dieser Knoten ist mit allen Bundesländerknoten verbunden, die an das Ausland oder das Meer (wie etwa Niedersachsen) grenzen.

"Landkartengraphen" kann man nach Konstruktion in die Ebene so zeichnen, dass sich je zwei Kanten (genauer: die Linien, die die Kanten in der Ebene repräsentieren) nicht schneiden (außer natürlich in ihren Endpunkten, wenn sie einen gemeinsamen Knoten besitzen). Landkartengraphen sind also planar. Das 4-Farben-Problem (in etwas allgemeinerer Form) lautet dann: "Kann man die Knoten eines planaren Graphen so färben,

dass je zwei benachbarte Knoten verschiedene Farben besitzen?"

Der Weg zur Lösung des 4-Farben-Problems war sehr lang, siehe hierzu Aigner (1984) . Die erste vollständige Lösung (unter Zuhilfenahme von Computerprogrammen) wurde 1976/1977 von K. Appel und W. Haken vorgelegt. Die Dokumentation eines transparenten Beweises von N. Robertson, D.P. Sanders, P. Seymour und R. Thomas, der weiterhin auf der Überprüfung vieler Einzelfälle durch Computerprogramme beruht, ist auf der Homepage von Robin Thomas zu finden:

`http://www.math.gatech.edu/~thomas/FC/fourcolor.html`. △

(3.7) Planarität Durch das 4-Farben-Problem gelangte die Frage, wann kann man einen Graphen so in die Ebene einbetten, dass sich je zwei Kanten nicht überschneiden, in den Fokus der Forschung. Natürlich wurde sofort verallgemeinert: "Finde eine 'gute' Charakterisierung dafür, dass ein Graph in die Ebene, auf dem Torus, in die projektive Ebene, auf Henkelflächen etc. überschneidungsfrei einbettbar ist."

Kuratowski gelang 1930 ein entscheidender Durchbruch. Es ist einfach zu sehen, dass weder der vollständige Graph K_5 noch der vollständige Graph $K_{3,3}$ planar sind. Kuratowski bewies, dass jeder nicht-planare Graph einen der Graphen K_5 oder $K_{3,3}$ "enthält". Das heißt, ist G nicht planar, so kann man aus G durch Entfernen und durch Kontraktion von Kanten entweder den K_5 oder den $K_{3,3}$ erzeugen. Dies ist auch heute noch ein keineswegs triviales Ergebnis. △

3.3 Graphentheoretische Optimierungsprobleme: Einige Beispiele

In diesem Abschnitt wollen wir mehrere Beispiele von kombinatorischen Optimierungsproblemen, die sich mit Hilfe von Graphentheorie formulieren lassen, und einige ihrer Anwendungen auflisten. Diese Sammlung ist nicht im geringsten vollständig, sondern umfasst nur einige in der Literatur häufig diskutierte oder besonders anwendungsnahe Probleme. Wir benutzen dabei gelegentlich englische Namen, die mittlerweile auch im Deutschen zu Standardbezeichnungen geworden sind. Fast alle der nachfolgend aufgeführten "Probleme" bestehen aus mehreren eng miteinander verwandten Problemtypen. Wir gehen bei unserer Auflistung so vor, dass wir meistens zunächst die graphentheoretische Formulierung geben und dann einige Anwendungen skizzieren.

(3.8) Kürzeste Wege Gegeben seien ein Digraph $D = (V, A)$ und zwei verschiedene Knoten $u, v \in V$, stelle fest, ob es einen gerichteten Weg von u nach v gibt. Falls das so ist, und falls "Entfernungen" $c_{ij} \geq 0$ für alle $(i, j) \in A$ bekannt sind, bestimme einen kürzesten gerichteten Weg von u nach v (d. h. einen (u, v)-Weg P, so dass $c(P)$ minimal ist). Dieses Problem wird üblicherweise *Problem des kürzesten Weges* (*shortest path problem*) genannt. Zwei interessante Varianten sind die folgenden: Finde einen kürzesten (u, v)-Weg gerader bzw. ungerader Länge (d. h. mit gerader bzw. ungerader Bogenzahl).

Das Problem des kürzesten Weges gibt es auch in einer ungerichteten Version. Hier sucht man in einem Graphen $G = (V, E)$ mit Entfernungen $c_e \geq 0$ für alle $e \in E$ bei

gegebenen Knoten $u, v \in V$ einen kürzesten $[u, v]$-Weg. Analog kann man nach einem kürzesten Weg gerader oder ungerader Länge fragen.

Natürlich kann man in allen bisher angesprochenen Problemen, das Wort "kürzester" durch "längster" ersetzen und erhält dadurch *Probleme der längsten Wege* verschiedener Arten. Hätten wir beim Problem des kürzesten Weges nicht die Beschränkung $c_{ij} \geq 0$ für die Zielfunktionskoeffizienten, wären die beiden Problemtypen offensichtlich äquivalent. Aber so sind sie es nicht! Ein Spezialfall (Zielfunktion $c_e = 1$ für alle $e \in E$) des Problems des längsten Weges ist das Problem zu entscheiden, ob ein Graph einen hamiltonschen Weg von u nach v enthält. △

Anwendungen dieses Problems und seiner Varianten sind offensichtlich. Alle Routenplaner, die im Internet zur Fahrstreckenplanung angeboten werden oder zur Unterstützung von Autofahrern in Navigationssysteme eingebaut sind, basieren auf Algorithmen zur Bestimmung kürzester Wege. Die Route jeder im Internet verschickten Nachricht wird ebenfalls durch (mehrfachen) Aufruf eines Kürzeste-Wege-Algorithmus ermittelt. Eine Anwendung aus der Wirtschafts- und Sozialgeographie, die nicht unbedingt im Gesichtsfeld von Mathematikern liegt, sei hier kurz erwähnt. Bei Fragen der Raumordnung und Landesplanung werden sehr umfangreiche Erreichbarkeitsanalysen angestellt, um Einzugsbereiche (bzgl. Straßen-, Nahverkehrs- und Bahnanbindung) festzustellen. Auf diese Weise werden Mittel- und Oberzentren des ländlichen Raumes ermittelt und Versorgungsgrade der Bevölkerung in Bezug auf Ärzte, Krankenhäuser, Schulen etc. bestimmt. Ebenso erfolgen Untersuchungen bezüglich des Arbeitsplatzangebots. Alle diese Analysen basieren auf einer genauen Ermittlung der Straßen-, Bus- und Bahnentfernungen (in Kilometern oder Zeiteinheiten) und Algorithmen zur Bestimmung kürzester Wege in den "Verbindungsnetzwerken".

(3.9) Das Zuordnungsproblem (assignment problem) Gegeben sei ein bipartiter Graph $G = (V, E)$ mit Kantengewichten $c_e \in \mathbb{R}$ für alle $e \in E$, gesucht ist ein Matching in G maximalen Gewichts. Man nennt dieses Problem das *Matchingproblem in bipartiten Graphen* oder kurz *bipartites Matchingproblem*. Haben die beiden Knotenmengen in der Bipartition von V gleiche Kardinalität und sucht man ein perfektes Matching minimalen Gewichts, so spricht man von einem *Zuordnungsproblem*. Es gibt noch eine weitere Formulierung des Zuordnungsproblems. Gegeben sei ein Digraph $D = (V, A)$, der auch Schlingen haben darf, mit Bogengewichten (meistens wird unterstellt, dass D vollständig ist und Schlingen hat), gesucht ist eine Bogenmenge minimalen Gewichts, so dass jeder Knoten von D genau einmal Anfangs- und genau einmal Endknoten eines Bogens aus B ist. (B ist also eine Menge knotendisjunkter gerichteter Kreise, so dass jeder Knoten auf genau einem Kreis liegt.) Wir wollen dieses Problem *gerichtetes Zuordnungsproblem* nennen. △

Das Zuordnungsproblem hat folgende "Anwendung". Gegeben seien n Männer und n Frauen, für $1 \leq i, j \leq n$ sei c_{ij} ein "Antipathiekoeffizient". Gesucht ist eine Zuordnung von Männern zu Frauen (Heirat), so dass die Summe der Antipathiekoeffizienten minimal ist. Dieses Problem wird häufig *Heiratsproblem* genannt.

Das Matchingproblem in bipartiten Graphen kann man folgendermaßen interpretieren. Ein Betrieb habe m offene Stellen und n Bewerber für diese Positionen. Durch Tests hat man herausgefunden, welche Eignung Bewerber i für die Stelle j hat. Diese "Kompetenz" sei mit c_{ij} bezeichnet. Gesucht wird eine Zuordnung von Bewerbern zu Positionen, so dass die "Gesamtkompetenz" maximal wird.

Das Zuordnungsproblem und das Matchingproblem in bipartiten Graphen sind offenbar sehr ähnlich, die Beziehungen zwischen dem Zuordnungsproblem und seiner gerichteten Version sind dagegen nicht ganz so offensichtlich. Dennoch sind diese drei Probleme in folgendem Sinne "äquivalent": man kann sie auf sehr einfache Weise ineinander transformieren, d. h. mit einem schnellen Algorithmus zur Lösung des einen Problems kann man die beiden anderen Probleme lösen, ohne komplizierte Transformationsalgorithmen einzuschalten.

Transformationstechniken, die einen Problemtyp in einen anderen überführen, sind außerordentlich wichtig und zwar sowohl aus theoretischer als auch aus praktischer Sicht. In der Theorie werden sie dazu benutzt, Probleme nach ihrem Schwierigkeitsgrad zu klassifizieren (siehe Kapitel 4), in der Praxis ermöglichen sie die Benutzung eines einzigen Algorithmus zur Lösung der verschiedensten Probleme und ersparen daher erhebliche Codierungs- und Testkosten. Anhand der drei vorgenannten Probleme wollen wir nun derartige Transformationstechniken demonstrieren.

Bipartites Matchingsproblem $\longrightarrow$ Zuordnungsproblem. Angenommen wir haben ein Matchingproblem in einem bipartiten Graphen und wollen es mit einem Algorithmus für Zuordnungsprobleme lösen. Das Matchingproblem ist gegeben durch einen bipartiten Graphen $G = (V, E)$ mit Bipartition V_1, V_2 und Kantengewichten $c_e \in \mathbb{R}$ für alle $e \in E$. O. B. d. A. können wir annehmen, dass $m = |V_1| \leq |V_2| = n$ gilt. Zur Menge V_1 fügen wir $n - m$ neue Knoten W (künstliche Knoten) hinzu. Wir setzen $V_1' := V_1 \cup W$. Für je zwei Knoten $i \in V_1'$ und $j \in V_2$, die nicht in G benachbart sind, fügen wir eine neue (künstliche) Kante ij hinzu. Die Menge der so hinzugefügten Kanten nennen wir E', und den Graphen $(V_1' \cup V_2, E \cup E')$ bezeichnen wir mit G'. G' ist der vollständige bipartite Graph $K_{n,n}$. Wir definieren neue Kantengewichte c_e' wie folgt:

$$c_e' := \begin{cases} 0 & \text{falls } e \in E' \\ 0 & \text{falls } e \in E \text{ und } c_e \leq 0 \\ -c_e & \text{falls } e \in E \text{ und } c_e > 0 \end{cases}$$

Lösen wir das Zuordnungsproblem bezüglich G' mit den Gewichten c_e', $e \in E \cup E'$, so erhalten wir ein perfektes Matching M' minimalen Gewichts bezüglich c'. Es ist nun einfach zu sehen, dass

$$M := \{e \in M' \mid c_e' < 0\}$$

ein Matching in G ist, das maximal bezüglich der Gewichtsfunktion c ist.

Zuordnungsproblem $\longrightarrow$ gerichtetes Zuordnungsproblem. Wir zeigen nun, dass man das Zuordnungsproblem mit einem Algorithmus für das gerichtete Zuordnungsproblem lösen kann. Gegeben sei also ein bipartiter Graph $G = (V, E)$ mit Bipartition V_1, V_2 und Kantengewichten c_e. Es gelte $V_1 = \{u_1, u_2, \ldots, u_n\}$, $V_2 = \{v_1, v_2, \ldots, v_n\}$. Wir definieren

einen Digraphen $D = (W, A)$ mit $W = \{w_1, \ldots, w_n\}$. Zwei Knoten $w_i, w_j \in W$ sind genau dann durch einen Bogen (w_i, w_j) verbunden, wenn $u_iv_j \in E$ gilt. Das Gewicht $c'((w_i, w_j))$ des Bogens (w_i, w_j) sei das Gewicht $c(u_iv_j)$ der Kante u_iv_j. Ist B eine minimale Lösung des gerichteten Zuordnungsproblems bezüglich D und c', so ist

$$M := \{u_iv_j \in E \mid (w_i, w_j) \in B\}$$

offenbar ein minimales perfektes Matching in G bezüglich der Gewichtsfunktion c. Es ist ebenfalls sofort klar, dass das gerichtete Zuordnungsproblem bezüglich D eine Lösung genau dann hat, wenn G ein perfektes Matching enthält.

Gerichtetes Zuordnungsproblem $\longrightarrow$ bipartites Matchingproblem. Schließlich wollen wir noch vorführen, dass man das gerichtete Zuordnungsproblem auf das Matchingproblem in bipartiten Graphen zurückführen kann. Gegeben sei also ein Digraph $D = (W, A)$ mit $W = \{w_1, \ldots, w_n\}$ und Bogengewichten $c((w_i, w_j))$ für alle $(w_i, w_j) \in A$. Wir definieren einen bipartiten Graphen $G = (V, E)$ mit Bipartition $V_1 = \{u_1, \ldots, u_n\}$, $V_2 = \{v_1, \ldots, v_n\}$ und Kantenmenge $E := \{u_iv_j \mid (w_i, w_j) \in A\}$. Es seien

$$z := n(\max\{|c((w_i, w_j))| : (w_i, w_j) \in A\}) + 1$$

und

$$c'(u_iv_j) := -c((w_i, w_j)) + z.$$

Nach Konstruktion gilt, dass jedes Matching in G mit k Kanten ein geringeres Gewicht hat als ein Matching mit $k+1$ Kanten, $k = 0, \ldots, n-1$. Daraus folgt, dass es eine Lösung des gerichteten Zuordnungproblems bezüglich D genau dann gibt, wenn jedes maximale Matching M bezüglich G und c' perfekt ist. Ist dies so, dann ist

$$B := \{(w_i, w_j) \in A \mid u_iv_j \in M\}$$

eine minimale Lösung des gerichteten Zuordnungsproblems mit Gewicht $c(B) = -c'(M) + nz$.

(3.10) Das Matchingproblem Die Grundversion dieses Problems ist die folgende. Gegeben sei ein Graph $G = (V, E)$ mit Kantengewichten c_e für alle $e \in E$. Ist ein Matching M von G maximalen Gewichts $c(M)$ gesucht, so heißt dieses Problem *Matchingproblem.* Sucht man ein perfektes Matching minimalen Gewichts, so wird es *perfektes Matchingproblem* genannt.

Diese Probleme können wie folgt verallgemeinert werden. Gegeben seien zusätzlich nichtnegative ganze Zahlen b_v für alle $v \in V$ (genannt Gradbeschränkungen) und u_e für alle $e \in E$ (genannt Kantenkapazitäten). Ein *(perfektes) b-Matching*ist eine Zuordnung x_e von nichtnegativen ganzen Zahlen zu den Kanten $e \in E$, so dass für jeden Knoten $v \in V$ die Summe der Zahlen x_e über die Kanten $e \in E$, die mit v inzidieren, höchstens (exakt) b_v ist. Das *unkapazitierte (perfekte) b-Matchingproblem* ist die Aufgabe ein (perfektes) b-Matching $(x_e)_{e\in E}$ zu finden, so dass $\sum_{e\in E} c_e x_e$ maximal (minimal) ist. Sollen die ganzzahligen Kantenwerte x_e für alle $e \in E$ zusätzlich noch die Kapazitätsschranken $0 \leq x_e \leq u_e$ erfüllen, so spricht man von einem *(perfekten) u-kapazitierten b-Matchingproblem.* △

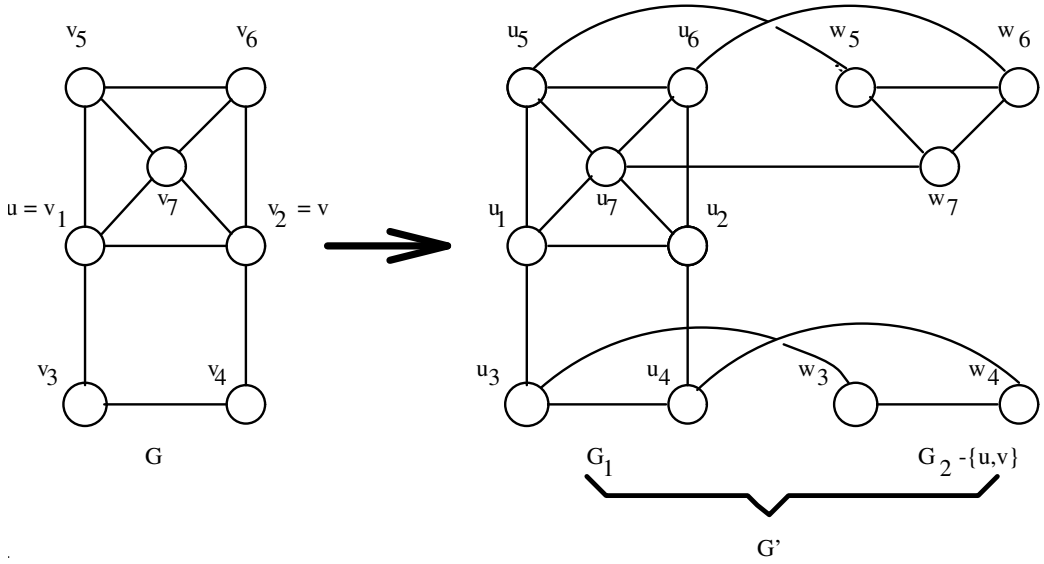


Abbildung 3.6: Die Abbildung entspricht z. B. dem ungeraden $[u, v]$-Weg (u, v_7, v_6, v) das perfekte Matching $M = \{u_1u_7, w_7w_6, u_6u_2, u_3w_3, u_4w_4, u_5w_5\}$ und umgekehrt.

An dieser Stelle wollen wir noch eine – nicht offensichtliche – Problemtransformation vorführen. Und zwar wollen wir zeigen, dass die Aufgabe, in einem ungerichteten Graphen $G = (V, E)$ mit Kantengewichten $c_e \geq 0$ für alle $e \in E$ einen kürzesten Weg ungerader Länge zwischen zwei Knoten $u, v \in V$ zu bestimmen, mit einem Algorithmus für das perfekte Matchingproblem gelöst werden kann. Und zwar konstruieren wir aus G einen neuen Graphen G' wie folgt. Nehmen wir an, dass $V = \{v_1, \ldots, v_n\}$ gilt. Die Graphen $G_1 = (U, E_1)$ mit $U := \{u_1, \ldots, u_n\}$ und $G_2 = (W, E_2)$ mit $W := \{w_1, \ldots, w_n\}$ seien knotendisjunkte isomorphe Bilder (also Kopien) von G, so dass die Abbildungen $v_i \mapsto u_i$ und $v_i \mapsto w_i$, $i = 1, \ldots, n$ Isomorphismen sind. Aus G_2 entfernen wir die Bilder der Knoten u und v, dann verbinden wir die übrigen Knoten $w_i \in W$ mit ihren isomorphen Bildern $u_i \in U$ durch eine Kante u_iw_i. Diese neuen Kanten u_iw_i erhalten das Gewicht $c(u_iw_i) = 0$. Die Kanten aus G_1 und $G_2 - \{u, v\}$, die ja Bilder von Kanten aus G sind, erhalten das Gewicht ihrer Urbildkanten. Der Graph G' entsteht also aus der Vereinigung von G_1 mit $G_2 - \{u, v\}$ unter Hinzufügung der Kanten u_iw_i, siehe Abbildung 2.6. Man überlegt sich leicht, dass jedes perfekte Matching in G' einer Kantenmenge in G entspricht, die einen ungeraden $[u, v]$-Weg in G enthält und dass jedes minimale perfekte Matching in G' einen minimalen ungeraden $[u, v]$-Weg bestimmt.

Hausaufgabe *Finden Sie eine ähnliche Konstruktion, die das Problem, einen kürzesten $[u, v]$-Weg gerader Länge zu bestimmen, auf ein perfektes Matchingproblem zurückführt!*

(3.11) Wälder, Bäume, Branchings, Arboreszenzen Gegeben sei ein Graph $G = (V, E)$ mit Kantengewichten $c_e \in \mathbb{R}$ für alle $e \in E$. Die Aufgabe, einen Wald $W \subseteq E$ zu finden, so dass $c(W)$ maximal ist, heißt *Problem des maximalen Waldes.*

Die Aufgabe einen Baum $T \subseteq E$ zu finden, der G aufspannt und dessen Gewicht $c(T)$ minimal ist, heißt *Problem des minimalen aufspannenden Baumes* (minimum spanning tree problem). Diese beiden Probleme haben auch eine gerichtete Version.

Gegeben sei ein Digraph $D = (V, A)$ mit Bogengewichten $c_a \in \mathbb{R}$ für alle $a \in A$. Die Aufgabe, ein Branching $B \subseteq A$ maximalen Gewichts zu finden, heißt *maximales Branching-Problem*, die Aufgabe, eine Arboreszenz (mit vorgegebener Wurzel r) von D minimalen Gewichts zu finden, heißt *minimales Arboreszenz-Problem (r-Arboreszenz-Problem).* △

Die im folgenden Punkt zusammengefaßten Probleme gehören zu den am meisten untersuchten und anwendungsreichsten Problemen.

(3.12) Routenplanung Gegeben seien n Städte und Entfernungen c_{ij} zwischen diesen, gesucht ist eine Rundreise (Tour), die durch alle Städte genau einmal führt und minimale Länge hat. Haben die Entfernungen die Eigenschaft, dass $c_{ij} = c_{ji}$ gilt, $1 \leq i < j \leq n$, so nennt man dieses Problem *symmetrisches Travelling-Salesman-Problem (TSP)*, andernfalls heißt es *asymmetrisches TSP*. Graphentheoretisch läßt sich das TSP wie folgt formulieren. Gegeben sei ein vollständiger Graph (oder Digraph) G mit Kantengewichten (oder Bogengewichten), gesucht ist ein (gerichteter) hamiltonscher Kreis minimaler Länge. Beim TSP geht man durch jeden Knoten genau einmal, beim (gerichteten) *Chinesischen Postbotenproblem* (Chinese postman problem) durch jede Kante (jeden Bogen) mindestens einmal, d. h. in einem Graphen (Digraphen) mit Kantengewichten (Bogengewichten) wird eine Kette (gerichtete Kette) gesucht, die jede Kante (jeden Bogen) mindestens einmal enthält und minimale Länge hat.

Zu diesen beiden Standardproblemen gibt es hunderte von Mischungen und Varianten. Z. B., man sucht eine Kette, die durch einige vorgegebene Knoten und Kanten mindestens einmal geht und minimale Länge hat; man legt verschiedene Ausgangspunkte (oder Depots) fest, zu denen man nach einer gewissen Streckenlänge wieder zurückkehren muss, etc. Eine relativ allgemeine Formulierung ist die folgende. Gegeben ist ein gemischter Graph mit Knotenmenge V, Kantenmenge E und Bogenmenge A. Ferner sind eine Menge von Depots $W \subseteq V$, von denen aus Reisen gestartet werden müssen, eine Menge $U \subseteq V$ von Knoten, die mindestens einmal besucht werden müssen, und eine Menge $B \subseteq E \cup A$ von Kanten und Bögen, die mindestens einmal durchlaufen werden müssen. Gesucht sind geschlossene Ketten von Kanten und gleichgerichteten Bögen, so dass jede dieser Folgen mindestens (oder genau) einen der Knoten aus W enthält und die Vereinigung dieser Ketten jeden Knoten aus U und jede Kante (Bogen) aus B mindestens einmal enthält und minimale Länge hat. △

Anwendungen dieser Probleme in der Routenplanung von Lieferwagen, von Straßenkehrmaschinen, der Müllabfuhr, von Speditionen etc. sind offensichtlich. Aber auch bei der Steuerung von NC-Maschinen (zum automatischen Bohren, Löten oder Schweißen) oder der Verdrahtung von Leiterplatten (z. B. von Testbussen) tritt das TSP (oder eine seiner Varianten) auf. Abbildung 3.7 zeigt eine Leiterplatte, durch die 441 Löcher gebohrt werden müssen. Links unten ist der Startpunkt, an den der Bohrkopf nach Beendigung des Arbeitsvorganges zurückkehrt, damit eine neue Platte in die Maschine eingelegt werden

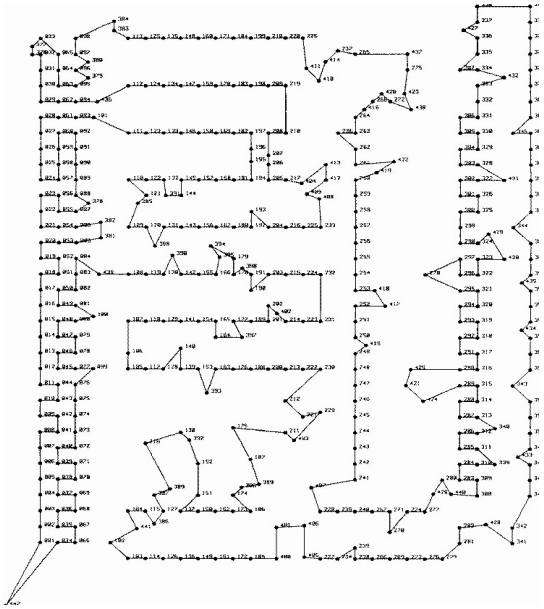


Abbildung 3.7: Optimaler Weg auf einer Leiterplatte um 441 Löcher zu bohren.

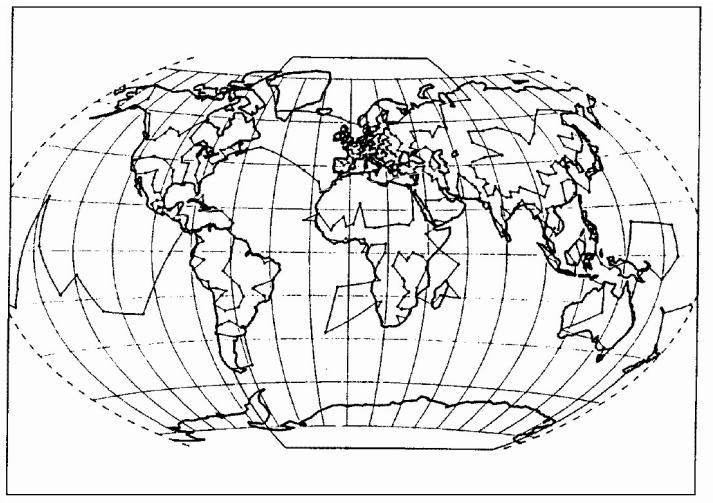


Abbildung 3.8: Optimale Tour für 666 Städte weltweit.

kann. Abbildung 3.7 zeigt eine optimale Lösung dieses 442-Städte-TSP. Die Bohrmaschine muss eine Weglänge von 50.069 Einheiten zurückzulegen.

Abbildung 3.8 zeigt 666 Städte auf der Weltkugel. Wählt man die "Luftliniendistanz" (bezüglich eines Großkreises auf der Kugel) als Entfernung zwischen zwei Städten, so zeigt Abbildung 3.8 eine kürzeste Rundreise durch die 666 Orte dieser Welt. Die Länge dieser Reise ist 294 358 km lang. Abbildungen 3.7 und 3.8 sind Grötschel and Holland

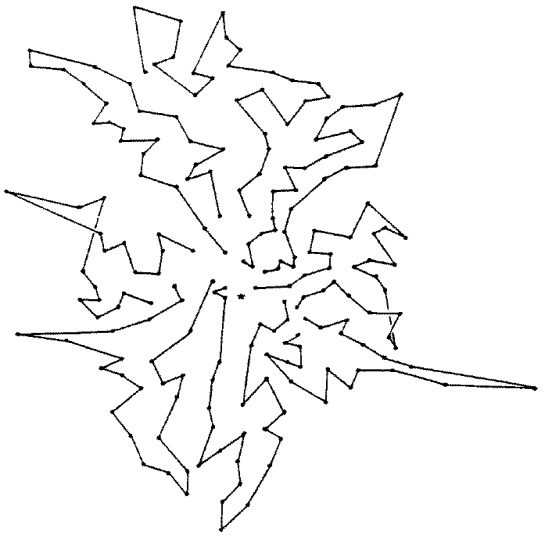


Abbildung 3.9: Optimale Routen für die Wartung der Telefonzellen in der holländischen Stadt Haarlem.

(1991) entnommen. Im Internet finden Sie unter der URL:
`http://www.math.princeton.edu/tsp/` interessante Informationen zum TSP sowie weitere Bilder von TSP-Beispielen. Daten zu vielen TSP-Beispielen wurden von G. Reinelt gesammelt und sind unter der folgenden URL zu finden:

`http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/`

Der beste zur Zeit verfügbare Code zur Lösung von TSPs ist zu finden unter

`http://www.tsp.gatech.edu/concorde/`

In Abbildung 3.9 sind die eingezeichneten Punkte Standorte von Telefonzellen in der holländischen Stadt Haarlem. Der Stern in der Mitte ist das Postamt. Die Aufgabe ist hier, eine Routenplanung für den sich wöchentlich wiederholenden Telefonzellenwartungsdienst zu machen. Die einzelnen Touren starten und enden am Postamt (diese Verbindungen sind nicht eingezeichnet) und führen dann so zu einer Anzahl von Telefonzellen, dass die Wartung aller Telefonzellen auf der Tour innerhalb einer Schicht durchgeführt werden kann.

Als Travelling-Salesman-Problem lassen sich auch die folgenden Anwendungsprobleme formulieren:

- Bestimmung einer optimalen Durchlaufreihenfolge der Flüssigkeiten (Chargen) in einer Mehrproduktenpipeline (Minimierung Reinigungszeiten),
- Bestimmung der optimalen Verarbeitungsfolge von Lacken in einer Großlackiererei (Minimierung Reinigungszeiten),

- Bestimmung einer Reihenfolge des Walzens von Profilen in einem Walzwerk, so dass die Umrüstzeiten der Walzstraße minimiert werden,
- Bestimmung der zeitlichen Reihenfolge von archäologischen Fundstätten (Grablegungsreihenfolge von Gräbern in einem Gräberfeld, Besiedlungsreihenfolge von Orten) aufgrund von Ähnlichkeitsmaßen (Distanzen), die durch die aufgefundenen Fundstücke definiert werden.

Umfangreiche Information über das TSP, seine Varianten und Anwendungen kann man in den Sammelbänden Lawler et al. (1985) und Gutin and Punnen (2002) finden.

(3.13) Bemerkung (Stabile Mengen, Cliquen,Knotenüberdeckungen) Gegeben sei ein Graph $G = (V, E)$ mit Knotengewichten $c_v \in \mathbb{R}$ für alle $v \in V$. Das *Stabile-Mengen-Problem* ist die Aufgabe, eine stabile Menge $S \subseteq V$ zu suchen, so dass $c(S)$ maximal ist, das *Cliquenproblem* die Aufgabe, eine Clique $Q \subseteq V$ zu suchen, so dass $c(Q)$ maximal ist, und das *Knotenüberdeckungsproblem* die Aufgabe, eine Knotenüberdeckung $K \subseteq V$ zu suchen, so dass $c(K)$ minimal ist. △

Die drei oben aufgeführten Probleme sind auf triviale Weise ineinander überführbar. Ist nämlich $S \subseteq V$ eine stabile Menge in G, so ist S eine Clique im komplementären Graphen $\overline{G}$ von G und umgekehrt. Also ist das Stabile-Menge-Problem für G mit Gewichtsfunktion c nicht anders als das Cliquenproblem für $\overline{G}$ mit derselben Gewichtsfunktion und umgekehrt. Ist ferner $S \subseteq V$ eine stabile Menge in G, so ist $V \setminus S$ eine Knotenüberdeckung von G. Daraus folgt, dass zu jeder gewichtsmaximalen stabilen Menge S die zugehörige Knotenüberdeckung $V \setminus S$ gewichtsminimal ist und umgekehrt. Das Stabile-Menge-Problem, das Cliquenproblem und das Knotenüberdeckungsproblem sind also drei verschiedene Formulierungen einer Aufgabe. Anwendungen dieser Probleme finden sich z. B. in folgenden Bereichen:

- Einsatzplanung von Flugzeugbesatzungen
- Busfahrereinsatzplanung
- Tourenplanung im Behindertentransport
- Auslegung von Fließbändern
- Investitionsplanung
- Zuordnung von Wirtschaftsprüfern zu Prüffeldern
- Entwurf von optimalen fehlerkorrigierenden Codes
- Schaltkreisentwurf
- Standortplanung
- Wiedergewinnung von Information aus Datenbanken

- Versuchsplanung
- Signalübertragung.

Aber auch das folgende Schachproblem kann als Stabile-Menge-Problem formuliert werden: Bestimme die maximale Anzahl von Damen (oder Türmen, oder Pferden etc.), die auf einem $n \times n$ Schachbrett so plaziert werden können, dass keine eine andere schlägt.

(3.14) Färbungsprobleme Gegeben sei ein Graph $G = (V, E)$. Zusätzlich seien Knotengewichte b_v für alle $v \in V$ gegeben. Die Aufgabe, eine Folge von (nicht notwendigerweise verschiedenen) stabilen Mengen $S_1, \ldots, S_t$ von G zu suchen, so dass jeder Knoten in mindestens b_v dieser stabilen Mengen enthalten und t minimal ist, heißt (gewichtetes) *Knotenfärbungsproblem* oder kurz *Färbungsproblem*. Beim (gewichteten) *Kantenfärbungsproblem* sind statt Knotengewichten Kantengewichte c_e, $e \in E$, gegeben und gesucht ist eine Folge von (nicht notwendigerweise verschiedenen) Matchings $M_1, \ldots, M_s$, so dass jede Kante in mindestens c_e dieser Matchings enthalten und s so klein wie möglich ist.

Das geographische Färbungsproblem ist uns schon in (3.6) begegnet.

Hat man eine Färbung der Länder, so dass je zwei benachbarte Länder verschieden gefärbt sind, so entspricht jede Gruppe von Ländern gleicher Farbe einer stabilen Menge in G. Hat man umgekehrt eine Zerlegung der Knotenmenge von G in stabile Mengen, so kann man jeweils die Länder, die zu den Knoten einer stabilen Menge gehören mit derselben Farbe belegen und erhält dadurch eine zulässige Landkartenfärbung. Das Landkartenfärbungsproblem ist also das Knotenfärbungsproblem des zugehörigen Graphen mit $b_v = 1$ für alle $v \in V$.

Die Aufgabe, in einer geographischen Region die Sendefrequenzen von Rundfunksendern (oder Mobilfunkantennen) so zu verteilen, dass sich die Sender gegenseitig nicht stören und alle Rundfunkteilnehmer, die für sie gedachten Programme auch empfangen können, kann man als Färbungsproblem (mit weiteren Nebenbedingungen) formulieren.△

(3.15) Schnitt-Probleme Gegeben sei ein Graph $G = (V, E)$ mit Kantengewichten $c_e \in \mathbb{R}$ für alle $e \in E$. Das Problem, einen Schnitt $\delta(W)$ in G zu finden mit maximalem Gewicht $c(\delta(W))$, heißt *Max-Cut-Problem*. Sind alle Kantengewichte c_e nicht-negativ, so nennt man das Problem, einen Schnitt minimalen Gewichts in G zu finden, *Min-Cut-Problem*.

Das Min-Cut-Problem ist in der Theorie der Netzwerkflüsse sehr wichtig (siehe Kapitel 7). Das Max-Cut-Problem hat z. B. eine interessante Anwendung in der Physik, und zwar kann man beim Studium magnetischer Eigenschaften von Spingläsern im Rahmen des Ising Modells die Aufgabe, einen Grundzustand (energieminimale Konfiguration bei 0° K) zu bestimmen, als Max-Cut-Problem formulieren. Ich will diese Anwendung kurz skizzieren.

Ein *Spinglas* besteht aus nichtmagnetischem Material, das an einigen Stellen durch magnetische Atome "verunreinigt" ist. Man interessiert sich für die Energie des Systems und die Orientierung der magnetischen Atome (Verunreinigungen) bei 0° K, also für den so genannten (gefrorenen) Grundzustand des Spinglases. Dieser Grundzustand ist experimentell nicht herstellbar, und die Physiker haben unterschiedliche, sich z. T. widersprechende Theorien über einige Eigenschaften dieses Grundzustandes.

Mathematisch wird dieses Problem wie folgt modelliert. Jeder Verunreinigung i wird ein Vektor $S_i \in \mathbb{R}^3$ zugeordnet, der (bei einem gegebenen Bezugssytem) die Orientierung des Atomes im Raum, d. h. den magnetischen Spin, beschreibt. Zwischen zwei Verunreinigungen i, j besteht eine magnetische Interaktion, die durch

$$H_{ij} = J(r_{ij})S_i \cdot S_j$$

beschrieben wird, wobei $J(r_{ij})$ eine Funktion ist, die vom Abstand r_{ij} der Verunreinigungen abhängt, und $S_i \cdot S_j$ das innere Produkt der Vektoren S_i, S_j ist. In der Praxis wird J (bei gewissen physikalischen Modellen) wie folgt bestimmt:

$$J(r_{ij}) := \cos(Kr_{ij})/r_{ij}^3\,,$$

wobei K eine materialabhängige Konstante ist (z. B. $K = 2.4 \times 10^8$). Die gesamte Energie einer Spinkonfiguration ist gegeben durch

$$H = -\sum J(r_{ij})S_i \cdot S_j + \sum F \cdot S_i\,,$$

wobei F ein äußeres magnetisches Feld ist. (Der Einfachheit halber nehmen wir im folgenden an $F = 0$.) Ein Zustand minimaler Energie ist also dadurch charakterisiert, dass $\sum J(r_{ij})S_i \cdot S_j$ maximal ist.

Das hierdurch gegebene Maximierungsproblem ist mathematisch kaum behandelbar. Von Ising wurde folgende Vereinfachung vorgeschlagen. Statt jeder beliebigen räumlichen Orientierung werden jeder Verunreinigung nur zwei Orientierungen erlaubt: "Nordpol oben" oder "Nordpol unten". Die dreidimensionalen Vektoren S_i werden dann in diesem Modell durch Variable s_i mit Werten in der zweielementigen Menge $\{1, -1\}$ ersetzt. Unter Physikern besteht Übereinstimmung darüber, dass dieses Ising-Modell das wahre Verhalten gewisser Spingläsern gut widerspiegelt. Das obige Maximierungsproblem lautet dann bezüglich des Ising Modells:

$$\max\{\sum J(r_{ij})s_i s_j \mid s_i \in \{-1, 1\}\}\,.$$

Nach dieser durch die Fachwissenschaftler vorgenommenen Vereinfachung ist der Schritt zum Max-Cut-Problem leicht. Wir definieren einen Graphen $G = (V, E)$, wobei jeder Knoten aus V eine Verunreinigung repräsentiert, je zwei Knoten i, j sind durch eine Kante verbunden, die das Gewicht $c_{ij} = -J(r_{ij})$ trägt. (Ist r_{ij} groß, so ist nach Definition c_{ij} sehr klein, und üblicherweise werden Kanten mit kleinen Gewichten c_{ij} gar nicht berücksichtigt). Eine Partition von V in V_1 und V_2 entspricht einer Orientierungsfestlegung der Variablen, z. B. $V_1 := \{i \in V \mid i$ repräsentiert eine Verunreinigung mit Nordpol oben$\}$, $V_2 := \{i \in V \mid$ der Nordpol von i ist unten$\}$. Bei gegebenen Orientierungen der Atome (Partition V_1, V_2 von V) ist die Energie des Spinglaszustandes also wie folgt definiert:

$$\sum_{i \in V_1, j \in V_2} c_{ij} - \sum_{i,j \in V_1} c_{ij} - \sum_{i,j \in V_2} c_{ij}\,.$$

Der Zustand minimaler Energie kann also durch Maximierung des obigen Ausdrucks bestimmt werden. Addieren wir zu diesem Ausdruck die Konstante $C := \sum_{i,j \in V} c_{ij}$, so

folgt daraus, dass der Grundzustand eines Spinglases durch die Lösung des Max-Cut Problems

$$\max\{\sum_{i\in V_1}\sum_{j\in V_2} c_{ij} \mid V_1, V_2 \text{ Partition von } V\}$$

bestimmt werden kann. Eine genauere Darstellung und die konkrete Berechnung von Grundzuständen von Spingläsern (und weitere Literaturhinweise) kann man in Barahona et al. (1988) finden. Dieses Paper beschreibt auch eine Anwendung des Max-Cut-Problems im VLSI-Design und bei der Leiterplattenherstellung: Die Lagenzuweisung von Leiterbahnen, so dass die Anzahl der Kontaktlöcher minimal ist. △

(3.16) Standortprobleme Probleme dieses Typs tauchen in der englischsprachigen Literatur z. B. unter den Namen Location oder Allocation Problems, Layout Planning, Facilities Allocation, Plant Layout Problems oder Facilities Design auf. Ihre Vielfalt ist (ähnlich wie bei (3.12)) kaum in wenigen Zeilen darstellbar. Ein (relativ allgemeiner) Standardtyp ist der folgende. Gegeben sei ein Graph (oder Digraph), dessen Knoten Städte, Wohnbezirke, Bauplätze, mögliche Fabrikationsstätten etc. repräsentieren, und dessen Kanten Verkehrsverbindungen, Straßen, Kommunikations- oder Transportmöglichkeiten etc. darstellen. Die Kanten besitzen "Gewichte", die z. B. Entfernungen etc. ausdrücken. Wo sollen ein Krankenhaus, ein Flughafen, mehrere Polizei- oder Feuerwehrstationen, Warenhäuser, Anlieferungslager, Fabrikationshallen ... errichtet werden, so dass ein "Optimalitätskriterium" erfüllt ist? Hierbei tauchen häufig Zielfunktionen auf, die nicht linear sind. Z. B. soll ein Feuerwehrdepot so stationiert werden, dass die maximale Entfernung vom Depot zu allen Wohnbezirken minimal ist; drei Auslieferungslager sollen so errichtet werden, dass jedes Lager ein Drittel der Kunden bedienen kann und die Summe der Entfernungen der Lager zu ihren Kunden minimal ist bzw. die maximale Entfernung minimal ist. △

(3.17) Lineare Anordnungen und azyklische Subdigraphen Gegeben sei ein vollständiger Digraph $D_n = (V, A)$ mit Bogengewichten $c((i,j))$ für alle $(i,j) \in A$. Das Problem, eine lineare Reihenfolge der Knoten, sagen wir $i_1, \ldots, i_n$, zu bestimmen, so dass die Summe der Gewichte der Bögen, die konsistent mit der linearen Ordnung sind (also $\sum_{p=1}^{n-1}\sum_{q=p+1}^{n} c((i_p, i_q))$), maximal ist, heißt *Linear-Ordering-Problem*. Das *Azyklische-Subdigraphen-Problem* ist die Aufgabe, in einem Digraphen $D = (V, A)$ mit Bogengewichten eine Bogenmenge $B \subseteq A$ zu finden, die keinen gerichteten Kreis enthält und deren Gewicht maximal ist. Beim *Feedback-Arc-Set-Problem* sucht man eine Bogenmenge minimalen Gewichts, deren Entfernung aus dem Digraphen alle gerichteten Kreise zerstört. △

Die drei in (3.17) genannten Probleme sind auf einfache Weise ineinander transformierbar. Diese Probleme haben interessante Anwendungen z. B. bei der Triangulation von Input-Output-Matrizen, der Rangbestimmung in Turniersportarten, im Marketing und der Psychologie. Weitergehende Informationen finden sich in Grötschel et al. (1984) und in Reinelt (1985). Einige konkrete Anwendungsbeispiele werden in den Übungen behandelt.

(3.18) Entwurf kostengünstiger und ausfallsicherer Telekommunikationsnetzwerke Weltweit wurden in den letzten Jahren (und das geschieht weiterhin) die Kupferkabel, die Telefongespräche etc. übertragen, durch Glasfaserkabel ersetzt. Da Glasfaserkabel enorm hohe Übertragungskapazitäten haben, wurden anfangs die stark "vermaschten" Kupferkabelnetzwerke durch Glasfasernetzwerke mit Baumstruktur ersetzt. Diese Netzwerkstrukturen haben jedoch den Nachteil, dass beim Ausfall eines Verbindungskabels (z. B. bei Baggerarbeiten) oder eines Netzknotens (z. B. durch einen Brand) große Netzteile nicht mehr miteinander kommunizieren können. Man ist daher dazu übergegangen, Telekommunikationsnetzwerke mit höherer Ausfallsicherheit wie folgt auszulegen. Zunächst wird ein Graph $G = (V, E)$ bestimmt; hierbei repräsentiert V die Knotenpunkte, die in einem Telekommunikationsnetz verknüpft werden sollen, und E stellt die Verbindungen zwischen Knoten dar, die durch das Ziehen eines direkten (Glasfaser-) Verbindungskabels realisiert werden können. Gleichzeitig wird geschätzt, was das Legen einer direkten Kabelverbindung kostet. Anschließend wird festgelegt, welche Sicherheitsanforderungen das Netz erfüllen soll. Dies wird so gemacht, dass man für je zwei Knoten bestimmt, ob das Netz noch eine Verbindung zwischen diesen beiden Knoten besitzen soll, wenn ein, zwei, drei ... Kanten oder einige andere Knoten ausfallen. Dann wird ein Netzwerk bestimmt, also eine Teilmenge F von E, so dass alle Knoten miteinander kommunizieren können, alle Sicherheitsanforderungen erfüllt werden und die Baukosten minimal sind. Mit Hilfe dieses Modells (und zu seiner Lösung entwickelter Algorithmen) werden z. B. in den USA Glasfasernetzwerke für so genannte LATA-Netze entworfen und ausgelegt, siehe Grötschel et al. (1992) und Grötschel et al. (1995). Abbildung 3.10(a) zeigt das Netzwerk der möglichen direkten Kabelverbindungen in einer großen Stadt in den USA, Abbildung 3.10(b) zeigt eine optimale Lösung. Hierbei sind je zwei durch ein Quadrat gekennzeichnete Knoten gegen den Ausfall eines beliebigen Kabels geschützt (d. h. falls ein Kabel durchschnitten wird, gibt es noch eine (nicht notwendig direkte, sondern auch über Zwischenknoten verlaufende) Verbindung zwischen je zwei dieser Knoten, alle übrigen Knotenpaare wurden als relativ unwichtig erachtet und mussten nicht gegen Kabelausfälle geschützt werden. △

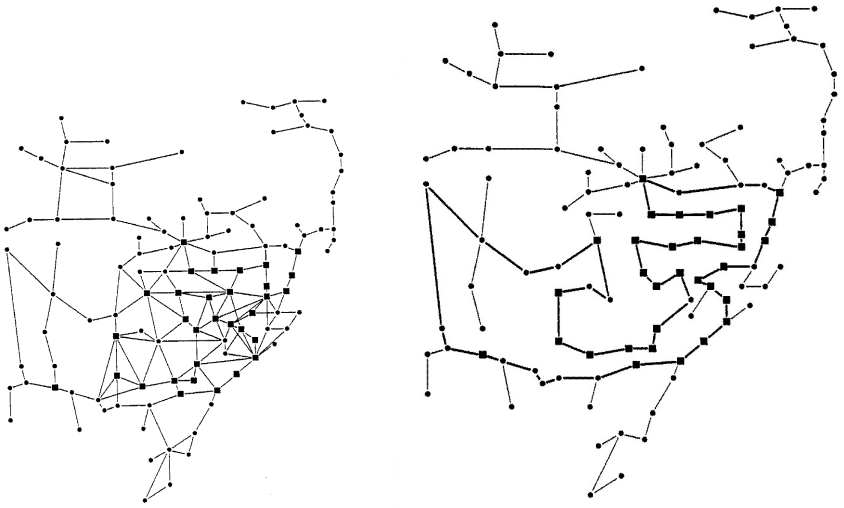
Literaturverzeichnis

M. Aigner. *Graphentheorie: Eine Entwicklung aus dem 4-Farben-Problem.* Teubner Verlag, Studienbücher : Mathematik, Stuttgart, 1984. ISBN 3-519-02068-8.

D. L. Applegate, R. E. Bixby, V. Chvátal, and W. J. Cook. *The Traveling Salesman Problem: A Computational Study.* Princeton University Press, 2006.

F. Barahona, M. Grötschel, M. Jünger, and G. Reinelt. An application of combinatorial optimization to statistical physics and circuit layout design. *Operations Research*, 36 (3):493–513, 1988.

C. Berge. *Hypergraphs, Combinatorics of finite sets*, volume 45. North-Holland Mathematical Library, Amsterdam, 1989.



(a) (b)

Abbildung 3.10: Abbildung (a) stellt ein Telekommunikationsnetz dar. In Abbildung (b) ist eine Optimallösung des kostengünstigsten Telekommunikationsnetzes in diesem Graphen zu sehen.

B. Bollobás. *Graph Theory: An Introductory Course.* Springer Verlag, New York, 1979.

J. A. Bondy and U. S. R. Murty. *Graph Theory.* Springer, Berlin, 2008.

W. J. Cook. *In Pursuit of the Traveling Salesman: Mathematics at the Limits of Computation.* Princeton University Press, 2012.

W. Domschke. *Logistik: Rundreisen und Touren.* Oldenbourg-Verlag, München - Wien, (4., erweiterte Aufl. 1997, 1982.

J. Ebert. Effiziente Graphenalgorithmen. *Studientexte: Informatik*, 1981.

M. C. Golombic. *Algorithmic Graph Theory and Perfect Graphs.* Academic Press, New York. ISBN 1980.

R. L. Graham, M. Grötschel, and L. Lovász, editors. *Handbook of Combinatorics, Volume I.* Elsevier (North-Holland); The MIT Press, Cambridge, Massachusetts, 1995. ISBN ISBN 0-444-82346-8/v.1 (Elsevier); ISBN 0-262-07170-3/v.1 (MIT).

M. Grötschel and O. Holland. Solution of large-scale symmetric travelling salesman problems. *Mathematical Programming, Series A*, 51(2):141–202, 1991.

M. Grötschel, M. Jünger, and G. Reinelt. A Cutting Plane Algorithm for the Linear Ordering Problem. *Operations Research*, 32(6):1195–1220, 1984.

M. Grötschel, C. L. Monma, and M. Stoer. Computational Results with a Cutting Plane Algorithm for Designing Communication Networks with Low-Connectivity Constraints. *Operations Research*, 40(2):309–330, 1992.

M. Grötschel, C. L. Monma, and M. Stoer. Design of Survivable Networks. In M. O. Ball, T. L. Magnanti, C. L. Monma, and G. L. Nemhauser, editors, *Network Models*, volume 7 of *Handbooks in Operations Research and Management Science*, pages 617–672. North-Holland, 1995.

G. Gutin and A. P. Punnen, editors. *The Traveling Salesman Problem and Its Variations.* Kluwer Academic Publishers, 2002.

R. Halin. *Graphentheorie.* Akademie-Verlag Berlin, 2. edition, 1989.

K. Hässig. *Graphentheoretische Methoden des Operations Research.* Teubner-Verlag, Stuttgart, 1979.

D. König. *Theorie der endlichen und unendlichen Graphen.* Akademische Verlagsgesellschaft, Leipzig, 1936. mehrfach auf deutsch und in englischer Übersetzung nachgedruckt.

E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, and D. B. Shmoys. *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization.* Wiley, Chichester, 1985.

T. Lengauer. *Combinatorial Algorithms for Integrated Circuit Layout.* Teubner, Stuttgart und Wiley, Chichester, 1990.

J. K. Lenstra. *Sequencing by Enumerative Methods.* PhD thesis, Mathematisch Centrum, Amsterdam, 1976.

J. G. Oxley. *Matroid Theory.* Oxford University Press, Oxford, 1992. ISBN 0-19-853563-5.

G. Reinelt. *The Linear Ordering Problem: Algorithms and Applications.* Heldermann Verlag, Berlin, 1985.

H. Sachs. *Einführung in die Theorie der endlichen Graphen.* Teubner, Leipzig, 1970, und Hanser, München, 1971, 1970.

M. Stoer. Design of survivable networks. *Lecture Notes for Mathematics*, 1992.

H. W. und G. Nägler. *Graphen, Algorithmen, Programme.* VEB Fachbuchverlag, Leipzig, 1987.

B. und Ghouila-Houri. *Programme, Spiele, Transportnetze.* Teubner Verlag, Leipzig, 1969.

K. Wagner. *Graphentheorie.* BI Wissenschaftsverlag, Mannheim, 1970.

4 Komplexitätstheorie