Diskrete Optimierung

(Algorithmische Diskrete Mathematik II, kurz ADM II)

Skriptum zur Vorlesung im SS 2013

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

Version vom 25. Juni 2013

Vorwort

Bei dem vorliegenden Skript handelt es sich um die Ausarbeitung der vierstündigen Vorlesung „Algorithmische Diskrete Mathematik II (Diskrete Optimierung)“ (mit zugehörigen Übungen und Tutorien), die die zweite Vorlesung des dreisemestrigen Zyklus „Algorithmische Diskrete Mathematik“ bildet. Diese Vorlesung wurde von mir im Sommersemester 2013 zusammen mit Axel Werner an der TU Berlin gehalten, der auch an der Ausarbeitung des vorliegenden Vorlesungsskripts beteiligt war.

Wir gehen in dieser Vorlesung davon aus, dass die Studierenden die Vorlesung „Algorithmische Mathematik I (Einführung in die Lineare und Kombinatorische Optimierung)“ gehört haben und das zugehörige Vorlesungsskriptum

`http://www.zib.de/groetschel/teaching/WS1213/Skriptum_ADM_I_130326.pdf`

kennen. Wir werden im vorliegenden Skript häufig auf das ADM I Skript Bezug nehmen und übernehmen die Notation aus diesem Skript.

Der Inhalt dieser Vorlesung besteht aus einer (bunten) Mischung von Polyedertheorie, Matroid- und Graphentheorie, linearer, kombinatorischer und gemischt-ganzzahliger Optimierung. Einige Themen aus ADM I (z. B. Polyeder, Simplex-Algorithmus, Approximationsverfahren) werden erneut aufgegriffen und vertieft. Wir versuchen dabei, verschiedene Aspekte der diskreten Mathematik miteinander zu verschränken und Bezüge zwischen den einzelnen Themenbereichen sichtbar zu machen. Die (algorithmische) diskrete Mathematik besteht nicht aus voneinander unabhängigen Einzelthemen und hochspezialisierten Werkzeugkästen, erst die Kombination der vielen Strukturen, Analysemethoden und algorithmischen Ansätzen macht diese sich stark entwickelnde Teildisziplin der Mathematik zu einem Gebiet, das neben schöner Theorie eine große Vielfalt an realen Anwendungen bietet.

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/SS2013/Lecture-SS2013deutsch.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 in der Vorlesung erfolgten Schilderungen komplexer Anwendungsfälle, der Schwierigkeiten bei der mathematischen 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 Polyedertheorie

Der zweite Teil des Vorlesungszyklus beginnt mit dem Ausbau der Polyedertheorie. Wir werden weitere Darstellungen von Polyedern angeben und neue Operationen mit Polyedern einführen, welche später an verschiedenen Stellen benötigt werden. Wir werden dabei meistens von der geometrischen Anschauung ausgehen und die Begriffe von dieser Sicht aus motivieren. Besonderes Augenmerk wird allerdings auch auf die analytische Beschreibung der geometrischen Konzepte gelegt. Weiterführende Details zur Polyedertheorie finden sich z. B. in Grünbaum (2003), Ziegler (2010) oder auch Matoušek (2002).

Wir erinnern daran, dass ein Polyeder P die Lösungsmenge eines linearen Ungleichungssystems $Ax \leq b$ ist und benutzen dafür die Bezeichnung $P = P(A, b)$. Polyeder der Form $\{x \in \mathbb{K}^n \mid Ax = b, x \geq 0\}$ bezeichnen wir mit $P^{=}(A, b)$. Ein Polytop ist ein beschränktes Polyeder.

1.1 Transformationen von Polyedern

Wir haben in der Vorlesung ADM I bereits Projektionen von Polyedern entlang eines Richtungsvektors c untersucht und in Satz (10.13) festgestellt, dass eine derartige Projektion wieder ein Polyeder ist. Wenn man mehrfach hintereinander projiziert, bleibt diese Eigenschaft natürlich erhalten. Sind $A \in \mathbb{K}^{(m,n)}$, $b \in \mathbb{K}^m$ und $k, r \geq 0$ mit $k + r = n$, so nennt man die Menge

$$Q := \left\{ x \in \mathbb{K}^k \mid \exists\, y \in \mathbb{K}^r \text{ mit } \begin{pmatrix} x \\ y \end{pmatrix} \in P(\tilde{A}, b) \right\}$$

eine *Projektion von $P(A, b)$ auf $\mathbb{K}^k$*. Hierbei ist $\tilde{A} \in \mathbb{K}^{(m,n)}$ eine Matrix, die durch Spaltenvertauschung aus A hervorgeht. Offensichtlich folgt aus Satz (10.13):

(1.1) Bemerkung. Jede Projektion eines Polyeders $P(A, b) \subseteq \mathbb{K}^n$ auf $\mathbb{K}^k$, $k \leq n$, ist ein Polyeder. △

Dieser Sachverhalt ist in größerer Allgemeinheit gültig. Erinnern wir uns daran, dass jede *affine Abbildung* $f : \mathbb{K}^n \to \mathbb{K}^k$ gegeben ist durch eine Matrix $D \in \mathbb{K}^{(k,n)}$ und einen Vektor $d \in \mathbb{K}^k$, so dass

$$f(x) = Dx + d \quad \forall\, x \in \mathbb{K}^n.$$

Für derartige Abbildungen gilt:

(1.2) Satz. *Affine Bilder von Polyedern sind Polyeder.* △

Beweis. Seien $P = P(A,b) \subseteq \mathbb{K}^n$ ein Polyeder und $f(x) = Dx + d$ eine affine Abbildung von $\mathbb{K}^n$ in den $\mathbb{K}^k$, dann gilt

$$\begin{aligned} f(P) &= \{y \in \mathbb{K}^k \mid \exists\, x \in \mathbb{K}^n \text{ mit } Ax \leq b \text{ und } y = Dx + d\} \\ &= \{y \in \mathbb{K}^k \mid \exists\, x \in \mathbb{K}^n \text{ mit } B\begin{pmatrix} x \\ y \end{pmatrix} \leq \overline{b}\}, \end{aligned}$$

wobei

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

Wenden wir nun das Projektionsverfahren (10.11) aus ADM I iterativ auf $B, \overline{b}$ und die Richtungsvektoren $e_1, e_2, \ldots, e_n$ an, so erhalten wir nach Satz (10.12) ein System $C\binom{x}{y} \leq c$ mit $C = (0, \overline{C})$, und es gilt:

$$\begin{aligned} \forall\, y \in \mathbb{K}^k : \big(\exists\, x \in \mathbb{K}^n \text{ mit } B\begin{pmatrix} x \\ y \end{pmatrix} \leq \overline{b} &\iff \exists\, x \in \mathbb{K}^n \text{ mit } C\begin{pmatrix} x \\ y \end{pmatrix} \leq c \\ &\iff \overline{C}y \leq c\big). \end{aligned}$$

Daraus folgt $f(P) = \{y \in \mathbb{K}^k \mid \overline{C}y \leq c\}$ ist ein Polyeder. □

Man beachte, dass der Beweis von Satz (1.2) durch die Anwendung der Fourier-Motzkin-Elimination sogar ein Verfahren zur expliziten Konstruktion des affinen Bildes von $P(A,b)$ beinhaltet.

Wir erinnern hier an unsere Konventionen zur Bildung von linearen, affinen, konvexen und konischen Hüllen von Mengen und Matrizen, die in Abschnitt 2.2.3 des ADM I Skripts zu finden sind. Aus Satz (1.2) ergibt sich dann direkt die folgende (auch aus anderen Gründen unmittelbar einsichtige) Beobachtung.

(1.3) Korollar (Satz von Weyl). *Für jede Matrix $A \in \mathbb{K}^{(m,n)}$ gilt:*

$$\left.\begin{array}{l} \operatorname{lin}(A) \\ \operatorname{aff}(A) \\ \operatorname{conv}(A) \\ \operatorname{cone}(A) \end{array}\right\} \quad \textit{ist ein Polyeder.}$$

△

Beweis. Wir zeigen die Behauptung für die konische Hülle. Alle anderen Fälle beweist man analog.

$$\begin{aligned} \operatorname{cone}(A) &= \{x \in \mathbb{K}^m \mid \exists\, y \geq 0 \text{ mit } x = Ay\} \\ &= \{x \in \mathbb{K}^m \mid \exists\, y \in \mathbb{K}^n \text{ mit } \begin{pmatrix} I & -A \\ -I & A \\ 0 & -I \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \leq 0\}. \end{aligned}$$

Die letzte Menge ist die Projektion eines Polyeders im $\mathbb{K}^{m+n}$ auf den $\mathbb{K}^m$, also nach (1.1) bzw. (1.2) ein Polyeder. □

Offenbar besagt die obige Folgerung nichts anderes als: Die lineare, affine, konvexe oder konische Hülle einer endlichen Teilmenge des $\mathbb{K}^n$ ist ein Polyeder. Für die konische Hülle hat dies Weyl (1935) gezeigt (daher der Name für Korollar (1.3)).

(1.4) Korollar. *Die Summe $P = P_1 + P_2$ zweier Polyeder P_1, P_2 ist ein Polyeder.* △

Beweis. Es seien $P_1 = P(A, a)$, $P_2 = P(B, b)$, dann gilt:

$$\begin{aligned} P = P_1 + P_2 &= \{x + y \in \mathbb{K}^n \mid Ax \leq a, By \leq b\} \\ &= \{z \in \mathbb{K}^n \mid \exists\, x, y \in \mathbb{K}^n \text{ mit } Ax \leq a, By \leq b, z = x + y\} \\ &= \{z \in \mathbb{K}^n \mid \exists\, x, y \in \mathbb{K}^n \text{ mit } A(z - y) \leq a, B(z - x) \leq b, z = x + y\} \\ &= \{z \in \mathbb{K}^n \mid \exists\, x, y \in \mathbb{K}^n \text{ mit } D \begin{pmatrix} x \\ y \\ z \end{pmatrix} \leq \begin{pmatrix} a \\ b \\ 0 \end{pmatrix}\} \end{aligned}$$

mit

$$D = \begin{pmatrix} 0 & -A & A \\ -B & 0 & B \\ 1 & 1 & -1 \\ -1 & -1 & 1 \end{pmatrix}.$$

Also ist P die Projektion eines Polyeders des $\mathbb{K}^{3n}$ auf den $\mathbb{K}^n$, und somit nach (1.1) ein Polyeder. □

Verbinden wir nun die Erkenntnis aus (1.3), dass $\operatorname{conv}(A)$ und $\operatorname{cone}(B)$ Polyeder sind, mit (1.4), so erhalten wir:

(1.5) Korollar. *Es seien $A \in \mathbb{K}^{(m,n)}$, $B \in \mathbb{K}^{(m,n')}$, dann gilt*

$$P = \operatorname{conv}(A) + \operatorname{cone}(B)$$

ist ein Polyeder. △

Die obige Folgerung erscheint (durch geschickte Vorbereitung) völlig trivial, sie ist jedoch eine durchaus beachtenswerte Erkenntnis, denn wir werden bald zeigen, dass in der Tat alle Polyeder von der Form $\operatorname{conv}(A) + \operatorname{cone}(B)$ sind.

1.2 Kegelpolarität

Es gibt mehrere Möglichkeiten die Umkehrung von (1.5) zu beweisen. Eine besondere elegante, die eine geometrische Version des Farkas-Lemmas benutzt, führt über die Kegelpolarität. Diese Operation mag zunächst nur als technisches Hilfsmittel erscheinen. Sie und ihre Verallgemeinerungen (allgemeine Polaritäten, Blocker, Antiblocker) sind jedoch bedeutende Methoden in der Polyedertheorie und der linearen sowie ganzzahligen Optimierung.

Wir beginnen mit einer Neuinterpretation des Farkas-Lemmas (11.2)(c) aus ADM I. Dieses besagt

$$\exists\, x \geq 0, Ax = b \iff \forall\, u\ (A^T u \geq 0 \Rightarrow u^T b \geq 0).$$

Durch diese Aussage sind offenbar auch alle rechten Seiten b charakterisiert, für die $x \geq 0$, $Ax = b$ eine Lösung hat. Nach Definition gilt $\text{cone}(A) = \{b \in \mathbb{K}^m \mid \exists\, x \geq 0 \text{ mit } Ax = b\}$, also können wir aus der Aussage (11.2)(c) des ADM I Skripts folgern:

(1.6) Bemerkung. Für alle Matrizen $A \in \mathbb{K}^{(m,n)}$ gilt:

$$\text{cone}(A) = \{b \in \mathbb{K}^m \mid u^T b \leq 0\ \forall u \in P(A^T, 0)\}. \qquad \triangle$$

Bemerkung (1.6) kann man geometrisch wie folgt beschreiben. Die Menge der zulässigen rechten Seiten b von $Ax = b$, $x \geq 0$ ist genau die Menge aller Vektoren $b \in \mathbb{K}^m$, welche einen stumpfen Winkel mit allen Vektoren des Kegels $P(A^T, 0)$ bilden. Allgemeiner definieren wir nun für jede beliebige Menge $S \subseteq \mathbb{K}^n$

$$S^\circ := \{y \in \mathbb{K}^n \mid y^T x \leq 0\ \forall x \in S\}.$$

S° ist die Menge aller Vektoren, die einen stumpfen Winkel mit allen Vektoren aus S bilden. S° heißt *polarer Kegel* von S. (Überzeugen Sie sich, dass S° ein Kegel ist!) Wir erinnern hier an das in der linearen Algebra definierte *orthogonale Komplement*

$$S^\perp := \{y \in \mathbb{K}^n \mid y^T x = 0\ \forall x \in S\}.$$

Offensichtlich gilt $S^\perp \subseteq S^\circ$. Unter Benutzung der obigen Definition können wir Bemerkung (1.6) nun auch wie folgt aufschreiben.

(1.7) Korollar. *Für alle Matrizen $A \in \mathbb{K}^{(m,n)}$ gilt*

$$P(A^T, 0)^\circ = \text{cone}(A) \quad \textit{und} \quad P(A, 0)^\circ = \text{cone}(A^T). \qquad \triangle$$

Korollar (1.7) und die vorher gemachten Beobachtungen wollen wir an einem Beispiel erläutern. Es sei

$$A = \begin{pmatrix} -3 & 1 \\ 1 & -2 \end{pmatrix},$$

dann sind die Kegel $P(A, 0)$ und $P(A, 0)^\circ$ in Abbildung 1.1 gezeichnet.

$P(A, 0)^\circ$ besteht also aus allen Vektoren, die mit den Elementen des Kegels $P(A, 0)$ einen stumpfen Winkel bilden, und das sind gerade diejenigen Vektoren, die als konische Kombination der Normalenvektoren $A_i.$ dargestellt werden können, also

$$P(A, 0)^\circ = \text{cone}\left(\begin{pmatrix} -3 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ -2 \end{pmatrix}\right).$$

Ferner gilt: $Ax = b$, $x \geq 0$ ist genau dann lösbar, wenn $b \in P(A, 0)^\circ$ gilt. Daraus folgt z. B., dass $Ax = \binom{0}{1}$, $x \geq 0$ nicht lösbar ist, während $Ax = \binom{-1}{0}$, $x \geq 0$ eine Lösung hat.

Aus der Definition des polaren Kegels und des orthogonalen Komplements ergeben sich unmittelbar einige triviale Beziehungen, deren Beweis wir dem Leser zur Übung überlassen. Wir schreiben im Weiteren

$$S^{\circ\circ} := (S^\circ)^\circ.$$

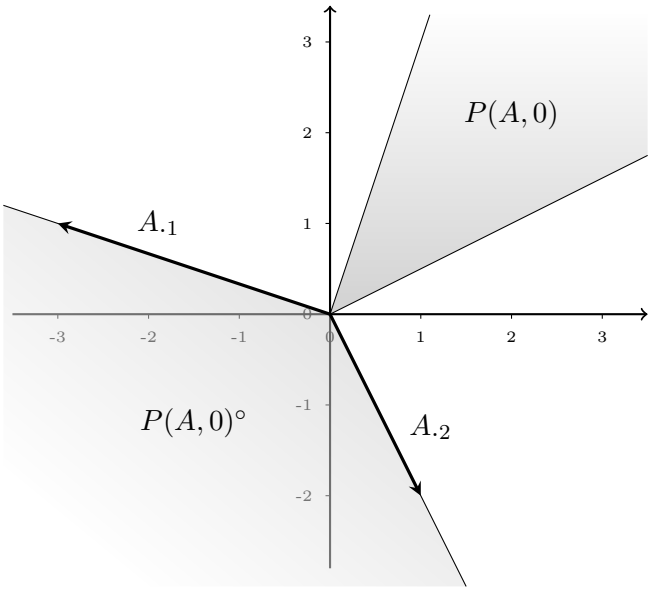


Abbildung 1.1: Kegel und polarer Kegel

(1.8) Bemerkung (Hausaufgabe). Für $S, S_i \subseteq \mathbb{K}^n$, $i = 1, \ldots, k$ gilt:

(a) $S_i \subseteq S_j \Longrightarrow S_j^\circ \subseteq S_i^\circ$

(b) $S \subseteq S^{\circ\circ}$

(c) $\left(\bigcup_{i=1}^k S_i\right)^\circ = \bigcap_{i=1}^k S_i^\circ$

(d) $S^\circ = \mathrm{cone}(S^\circ) = (\mathrm{cone}(S))^\circ$

(e) $S = \mathrm{lin}(S) \Longrightarrow S^\circ = S^\perp$. Gilt die Umkehrung?

(f) Ersetzen wir in (a), ..., (d) "$\circ$" durch "$\perp$", sind dann auch noch alle Behauptungen wahr? △

(1.9) Bemerkung (Hausaufgabe). Für welche Mengen $S \subseteq \mathbb{K}^n$ gilt

(a) $S^\circ = S^{\circ\circ\circ}$,

(b) $S = S^\circ$? △

Die Aussage (1.8)(d) impliziert insbesondere:

(1.10) Korollar. $\mathrm{cone}(A^T)^\circ = P(A, 0)$. △

Beweis. $(\mathrm{cone}(A^T))^\circ = \mathrm{cone}((A^T)^\circ) = (A^T)^\circ = \{x \mid Ax \leq 0\} = P(A, 0)$. □

Das folgende Korollar aus (1.7) und (1.10) wird in der Literatur häufig mit einem Namen belegt.

(1.11) Satz (Polarensatz). *Für jede Matrix $A \in \mathbb{K}^{(m,n)}$ gilt:*

$$P(A,0)^{\circ\circ} = P(A,0),$$
$$\operatorname{cone}(A)^{\circ\circ} = \operatorname{cone}(A).$$

△

Beweis.

$$P(A,0) \overset{(1.10)}{=} \operatorname{cone}(A^T)^\circ \overset{(1.7)}{=} P(A,0)^{\circ\circ},$$
$$\operatorname{cone}(A) \overset{(1.7)}{=} P(A^T,0)^\circ \overset{(1.10)}{=} \operatorname{cone}(A)^{\circ\circ}.$$ □

Unser kurzer Exkurs über Kegelpolarität ist damit beendet.

1.3 Darstellungssätze

Wir wollen nun zeigen, dass Polyeder nicht nur in der Form $P(A,b)$ dargestellt werden können und benutzen dazu die bisher entwickelte Maschinerie.

(1.12) Satz (Minkowski (1896)). *Eine Teilmenge $K \subseteq \mathbb{K}^n$ ist genau dann ein polyedrischer Kegel, wenn K die konische Hülle von endlich vielen Vektoren ist. Mit anderen Worten: Zu jeder Matrix $A \in \mathbb{K}^{(m,n)}$ gibt es eine Matrix $B \in \mathbb{K}^{(n,k)}$, so dass*

$$P(A,0) = \operatorname{cone}(B)$$

gilt und umgekehrt. △

Beweis. $P(A,0) \overset{(1.10)}{=} \operatorname{cone}(A^T)^\circ \overset{(1.3)}{=} P(B^T,0)^\circ \overset{(1.7)}{=} \operatorname{cone}(B)$. □

(1.13) Satz. *Es seien $A \in \mathbb{K}^{(m,n)}$, $b \in \mathbb{K}^m$, dann existieren endliche Mengen $V, E \subseteq \mathbb{K}^n$ mit*

$$P(A,b) = \operatorname{conv}(V) + \operatorname{cone}(E).$$ △

Beweis. Setze

$$H := P\left(\begin{pmatrix} A & -b \\ 0^T & -1 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \end{pmatrix}\right),$$

dann gilt: $x \in P(A,b) \iff \binom{x}{1} \in H$. H ist nach Definition ein polyedrischer Kegel. Also gibt es nach Satz (1.12) eine Matrix $B \in \mathbb{K}^{(n+1,k)}$ mit $H = \operatorname{cone}(B)$. Aufgrund der Definition von H hat die letzte Zeile von B nur nichtnegative Elemente. Durch Skalieren

der Spalten von B und Vertauschen von Spalten können wir B in eine Matrix $\overline{B}$ so umformen, dass gilt

$$\overline{B} = \begin{pmatrix} V & E \\ \mathbb{1}^T & 0^T \end{pmatrix}, \quad \operatorname{cone}(\overline{B}) = H.$$

Daraus folgt:

$$\begin{aligned} x \in P(A,b) &\iff \begin{pmatrix} x \\ 1 \end{pmatrix} \in H \\ &\iff x = V\lambda + E\mu \text{ mit } \lambda^T\mathbb{1} = 1,\ \lambda, \mu \geq 0 \\ &\iff x \in \operatorname{conv}(V) + \operatorname{cone}(E). \end{aligned}$$

□

(1.14) Korollar. *Eine Teilmenge $P \subseteq \mathbb{K}^n$ ist genau dann ein Polytop, wenn P die konvexe Hülle endlich vieler Vektoren ist.* △

Beweis. Sei $V \subseteq \mathbb{K}^n$ endlich und $P = \operatorname{conv}(V)$, dann ist P nach (1.3) ein Polyeder. Ist $x \in P$, so gilt $x = \sum_{i=1}^k \lambda_i v_i$, $v_i \in V$, $\lambda_i \geq 0$, $\sum_{i=1}^k \lambda_i = 1$, und somit $\|x\| \leq \sum_{i=1}^k \|v_i\|$, d. h. $P \subseteq \{x \mid \|x\| \leq \sum_{v \in V} \|v\|\}$. Also ist P beschränkt, d. h. P ist ein Polytop.

Ist umgekehrt P ein Polytop, so gibt es nach Satz (1.13) endliche Mengen V, E mit $P = \operatorname{conv}(V) + \operatorname{cone}(E)$. Gibt es einen Vektor $e \in E$ mit $e \neq 0$, so gilt für alle $n \in \mathbb{N}$: $x + ne \in P$ für alle $x \in \operatorname{conv}(V)$. Also ist P unbeschränkt, falls $E \setminus \{0\} \neq \emptyset$. Daraus folgt $E \in \{\emptyset, \{0\}\}$, und dann gilt trivialerweise $\operatorname{conv}(V) = \operatorname{conv}(V) + \operatorname{cone}(E) = P$. □

(1.15) Satz (Darstellungssatz). *Eine Teilmenge $P \subseteq \mathbb{K}^n$ ist genau dann ein Polyeder, wenn P die Summe eines Polytops und eines polyedrischen Kegels ist, d. h. wenn es endliche Mengen $V, E \subseteq \mathbb{K}^n$ gibt mit*

$$P = \operatorname{conv}(V) + \operatorname{cone}(E).$$ △

Beweis. Kombiniere (1.12), (1.13), (1.14) und (1.5). □

Ist $P \subseteq \mathbb{K}^n$ ein Polyeder, so wissen wir nunmehr, dass es für P zwei mögliche Darstellungen gibt. Es gilt nämlich

$$P = P(A,b) = \operatorname{conv}(V) + \operatorname{cone}(E),$$

wobei A eine (m,n)-Matrix, $b \in \mathbb{K}^m$ und V, E endliche Mengen sind. Diese beiden Darstellungen sind grundsätzlich verschieden, was natürlich in vielerlei Hinsicht nützlich sein kann. Manche Aussagen über Polyeder sind völlig trivial, wenn man von der einen Beschreibung ausgeht, während sie aus der anderen Beschreibung nicht unmittelbar folgen.

Die Darstellung $P(A,b)$ nennt man auch *äußere Beschreibung* von P. Der Grund für diese Bezeichnung liegt darin, dass man wegen

$$P \;=\; \bigcap_{i=1}^m \{x \mid A_i.x \leq b_i\} \;\subseteq\; \{x \mid A_i.x \leq b_i\},$$

das Polyeder P als Durchschnitt von größeren Mengen betrachten kann. P wird sozusagen „von außen“ durch sukzessives Hinzufügen von Ungleichungen (bzw. Halbräumen) konstruiert.

Hingegen nennt man $\operatorname{conv}(V) + \operatorname{cone}(E)$ eine *innere Beschreibung* von P. Ist $E = \emptyset$, so ist die Bezeichnung offensichtlich, denn $V \subseteq P$ und somit wird P durch konvexe Hüllenbildung von Elementen von sich selbst erzeugt. Analoges gilt, wenn P ein polyedrischer Kegel ist. Sind jedoch V und E nicht leer, dann ist E nicht notwendigerweise eine Teilmenge von P, jedoch gelingt es eben aus den Vektoren $v \in V$ zusammen mit den Vektoren $e \in E$ das Polyeder P „von innen her“ zu konstruieren.

Die Sätze (1.12), (1.14) und (1.15) beinhalten weitere wichtige Charakterisierungen von polyedrischen Kegeln, Polytopen und Polyedern. Wir erinnern uns aus der linearen Algebra daran, dass jeder lineare Teilraum L des $\mathbb{K}^n$ eine endliche Basis hat, d. h. eine endliche Teilmenge B besitzt, so dass B linear unabhängig ist und $L = \operatorname{lin}(B)$ gilt. Die linearen Teilräume des $\mathbb{K}^n$ sind also diejenigen Teilmengen des $\mathbb{K}^n$, deren Elemente durch Linearkombinationen einer endlichen Menge erzeugt werden können. Nach (1.14) sind Polytope genau diejenigen Teilmengen des $\mathbb{K}^n$, die durch Konvexkombinationen einer endlichen Menge erzeugt werden können.

Wir werden später sehen, dass es sogar eine eindeutig bestimmte minimale (im Sinne der Mengeninklusion) endliche Menge $V \subseteq \mathbb{K}^n$ gibt mit $P = \operatorname{conv}(V)$, d. h. Polytope haben sogar eine eindeutig bestimmte „konvexe Basis“. Nach (1.12) sind polyedrische Kegel genau diejenigen Teilmengen des $\mathbb{K}^n$, die ein endliches „Kegelerzeugendensystem“ haben. Auch hier gibt es natürlich minimale endliche Mengen, die die Kegel konisch erzeugen. Aber nur unter zusätzlichen Voraussetzungen haben zwei minimale konische Erzeugendensysteme auch gleiche Kardinalität, und Eindeutigkeit gilt lediglich bis auf Multiplikation mit positiven Skalaren. Häufig nennt man eine Teilmenge T des $\mathbb{K}^n$ *endlich erzeugt*, falls $T = \operatorname{conv}(V) + \operatorname{cone}(E)$ für endliche Mengen V, E gilt. Nach (1.15) sind also die Polyeder gerade die endlich erzeugten Teilmengen des $\mathbb{K}^n$. Fassen wir zusammen, so gilt:

(1.16) Bemerkung. Ist $T \subseteq \mathbb{K}^n$, so gilt

(a) T ist ein linearer Teilraum $\iff$ T ist die lineare Hülle einer endlichen Menge.

(b) T ist ein affiner Teilraum $\iff$ T ist die affine Hülle einer endlichen Menge.

(c) T ist ein polyedrischer Kegel $\iff$ T ist die konische Hülle einer endlichen Menge.

(d) T ist ein Polytop $\iff$ T ist die konvexe Hülle einer endlichen Menge.

(e) T ist ein Polyeder $\iff$ T ist endlich erzeugt. △

Exkurs: Andere Darstellungsformen von Polyedern

Satz (1.15) und Bemerkung (1.16) zeigen, dass Polyeder auch durch Hüllenbildungsprozesse (linear, affin, konisch, konvex) und nicht nur durch Durchschnitte (Halbräume,

Hyperebenen), die uns in ADM I (siehe (2.1)) zur Definition gedient haben, charakterisiert werden können. Dies sind jedoch nicht die einzigen Möglichkeiten, Polyeder zu beschreiben. Wir können hierauf nicht vertieft eingehen, sondern erwähnen nur zwei Beispiele.

Der harmlos aussehende absolute Betrag $|.|$ ermöglicht in manchen Fällen enorm kompakte Darstellungen. Wir betrachten als Beispiel

$$K(n) := \text{conv}\{e_1, \ldots, e_n, -e_1, \ldots, -e_n\},$$

wobei e_i den i-ten Einheitsvektor im $\mathbb{K}^n$ bezeichnet. $K(n)$ wird in der Literatur *Kreuzpolytop* genannt. Zur Definition des Kreuzpolytops $K(n)$ durch Hüllenbildung benötigt man also $2n$ Vektoren. Will man $K(n)$ als Durchschnitt von Halbräumen darstellen, so sind (beweisbar) 2^n Ungleichungen erforderlich:

$$K(n) = \{x \in \mathbb{K}^n \mid a^T x \leq 1 \; \forall a \in \{-1, 1\}^n\}.$$

Erlaubt man die Benutzung des Absolutbetrages, so ergibt sich

$$K(n) = \{x \in \mathbb{K}^n \mid \sum_{i=1}^{n} |x_i| \leq 1\}.$$

Eine einzige Ungleichung genügt in diesem Falle also zur Darstellung des Kreuzpolytops. Das Kreuzpolytop $K(3)$ im dreidimensionalen Raum ist das bekannte Oktaeder.

Tiefliegende Sätze der reellen algebraischen Geometrie, die auf Bröcker (1991) und Scheiderer (1989) zurückgehen, siehe hierzu Bochnak et al. (1998), zeigen, dass der *Stabilitätsindex* jeder „basic closed semi-algebraic set“ im Raum $\mathbb{R}^n$ den Wert $m := \frac{n(n+1)}{2}$ hat.

Polyeder sind spezielle „basic closed semi-algebraic sets“. Übersetzt in „unsere“ Sprache und bezogen auf Polyeder besagt das Resultat von Bröcker und Scheiderer, dass es zu jedem Polyeder P Polynome $p_1, \ldots, p_m$ in n reellen Variablen mit reellen Koeffizienten gibt, so dass

$$P = \{x \in \mathbb{R}^n \mid p_i(x) \geq 0, i = 1, \ldots, \frac{n(n+1)}{2}\}$$

gilt. Der Beweis ist rein „existenziell“ und liefert kein Konstruktionsverfahren für diese m Polynome. Es gibt allgemeine semi-algebraische Mengen, bei denen man auch beweisbar m Polynome braucht.

Für den Spezialfall von Polyedern wurde in Bosse et al. (2005) gezeigt, dass man im $\mathbb{R}^n$ die benötigten Polynome algorithmisch bestimmen kann und dass man sogar mit $2n$ Polynomen auskommt. Es wird vermutet, dass sogar n Polynome zur Darstellung von Polyedern ausreichen (und dass diese auch konstruiert werden können). Für einen wichtigen Spezialfall haben dies Averkov und Henk bewiesen, der allgemeine Fall ist noch offen.

Eine Konsequenz der oben geschilderten Ergebnisse ist, dass das Kreuzpolytop $K(n)$ statt mit 2^n linearen Ungleichungen mit lediglich $2n$ Polynomialgleichungen beschrieben werden kann.

1.4 Gültige Ungleichungen, Seitenflächen und Dimension

In Abschnitt 1.2 haben wir bereits einen Polarentyp, die Kegelpolare, zur Beweisvereinfachung eingeführt. Hier wollen wir eine weitere Polare betrachten, die es uns ermöglichen wird, eine Charakterisierung bezüglich $P(A,b)$ in eine Charakterisierung bezüglich $\text{conv}(V) + \text{cone}(E)$ zu übertragen.

(1.17) Definition. *Es seien $S \subseteq \mathbb{K}^n$, $a \in \mathbb{K}^n$, $\alpha \in \mathbb{K}$. Die Menge*

$$S^\gamma := \{\begin{pmatrix} a \\ \alpha \end{pmatrix} \in \mathbb{K}^{n+1} \mid a^T x \leq \alpha \; \forall x \in S\}$$

heißt γ-Polare von S. △

Die γ-Polare S^γ kann als die „Menge aller gültigen Ungleichungen bezüglich S“ betrachtet werden. Wir wollen nun die γ-Polare eines Polyeders charakterisieren.

(1.18) Satz. *Es sei $P \subseteq \mathbb{K}^n$, $P \neq \emptyset$, ein Polyeder mit den Darstellungen $P = P(A,b) = \text{conv}(V) + \text{cone}(E)$, dann gilt:*

(a) $P^\gamma = \left\{ \begin{pmatrix} a \\ \alpha \end{pmatrix} \in \mathbb{K}^{n+1} \mid \exists u \geq 0, u^T A = a^T, u^T b \leq \alpha \right\} = \text{cone} \begin{pmatrix} A^T & 0 \\ b^T & 1 \end{pmatrix}.$

(b) $P^\gamma = \left\{ \begin{pmatrix} a \\ \alpha \end{pmatrix} \in \mathbb{K}^{n+1} \mid \begin{pmatrix} V^T & -\mathbb{1} \\ E^T & 0 \end{pmatrix} \cdot \begin{pmatrix} a \\ \alpha \end{pmatrix} \leq 0 \right\} = P\left(\begin{pmatrix} V^T & -\mathbb{1} \\ E^T & 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \end{pmatrix} \right).$ △

Beweis.

(a) $\begin{pmatrix} a \\ \alpha \end{pmatrix} \in P^\gamma \iff Ax \leq b, a^T x > \alpha$ inkonsistent

$\iff \exists u \geq 0, v > 0$ mit $u^T A - v a^T = 0, u^T b - v\alpha \leq 0$ (ADM I, (11.5))

$\iff \exists u \geq 0$ mit $u^T A = a^T, u^T b \leq \alpha$

$\iff \exists u \geq 0, \lambda \geq 0$ mit $\begin{pmatrix} a \\ \alpha \end{pmatrix} = \begin{pmatrix} A^T & 0 \\ b^T & 1 \end{pmatrix} \begin{pmatrix} u \\ \lambda \end{pmatrix}$

$\iff \begin{pmatrix} a \\ \alpha \end{pmatrix} \in \text{cone} \begin{pmatrix} A^T & 0 \\ b^T & 1 \end{pmatrix}.$

(b) $\begin{pmatrix} a \\ \alpha \end{pmatrix} \in P^\gamma \implies a^T v \leq \alpha \; \forall v \in V$ und $a^T(v + \lambda e) \leq \alpha \; \forall v \in V, e \in E, \lambda \geq 0$

$\implies a^T e \leq 0$ (andernfalls wäre $a^T(v + \lambda e) > \alpha$ für genügend großes λ)

$\implies \begin{pmatrix} V^T & -\mathbb{1} \\ E^T & 0 \end{pmatrix} \begin{pmatrix} a \\ \alpha \end{pmatrix} \leq 0.$

Gilt umgekehrt das letztere Ungleichungssystem, und ist $x \in P$, so existieren $v_1, \ldots, v_p \in V$ und $e_1, \ldots, e_q \in E$, $\lambda_1, \ldots, \lambda_p \geq 0$, $\sum_{i=1}^p \lambda_i = 1$, $\mu_1, \ldots, \mu_q \geq 0$, so dass

$$x = \sum_{i=1}^p \lambda_i v_i + \sum_{j=1}^q \mu_j e_j.$$

Und daraus folgt

$$a^T x \;=\; \sum_{i=1}^{p} \lambda_i a^T v_i + \sum_{j=1}^{q} \mu_j a^T e_j \;\leq\; \sum_{i=1}^{q} \lambda_i \alpha + \sum_{j=1}^{q} \mu_j 0 \;=\; \alpha,$$

also gilt $\begin{pmatrix} a \\ \alpha \end{pmatrix} \in P^\gamma$. □

(1.19) Korollar. *Die γ-Polare eines Polyeders $\emptyset \neq P \subseteq \mathbb{K}^n$ ist ein polyedrischer Kegel im $\mathbb{K}^{n+1}$.* △

(1.20) Korollar. *Ist $\emptyset \neq P = P(A,b) = \text{conv}(V) + \text{cone}(E)$ ein Polyeder und $a^T x \leq \alpha$ eine Ungleichung, dann sind äquivalent:*

(i) *$a^T x \leq \alpha$ ist gültig bezüglich P.*

(ii) *$\exists u \geq 0$ mit $u^T A = a^T$, $u^T b \leq \alpha$.*

(iii) *$a^T v \leq \alpha \; \forall v \in V$ und $a^T e \leq 0 \; \forall e \in E$.*

(iv) $\begin{pmatrix} a \\ \alpha \end{pmatrix} \in P^\gamma$. △

Wir erinnern an einige Begriffsbildungen aus der ADM I (siehe Definitionen (8.2) und (8.6) des ADM I Skripts): Eine Teilmenge $F \subseteq P$ eines Polyeders P ist eine *Seitenfläche von P*, wenn es eine gültige Ungleichung $c^T x \leq c_0$ bezüglich P gibt mit $F = P \cap \{x \mid c^T x = c_0\}$. Ist $P = P(A,b)$ und M die Zeilenindexmenge von A, dann ist für eine Teilmenge $F \subseteq P$ die *Gleichheitsmenge* $\text{eq}(F) := \{i \in M \mid A_{i.} x = b_i \; \forall x \in F\}$, die Menge der für alle $x \in F$ bindenden Restriktionen. Für $I \subseteq M$ ist $\text{fa}(I) := \{x \in P \mid A_{I.} x = b_I\}$ die *von I induzierte Seitenfläche.*

Wir zeigen nun, dass man die Gleichheitsmenge einer Seitenfläche explizit berechnen kann.

(1.21) Satz. *Seien $P = P(A,b)$ ein Polyeder und $\emptyset \neq F = \{x \in P \mid c^T x = c_0\}$ eine Seitenfläche von P. Dann gilt*

$$\text{eq}(F) = \{i \in M \mid \exists u \geq 0 \text{ mit } u_i > 0 \text{ und } u^T A = c^T, u^T b = c_0\}.$$ △

Beweis. Nach Voraussetzung und Folgerung (11.19) aus ADM I haben die beiden linearen Programme

$$\text{(P)} \quad \begin{array}{l} \max\; c^T x \\ \quad Ax \leq b \end{array} \qquad \text{und} \qquad \text{(D)} \quad \begin{array}{l} \min\; u^T b \\ \quad u^T A = c^T \\ \quad u \geq 0 \end{array}$$

optimale Lösungen mit gleichem Zielfunktionswert c_0, und F ist die Menge der Optimallösungen von (P). Sei nun $i \in \mathrm{eq}(F)$. Aufgrund des Satzes (11.26) vom starken komplementären Schlupf existieren Optimallösungen $\bar{x}$, $\bar{u}$ von (P), (D) mit $\bar{u}_j > 0 \Leftrightarrow A_{j.}\bar{x} = b_{j.}$. Wegen $\bar{x} \in F$ gilt $A_{i.}\bar{x} = b_i$, also gibt es einen Vektor u mit den geforderten Eigenschaften.

Gibt es umgekehrt einen Vektor $u \geq 0$ mit $u_i > 0$ und $u^T A = c^T$, $u^T b = c_0$, so ist u optimal für (D), und aus dem Satz vom schwachen komplementären Schlupf (11.25) folgt $A_{i.}x = b_i$ für alle $x \in F$, d. h. $i \in \mathrm{eq}(F)$. □

(1.22) Satz. *Seien $P = P(A, b)$ ein Polyeder und $F \neq \emptyset$ eine Teilmenge von P, dann sind äquivalent:*

(i) *F ist eine Seitenfläche von P.*

(ii) *$\exists I \subseteq M$ mit $F = \mathrm{fa}(I) = \{x \in P \mid A_{I.}x = b_I\}$.*

(iii) *$F = \mathrm{fa}(\mathrm{eq}(F))$.* △

Beweis. Gelten (ii) oder (iii), dann ist F offenbar eine Seitenfläche von P.

(i) $\Longrightarrow$ (ii): Sei $F = \{x \in P \mid c^T x = c_0\}$ eine Seitenfläche. Setzen wir $I := \mathrm{eq}(F)$, dann gilt nach Definition $F \subseteq \{x \in P \mid A_{I.}x = b_I\} =: F'$. Ist $x \in F'$, so gilt $x \in P$; es bleibt zu zeigen, dass $c^T x = c_0$ gilt. Zu jedem $i \in I$ gibt es nach (1.21) einen Vektor $u^{(i)}$ mit $u^{(i)} \geq 0$, $u_i^{(i)} > 0$, $(u^{(i)})^T A = c^T$ und $(u^{(i)})^T b = c_0$. Setze

$$u := \sum_{i \in I} \frac{1}{|I|} u^{(i)},$$

dann gilt nach Konstruktion $u_i > 0 \; \forall i \in I$ und ferner $u_i = 0 \; \forall i \in M \setminus I$ (andernfalls wäre $i \in I$ nach (1.21)). Die Vektoren x und u sind zulässig für die linearen Programme (P), (D) des Beweises von (1.21). Aus dem Satz vom schwachen komplementären Schlupf (11.25) aus ADM I folgt, dass sie auch optimal sind. Daraus folgt $c^T x = c_0$ und somit $x \in F$.

(ii) $\Longrightarrow$ (iii) folgt direkt aus dem obigen Beweis. □

Aus Satz (1.22) folgt, dass zur Darstellung einer Seitenfläche von $P(A, b)$ keine zusätzliche Ungleichung benötigt wird. Man braucht lediglich in einigen der Ungleichungen des Systems $Ax \leq b$ Gleichheit zu fordern. Da jede nichtleere Seitenfläche auf diese Weise erzeugt werden kann, folgt:

(1.23) Korollar. *Sind $A \in \mathbb{K}^{(m,n)}$, $b \in \mathbb{K}^m$, dann hat das Polyeder $P(A, b)$ höchstens $2^m + 1$ Seitenflächen.* △

Beweis. $M = \{1, \ldots, m\}$ hat 2^m Teilmengen. Für jede Teilmenge $I \subseteq M$ ist $P \cap \{x \mid A_{I.}x = b_I\}$ eine Seitenfläche von P. Dazu kommt u. U. noch die leere Seitenfläche. □

Man kann Seitenflächen auf ähnliche Weise durch Einführung von Abbildungen analog zu eq bzw. fa bezüglich der Darstellung $P = \text{conv}(V) + \text{cone}(E)$ charakterisieren. Diese Kennzeichnungen von Seitenflächen sind jedoch technisch aufwendiger. Der interessierte Leser sei dazu auf Bachem and Grötschel (1982) verwiesen.

Wir wollen nun zeigen, dass man auch die Dimension einer Seitenfläche eines Polyeders explizit berechnen kann. Zunächst führen wir einen Hilfsbegriff ein.

(1.24) Definition. *Ein Element x eines Polyeders P heißt* innerer Punkt von P, *wenn x in keiner echten Seitenfläche von P enthalten ist.* △

Achtung! Innere Punkte eines Polyeders P sind nicht notwendig auch topologisch innere Punkte im Sinne der natürlichen Topologie des $\mathbb{K}^n$. Unsere inneren Punkte sind topologisch innere Punkte im Sinne der Relativtopologie auf P.

(1.25) Satz. *Jedes nichtleere Polyeder besitzt innere Punkte.* △

Beweis. Sei $P = P(A, b)$ und $I = \text{eq}(P(A, b))$, $J = M \setminus I$. Gilt $I = M$, so hat P keine echten Seitenflächen, also ist jedes Element von P ein innerer Punkt. Andernfalls ist das System $Ax \leq b$ äquivalent zu

$$A_I.x = b_I,$$
$$A_J.x \leq b_J.$$

P hat innere Punkte heißt dann, dass es ein x gibt mit $A_I.x = b_I$ und $A_J.x < b_J$. Zu jedem $i \in J$ existiert nach Definition ein Vektor $y^{(i)} \in P$ mit $A_i.y^{(i)} < b_i$. Setze $y := \frac{1}{|J|} \sum_{i \in J} y^{(i)}$, dann ist y Konvexkombination von Elementen von P, also $y \in P$, und es gilt $A_J.y < b_J$. Mithin ist y ein innerer Punkt von P. □

(1.26) Satz. *Sei F Seitenfläche eines Polyeders $P(A, b)$ und $\bar{x} \in F$. Der Vektor $\bar{x}$ ist ein innerer Punkt von F genau dann, wenn* $\text{eq}(\{\bar{x}\}) = \text{eq}(F)$. △

Beweis. $\bar{x}$ ist genau dann ein innerer Punkt von F, wenn die kleinste (im Sinne der Mengeninklusion) Seitenfläche von F, die $\bar{x}$ enthält, F selbst ist. Offenbar ist $\text{fa}(\text{eq}(\{\bar{x}\}))$ die minimale Seitenfläche von P, die $\bar{x}$ enthält. Daraus folgt die Behauptung. □

(1.27) Satz. *Ist $F \neq \emptyset$ eine Seitenfläche des Polyeders $P(A, b) \subseteq \mathbb{K}^n$, dann gilt*

$$\dim(F) = n - \text{rang}(A_{\text{eq}(F)}.).$$ △

Beweis. Sei $I := \text{eq}(F)$. Aus der linearen Algebra wissen wir, dass $n = \text{rang}(A_I.) + \dim(\text{kern}(A_I.))$ gilt. Zu zeigen ist also: $\dim(F) = \dim(\text{kern}(A_I.))$. Seien

$$r := \dim(\text{kern}(A_I.)) \quad \text{und} \quad s := \dim(F).$$

„$r \geq s$“: Da $\dim(F) = s$, gibt es $s+1$ affin unabhängige Vektoren $x_0, x_1, \ldots, x_s \in F$. Dann sind die Vektoren $x_1 - x_0, \ldots, x_s - x_0$ linear unabhängig und erfüllen $A_I.(x_i - x_0) = 0$. $\text{kern}(A_I.)$ enthält also mindestens s linear unabhängige Vektoren, also gilt $r \geq s$.

„$s \geq r$“: Nach (1.25) besitzt F einen inneren Punkt $\bar{x} \in F$. Nach (1.26) gilt $\mathrm{eq}(\{\bar{x}\}) = \mathrm{eq}(F) = I$, und daraus folgt für $J := M \setminus I$:

$$A_I.\bar{x} = b_I,$$
$$A_J.\bar{x} < b_J.$$

Ist $r = 0$, so gilt $s \geq 0$ wegen $\bar{x} \in F$. Sei also $r \geq 1$, und $\{x_1, \ldots, x_r\}$ sei eine Basis von $\mathrm{kern}(A_I.)$. Für $p = 1, \ldots, r$ und $j \in J$ setze:

$$\delta_{jp} := \begin{cases} \infty, & \text{falls } A_j.x_p = 0 \\ \dfrac{b_j - A_j.\bar{x}}{A_j.x_p} & \text{andernfalls,} \end{cases}$$
$$\varepsilon := \min\{\delta_{jp} \mid j \in J, p \in \{1, \ldots, r\}\}.$$

(Setze $\varepsilon \neq 0$ beliebig, falls $\delta_{jp} = \infty$ für alle j, p.) Für $i \in I$ und alle $p \in \{1, \ldots, r\}$ gilt nun

$$A_i.(\bar{x} + \varepsilon x_p) \;=\; A_i.\bar{x} + \varepsilon A_i.x_p \;=\; A_i.\bar{x} \;=\; b_i,$$

da $A_i.x_p = 0$. Für $j \in J$ gilt

$$\begin{aligned} A_j.(\bar{x} + \varepsilon x_p) &= A_j.\bar{x} + \varepsilon A_j.x_p \\ &\leq A_j.\bar{x} + \delta_{jp} A_j.x_p \\ &= A_j.\bar{x} + b_j - A_j.\bar{x} \\ &= b_j. \end{aligned}$$

Daraus folgt $\bar{x} + \varepsilon x_p \in F$ für alle $p \in \{1, \ldots, r\}$. Da die Vektoren $\varepsilon x_1, \ldots, \varepsilon x_r$ linear unabhängig sind, sind die Vektoren $\bar{x}, \bar{x} + \varepsilon x_1, \ldots, \bar{x} + \varepsilon x_r$ affin unabhängig. Das heißt, F enthält mindestens $r + 1$ affin unabhängige Vektoren, und somit gilt $\dim(F) = s \geq r$. □

(1.28) Korollar. *$P = P(A, b) \subseteq \mathbb{K}^n$ sei ein nichtleeres Polyeder, dann gilt:*

(a) $\dim(P) = n - \mathrm{rang}(A_{\mathrm{eq}(P)}.)$.

(b) *Ist* $\mathrm{eq}(P) = \emptyset$, *dann ist P volldimensional (d. h.* $\dim(P) = n$*).*

(c) *Ist F eine echte Seitenfläche von P, dann gilt* $\dim(F) \leq \dim(P) - 1$. △

Mit Hilfe von Satz (1.27) kann man auch die affine Hülle einer Seitenfläche auf einfache Weise bestimmen.

(1.29) Satz. *Sei $F \neq \emptyset$ eine Seitenfläche des Polyeders $P(A, b)$, dann gilt*

$$\mathrm{aff}(F) = \{x \mid A_{\mathrm{eq}(F)}.x = b_{\mathrm{eq}(F)}\}.$$ △

Beweis. Es seien $I := \mathrm{eq}(F)$ und $T := \{x \mid A_I.x = b_I\}$. Offenbar ist T ein affiner Raum und wegen $F \subseteq T$ gilt $\mathrm{aff}(F) \subseteq \mathrm{aff}(T) = T$. Sei $s = \dim(F)$, dann folgt aus Satz (1.27), dass $\dim(\mathrm{kern}(A_I.)) = s$ und somit $\dim(T) = s$ gilt. Aus $\dim(\mathrm{aff}(F)) = \dim T$ und $\mathrm{aff}(F) \subseteq T$ folgt $\mathrm{aff}(F) = T$. □

1.5 Facetten und Redundanz

Wie wir bereits bemerkt haben, kann man zu einem Ungleichungssystem $Ax \leq b$ beliebig viele Ungleichungen hinzufügen, ohne die Lösungsmenge des Systems zu ändern. Wir wollen nun untersuchen, wie man ein gegebenes Polyeder mit möglichst wenigen Ungleichungen darstellen kann. Dies ist speziell für die lineare Optimierung wichtig, da der Rechenaufwand zur Auffindung einer Optimallösung in der Regel von der Anzahl der vorgelegten Ungleichungen abhängt. Gesucht wird also eine Minimaldarstellung eines Polyeders, um rechentechnische Vorteile zu haben. Es wird sich zeigen, dass hierbei diejenigen Ungleichungen, die maximale echte Seitenflächen eines Polyeders definieren, eine wesentliche Rolle spielen. Deshalb wollen wir derartige Seitenflächen untersuchen.

(1.30) Definition. *$Ax \leq b$ sei ein Ungleichungssystem, und M sei die Zeilenindexmenge von A.*

(a) *Sei $I \subseteq M$, dann heißt das System $A_I.x \leq b_I$* unwesentlich *oder* redundant bezüglich *$Ax \leq b$, wenn $P(A,b) = P(A_{M\setminus I}., b_{M\setminus I})$ gilt.*

(b) *Enthält $Ax \leq b$ ein unwesentliches Teilsystem $A_I.x \leq b_I$, dann heißt $Ax \leq b$* redundant*, andernfalls* irredundant*.*

(c) *Eine Ungleichung $A_i.x \leq b_i$ heißt* wesentlich *oder* nicht redundant bezüglich *$Ax \leq b$, wenn $P(A,b) \neq P(A_{M\setminus\{i\}}., b_{M\setminus\{i\}})$ gilt.*

(d) *Eine Ungleichung $A_i.x \leq b_i$ heißt* implizite Gleichung bezüglich *$Ax \leq b$, wenn $i \in \mathrm{eq}(P(A,b))$ gilt.*

(e) *Ein System $Ax \leq a$, $Bx = b$ heißt* irredundant*, wenn $Ax \leq a$ keine unwesentliche Ungleichung bezüglich des Systems $Ax \leq a$, $Bx \leq b$, $-Bx \leq -b$ enthält und B vollen Zeilenrang hat.*

(f) *Eine nichttriviale Seitenfläche F von $P(A,b)$ heißt* Facette von *$P(A,b)$, falls F in keiner anderen echten Seitenfläche von $P(A,b)$ enthalten ist.* △

Wir weisen ausdrücklich darauf hin, dass Redundanz bzw. Irredundanz keine Eigenschaft des Polyeders $P(A,b)$ ist, sondern eine Eigenschaft des Ungleichungssystems $Ax \leq b$. Wir werden sehen, dass ein Polyeder viele irredundante Beschreibungen haben kann. Ferner ist auch die Annahme falsch, dass man immer durch das gleichzeitige Weglassen aller unwesentlichen Ungleichungen eines Systems $Ax \leq b$ eine irredundante Beschreibung von $P(A,b)$ enthält. Wir wollen nun zunächst unwesentliche Ungleichungen charakterisieren.

(1.31) Satz. *Ein Ungleichungssystem $A_I.x \leq b_I$ ist unwesentlich bezüglich $Ax \leq b$ genau dann, wenn es eine Matrix $U \in \mathbb{K}_+^{(|I|,m)}$ gibt mit $UA = A_I.$, $Ub \leq b_I$ und $U_{.I} = 0$.* △

Beweis. Für jedes $i \in I$ ist nach (1.20) die Ungleichung $A_i.x \leq b_i$ gültig bezüglich $P(A_{M\setminus I}., b_{M\setminus I})$ genau dann, wenn es einen Vektor $\bar{u}_i \geq 0$ gibt mit $\bar{u}_i^T A_{M\setminus I}. = A_i.$ und

$\bar{u}_i^T b \leq b_i$. Dies ist genau dann der Fall, wenn es eine Matrix $U \in \mathbb{K}_+^{(|I|,m)}$ gibt mit $UA = A_{I.}$, $Ub \leq b_I$ und $U_{.I} = 0$. □

(1.32) Korollar. *$A_{i.}x \leq b_i$ ist genau dann redundant bezüglich $Ax \leq b$, wenn es einen Vektor $u \in \mathbb{K}_+^m$ gibt mit $u^T A = A_{i.}$, $u^T b \leq b_i$, $u_i = 0$.* △

Der nächste Satz zeigt, wann man Ungleichungen nicht mehr weglassen kann, ohne das Polyeder zu ändern.

(1.33) Satz. *$P = P(A,b) \neq \emptyset$ sei ein Polyeder. Sei $\emptyset \neq I \subseteq M \setminus \mathrm{eq}(P)$ und $P' := P(A_{M\setminus I.}, b_{M\setminus I})$. Dann gilt*

$$P \neq P' \iff \exists \text{ nichttriviale Seitenfläche } F \subseteq P \text{ mit } \mathrm{eq}(F) \subseteq I \cup \mathrm{eq}(P). \qquad \triangle$$

Beweis. Im Weiteren bezeichnen wir mit $\mathrm{eq}_{P'}$ die „equality set"-Abbildung bezüglich P'. Es gilt offenbar $\mathrm{eq}_{P'}(F) \subseteq \mathrm{eq}(F)$.

„$\Longleftarrow$" Angenommen, es gilt $P = P'$, und F sei eine beliebige nichttriviale Seitenfläche von P (und somit auch von P'). Da F eine nichttriviale Seitenfläche von P' ist, gibt es ein $i \in (M \setminus I) \setminus \mathrm{eq}_{P'}(P')$ mit $i \in \mathrm{eq}_{P'}(F)$. Daraus folgt $\mathrm{eq}(F) \not\subseteq I \cup \mathrm{eq}(P)$.

„$\Longrightarrow$" Angenommen, es gilt $P \neq P'$. Wegen $P \subseteq P'$ heißt dies, es existiert ein Vektor $v \in P' \setminus P$, und somit gibt es eine Indexmenge $\emptyset \neq K \subseteq I$ mit der Eigenschaft $A_{i.}v \leq b_i \; \forall i \in M \setminus K$ und $A_{i.}v > b_i \; \forall i \in K$. Nach Satz (1.25) hat P einen inneren Punkt, sagen wir w, d. h. es gilt $A_{i.}w = b_i \; \forall i \in \mathrm{eq}(P)$ und $A_{i.}w < b_i \; \forall i \in M \setminus \mathrm{eq}(P)$.

Wir betrachten nun einen Punkt y auf der Strecke zwischen v und w, d. h. $y = \lambda w + (1-\lambda)v$ mit $0 \leq \lambda \leq 1$. Aufgrund der Voraussetzungen gilt:

$$\begin{aligned} A_{i.}y &= b_i \quad \forall i \in \mathrm{eq}(P) \\ A_{i.}y &< b_i \quad \forall i \in M \setminus (\mathrm{eq}(P) \cup K), \text{ falls } \lambda > 0. \end{aligned}$$

Ist $i \in K$, so gilt

$$\begin{aligned} A_{i.}y \leq b_i &\iff \lambda A_{i.}w + (1-\lambda)A_{i.}v \leq b_i \\ &\iff \lambda A_{i.}(w-v) \leq b_i - A_{i.}v \\ &\iff \lambda \geq \frac{b_i - A_{i.}v}{A_{i.}(w-v)} \quad (\text{da } A_{i.}(w-v) < 0). \end{aligned}$$

Setzen wir

$$\mu := \max\left\{ \frac{b_i - A_{i.}v}{A_{i.}(w-v)} \mid i \in K \right\},$$
$$L := \left\{ i \in K \mid \mu = \frac{b_i - A_{i.}v}{A_{i.}(w-v)} \right\},$$

dann gilt $z := \mu w + (1-\mu)v \in P$, $\emptyset \neq L \subseteq K \subseteq I$ und

$$\begin{aligned} A_{i.}z &= b_i \; \forall i \in L \\ A_{i.}z &< b_i \; \forall i \in K \setminus L. \end{aligned}$$

Daraus folgt, z ist ein innerer Punkt von $F := \mathrm{fa}(L \cup \mathrm{eq}(P))$. Nach (1.26) gilt dann $\mathrm{eq}(F) = \mathrm{eq}(\{z\}) = L \cup \mathrm{eq}(P)$, und das bedeutet, dass F eine nichttriviale Seitenfläche von P mit $\mathrm{eq}(F) \subseteq I \cup \mathrm{eq}(P)$ ist. □

Wir werden nun wichtige Eigenschaften von Facetten bestimmen, Nichtredundanz kennzeichnen und Facetten charakterisieren.

(1.34) Satz. *Sei F eine Facette von $P = P(A, b)$, dann gilt:*

(a) $\mathrm{eq}(P) \subset \mathrm{eq}(F)$.

(b) *Für alle $i \in \mathrm{eq}(F) \setminus \mathrm{eq}(P)$ gilt*

$$F = \mathrm{fa}(\{i\}) = \{x \in P \mid A_i.x = b_i\}. \qquad \triangle$$

Beweis. (a) gilt offensichtlich für alle nichttrivialen Seitenflächen von P.
(b) Die Abbildung fa ist inklusionsumkehrend, d. h.

$$I \subseteq J \implies \mathrm{fa}(I) \supseteq \mathrm{fa}(J).$$

Daraus folgt $F = \mathrm{fa}(\mathrm{eq}(F)) \subseteq \mathrm{fa}(\{i\})$. Da $i \notin \mathrm{eq}(P)$, muss $\mathrm{fa}(\{i\})$ eine echte Seitenfläche von P sein. Aus der Maximalität von F folgt die Behauptung. □

(1.35) Korollar. *Sei $P = P(A, b)$ ein Polyeder und $\mathcal{F}$ die Menge der Facetten von P. Dann gilt:*

(a) $F_1, F_2 \in \mathcal{F},\ F_1 \neq F_2 \implies \mathrm{eq}(F_1) \cap \mathrm{eq}(F_2) = \mathrm{eq}(P)$.

(b) $|\mathcal{F}| \leq m - |\,\mathrm{eq}(P)|$.

(c) *Es gibt eine Menge $I \subseteq M$ mit folgenden Eigenschaften*
   (c_1) $I \subseteq M \setminus \mathrm{eq}(P)$,
   (c_2) $|I| = |\mathcal{F}|$,
   (c_3) $F \in \mathcal{F} \iff \exists$ *genau ein* $i \in I$ *mit* $F = \mathrm{fa}(\{i\})$. $\triangle$

Jede Menge $I \subseteq M$ mit den Eigenschaften (c_1), (c_2), (c_3) wollen wir *Facetten-Indexmenge* nennen. Satz (1.34)(b) zeigt, dass man Facetten von P dadurch erhält, dass man in nur einer Ungleichung $A_i.x \leq b_i$ des Systems $Ax \leq b$ Gleichheit fordert. Jedoch ist es keineswegs so, dass für alle $i \in M$ die Menge $\mathrm{fa}(\{i\})$ eine Facette von P ist! Dies gilt nur für solche $i \in M$, die in einer Facettenindexmenge enthalten sind.

(1.36) Satz. *Seien $P = P(A, b) \neq \emptyset$ ein Polyeder und $\mathcal{F}$ die Menge der Facetten von P. Seien M die Zeilenindexmenge von A, $I \subseteq M \setminus \mathrm{eq}(P)$ und $J \subseteq \mathrm{eq}(P)$. Sei $P' := \{x \mid A_J.x = b_J, A_I.x \leq b_I\}$, dann gilt:*

(a) $P = P' \iff$ (a_1) $\forall F \in \mathcal{F}$ *gilt* $I \cap \mathrm{eq}(F) \neq \emptyset$ *und*
   (a_2) $\mathrm{rang}(A_J.) = \mathrm{rang}(A_{\mathrm{eq}(P)}.)$.

(b) $P = P(A_{I\cup \mathrm{eq}(P)}., b_{I\cup \mathrm{eq}(P)}) \iff \forall F \in \mathcal{F}$ *gilt* $I \cap \mathrm{eq}(F) \neq \emptyset$. △

Beweis. Mit $J = \mathrm{eq}(P)$ folgt (b) direkt aus (a). Wir beweisen (a).

„$\Longrightarrow$“ Nach Definition gilt offenbar $J = \mathrm{eq}_{P'}(P')$. Angenommen (a_2) ist nicht erfüllt, d. h. $\mathrm{rang}(A_J.) < \mathrm{rang}(A_{\mathrm{eq}(P)}.)$. Dann folgt aus der Dimensionsformel (1.28)(a) $\dim(P') > \dim(P)$ und somit muss $P \neq P'$ gelten. Widerspruch!

Angenommen (a_1) ist nicht erfüllt. Dann gibt es eine Facette F von P mit $\mathrm{eq}(F) \subseteq M \setminus I = (M \setminus I) \cup \mathrm{eq}(P)$. Folglich gilt $P \neq P'$ nach Satz (1.33). Widerspruch!

„$\Longleftarrow$“ Wir zeigen zunächst, dass unter der Voraussetzung (a_2) gilt:

$$A_J.x = b_J \implies A_{\mathrm{eq}(P)}.x = b_{\mathrm{eq}(P)}.$$

Da $P' \neq \emptyset$, gilt $\mathrm{rang}(A_J., b_J) = \mathrm{rang}(A_J.) = \mathrm{rang}(A_{\mathrm{eq}(P)}.) = \mathrm{rang}(A_{\mathrm{eq}(P)}., b_{\mathrm{eq}(P)})$. Das heißt, für alle $i \in \mathrm{eq}(P)$ existieren $K \subseteq J$ und λ_k, $k \in K$, mit $A_i. = \sum_{k\in K} \lambda_k A_k.$, $b_i = \sum_{k\in K} \lambda_k b_k$. Erfüllt also der Vektor x das System $A_J.x = b_J$, so gilt für alle $i \in \mathrm{eq}(P)$

$$A_i.x = \sum_{k\in K} \lambda_k A_k.x = \sum_{k\in K} \lambda_k b_k = b_i.$$

Nach (a_1) gilt für jede Facette F von P: $\mathrm{eq}(F) \not\subseteq M \setminus I$, und da Facetten maximale echte Seitenflächen sind und eq inklusionsumkehrend ist, folgt daraus $\mathrm{eq}(G) \not\subseteq M \setminus I$ für alle echten Seitenflächen G von P. Aus Satz (1.33) folgt daher $P = P'$. □

(1.37) Korollar. *Seien* $P = P(A, b) \neq \emptyset$ *ein Polyeder,* $I \subseteq M \setminus \mathrm{eq}(P)$, $J \subseteq \mathrm{eq}(P)$ *und* $P = \{x \mid A_J.x = b_J, A_I.x \leq b_I\}$. *Diese Darstellung von* P *ist genau dann irredundant, wenn gilt:*

(a) I *ist eine Facetten-Indexmenge von* P.

(b) $A_J.$ *ist eine* $(\mathrm{rang}(A_{\mathrm{eq}(P)}.), n)$*-Matrix mit vollem Zeilenrang.* △

(1.38) Korollar. *Sei* $P = P(A, b) \subseteq \mathbb{K}^n$ *ein volldimensionales Polyeder (also* $\mathrm{eq}(P) = \emptyset$*, bzw.* $\dim(P) = n$*), dann gilt für alle* $I \subseteq M$

$P(A_I., b_I)$ *ist eine irredundante Beschreibung von* P
$\iff$ I *ist Facetten-Indexmenge von* P. △

(1.39) Satz. *Sei* $P = P(A, b)$ *ein Polyeder, und* F *sei eine nichttriviale Seitenfläche von* P. *Dann sind äquivalent:*

(i) F *ist eine Facette von* P.

(ii) F *ist eine maximale echte Seitenfläche von* P.

(iii) $\dim(F) = \dim(P) - 1$.

(iv) *F enthält* $\dim(P)$ *affin unabhängige Vektoren.*

(v) *Sei $c^T x \leq c_0$ eine bezüglich P gültige Ungleichung mit $F = \{x \in P \mid c^T x = c_0\}$, dann gilt für alle gültigen Ungleichungen $d^T x \leq \delta$ mit $F \subseteq \{x \in P \mid d^T x = \delta\}$: Es gibt einen Vektor $u \in \mathbb{K}^{\mathrm{eq}(P)}$ und $\alpha \in \mathbb{K}$, $\alpha \geq 0$ mit*

$$d^T = \alpha c^T + u^T A_{\mathrm{eq}(P)\cdot},$$
$$\delta = \alpha c_0 + u^T b_{\mathrm{eq}(P)}. \qquad \triangle$$

Beweis. (i) $\iff$ (ii): nach Definition.

(iv) $\iff$ (iii): trivial.

(iii) $\implies$ (ii): Angenommen F ist keine Facette, dann existiert eine echte Seitenfläche G von P mit $F \subset G \subset P$. Aus (1.28)(c) folgt dann $\dim(F) \leq \dim(G) - 1 \leq \dim(P) - 2$, Widerspruch!

(i) $\implies$ (v): Sei F eine beliebige Facette von P. Wir nehmen zunächst an, dass $A_I.x \leq b_I$, $A_J.x = b_J$ eine irredundante Darstellung von $P(A, b)$ ist mit $1 \in I$ und dass $F = \{x \in P \mid A_1.x = b_1\}$ gilt. Sei nun $d^T x \leq \delta$ eine gültige Ungleichung mit $F \subseteq \{x \in P \mid d^T x = \delta\}$. Aufgrund von Folgerung (1.20) gibt es Vektoren $v \geq 0$ und w mit $v^T A_{I\cdot} + w^T A_{J\cdot} = d^T$ und $v^T b_I + w^T b_J \leq \delta$ (in der Tat gilt hier Gleichheit, da $\{x \mid d^T x = \delta\}$ eine Stützhyperebene ist). Angenommen, es gibt einen Index $i \in I \setminus \{1\}$ mit $v_i > 0$, dann gilt nach (1.21) $i \in \mathrm{eq}(F)$. Dies ist aber ein Widerspruch dazu, dass I eine Facettenindexmenge ist. Hieraus folgt (v).

(v) $\implies$ (iii): Da F eine echte Seitenfläche von P ist, gilt $\dim(F) \leq \dim(P) - 1$. Angenommen $\dim(F) \leq \dim(P) - 2$. O. B. d. A. können wir annehmen, dass $F = \{x \in P \mid A_1.x = b_1\}$ gilt. Aus (1.27) folgt

$$\mathrm{rang}(A_{\mathrm{eq}(F)\cdot}) \geq \mathrm{rang}(A_{\mathrm{eq}(P)\cdot}) + 2.$$

Mithin gibt es einen Index $i \in \mathrm{eq}(F) \setminus (\mathrm{eq}(P) \cup \{1\})$, so dass der Zeilenvektor $A_i.$ linear unabhängig von den Zeilenvektoren $A_j.$, $j \in \mathrm{eq}(P) \cup \{1\}$, ist. Das aber heißt, dass das System

$$A_i. = \alpha A_1. + u^T A_{\mathrm{eq}(P)\cdot}$$

keine Lösung α, u hat. Wegen $F \subseteq \{x \in P \mid A_i.x = b_i\}$ ist dies ein Widerspruch zu (v). $\square$

(1.40) Korollar. *Seien $P = P(A, b) \subseteq \mathbb{K}^n$ ein volldimensionales Polyeder und $F = \{x \in P \mid c^T x = c_0\}$ eine Seitenfläche von P. Dann sind äquivalent:*

(i) *F ist Facette von P.*

(ii) $\dim(F) = n - 1$.

(iii) *Für alle gültigen Ungleichungen* $d^T x \leq \delta$, $d \neq 0$, *mit* $F \subseteq \{x \in P \mid d^T x = \delta\}$ *gilt: Es existiert ein* $\alpha > 0$ *mit*

$$d^T = \alpha c^T,$$
$$\delta = \alpha c_0.$$

△

(1.41) Beispiel. Wir betrachten das Polyeder $P = P(A, b) \subseteq \mathbb{R}^2$, das wie folgt gegeben ist (siehe Abbildung 1.2).

$$A = \begin{pmatrix} A_{1.} \\ A_{2.} \\ A_{3.} \\ A_{4.} \\ A_{5.} \\ A_{6.} \end{pmatrix} = \begin{pmatrix} 1 & -1 \\ -2 & 2 \\ 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ -1 & -1 \end{pmatrix}, \qquad b = \begin{pmatrix} 0 \\ 0 \\ 2 \\ 2 \\ -1 \\ -2 \end{pmatrix}.$$

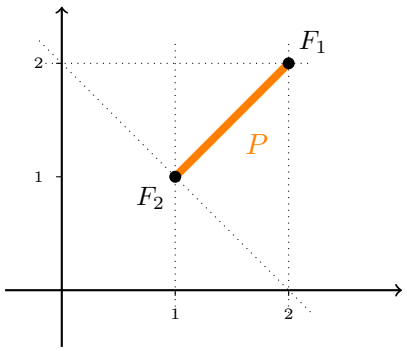


Abbildung 1.2: Ein 1-dimensionales Polyeder in $\mathbb{R}^2$

P hat 4 Seitenflächen, nämlich $\emptyset$, P und $F_1 = \{\binom{2}{2}\}$, $F_2 = \{\binom{1}{1}\}$. F_1 und F_2 sind Facetten von P. Es gilt $\text{eq}(P) = \{1, 2\}$, $\text{eq}(F_1) = \{1, 2, 3, 4\}$, $\text{eq}(F_2) = \{1, 2, 5, 6\}$, $\text{eq}(\emptyset) = \{1, \ldots, 6\}$. Die Mengen $\{3, 5\}$, $\{3, 6\}$, $\{4, 5\}$, $\{4, 6\}$ sind die Facettenindexmengen von P. Eine irredundante Beschreibung von P ist z. B. gegeben durch

$$P = \{x \mid A_{1.}x = 0, A_{3.}x \leq b_3, A_{5.}x \leq b_5\}.$$

Übrigens sind die Ungleichungen $A_{i.}x \leq b_i$, $i = 3, 4, 5, 6$ redundant bezüglich $P(A, b)$. Die Ungleichungssysteme $A_{I.}x \leq b_I$ mit $I = \{3, 5\}$ oder $I = \{4, 6\}$ sind z. B. ebenfalls redundant. Aber $A_{I.}x \leq b_I$ ist nicht redundant bezüglich $P(A, b)$, falls $I = \{3, 4, 5\}$. △

1.6 Rezessionskegel, Linienraum und Homogenisierung

An dieser Stelle ist es nützlich einige weitere Objekte einzuführen, die man Polyedern (bzw. allgemeinen Mengen) zuordnen kann. Das Studium dieser Objekte ist für sich

selbst betrachtet sehr interessant. Wir wollen diese Mengen jedoch nur als Hilfsmittel zur Vereinfachung von Beweisen verwenden, weswegen wir nicht weiter auf theoretische Untersuchungen dieser Mengen eingehen werden.

(1.42) Definition. *Sei $S \subseteq \mathbb{K}^n$ eine beliebige Menge. Wir definieren*

(a) $\operatorname{rec}(S) := \{y \in \mathbb{K}^n \mid \exists x \in S,$ *so dass* $\forall \lambda \geq 0$ *gilt* $x + \lambda y \in S\}$,

(b) $\operatorname{lineal}(S) := \{y \in \mathbb{K}^n \mid \exists x \in S,$ *so dass* $\forall \lambda \in \mathbb{K}$ *gilt* $x + \lambda y \in S\}$,

(c) $\operatorname{hog}(S) := \{\binom{x}{1} \in \mathbb{K}^{n+1} \mid x \in S\}^{\circ\circ}$.

Die Menge $\operatorname{rec}(S)$ *heißt* Rezessionskegel *von* S, $\operatorname{lineal}(S)$ *heißt* Linealitätsraum *oder* Linienraum von S, *und* $\operatorname{hog}(S)$ *heißt* Homogenisierung von S. △

Wir wollen nun die oben eingeführten Mengen bezüglich Polyedern charakterisieren. Nennen wir einen Vektor y mit $x + \lambda y \in S$ für alle $\lambda \geq 0$ eine „Richtung nach Unendlich", so besteht der Rezessionskegel einer Menge S aus allen Richtungen nach Unendlich. Für Polyeder gilt Folgendes:

(1.43) Satz. *Sei $P = P(A, b) = \operatorname{conv}(V) + \operatorname{cone}(E)$ ein nichtleeres Polyeder, dann gilt*

$$\operatorname{rec}(P) = P(A, 0) = \operatorname{cone}(E). \qquad \triangle$$

Beweis. (a) $\operatorname{rec}(P) = P(A, 0)$.

Ist $y \in \operatorname{rec}(P)$, so existiert ein $x \in P$ mit $x + \lambda y \in P$ $\forall \lambda \geq 0$. Daraus folgt $b \geq A(x + \lambda y) = Ax + \lambda Ay$. Gäbe es eine Komponente von Ay, die größer als Null ist, sagen wir $(Ay)_i > 0$, so wäre der Vektor $x + \lambda_0 y$ mit

$$\lambda_0 = \frac{b_i - (Ax)_i}{(Ay)_i} + 1$$

nicht in $P(A, b)$, Widerspruch!

Ist $y \in P(A, 0)$, so gilt für alle $x \in P(A, b)$ und $\lambda \geq 0$, $A(x+\lambda y) = Ax+\lambda Ay \leq b+0 = b$, also ist $y \in \operatorname{rec}(P)$.

(b) $\operatorname{rec}(P) = \operatorname{cone}(E)$.

Die Inklusion $\operatorname{cone}(E) \subseteq \{y \in \mathbb{K}^n \mid \forall x \in S, \forall \lambda \geq 0 : x + \lambda y \in S\} \subseteq \operatorname{rec}(P)$ ist offensichtlich. Umgekehrt sei $y \in \operatorname{rec}(P)$, dann existiert wieder ein $x \in P$ wie oben. Angenommen $y \notin \operatorname{cone}(E)$, dann gibt es nach dem Farkas-Lemma (ADM I Skript (11.2)(c)) ein u mit $u^T E \leq 0$ und $u^T y > 0$. Für jedes $z \in P$ folgt dann mit gewissen λ_i, $0 \leq \lambda_i \leq 1$, $\sum_i \lambda_i = 1$ und $\mu_i \geq 0$:

$$\begin{aligned} u^T z &= u^T \sum_i \lambda_i V_{\cdot i} + u^T \sum_i \mu_i E_{\cdot i} = \sum_i \lambda_i u^T V_{\cdot i} + u^T E \begin{pmatrix} \mu_1 \\ \vdots \\ \mu_{|E|} \end{pmatrix} \leq \sum_i \lambda_i u^T V_{\cdot i} \\ &\leq \max_i u^T V_{\cdot i} \end{aligned}$$

Andererseits gilt aber $u^T(x + \lambda y) = u^T x + \lambda u^T y \to \infty$, für $\lambda \to \infty$, ein Widerspruch zu $x + \lambda y \in P$ für alle $\lambda \geq 0$. □

Insbesondere folgt aus dem Beweis auch, dass $\operatorname{rec}(P) = \{y \in \mathbb{K}^n \mid \forall x \in P \text{ und } \forall \lambda \geq 0 \text{ gilt } x + \lambda y \in P\}$ für Polyeder P gilt. Ist P ein Kegel, so gilt natürlich $P = \operatorname{rec}(P)$, und offenbar ist ein Polyeder P genau dann ein Polytop, wenn $\operatorname{rec}(P) = \{0\}$. Abbildung 1.3 zeigt ein Polyeder und seinen Rezessionskegel.

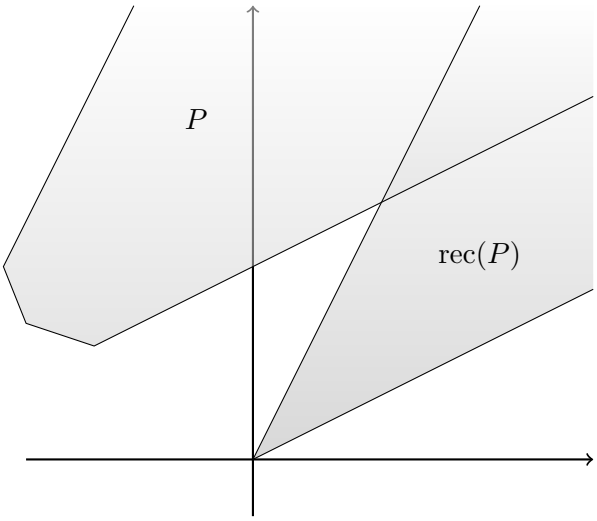


Abbildung 1.3: Ein Polyeder und sein Rezessionskegel

Aus Definition (1.42) folgt $\operatorname{lineal}(P) = \operatorname{rec}(P) \cap (-\operatorname{rec}(P))$. Offenbar ist $\operatorname{lineal}(P)$ ein linearer Teilraum des $\mathbb{K}^n$, und zwar ist es der größte lineare Teilraum $L \subseteq \mathbb{K}^n$, so dass $x + L \subseteq P$ für alle $x \in P$ gilt. Analytisch können wir $\operatorname{lineal}(P)$ wie folgt darstellen.

(1.44) Satz. *Sei $P = P(A, b) = \operatorname{conv}(V) + \operatorname{cone}(E)$ ein nichtleeres Polyeder, dann gilt*

$$\operatorname{lineal}(P) = \{x \mid Ax = 0\} = \operatorname{cone}(\{e \in E \mid -e \in \operatorname{cone}(E)\}). \qquad \triangle$$

Beweis. Wegen $\operatorname{lineal}(P) = \operatorname{rec}(P) \cap (-\operatorname{rec}(P))$ folgt die Behauptung direkt aus (1.43).□

Wir kommen nun zur Homogenisierung. Die Definition der Homogenisierung erscheint etwas kompliziert: Man wende zweimal die Kegelpolarität auf die Menge S an! Geometrisch betrachtet ist $\operatorname{hog}(S)$ der Durchschnitt aller Ungleichungen mit rechter Seite 0, die gültig bezüglich $\{\binom{x}{1} \mid x \in S\}$ sind.

(1.45) Satz. *Sei $P = P(A, b) = \operatorname{conv}(V) + \operatorname{cone}(E)$ ein nichtleeres Polyeder. Sei*

$$B = \begin{pmatrix} A & -b \\ 0 & -1 \end{pmatrix},$$

dann gilt

$$\operatorname{hog}(P) = P(B, 0) = \operatorname{cone}(\{\tbinom{v}{1} \mid v \in V\}) + \operatorname{cone}(\{\tbinom{e}{0} \mid e \in E\}). \qquad \triangle$$

Beweis. Setzen wir $P_1 := \{\binom{x}{1} \in \mathbb{K}^{n+1} \mid x \in P\}$, so gilt offensichtlich

$$P_1 = \operatorname{conv}(\{\tbinom{v}{1} \mid v \in V\}) + \operatorname{cone}(\{\tbinom{e}{0} \mid e \in E\}).$$

Aus Folgerung (1.20)(iii) ergibt sich dann:

$$\begin{aligned} P_1^\circ &= \{z \in \mathbb{K}^{n+1} \mid z^T u \le 0 \ \forall u \in P_1\} \\ &= \{z \in \mathbb{K}^{n+1} \mid z^T \tbinom{v}{1} \le 0 \ \forall v \in V, z^T \tbinom{e}{0} \le 0 \ \forall e \in E\} \\ &= \left\{z \mid \begin{pmatrix} V^T & \mathbb{1} \\ E^T & 0 \end{pmatrix} z \le 0\right\} = P\left(\begin{pmatrix} V^T & \mathbb{1} \\ E^T & 0 \end{pmatrix}, 0\right). \end{aligned}$$

Mit Folgerung (1.7) $P(A,0)^\circ = \operatorname{cone}(A^T)$ erhalten wir nun

$$\operatorname{hog}(P) = P_1^{\circ\circ} = P\left(\begin{pmatrix} V^T & \mathbb{1} \\ E^T & 0 \end{pmatrix}, 0\right)^\circ = \operatorname{cone}\begin{pmatrix} V & E \\ \mathbb{1}^T & 0^T \end{pmatrix}.$$

Die zweite Charakterisierung von $\operatorname{hog}(P)$ folgt aus einer anderen Darstellung von P_1°. Es gilt nämlich mit Satz (1.18):

$$\begin{aligned} P_1^\circ &= \{\tbinom{y}{\lambda} \in \mathbb{K}^{n+1} \mid y^T x + \lambda 1 \le 0 \ \forall x \in P\} = \{\tbinom{y}{\lambda} \in \mathbb{K}^{n+1} \mid y^T x \le -\lambda \ \forall x \in P\} \\ &= \{\tbinom{y}{\lambda} \in \mathbb{K}^{n+1} \mid \tbinom{y}{-\lambda} \in P^\gamma\} = \{\tbinom{y}{\lambda} \in \mathbb{K}^{n+1} \mid \tbinom{y}{-\lambda} \in \operatorname{cone}\begin{pmatrix} A^T & 0 \\ b^T & 1 \end{pmatrix}\} \\ &= \operatorname{cone}\begin{pmatrix} A^T & 0 \\ -b^T & -1 \end{pmatrix}. \end{aligned}$$

Folgerung (1.10) impliziert nun

$$\operatorname{hog}(P) = P_1^{\circ\circ} = \left(\operatorname{cone}\begin{pmatrix} A^T & 0 \\ -b^T & -1 \end{pmatrix}\right)^\circ = P\left(\begin{pmatrix} A & -b \\ 0 & -1 \end{pmatrix}, 0\right). \qquad \square$$

In Abbildung 1.4 sind ein Polyeder $P \subseteq \mathbb{R}^1$, die im obigen Beweis definierte Menge P_1 und $\operatorname{hog}(P)$ dargestellt.

(1.46) Bemerkung. Sei $P \subseteq \mathbb{K}^n$ ein Polyeder, dann gilt:

(a) $x \in P \iff \binom{x}{1} \in \operatorname{hog}(P)$.

(b) $x \in \operatorname{rec}(P) \iff \binom{x}{0} \in \operatorname{hog}(P)$. $\triangle$

Beweis. (a) ist trivial.

(b) Sei $P = P(A,b)$ eine Darstellung von P, dann gilt $\operatorname{hog}(P) = P(B,0)$ mit $B = \begin{pmatrix} A & -b \\ 0 & -1 \end{pmatrix}$. Folglich gilt nach (1.45) und (1.43)

$$\begin{pmatrix} x \\ 0 \end{pmatrix} \in \operatorname{hog}(P) \iff \begin{pmatrix} x \\ 0 \end{pmatrix} \in P(B,0) \iff Ax \le 0 \iff x \in \operatorname{rec}(P). \qquad \square$$

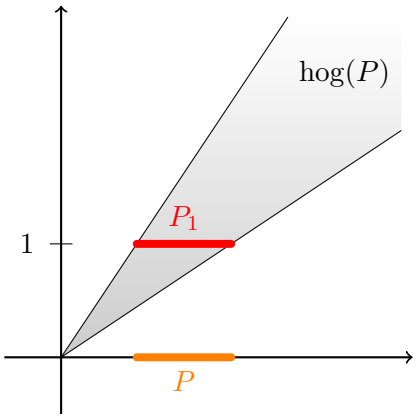


Abbildung 1.4: 1-dimensionaler Polyeder und seine Homogenisierung

1.7 Extremalen von spitzen Polyedern

Wir wollen nachfolgend einige Aussagen über spitze Polyeder beweisen, die sich – entsprechend modifiziert – auch für allgemeine Polyeder zeigen lassen. Dabei treten jedoch einige unschöne technische Komplikationen auf, so dass wir hier auf die Behandlung dieser Verallgemeinerung verzichten.

Wir erinnern daran, dass ein Polyeder spitz genannt wird, wenn es eine Ecke (nulldimensionale Seitenfläche) besitzt. Die folgende Aussage erweitert Satz (8.11) aus dem ADM I Skript.

(1.47) Satz. *Sei $P = P(A, b) \subseteq \mathbb{K}^n$ ein nichtleeres Polyeder, dann sind äquivalent:*

(1) *P ist spitz.*

(2) $\operatorname{rang}(A) = n$.

(3) $\operatorname{rec}(P)$ *ist spitz, d. h.* 0 *ist eine Ecke von* $\operatorname{rec}(P)$.

(4) *Jede nichtleere Seitenfläche von P ist spitz.*

(5) $\operatorname{hog}(P)$ *ist spitz.*

(6) *P enthält keine Gerade.*

(7) $\operatorname{rec}(P)$ *enthält keine Gerade.*

(8) $\operatorname{lineal}(P) = \{0\}$. △

Beweis. Die Äquivalenz von (1), (2) und (4) wurde schon in ADM I in Satz (8.11) gezeigt.

Aus der Äquivalenz von (1) und (2) folgt direkt die Äquivalenz der Aussagen (2), (3) und (4), da

$$\begin{aligned} \operatorname{rec}(P) &= P(A, 0), \text{ nach (1.43)}, \\ \operatorname{hog}(P) &= P\left(\begin{pmatrix} A & -b \\ 0 & -1 \end{pmatrix}, 0\right), \text{ nach (1.45)}. \end{aligned}$$

(3) $\Longrightarrow$ (6). Angenommen P enthält eine Gerade $G = \{u + \lambda v \mid \lambda \in \mathbb{K}\}$, $v \neq 0$, dann gilt $b \geq A(u + \lambda v) = Au + \lambda Av$ für alle $\lambda \in \mathbb{K}$. Daraus folgt $A(\lambda v) \leq 0$ für alle $\lambda \in \mathbb{K}$ und somit $v, -v \in \operatorname{rec}(P)$, d. h. $0 = \frac{1}{2}v + \frac{1}{2}(-v)$ ist eine Konvexkombination. Also ist 0 keine Ecke von $\operatorname{rec}(P)$.

(6) $\Longrightarrow$ (3). Ist $\operatorname{rec}(P)$ nicht spitz, so ist 0 echte Konvexkombination von Vektoren aus $\operatorname{rec}(P)$, sagen wir $0 = \lambda u + (1 - \lambda)v$, $u \neq 0 \neq v$, $0 < \lambda < 1$. Dann aber ist neben u auch $-\lambda u = (1 - \lambda)v \in \operatorname{rec}(P)$ und folglich ist $G = \{\mu u \mid \mu \in \mathbb{K}\}$ eine Gerade in $\operatorname{rec}(P)$, und für alle $x \in P$ ist $x + G$ eine Gerade in P.

Die Äquivalenz von (7) und (8) zu den übrigen Aussagen ist nun offensichtlich. □

Der folgende Hilfssatz über innere Punkte wird im Weiteren benötigt.

(1.48) Lemma. *Ist F eine nichtleere Seitenfläche von $P = P(A, b)$, gilt $I = \operatorname{eq}(F)$, und ist $B = \{y^1, \ldots, y^k\}$ eine Basis des Kerns von $A_{I.}$, dann gibt es zu jedem inneren Punkt $x \in F$ von F ein $\varepsilon > 0$, so dass $x \pm \varepsilon y^j \in P$ für alle $j = 1, \ldots, k$ gilt.* △

Beweis. Übungsaufgabe. □

(1.49) Definition. *Sei P ein Polyeder. Ein Vektor $z \in \operatorname{rec}(P) \setminus \{0\}$ heißt* Extremale *(oder* Extremalvektor*) von P, wenn $\operatorname{cone}(\{z\})$ ein Extremalstrahl von $\operatorname{rec}(P)$ ist.* △

Nur spitze Polyeder haben Extremalen. Denn ist P nicht spitz, so ist nach (1.47) $\operatorname{rec}(P)$ nicht spitz, also ist der Nullvektor eine echte Konvexkombination zweier von Null verschiedener Vektoren, sagen wir $0 = \lambda u + (1 - \lambda)v$, $0 < \lambda < 1$. Ist $F = \operatorname{cone}(\{z\})$ ein Extremalstrahl von $\operatorname{rec}(P)$, so gibt es eine bezüglich $\operatorname{rec}(P)$ gültige Ungleichung $c^T x \leq 0$ mit $F = \{x \in \operatorname{rec}(P) \mid c^T x = 0\}$. Nun gilt $0 = c^T 0 = c^T(\lambda u + (1 - \lambda)v) = \lambda c^T u + (1 - \lambda)c^T v \leq 0$. Aus $c^T u \leq 0$, $c^T v \leq 0$ folgt $c^T u = c^T v = 0$ und somit $u, v \in F$, ein Widerspruch. Aussagen über Extremalen machen also nur für spitze Polyeder Sinn.

Ist K speziell ein spitzer polyedrischer Kegel, so ist (wegen $\operatorname{rec}(K) = K$) eine Extremale von K ein Vektor $z \in K$, so dass $\operatorname{cone}(\{z\})$ ein Extremalstrahl von K ist. Das heißt, jeder auf einem Extremalstrahl von K gelegener und von Null verschiedener Vektor ist eine Extremale von K.

(1.50) Satz. *Seien $P = P(A, b) \subseteq \mathbb{K}^n$ ein spitzes Polyeder und $z \in \operatorname{rec}(P) \setminus \{0\}$. Dann sind äquivalent:*

(1) *z ist eine Extremale von P.*

(2) *$\operatorname{cone}(\{z\})$ ist ein Extremalstrahl von $\operatorname{rec}(P)$.*

(3) *z läßt sich nicht als echte konische Kombination zweier linear unabhängiger Elemente von* $\mathrm{rec}(P)$ *darstellen.*

(4) $(\mathrm{rec}(P) \setminus \mathrm{cone}\{z\}) \cup \{0\}$ *ist ein Kegel.*

(5) $\mathrm{rang}(A_{\mathrm{eq}(\{z\}).}) = n - 1$ *(wobei sich* eq *auf das System* $Ax \leq 0$ *bezieht).* △

Beweis. (1) $\iff$ (2). Definition.

(2) $\implies$ (3). Ist $F := \mathrm{cone}(\{z\})$ ein Extremalstrahl von $\mathrm{rec}(P)$, so ist F eine eindimensionale Seitenfläche von $\mathrm{rec}(P)$, d. h. F kann keine zwei linear unabhängigen Vektoren enthalten. Insbesondere gibt es eine bezüglich $\mathrm{rec}(P)$ gültige Ungleichung $c^T x \leq 0$ mit $F = \{x \in \mathrm{rec}(P) \mid c^T x = 0\}$. Gibt es zwei linear unabhängige Vektoren $u, v \in \mathrm{rec}(P)$ und $\lambda, \mu > 0$ mit $z = \lambda u + \mu v$, so gilt $0 = c^T z = c^T(\lambda u + \mu v) = \lambda c^T u + \mu c^T v \leq 0$. Daraus folgt $c^T u = c^T v = 0$, d. h. $u, v \in F$, ein Widerspruch.

(3) $\iff$ (4). Trivial.

(3) $\implies$ (5). Sei $I = \mathrm{eq}(\{z\})$, dann ist z innerer Punkt von $F := \{x \in \mathrm{rec}(P) \mid A_I.x = 0\}$. Ist $\mathrm{rang}(A_I.) < n - 1$, dann enthält der Kern von $A_I.$ einen von z linear unabhängigen Vektor u. Nach Lemma (1.48) gibt es ein $\varepsilon > 0$, so dass $z \pm \varepsilon u \in \mathrm{rec}(P)$ gilt. Dann aber gilt $z = \frac{1}{2}(z+\varepsilon u)+\frac{1}{2}(z-\varepsilon u)$, d. h. z ist echte konische Kombination von zwei linear unabhängigen Elementen von $\mathrm{rec}(P)$, Widerspruch. Offenbar ist $\mathrm{rang}(A_I.) \neq n$.

(5) $\implies$ (2). Sei $I = \mathrm{eq}(\{z\})$, dann folgt für $F = \{x \in \mathrm{rec}(P) \mid A_I.x = 0\}$ aus der Voraussetzung, dass $\dim(F) = 1$ gilt. Da $\mathrm{rec}(P)$ spitz ist, enthält nach (1.47) $\mathrm{rec}(P)$ keine Gerade, also muss die eindimensionale Seitenfläche F der Strahl $\mathrm{cone}(\{z\})$ sein. □

Wir wollen nun noch eine Beziehung zwischen den Ecken und Extremalen eines spitzen Polyeders und den Extremalen seiner Homogenisierung aufzeigen.

(1.51) Satz. *Sei* $P \subseteq \mathbb{K}^n$ *ein spitzes Polyeder, dann gilt:*

(a) *x ist Ecke von* $P \iff \binom{x}{1}$ *ist Extremale von* $\mathrm{hog}(P)$.

(b) *z ist Extremale von* $P \iff \binom{z}{0}$ *ist Extremale von* $\mathrm{hog}(P)$. △

Beweis. Sei $P = P(A, b)$, dann gilt nach Satz (1.45) $\mathrm{hog}(P) = P(B, 0)$ mit

$$B = \begin{pmatrix} A & -b \\ 0 & -1 \end{pmatrix}.$$

(a) Sei $I = \mathrm{eq}(\{x\})$ bezüglich $P(A, b)$. x ist Ecke von $P \iff \mathrm{rang}(A_I.) = n$ (nach Satz (8.8) aus dem ADM I Skript) $\iff \mathrm{rang}(B_{\mathrm{eq}(\{\binom{x}{1}\}).}) = n$ (denn die neu hinzugekommene Ungleichung ist nicht mit Gleichheit erfüllt) $\overset{(1.50)}{\iff} \binom{x}{1}$ ist Extremale von $\mathrm{hog}(P)$.

(b) z ist Extremale von $P \iff z$ ist Extremale von $\mathrm{rec}(P) \overset{(1.50)}{\iff} \mathrm{rang}(A_{\mathrm{eq}(\{z\}).}) = n-1 \iff \mathrm{rang}(B_{\mathrm{eq}(\{\binom{z}{0}\}).}) = n \overset{(1.50)}{\iff} \binom{z}{0}$ ist Extremale von $\mathrm{hog}(P)$. □

1.8 Weitere Darstellungssätze

Wir knüpfen hier an Abschnitt 1.3 an, wo wir bereits verschiedene Darstellungssätze bewiesen haben. Einige dieser Sätze können wir mit Hilfe der nun gewonnenen Erkenntnisse verschärfen.

Ist K ein polyedrischer Kegel und gilt $K = \mathrm{cone}(E)$, dann nennen wir E eine *Kegelbasis von K*, wenn es keine echte Teilmenge E' von E gibt mit $K = \mathrm{cone}(E')$ und wenn jede andere minimale Menge F mit $K = \mathrm{cone}(F)$ dieselbe Kardinalität wie E hat. Ist P ein Polytop, dann heißt eine Menge V mit $P = \mathrm{conv}(V)$ *konvexe Basis von P*, wenn V keine echte Teilmenge besitzt, deren konvexe Hülle P ist, und wenn jede andere minimale Menge W mit $P = \mathrm{conv}(W)$ dieselbe Kardinalität wie V hat.

Trivialerweise sind die Elemente einer Kegelbasis E *konisch unabhängig*, d. h. kein $e \in E$ ist konische Kombination der übrigen Elemente von E; und die Elemente einer konvexen Basis sind *konvex unabhängig*, d. h. kein Element von V ist Konvexkombination der übrigen Elemente von V. Es gilt aber keineswegs, dass jeder Vektor $x \in \mathrm{cone}(E)$ bzw. $x \in \mathrm{conv}(V)$ eine eindeutige konische bzw. konvexe Darstellung durch Vektoren aus E bzw. V hat. In dieser Hinsicht unterscheiden sich also Kegelbasen und konvexe Basen von Vektorraumbasen.

(1.52) Satz. *Sei $\{0\} \neq K \subseteq \mathbb{K}^n$ ein spitzer polyedrischer Kegel, dann sind äquivalent:*

(1) *E ist eine Kegelbasis von K.*

(2) *E ist eine Menge, die man dadurch erhält, dass man aus jedem Extremalstrahl von K genau einen von Null verschiedenen Vektor (also eine Extremale von K) auswählt.* △

Beweis. Ist z Extremale von K, so ist $K' := (K \setminus \mathrm{cone}(\{z\})) \cup \{0\}$ nach (1.50) ein Kegel. Folglich gilt $\mathrm{cone}(E) \subseteq K'$ für alle Teilmengen E von K'. Also muss jede Kegelbasis von K mindestens ein (aus der Basiseigenschaft folgt sofort „genau ein“) Element von $\mathrm{cone}(\{z\}) \setminus \{0\}$ enthalten.

Zum Beweis, dass jede wie in (2) spezifizierte Menge eine Kegelbasis ist, benutzen wir Induktion über $d = \dim K$. Für Kegel der Dimension 1 ist die Behauptung trivial. Sei die Behauptung für Kegel der Dimension d richtig und K ein Kegel mit $\dim K = d+1$. Sei $y \in K \setminus \{0\}$ beliebig und $c \in \mathbb{K}^n \setminus \{0\}$ ein Vektor, so dass die Ecke 0 von K die eindeutig bestimmte Lösung von $\max\{c^T x \mid x \in K\}$ ist (c existiert nach Satz (8.8) aus dem ADM I Skript). Sei $z \in \{x \mid c^T x = 0\} \setminus \{0\}$. Dann ist für die Gerade

$$G = \{y + \lambda z \mid \lambda \in \mathbb{K}\}$$

die Menge $K \cap G$ ein endliches Streckenstück (andernfalls wäre $z \in \mathrm{rec}(K) = K$, und wegen $c^T z = 0$ wäre 0 nicht der eindeutige Maximalpunkt). Folglich gibt es zwei Punkte

z_1 und z_2, die auf echten Seitenflächen, sagen wir F_1 und F_2, von K liegen, so dass $K \cap G = \text{conv}(\{z_0, z_1\})$. Die Dimensionen der Seitenflächen F_1, F_2 sind höchstens d, F_1 und F_2 sind Kegel, und die Extremalstrahlen von F_1 und F_2 sind Extremalstrahlen von K. Nach Induktionsvoraussetzung werden z_1 und z_2 durch die in (2) festgelegten Mengen bezüglich F_1 und F_2 konisch erzeugt. Daraus folgt, dass y durch jede Menge des Typs (2) konisch erzeugt werden kann. Dies impliziert die Behauptung. □

(1.53) Korollar. *Jeder spitze polyedrische Kegel besitzt eine – bis auf positive Skalierung der einzelnen Elemente – eindeutige Kegelbasis.* △

(1.54) Korollar. *Jeder spitze polyedrische Kegel $K \neq \{0\}$ ist die Summe seiner Extremalstrahlen, d. h. sind* $\text{cone}(\{e_i\})$, $i = 1, \ldots, k$ *die Extremalstrahlen von K, so gilt*

$$K = \text{cone}(\{e_1, \ldots, e_k\}) = \sum_{i=1}^{k} \text{cone}(\{e_i\}).$$

△

Der folgende Satz verschärft (1.13) für spitze Polyeder.

(1.55) Satz. *Jedes spitze Polyeder P lässt sich darstellen als die Summe der konvexen Hülle seiner Ecken und der konischen Hülle seiner Extremalen, d. h. sind V die Eckenmenge von P und* $\text{cone}(\{e\})$, $e \in E$, *die Extremalstrahlen von* $\text{rec}(P)$ *(bzw. ist E eine Kegelbasis von* $\text{rec}(P)$*), so gilt*

$$P = \text{conv}(V) + \text{cone}(E).$$

△

Beweis. Sei $\text{hog}(P)$ die Homogenisierung von P. Da P spitz ist, ist $\text{hog}(P)$ nach (1.47) ein spitzer Kegel. Nach (1.54) ist $\text{hog}(P)$ die Summe seiner Extremalstrahlen $\text{cone}(\{e_i'\})$, $i = 1, \ldots, k$. O. B. d. A. können wir annehmen, dass $e_i' = \binom{v_i}{1}$, $i = 1, \ldots, p$, und $e_i' = \binom{e_i}{0}$, $i = p+1, \ldots, k$ gilt. Aus (1.51) folgt: $V = \{v_1, \ldots, v_p\}$ ist die Eckenmengen von P und $E = \{e_{p+1}, \ldots, e_k\}$ ist die Extremalenmenge von P. Nach (1.46) gilt $x \in P \iff \binom{x}{1} \in \text{hog}(P)$ und somit folgt $x \in P \iff x = \sum_{i=1}^{p} \lambda_i v_i + \sum_{i=p+1}^{k} \mu_i e_i$, $\lambda_i, \mu_i \geq 0$, $\sum_{i=1}^{p} \lambda_i = 1$, d. h. $x \in \text{conv}(V) + \text{cone}(E)$. □

(1.56) Korollar (Satz von Krein-Milman). *Sei P ein Polyeder und V die Menge seiner Ecken, dann gilt*

$$P \text{ ist ein Polytop} \iff P = \text{conv}(V).$$

△

(1.57) Korollar. *Polytope haben eine eindeutige konvexe Basis.* △

Für die lineare Optimierung ist die folgende Beobachtung wichtig.

(1.58) Satz. *Seien $P \subseteq \mathbb{K}^n$ ein spitzes Polyeder und $c \in \mathbb{K}^n$. Das lineare Programm $\max c^T x$, $x \in P$ ist genau dann unbeschränkt, wenn es eine Extremale e von P gibt mit $c^T e > 0$.* △

Beweis. Seien V die Eckenmenge von P und E eine Kegelbasis von $\mathrm{rec}(P)$, dann gilt nach (1.55) $P = \mathrm{conv}(V)+\mathrm{cone}(E)$. Es ist $\gamma := \max\{c^T v \mid v \in V\} < \infty$, da V endlich und $\gamma = \max\{c^T x \mid x \in \mathrm{conv}(V)\}$. Gilt $c^T e \leq 0 \ \forall e \in E$, so ist $\gamma = \max\{c^T x \mid x \in P\} < \infty$. Falls also $\max\{c^T x \mid x \in P\}$ unbeschränkt ist, muss für mindestens ein $e \in E$ gelten $c^T e > 0$. Die umgekehrte Richtung ist trivial. □

Wie bereits bemerkt, haben Elemente von spitzen Polyedern P i. A. keine eindeutige Darstellung als konvexe und konische Kombination von Ecken und Extremalen. Man kann jedoch zeigen, dass zu einer derartigen Darstellung von Elementen von P nicht allzu viele Ecken und Extremalen benötigt werden.

(1.59) Satz. *Es seien $K \subseteq \mathbb{K}^n$ ein spitzer Kegel und $0 \neq x \in K$. Dann gibt es Extremalen $y_1, \ldots, y_d$ von K, wobei $d \leq \dim(K) \leq n$ gilt, mit*

$$x = \sum_{i=1}^{d} y_i.$$

△

Beweis. Es seien $\mathrm{cone}(\{e_i\})$ die Extremalstrahlen von K, $i = 1, \ldots, k$, dann gibt es nach (1.54) Skalare $\lambda_i \geq 0$, $i = 1, \ldots, k$, mit

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

Unter allen möglichen Darstellungen von x dieser Art sei die obige eine, so dass $I = \{i \in \{1, \ldots, k\} \mid \lambda_i > 0\}$ minimal ist. Sagen wir, es gilt $I = \{1, \ldots, d\}$. Angenommen die Vektoren e_i, $i \in I$ sind linear abhängig, dann gibt es $\mu_1, \ldots, \mu_d \in \mathbb{K}$, nicht alle μ_i Null, so dass gilt

$$\sum_{i=1}^{d} \mu_i e_i = 0.$$

Angenommen $\mu_i \geq 0$ für $i = 1, \ldots, d$, und o. B. d. A. sei $\mu_1 > 0$. Dann ist $-e_1 = \sum_{i=2}^{d} \frac{\mu_i}{\mu_1} e_i$ eine konische Kombination, und nach Satz (1.44) gilt $e_1 \in \mathrm{lineal}(K)$. Dies widerspricht nach (1.47) der Voraussetzung K ist spitz.

O. B. d. A. können wir daher annehmen, dass gilt $\mu_1 < 0$ und

$$\frac{\lambda_1}{\mu_1} \ = \ \max\left\{\frac{\lambda_i}{\mu_i} \mid \mu_i < 0\right\}.$$

Daraus folgt

$$e_1 = -\sum_{i=2}^{d} \frac{\mu_i}{\mu_1} e_i,$$

$$x = \sum_{i=2}^{d} (\lambda_i - \frac{\lambda_1}{\mu_1}\mu_i) e_i.$$

Diese Darstellung von x ist eine konische Kombination, denn

$$\mu_i \geq 0 \implies \lambda_i - \frac{\lambda_1}{\mu_1}\mu_i \geq 0,$$
$$\mu_i < 0 \implies \frac{\lambda_i}{\mu_i} \leq \frac{\lambda_1}{\mu_1} \implies \lambda_i \geq \frac{\lambda_1}{\mu_1}\mu_i,$$

also kann x mit weniger als d Extremalen konisch dargestellt werden, Widerspruch zur Minimalität! Da ein Kegel höchstens $\dim(K)$ linear unabhängige Vektoren enthält, folgt $d \leq \dim(K)$. Setzen wir $y_i = \lambda_i e_i$, $i = 1, \ldots, d$, so sind die Vektoren y_i Extremalen von K mit der gewünschten Eigenschaft. □

(1.60) Korollar. *Ist $P \subseteq \mathbb{K}^n$ ein spitzes Polyeder und ist $x \in P$, dann gibt es Ecken $v_0, v_1, \ldots, v_k$ und Extremalen $e_{k+1}, e_{k+2}, \ldots, e_d$ von P mit $d \leq \dim(P)$ und nichtnegative Skalare $\lambda_0, \ldots, \lambda_k$ mit $\sum_{i=0}^{k} \lambda_i = 1$, so dass gilt*

$$x = \sum_{i=0}^{k} \lambda_i v_i + \sum_{i=k+1}^{d} e_i.$$

△

Beweis. Nach (1.47) ist $\mathrm{hog}(P)$ ein spitzer Kegel, und die Dimension von $\mathrm{hog}(P)$ ist $\dim(P)+1$. Nach (1.46) gilt $x \in P \iff \binom{x}{1} \in \mathrm{hog}(P)$. Nach Satz (1.59) ist $\binom{x}{1}$ konische Kombination von $d+1 \leq \dim(P)+1$ Extremalen von $\mathrm{hog}(P)$. O. B. d. A. können wir annehmen, dass gilt

$$\binom{x}{1} = \sum_{i=0}^{k} \binom{y_i}{\lambda_i} + \sum_{i=k+1}^{d} \binom{e_i}{0},$$

wobei $\lambda_i > 0$, $i = 0, \ldots, k$. Für $v_i := \frac{1}{\lambda_i} y_i$ gilt dann

$$x = \sum_{i=0}^{k} \lambda_i v_i + \sum_{i=k+1}^{d} e_i, \qquad \sum_{i=0}^{k} \lambda_i = 1.$$

Ferner sind nach (1.51) die Vektoren v_i Ecken von P und die Vektoren e_i Extremalen von P. Also haben wir die gewünschte Kombination von x gefunden. □

Das folgende direkte Korollar von (1.60) wird in der Literatur häufig **Satz von Caratheodory** genannt.

(1.61) Korollar. *Ist $P \subseteq \mathbb{K}^n$ ein Polytop, dann ist jedes Element von P Konvexkombination von höchstens $\dim(P)+1$ Ecken von P.* △

Literaturverzeichnis

A. Bachem and M. Grötschel. New aspects of polyhedral theory. In B. Korte, editor, *Modern Applied Mathematics – Optimization and Operations Research*. North-Holland, Amsterdam, 1982.

J. Bochnak, M. Coste, and M.-F. Roy. *Real algebraic geometry*. Springer, 1998.

H. Bosse, M. Grötschel, and M. Henk. Polynomial inequalities representing polyhedra. *Mathematical Programming*, 103(1):35–44, May 2005.

B. Grünbaum. *Convex Polytopes*, volume 221 of *Graduate Texts in Mathematics*. Springer, 2nd edition, 2003.

J. Matoušek. *Lectures on Discrete Geometry*, volume 212 of *Graduate Texts in Mathematics*. Springer, 2002.

G. M. Ziegler. *Lectures on Polytopes*, volume 152 of *Graduate Texts in Mathematics*. Springer, 2010.

2 Matroide und Unabhängigkeitssysteme

Wir werden nun einige sehr allgemeine Klassen von kombinatorischen Optimierungsproblemen kennenlernen. Diese enthalten die in den Kapiteln 5 bis 7 in ADM I betrachteten Probleme. Der in Kapitel 5 von ADM I eingeführte Greedy-Algorithmus wird hier von besonderer Bedeutung sein. Allerdings treten bei der algorithmischen Behandlung der hier besprochenen allgemeinen Probleme einige Schwierigkeiten auf, die wir eingehend diskutieren müssen.

Gute Bücher über Matroidtheorie sind Oxley (1992), Truemper (1992) und Welsh (1976), ein lesenswerter Übersichtsartikel ist Welsh (1995). Optimierungsprobleme auf Matroiden werden z. B. in Bixby and Cunningham (1995) und Lee (2004), Seiten 49–74, ausführlich behandelt. Eine umfassende Übersicht über Matroide, submodulare Funktionen und verwandte Strukturen findet sich in Part IV, Volume B von Schrijver (2003) auf den Seiten 649–852.

2.1 Allgemeine Unabhängigkeitssysteme

E sei im folgenden immer eine endliche Menge, 2^E bezeichne die Menge aller Teilmengen von E.

(2.1) Definition. *Eine Menge $\mathcal{I} \subseteq 2^E$ heißt* Unabhängigkeitssystem *(oder* monotones Mengensystem*) auf E, wenn $\mathcal{I}$ die folgenden Axiome erfüllt:*

(I.1) $\emptyset \in \mathcal{I}$,

(I.2) $F \subseteq G \in \mathcal{I} \Longrightarrow F \in \mathcal{I}$.

Häufig wird auch das Paar $(E, \mathcal{I})$ Unabhängigkeitssystem genannt. Die Teilmengen von E, die in $\mathcal{I}$ enthalten sind, heißen unabhängige Mengen*, alle übrigen Teilmengen von E heißen* abhängige Mengen. △

Mit jedem Unabhängigkeitssystem $(E, \mathcal{I})$ sind auf kanonische Weise andere Mengensysteme verbunden, die wir nun kurz einführen wollen.

Die bezüglich Mengeninklusion minimalen abhängigen Teilmengen von E heißen *Zirkuits* (oder *Kreise*), d. h. $C \subseteq E$ ist ein Zirkuit, wenn C abhängig ist und wenn C keine von sich selbst verschiedene abhängige Teilmenge enthält. Die Menge aller Zirkuits (bzgl. eines Unabhängigkeitssystems $\mathcal{I}$) heißt *Zirkuitsystem* und wird mit $\mathcal{C}$ bezeichnet.

Ist $F \subseteq E$, so heißt jede Teilmenge von F, die unabhängig ist und die in keiner anderen unabhängigen Teilmenge von F enthalten ist, *Basis* von F, d. h.

$$B \text{ Basis von } F \iff (B, B' \in \mathcal{I}, B \subseteq B' \subseteq F \Longrightarrow B = B').$$

Die Menge aller Basen der Grundmenge E heißt *Basissystem* (bzgl. $\mathcal{I}$) und wird mit $\mathcal{B}$ bezeichnet.

Für jede Menge $F \subseteq E$ heißt die ganze Zahl

$$r(F) := \max\{|B| \mid B \text{ Basis von } F\}$$

Rang von F. Die Rangfunktion r ist also eine Funktion, die 2^E in die nicht-negativen ganzen Zahlen abbildet.

Offenbar induziert jedes Unabhängigkeitssystem $\mathcal{I}$ auf E ein eindeutig bestimmtes Zirkuitsystem, ein eindeutig bestimmtes Basissystem und eine eindeutig bestimmte Rangfunktion. Es gilt auch die Umkehrung, wie wir nachfolgend (ohne Beweis) skizzieren.

Zirkuitsysteme und Basissysteme sind nach Definition *Antiketten (Clutter)*, d. h. Systeme von Mengen, so dass keine zwei Mengen ineinander enthalten sind.

Ist $\mathcal{B} \neq \emptyset$ eine Antikette auf E, so ist

$$\mathcal{I} := \{I \subseteq E \mid \exists B \in \mathcal{B} \text{ mit } I \subseteq B\}$$

ein Unabhängigkeitssystem auf E, und $\mathcal{B}$ ist das zu $\mathcal{I}$ gehörige Basissystem.

Ist $\mathcal{C} \neq \{\emptyset\}$ eine Antikette auf E, so ist

$$\mathcal{I} := \{I \subseteq E \mid I \text{ enthält kein Element von } \mathcal{C}\} \tag{2.2}$$

ein Unabhängigkeitssystem, und $\mathcal{C}$ ist das zu $\mathcal{I}$ gehörige Zirkuitsystem.

Die oben definierte Rangfunktion hat folgende Eigenschaften. Sie ist *subkardinal*, d. h. für alle $F \subseteq E$ gilt

$$r(F) \leq |F|,$$

sie ist *monoton*, d. h. für alle $F, G \subseteq E$ gilt

$$F \subseteq G \Longrightarrow r(F) \leq r(G),$$

und sie ist *stark subadditiv*, d. h. für alle $F \subseteq E$, für alle ganzen Zahlen $k \geq 1$, für alle endlichen Indexmengen $K \subseteq \mathbb{N}$ und für alle Familien $(F_i)_{i \in K}$ von Teilmengen von F mit der Eigenschaft, dass $|\{i \in K \mid e \in F_i\}| = k$ für alle $e \in F$, gilt

$$k \cdot r(F) \leq \sum_{i \in K} r(F_i).$$

Ist $r : 2^E \to \mathbb{Z}_+$ eine subkardinale, monotone, stark subadditive Funktion, so ist

$$\mathcal{I} := \{I \subseteq E \mid r(I) = |I|\}$$

ein Unabhängigkeitssystem, dessen Rangfunktion die Funktion r ist.

Unabhängigkeitssysteme $\mathcal{I}$ auf einer Grundmenge E definieren also mathematische Strukturen, die äquivalent durch Zirkuitsysteme, Basissysteme oder Rangfunktionen gegeben werden können. Unabhängigkeitssysteme sind sehr allgemeine Objekte und besitzen zu wenig Struktur, um tiefliegende Aussagen über sie machen zu können.

Sei für jedes Element $e \in E$ ein Gewicht $c_e \in \mathbb{R}$ gegeben. Für $F \subseteq E$ setzen wir wie üblich

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

Das Problem, eine unabhängige Menge $I^* \in \mathcal{I}$ zu finden, so dass $c(I^*)$ maximal ist, heißt *Optimierungsproblem über einem Unabhängigkeitssystem* $\mathcal{I}$, d. h. wir suchen

$$\max\{c(I) \mid I \in \mathcal{I}\}. \tag{2.3}$$

Offenbar macht es hier keinen Sinn, Gewichte c_e zu betrachten, die nicht positiv sind. Denn wenn $I^* \in \mathcal{I}$ optimal ist, gilt $I' := I^* \setminus \{e \in E \mid c_e \leq 0\} \in \mathcal{I}$ (wegen (I.2)) und $c(I') \geq c(I^*)$, also ist I' ebenfalls eine optimale unabhängige Menge. Deswegen werden wir uns im Weiteren bei Optimierungsproblemen über Unabhängigkeitssystemen auf Gewichtsfunktionen beschränken, die positiv oder nicht-negativ sind.

Bei Optimierungsproblemen über Basissystemen ist es dagegen auch sinnvoll, negative Gewichte zuzulassen. Ist ein Basissystem $\mathcal{B}$ auf der Grundmenge E gegeben und sind $c_e \in \mathbb{R}$ Gewichte, so nennen wir

$$\min\{c(B) \mid B \in \mathcal{B}\} \tag{2.4}$$

Optimierungsproblem über einem Basissystem $\mathcal{B}$.

(2.5) Beispiel.

(a) Sei $G = (V, E)$ ein Graph. Eine Knotenmenge $S \subseteq V$ heißt *stabil (Clique)*, falls je zwei Knoten aus S nicht benachbart (benachbart) sind. Die Menge der stabilen Knotenmengen (Cliquen) ist ein Unabhängigkeitssystem auf V. Die Aufgabe – bei gegebenen Knotengewichten – eine gewichtsmaximale stabile Menge (Clique) zu finden, ist ein Optimierungsproblem über einem Unabhängigkeitssystem, vergleiche ADM I (3.13).

(b) Ein *Wald* in einem Graphen $G = (V, E)$ ist eine Kantenmenge, die keinen Kreis enthält. Die Menge aller Wälder bildet ein Unabhängigkeitssystem auf E. Das Problem, einen maximalen Wald in G zu finden, war Hauptthema von ADM I, Abschnitt 5.2. Offenbar ist in einem zusammenhängenden Graphen G die Menge der aufspannenden Bäume genau die Menge der Basen des Unabhängigkeitssystems der Wälder. Das Problem, einen minimalen aufspannenden Baum zu finden (siehe ADM I, Satz (5.13)), ist somit ein Optimierungsproblem über einem Basissystem.

(c) Ein *Matching* in einem Graphen $G = (V, E)$ ist eine Kantenmenge $M \subseteq E$, so dass jeder Knoten aus V in höchstens einer Kante aus M enthalten ist. Die Menge aller Matchings ist ein Unabhängigkeitssystem auf E. Das Matchingproblem ADM I, (3.10) ist also ein Optimierungsproblem über einem Unabhängigkeitssystem. Die Aufgabe, in einem vollständigen Graphen mit Kantengewichten ein minimales perfektes Matching zu finden, ist ein Optimierungsproblem über einem Basissystem. Analog bilden die zulässigen Lösungen eines 1-kapazitierten b-Matchingproblems ein Unabhängigkeitssystem.

(d) Gegeben sei ein vollständiger Digraph $D_n = (V, A)$ mit n Knoten und Bogenlängen c_{ij} für alle $(i, j) \in A$. Eine *Tour (gerichteter Hamiltonkreis)* ist ein gerichteter Kreis in D_n, der jeden Knoten enthält. Die Aufgabe, eine Tour mit minimalem Gewicht zu finden, heißt *asymmetrisches Travelling-Salesman-Problem*, siehe ADM I, (3.12). Die Menge $\mathcal{T}$ aller Touren ist kein Unabhängigkeitssystem, jedoch eine Antikette, also Basissystem eines Unabhängigkeitssystems. Das asymmetrische TSP ist somit ein Optimierungsproblem über einem Basissystem. Wir können es aber auch als Optimierungsproblem über einem Unabhängigkeitssystem auffassen. Dies geht wie folgt: Setzen wir

$$\tilde{\mathcal{T}} := \{I \subseteq A \mid \exists T \in \mathcal{T} \text{ mit } I \subseteq T\},$$
$$c'_{ij} := \max\{|c_{ij}| \mid (i, j) \in A\} + 1 - c_{ij},$$

so ist $\tilde{\mathcal{T}}$ ein Unabhängigkeitssystem, und jede Lösung von $\max\{c'(I) \mid I \in \tilde{\mathcal{T}}\}$ ist eine Tour, die – bezüglich der Gewichte c_{ij} – minimales Gewicht hat. (Auf die gleiche Weise kann man viele andere Optimierungsprobleme über Basissystemen in Optimierungsprobleme über Unabhängigkeitssystemen überführen.) Ebenso ist das symmetrische TSP ein Optimierungsproblem über einem Basissystem.

(e) Gegeben sei ein gerichteter Graph $D = (V, A)$. Ein *Branching* in D ist eine Bogenmenge $B \subseteq A$, die keinen Kreis (im ungerichteten Sinne) enthält, und die die Eigenschaft hat, dass jeder Knoten $v \in V$ Endknoten von höchstens einem Bogen aus B ist. Die Menge aller Branchings ist ein Unabhängigkeitssystem auf A. Das Problem, in einem vollständigen Digraphen eine minimale aufspannende Arboreszenz zu finden, ist ein Optimierungsproblem über einem Basissystem, siehe ADM I, (3.11).△

Überlegen Sie sich, welche der übrigen Beispiele aus ADM I, Abschnitt 3.3 als Optimierungsprobleme über Unabhängigkeits- oder Basissystemen aufgefasst werden können und welche nicht.

Verschiedene praktische Fragestellungen führen auch zu Optimierungsproblemen über Zirkuits. Sind die Elemente $e \in E$ durch Gewichte c_e bewertet, so kann man das Problem untersuchen, ein Zirkuit $C \in \mathcal{C}$ zu finden, das minimales Gewicht $c(C)$ hat. Die Aufgabe, in einem Graphen einen kürzesten Kreis zu bestimmen, ist z. B. von diesem Typ.

Allgemeiner noch ist folgende Frage von Interesse. Wir sagen, dass eine Menge $Z \subseteq E$ ein *Zyklus* ist, wenn Z die Vereinigung von paarweise disjunkten Zirkuits ist, d. h. wenn es Zirkuits $C_1, \ldots, C_k$ gibt mit $C_i \cap C_j = \emptyset$, $1 \leq i < j \leq k$, so dass $Z = \bigcup_{i=1}^{k} C_i$. Sind die Elemente $e \in E$ mit Gewichten c_e belegt, so sucht man nach einem Zyklus maximalen Gewichts. Das Chinesische Postbotenproblem (siehe ADM I, (3.12)) und das Max-Cut-Problem (siehe ADM I, (3.15)) sind z. B. von diesem Typ. Aus Zeitgründen können wir auf Optimierungsprobleme über Zirkuits bzw. Zyklen nicht eingehen, siehe hierzu Barahona and Grötschel (1986) und Grötschel and Truemper (1989).

2.2 Matroide

Wie die Beispiele aus dem vorigen Abschnitt zeigen, enthalten Optimierungsprobleme über Unabhängigkeitssystemen sowohl polynomial lösbare als auch $\mathcal{NP}$-vollständige Probleme. Man wird daher nicht erwarten können, dass für diese Probleme eine „gute“ Lösungstheorie existiert. Wir wollen nun eine Spezialklasse von Unabhängigkeitssystemen einführen, für die es so etwas gibt. In einem noch zu präzisierenden Sinn (siehe Folgerung (2.15)) ist dies die Klasse der Unabhängigkeitssysteme, für die der Greedy-Algorithmus (siehe ADM I, (5.7)) eine Optimallösung liefert.

(2.6) Definition. *Ein* Matroid M *besteht aus einer Grundmenge* E *zusammen mit einem Unabhängigkeitssystem* $\mathcal{I} \subseteq 2^E$*, das eine der folgenden äquivalenten Bedingungen erfüllt:*

$$\text{(I.3)} \quad I, J \in \mathcal{I}, |I| = |J| - 1 \implies \exists j \in J \setminus I \text{ mit } I \cup \{j\} \in \mathcal{I},$$

$$\text{(I.3}') \quad I, J \in \mathcal{I}, |I| < |J| \implies \exists K \subseteq J \setminus I \text{ mit } |I \cup K| = |J|, \text{ so dass } I \cup K \in \mathcal{I},$$

$$\text{(I.3}'') \quad F \subseteq E \text{ und } B, B' \text{ Basen von } F \implies |B| = |B'|. \qquad \triangle$$

Das heißt also, das Unabhängigkeitssystem eines Matroids auf E ist ein Mengensystem $\mathcal{I} \subseteq 2^E$, das die Axiome (I.1), (I.2) und eines der Axiome (I.3), (I.3′), (I.3″) erfüllt. Ein solches System erfüllt automatisch auch die beiden übrigen der drei Axiome (I.3), (I.3′), (I.3″).

Ein Wort zur Terminologie! Nach der obigen Definition ist ein Matroid M ein Paar $(E, \mathcal{I})$ mit den oben aufgeführten Eigenschaften. Wenn klar ist, um welche Grundmenge E es sich handelt, spricht man häufig auch einfach von dem Matroid $\mathcal{I}$, ohne dabei E explizit zu erwähnen. Man sagt auch, M (bzw. $\mathcal{I}$) ist ein Matroid auf E.

In Definition (2.6) haben wir von den „äquivalenten Bedingungen“ (I.3), (I.3′), (I.3″) gesprochen. Diese Äquivalenz muss natürlich bewiesen werden. Da wir hier jedoch keine Vorlesung über Matroide halten wollen, können wir auf die Beweise (aus Zeitgründen) nicht eingehen. Das gleiche gilt für die nachfolgend gemachten Aussagen über Zirkuit- und Basissysteme und Rangfunktionen. Beweise der hier gemachten Aussagen findet der interessierte Leser z. B. in Oxley (1992) und Welsh (1976).

Wie wir bereits gesehen haben, können Unabhängigkeitssysteme über Zirkuits, Basen oder Rangfunktionen beschrieben werden. Ist $M = (E, \mathcal{I})$ ein Matroid und sind $\mathcal{C}, \mathcal{B}, r$ das zugehörige Zirkuit-, Basissystem bzw. die zugehörige Rangfunktion, so ist natürlich $\mathcal{I}$ durch die Angabe von $\mathcal{C}$, $\mathcal{B}$ oder r eindeutig beschrieben. Gelegentlich werden daher auch die Paare $(E, \mathcal{C})$, $(E, \mathcal{B})$, (E, r) als Matroide bezeichnet, meistens dann, wenn es bei einer speziellen Untersuchung sinnvoller erscheint, mit Zirkuits, Basen oder Rangfunktionen statt mit Unabhängigkeitssystemen zu arbeiten.

Sicherlich induzieren nicht alle Zirkuit- oder Basissysteme oder alle Rangfunktionen Unabhängigkeitssysteme von Matroiden. Diese Antiketten bzw. Funktionen müssen spezielle Eigenschaften haben, damit das zugehörige Unabhängigkeitssystem das Axiom (I.3) erfüllt. Einige solcher Eigenschaften wollen wir kurz auflisten.

(2.7) Satz.

(a) *Eine Antikette $\mathcal{C} \subseteq 2^E$, $\mathcal{C} \neq \{\emptyset\}$, ist das* **Zirkuitsystem eines Matroids** *auf E genau dann, wenn eine der beiden folgenden äquivalenten Bedingungen erfüllt ist:*

(C.1) $C_1, C_2 \in \mathcal{C}$, $C_1 \neq C_2$, $z \in C_1 \cap C_2 \implies \exists C_3 \in \mathcal{C}$ *mit* $C_3 \subseteq (C_1 \cup C_2) \setminus \{z\}$,

(C.1′) $C_1, C_2 \in \mathcal{C}$, $C_1 \neq C_2$, $y \in C_1 \setminus C_2 \implies \forall x \in C_1 \cap C_2\ \exists C_3 \in \mathcal{C}$ *mit* $y \in C_3 \subseteq (C_1 \cup C_2) \setminus \{x\}$.

(b) *Eine Antikette $\mathcal{B} \subseteq 2^E$, $\mathcal{B} \neq \emptyset$, ist das* **Basissystem eines Matroids** *auf E genau dann, wenn das folgende Axiom erfüllt ist:*

(B.1) $B_1, B_2 \in \mathcal{B}$, $x \in B_1 \setminus B_2 \implies \exists y \in B_2 \setminus B_1$ *mit* $(B_1 \cup \{y\}) \setminus \{x\} \in \mathcal{B}$.

(c) *Eine Funktion $r : 2^E \to \mathbb{Z}$ ist die* **Rangfunktion eines Matroids** *auf E genau dann, wenn eines der beiden folgenden äquivalenten Axiomensysteme erfüllt ist:*

(R.1) $r(\emptyset) = 0$,

(R.2) $F \subseteq E$, $e \in E \implies r(F) \leq r(F \cup \{e\}) \leq r(F) + 1$,

(R.3) $F \subseteq E$, $f, g \in E$ *mit* $r(F \cup \{g\}) = r(F \cup \{f\}) = r(F) \implies r(F \cup \{g, f\}) = r(F)$,

beziehungsweise

(R.1′) $F \subseteq E \implies 0 \leq r(F) \leq |F|$ *(r ist subkardinal),*

(R.2′) $F \subseteq G \subseteq E \implies r(F) \leq r(G)$ *(r ist monoton),*

(R.3′) $F, G \subseteq E \implies r(F \cup G) + r(F \cap G) \leq r(F) + r(G)$ *(r ist submodular).*△

Es gibt noch einige hundert weitere äquivalente Definitionen von Matroiden (die Äquivalenz ist – auch in den obigen Fällen – nicht immer offensichtlich). Wir wollen uns jedoch mit den obigen begnügen.

Ist $\mathcal{I}_1$ ein Matroid auf einer Grundmenge E_1 und $\mathcal{I}_2$ ein Matroid auf E_2, so heißen $\mathcal{I}_1$ und $\mathcal{I}_2$ *isomorph*, falls es eine bijektive Abbildung $\varphi : E_1 \to E_2$ gibt mit

$$\varphi(F) \text{ ist unabhängig in } \mathcal{I}_2 \iff F \text{ ist unabhängig in } \mathcal{I}_1.$$

Eine Teilmenge $F \subseteq E$ heißt *abgeschlossen (bezüglich $\mathcal{I}$)*, falls gilt

$$r(F) < r(F \cup \{e\}) \text{ für alle } e \in E \setminus F.$$

Die Matroidtheorie kann man als eine gemeinsame Verallgemeinerung gewisser Aspekte der Graphentheorie und der linearen Algebra ansehen. Die beiden Beispiele, aus denen die Matroidtheorie entstanden ist, wollen wir daher zuerst vorstellen.

(2.8) Beispiel.

(a) **Graphische Matroide**
Das in (2.5)(b) definierte Unabhängigkeitssystem der Wälder eines Graphen $G = (V, E)$ ist ein Matroid. Ist $F \subseteq E$, so zerfällt (V, F) in Zusammenhangskomponenten $G_1 = (V_1, F_1), \ldots, G_k = (V_k, F_k)$. Die Basen der Zusammenhangskomponenten $G_1, \ldots, G_k$ sind die aufspannenden Bäume von $G_1, \ldots, G_k$. Jede Vereinigung von aufspannenden Bäumen von $G_1, \ldots, G_k$ ist eine Basis von F. Die Zirkuits von $\mathcal{I}$ sind die Kreise des Graphen G (daher der Name!). Der Rang einer Menge $F \subseteq E$ ist gegeben durch

$$r(F) = |V(F)| - \text{ Anzahl } k \text{ der Komponenten von } (V(F), F),$$

wobei $V(F)$ die Menge aller Knoten $v \in V$ bezeichnet, die in mindestens einer Kante aus F enthalten sind, vergleiche ADM I, Korollar (5.5).
Die Matroide, die wie oben angegeben auf einem Graphen definiert werden können (bzw. isomorph zu solchen sind), heißen *graphische Matroide*.

(b) **Matrix-Matroide**
Sei $A = (a_{ij})$ eine (m, n)-Matrix über einem beliebigen Körper K mit Spaltenvektoren $A_{\cdot 1}, \ldots, A_{\cdot n}$. $E = \{1, \ldots, n\}$ sei die Menge der Spaltenindizes von A. Eine Teilmenge $F \subseteq E$ heißt unabhängig, wenn die Vektoren $A_{\cdot j}$, $j \in F$, linear unabhängig in K^m sind. Da jede Teilmenge einer linear unabhängigen Menge wiederum linear unabhängig ist, ist das so definierte Mengensystem $\mathcal{I}$ offenbar ein Unabhängigkeitssystem. Die Menge aller Basen von E ist die Menge aller $B \subseteq E$, so dass die Vektoren $A_{\cdot j}$, $j \in B$, eine Basis des durch die Spaltenvektoren von A aufgespannten linearen Teilraums von K^m bilden. Das Basisaxiom (B.1) ist in diesem Falle aufgrund des Steinitz'schen Austauschsatzes erfüllt. (Dieser Satz war die Motivation für (B.1).)
Matroide, die auf die hier angegebene Weise konstruiert werden können, heißen *Matrix-Matroide*. Die Rangfunktion von $\mathcal{I}$ entspricht der aus der linearen Algebra bekannten Rangfunktion des K^m beschränkt auf die Spalten von A. Ist $\mathcal{I}$ ein Matroid auf E und gibt es einen Körper K und eine (m, n)-Matrix A über K, so dass $\mathcal{I}$ zu dem Matrix-Matroid bezüglich A isomorph ist, dann heißt $\mathcal{I}$ *repräsentierbar (über K)*. (Natürlich kann man hier etwas verallgemeinern und anstelle von Körpern Schiefkörper oder andere geeignete Objekte betrachten und Repräsentierbarkeit über diesen studieren.) Matroide, die über dem zweielementigen Körper $GF(2)$ repräsentierbar sind, heißen *binär*. Eine umfassende Untersuchung dieser wichtigen Klasse von Matroiden findet sich in Truemper (1992). Matroide, die über allen Körpern repräsentierbar sind, nennt man *regulär*.

(c) **Binäre Matroide**
Die über dem zweielementigen Körper $GF(2)$ repräsentierbaren Matroide kann man auf sehr einfache Weise durch eine 0/1-Matrix repräsentieren. Seien M ein binäres Matroid auf E, T eine Basis von E und $S := E \setminus T$. Wir können o. B. d. A. annehmen, dass $T = \{e_1, \ldots, e_m\}$ und $S = \{e_{m+1}, \ldots, e_n\}$ gilt. Eine das Matroid M (mit Rang

m) repräsentierende Matrix A erhält man wie folgt: A hat m Zeilen und n Spalten. Die i-te Zeile „repräsentiert“ das i-te Basiselement e_i, $1 \leq i \leq m$, die j-te Spalte das j-te Element e_j, $1 \leq j \leq n$, von E; A hat die folgende Form (genannt *Standardform* oder *Standardrepräsentation*):

$$A = (I_m, B),$$

wobei I_m die (m, m)-Einheitsmatrix ist. Die j-te Spalte von A, $m+1 \leq j \leq n$, wird wie folgt konstruiert. Fügt man das Element e_j zur Basis T hinzu, so kann man beweisen, dass genau ein Zirkuit entsteht, genannt das Fundamentalzirkuit zu e_j. Für das graphische Matroid des Beispielgraphen in Abbildung 2.1 ergibt sich so die folgende Standardrepräsentation:

	1	2	3	4	5	6	7	8	9	10
1	1					0	0	0	0	0
2		1			0	0	1	1	1	0
3			1			0	1	0	1	0
4	0			1		1	1	1	0	0
5					1	1	1	0	0	0

△

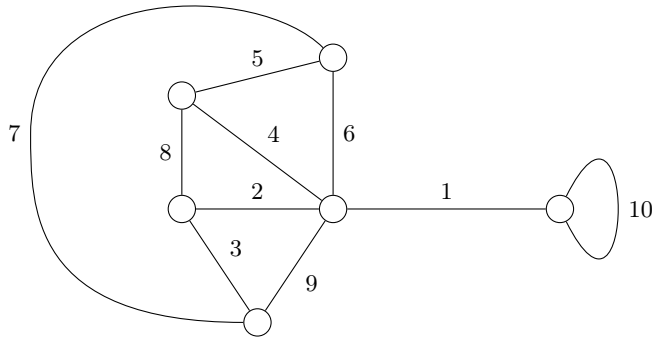


Abbildung 2.1: Graph mit 10 Kanten

Es folgt nun eine Liste weiterer interessanter Matroide.

(2.9) Beispiel.

(a) **Cographische Matroide**
Gegeben sei ein Graph $G = (V, E)$. Ein *Cokreis* ist eine Kantenmenge, deren Entfernung aus G die Komponentenzahl erhöht und die (mengeninklusionsweise) minimal bezüglich dieser Eigenschaft ist. Ein *Schnitt* ist eine Kantenmenge der Form

$$\delta(W) = \{ij \in E \mid i \in W, j \in V \setminus W\}, \quad W \subseteq V.$$

Jeder Cokreis ist offenbar ein Schnitt, und es ist einfach einzusehen, dass die Cokreise gerade die minimalen nicht-leeren Schnitte von G sind. Sind $\delta(W)$ und $\delta(W')$ verschiedene Cokreise und ist die Kante ij in beiden enthalten, so ist $\delta(W \triangle W')$ ein Schnitt, der ij nicht enthält. (Hier bezeichnet $W \triangle W'$ die *symmetrische Differenz* $(W \cup W') \setminus (W \cap W')$.) Ist $\delta(W \triangle W')$ kein Cokreis, so enthält er einen Cokreis, der natürlich ij auch nicht enthält. Daraus folgt, dass die Antikette der Cokreise das Axiom (C.1) erfüllt und somit das Zirkuitsystem eines Matroids auf E ist. Ein Matroid, das isomorph zu einem so definierten Matroid ist, heißt *cographisch*. Ist $G = (V, E)$ ein zusammenhängender Graph und M das cographische Matroid auf G, so ist das Basissystem $\mathcal{B}$ von M gegeben durch

$$\mathcal{B} = \{B \subseteq E \mid \exists \text{ aufspannender Baum } T \subseteq E \text{ mit } B = E \setminus T\}.$$

(b) **Uniforme Matroide**
Sei E eine Menge mit n Elementen, dann ist die Menge aller Teilmengen von E mit höchstens k Elementen ein Matroid auf E. Dieses Matroid heißt *uniform* und ist durch die Angabe von k und n bis auf Isomorphie eindeutig bestimmt. Dieses Matroid wird mit $U_{k,n}$ bezeichnet. Das Basissystem von $U_{k,n}$ wird durch die Menge der Teilmengen von E mit genau k Elementen gebildet. Das Zirkuitsystem von $U_{k,n}$ besteht aus den Teilmengen von E mit $k+1$ Elementen. Die Matroide $U_{n,n}$ (d. h. die Matroide, in denen alle Mengen unabhängig sind) heißen *frei* (oder *trivial*). Für freie Matroide gilt $\mathcal{C} = \emptyset$, $\mathcal{B} = \{E\}$, $r(F) = |F|$ für alle $F \subseteq E$. Das uniforme Matroid $U_{2,4}$ ist das kleinste nicht binäre Matroid. Überlegen Sie sich einen Beweis hierfür!

(c) **Partitionsmatroide**
Sei E eine endliche Menge, und $E_1, \ldots, E_k$ seien nicht-leere Teilmengen von E mit $E_i \cap E_j = \emptyset$, $i \neq j$, und $\bigcup_{i=1}^{k} E_i = E$. Seien $b_1, \ldots, b_k$ nicht-negative ganze Zahlen, dann ist $\mathcal{I} := \{I \subseteq E \mid |I \cap E_i| \leq b_i, i = 1, \ldots, k\}$ ein Matroid auf E, genannt *Partitionsmatroid*.

(d) **Transversalmatroide**
Sei E eine endliche Menge, $(E_i)_{i \in I}$ sei eine endliche Familie von Teilmengen von E. Eine Teilmenge $T \subseteq E$ ist eine *teilweise Transversale* (oder partielles Repräsentantensystem) von $(E_i)_{i \in I}$, falls es eine Indexmenge $J \subseteq I$ gibt mit $|J| = |T|$ und eine Bijektion $\pi : T \to J$, so dass $t \in E_{\pi(t)}$ für alle $t \in T$. Die Menge aller teilweisen Transversalen ist das Unabhängigkeitssystem eines Matroids auf E. (Dieses Ergebnis ist nicht trivial und hatte einen wesentlichen Einfluss auf die Transversaltheorie.) $\triangle$

Wir wollen nun noch einige konkrete Beispiele von Matroiden angeben und ihre Basis-, Zirkuitsysteme etc. explizit auflisten.

Betrachten wir den folgenden, in Abbildung 2.2 dargestellten Graphen $G = (V, E)$ mit $E = \{1, 2, \ldots, 13\}$. Das Zirkuitsystem $\mathcal{C}$ des graphischen Matroids auf E ist gegeben durch die Menge aller Kreise in G, d. h.

$$\mathcal{C} = \{\{1, 2, 3\}, \{4, 5, 6, 7\}, \{9, 10, 11\}, \{11, 12, 13\}, \{9, 10, 12, 13\}\}.$$

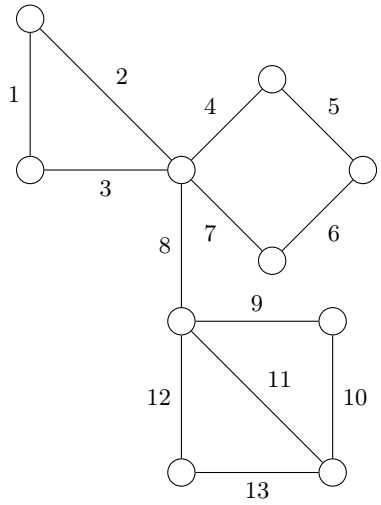


Abbildung 2.2: Graph G mit 13 Kanten

Das Zirkuitsystem $\mathcal{C}^*$ des cographischen Matroids auf E ist gegeben durch die Menge aller minimalen Schnitte, d. h.

$$\begin{aligned}\mathcal{C}^* = \ & \{\{1,2\},\{1,3\},\{2,3\},\{4,5\},\{4,6\},\{4,7\},\{5,6\},\{5,7\},\{6,7\},\{8\},\\ & \{9,10\},\{9,11,12\},\{9,11,13\},\{10,11,12\},\{10,11,13\},\{12,13\}\}.\end{aligned}$$

Das Basissystem $\mathcal{B}$ des graphischen Matroids des folgenden Graphen $G = (V, E)$ (siehe Abbildung 2.3) mit $E = \{1, 2, 3, 4\}$ ist gegeben durch

$$\mathcal{B} = \{\{1,2,3\},\{1,2,4\},\{1,3,4\}\},$$

und das Basissystem $\mathcal{B}^*$ des cographischen Matroids bezüglich G ist die folgende Anti-

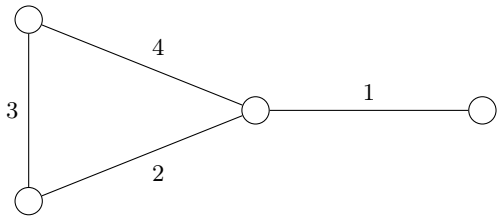


Abbildung 2.3: Graph G mit 4 Kanten

kette:

$$\mathcal{B}^* = \{\{4\},\{3\},\{2\}\}.$$

Das graphische Matroid des in Abbildung 2.4 dargestellten Graphen ist das uniforme Matroid $U_{2,3}$. Uniforme Matroide können also auch isomorph zu graphischen sein. Das uniforme Matroid $U_{2,4}$ ist jedoch nicht graphisch (Übungsaufgabe).

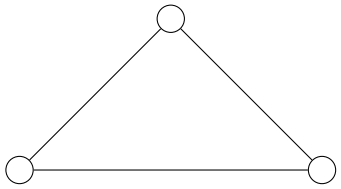


Abbildung 2.4: Graph zum uniformen Matroid $U_{2,3}$

Betrachten wir die Matrix

$$A = \begin{pmatrix} 1 & -1 & 1 & -1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

als Matrix über dem Körper $\mathbb{R}$ oder $\mathbb{Q}$. Das Zirkuitsystem $\mathcal{C}$ des Matrix-Matroids M bezüglich A ist offenbar gegeben durch

$$\mathcal{C} = \{\{1,2,3\},\{2,3,4\},\{1,4\}\},$$

und das Basissystem $\mathcal{B}$ des Matrix-Matroids M ist

$$\mathcal{B} = \{\{1,2\},\{1,3\},\{2,3\},\{2,4\},\{3,4\}\}.$$

Ein aus der endlichen Geometrie stammendes Beispiel ist das **Fano-Matroid**, das häufig mit dem Symbol F_7 bezeichnet wird. Betrachten Sie Abbildung 2.5 Dies ist eine

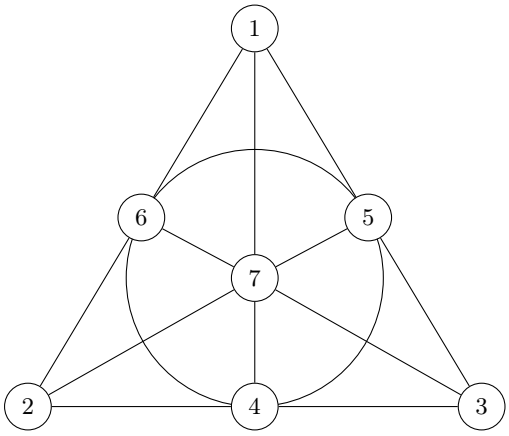


Abbildung 2.5: Fano-Ebene

graphische Darstellung der Fano-Ebene. Die Fano-Ebene hat 7 Punkte und 7 Geraden. Die 7 Geraden werden durch die 6 geraden Linien $\{1,2,6\}$, ..., $\{3,6,7\}$ und den Kreis, der durch die Punkte $\{4,5,6\}$ geht, repräsentiert. Das Fano-Matroid F_7 ist auf der Menge $E = \{1,\ldots,7\}$ definiert. Eine Teilmenge B von E ist genau dann eine Basis von F_7, wenn $|B| = 3$ und wenn die drei zu B gehörigen Punkte nicht kollinear sind, also nicht auf einer Geraden liegen. Die Menge $\{1,2,3\}$ ist somit eine Basis, $\{4,5,6\}$ dagegen nicht.

Das Fano-Matroid ist binär. Wählen wir die Basis $T = \{1, 2, 3\}$, so ergibt die in (2.8)(c) beschriebene Konstruktion der Standardrepräsentation die folgende Matrix:

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{pmatrix} \end{array}$$

die das Fano-Matroid F_7 über $GF(2)$ repräsentiert.

Wir wollen nun noch einen einfachen, aber interessanten Zusammenhang zwischen Unabhängigkeitssystemen und Matroiden erwähnen.

(2.10) Satz. *Jedes Unabhängigkeitssystem ist als Durchschnitt von Matroiden darstellbar, d. h. ist $\mathcal{I}$ ein Unabhängigkeitssystem auf E, dann gibt es Matroide $\mathcal{I}_1, \ldots, \mathcal{I}_k$ auf E mit*

$$\mathcal{I} = \bigcap_{i=1}^{k} \mathcal{I}_i.$$

△

Beweis. Sei $\mathcal{C}$ das zu $\mathcal{I}$ gehörige Zirkuitsystem. Jedes Zirkuit $C \in \mathcal{C}$ definiert eine Antikette $\{C\} \subseteq 2^E$, die trivialerweise das Axiom (C.1) aus (2.7) erfüllt. Also ist das zu dem Zirkuitsystem $\{C\}$ gehörige Unabhängigkeitssystem $\mathcal{I}_C$ ein Matroid. Wir behaupten nun

$$\mathcal{I} = \bigcap_{C \in \mathcal{C}} \mathcal{I}_C.$$

Ist $I \in \mathcal{I}$, so ist kein Zirkuit $C \in \mathcal{C}$ in I enthalten, folglich ist nach (2.2) $I \in \mathcal{I}_C$ für alle $C \in \mathcal{C}$. Sei umgekehrt $I \in \mathcal{I}_C$ für alle $C \in \mathcal{C}$, so heißt dies, dass kein Zirkuit $C \in \mathcal{C}$ in I enthalten ist, und somit, dass I ein Element von $\mathcal{I}$ ist. □

Die im Beweis von Satz (2.10) angegebene Konstruktion zur Darstellung eines Unabhängigkeitssystems als Durchschnitt von Matroiden produziert i. A. eine riesige Zahl von Matroiden, die das Gewünschte leisten. Häufig kommt man mit viel weniger Matroiden aus.

Betrachten wir z. B. die Menge der Branchings $\mathcal{I} \subseteq 2^A$ in einem Digraphen $D = (V, A)$. Man kann einfach zeigen, dass das zugehörige Zirkuitsystem $\mathcal{C}$ aus den inklusionsminimalen Mengen der Vereinigung der folgenden Antiketten $\mathcal{C}_1$ und $\mathcal{C}_2$ besteht:

$\mathcal{C}_1 := \{C \subseteq A \mid |C| = 2$ und die Endknoten der beiden Bögen in C sind identisch$\}$,
$\mathcal{C}_2 := \{C \subseteq A \mid C$ ist ein Kreis (die Bogenrichtungen spielen keine Rolle)$\}$.

$\mathcal{C}_1$ ist das Zirkuitsystem eines Partitionsmatroids auf A, dessen Unabhängigkeitssystem gegeben ist durch $\{B \subseteq A \mid |B \cap \delta^-(v)| \leq 1 \; \forall v \in V\}$, und $\mathcal{C}_2$ ist das Zirkuitsystem des graphischen Matroids auf D (hierbei wird D als Graph aufgefasst, d. h. die Bogenrichtungen werden ignoriert). Daraus folgt, dass das Unabhängigkeitssystem der Branchings

Durchschnitt von 2 Matroiden ist. Für den vollständigen Digraphen D mit n Knoten hätte die Konstruktion im Beweis von Satz (2.10) insgesamt

$$n\binom{n-1}{2} + \sum_{k=2}^{n} \binom{n}{k}(k-1)!$$

verschiedene Matroide geliefert.

Das Unabhängigkeitssystem $\tilde{\mathcal{T}}$ der Teilmengen von Touren im vollständigen Digraphen D_n mit n Knoten, siehe (2.5)(d), ist als Durchschnitt von 3 Matroiden darstellbar (Übungsaufgabe). Also ist das asymmetrische Travelling-Salesman-Problem als Optimierungsproblem über dem Durchschnitt von 3 Matroiden formulierbar.

2.3 Orakel

Um Eigenschaften von Matroiden oder Unabhängigkeitssystemen überprüfen zu können, muss man sich natürlich fragen, wie man Matroide geeignet darstellt, um z. B. ein Computerprogramm schreiben zu können, das Matroide als Input akzeptiert.

Betrachten wir zum Beispiel das in (2.3) formulierte Optimierungsproblem $\max\{c(I) \mid I \in \mathcal{I}\}$ über einem Unabhängigkeitssystem $\mathcal{I}$ auf E (Spezialfall: $(E, \mathcal{I})$ ist ein Matroid). Ist $\mathcal{I}$ als Liste aller unabhängigen Mengen gegeben, so ist das Optimierungsproblem völlig trivial. Wir durchlaufen die Liste, rechnen für jede Menge I der Liste den Wert $c(I)$ aus und wählen eine Menge I^*, so dass $c(I^*)$ maximal ist. Die Laufzeit dieses Enumerationsalgorithmus ist linear in der Inputlänge des Unabhängigkeitssystems.

Viele der Matroide und Unabhängigkeitssysteme, die wir in den voraufgegangenen Abschnitten definiert haben, sind jedoch in wesentlich kompakterer Form gegeben. Zum Beispiel ist ein Matrix-Matroid (2.8)(b) durch eine Matrix A (mit der Information, dass $I \subseteq E$ unabhängig genau dann ist, wenn die Spalten $A_{\cdot i}$, $i \in I$, linear unabhängig sind) gegeben, ein graphisches Matroid (2.8)(a) durch einen Graphen $G = (V, E)$ (zusammen mit der Information, dass $I \subseteq E$ unabhängig ist genau dann, wenn I keinen Kreis enthält), ein cographisches Matroid (2.9)(a) durch einen Graphen $G = (V, E)$ (zusammen mit der Information, dass $C \subseteq E$ ein Zirkuit ist genau dann, wenn C ein Cokreis ist). Würden wir die Inputlänge des Matroids als die Länge der Kodierung der Matrix A (für (2.8)(b)) bzw. als die Länge der Kodierung des Graphen G (für (2.8)(a)) definieren, hätte unser trivialer Enumerationsalgorithmus exponentielle Laufzeit in der Kodierungslänge dieses kompakten Inputs.

Die Frage ist also: Was ist eine „geeignete“ Kodierung eines Matroids? Motiviert durch die gerade angegebenen Beispiele könnte man glauben, dass die Angabe einer Liste aller unabhängigen Mengen eine unökonomische Methode sei. Jedoch geht es im allgemeinen nicht viel besser. Man kann nämlich zeigen, dass es mindestens

$$2^{2^{n/2}} \text{ Matroide mit } n \text{ Elementen}$$

gibt. Daraus folgt, dass es bei jeder beliebigen Kodierungsart immer Matroide gibt, deren Kodierung eine Länge hat, die mindestens $2^{n/2}$ beträgt.

Aufgrund dieser Tatsache hat sich ein anderes Konzept der Repräsentation von Matroiden durchgesetzt und als außerordentlich nützlich erwiesen. Matroide werden durch „Orakel“ dargestellt.

Wir treffen folgende Definition. Die *Kodierungslänge eines Matroids* $M = (E, \mathcal{I})$ ist die Anzahl der Elemente von E. Das Matroid selbst ist in einem Orakel „versteckt“, wobei das Orakel als ein Computerprogramm (Subroutine) interpretiert werden kann, das Fragen eines speziellen Typs beantwortet. Wir geben einige Beispiele (diese gelten natürlich nicht nur für Matroide, sondern analog auch für Unabhängigkeitssysteme) und gehen davon aus, dass wir die Grundmenge E kennen.

(2.11) Beispiel.

(a) **Unabhängigkeitsorakel**
Für jede Menge $F \subseteq E$ können wir das Orakel fragen, ob F unabhängig ist. Das Orakel antwortet mit „ja“ oder „nein“.

(b) **Zirkuitorakel**
Für jede Menge $F \subseteq E$ können wir das Orakel fragen, ob F ein Zirkuit ist. Das Orakel antwortet mit „ja“ oder „nein“.

(c) **Basisorakel**
Für jede Menge $F \subseteq E$ können wir das Orakel fragen, ob F eine Basis von E ist. Das Orakel antwortet mit „ja“ oder „nein“.

(d) **Rangorakel**
Für jede Menge $F \subseteq E$ können wir das Orakel fragen, wie groß der Rang von F ist. Die Antwort des Orakels ist „$r(F)$“. △

Man sieht sofort, dass dieses theoretische Orakelkonzept dem Konzept von Unterprogrammen der Programmierung entspricht.

Wenn wir also Algorithmen betrachten wollen, die Eigenschaften von Matroiden überprüfen, so gehen wir immer davon aus, dass ein Matroid (oder Unabhängigkeitssystem) durch die Grundmenge E und ein Orakel gegeben ist, das im Verlaufe des Algorithmus befragt werden kann. Bei der Komplexitätsanalyse von Algorithmen für Matroide (Unabhängigkeitssysteme) wird dann ein Orakelaufruf (Unterprogrammaufruf) als ein Schritt gezählt. Achtung! Die Laufzeit beim Lesen der Antwort wird wie üblich berechnet. Gibt also ein Orakel eine Antwort, die z. B. $2^{|E|}$ Speicherplätze benötigt, so hat der Algorithmus automatisch eine exponentielle Laufzeit. Bei den in (2.11) angegebenen Orakeln kommt so ein Fall allerdings nie vor! Man nennt Verfahren, die Orakel aufrufen können, *Orakelalgorithmen*. Ist ihre Laufzeit in dem oben angegebenen Sinne polynomial in $|E|$, so sagt man, dass die Verfahren *orakelpolynomial* sind.

Hat man einen Algorithmus, dessen Laufzeit orakelpolynomial in $|E|$ ist, so ist der Algorithmus für alle die Matroide (Unabhängigkeitssysteme) im üblichen Sinne polynomial, für die das Orakel durch einen in $|E|$ polynomialen Algorithmus realisiert werden kann. Dies ist zum Beispiel für Matrix-Matroide der Fall. Ein Unabhängigkeitsorakel kann man bei einer gegebenen Matrix durch Rangbestimmung mit Gauß-Elimination realisieren

(analog die drei anderen Orakel). Dies gilt auch für graphische und cographische Matroide, die durch einen Graphen $G = (V, E)$ gegeben sind. Zum Beispiel kann man die Unabhängigkeit einer Menge F im graphischen Matroid (F enthält keinen Kreis) durch Depth-First-Search testen; auch die übrigen drei Orakel kann man durch Algorithmen realisieren, deren Laufzeit polynomial in $|E|$ ist.

Eine wichtige Frage ergibt sich sofort. Sind die verschiedenen Orakel algorithmisch „äquivalent“? Das heißt, kann man ein Orakel durch ein anderes Orakel in orakelpolynomialer Zeit simulieren und umgekehrt? Zum Beispiel: Ist ein Unabhängigkeitsorakel gegeben, kann man dann einen Algorithmus entwerfen (der nur dieses Orakel benutzt und orakelpolynomial ist), der für jede Menge $F \subseteq E$ korrekt entscheidet, ob F ein Zirkuit ist oder nicht?

Für Matroide und die vier oben angegebenen Orakel wurde diese Frage von Hausmann and Korte (1981) wie folgt beantwortet.

(2.12) Satz. *Sei $M = (E, \mathcal{I})$ ein beliebiges Matroid.*

(a) *Das Unabhängigkeitsorakel und das Rangorakel sind bezüglich M „äquivalent“.*

(b) *Das Basisorakel und das Zirkuitorakel können durch das Unabhängigkeitsorakel sowie durch das Rangorakel in orakelpolynomialer Zeit simuliert werden.*

(c) *Es gibt keine anderen orakelpolynomialen Beziehungen zwischen diesen Orakeln.* △

Man kann diesen Satz graphisch durch Abbildung 2.6 veranschaulichen. Ein Pfeil in diesem Diagramm besagt, dass das Orakel am Ende des Pfeils durch polynomial viele Aufrufe des Orakels an der Spitze des Pfeils simuliert werden kann. Ist zwischen zwei Kästchen kein Pfeil (in irgendeiner der beiden Richtungen), so besagt dies auch, dass es keine orakelpolynomiale Simulation dieser Art gibt. Z. B. kann man das Basisorakel nicht durch das Zirkuitorakel in orakelpolynomialer Zeit simulieren und umgekehrt. Dies zeigt man dadurch, dass man eine Klasse von Beispielen angibt, bei denen zur Entscheidung der Frage, ob eine Menge I eine Basis (ein Zirkuit) ist, eine Anzahl von Aufrufen des Zirkuitorakels (Basisorakels) notwendig ist, die exponentiell in $|E|$ ist.

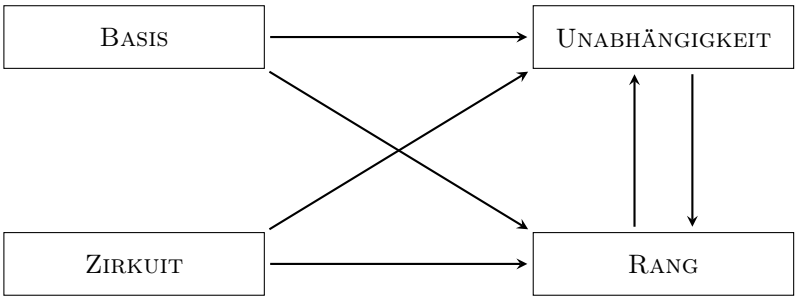


Abbildung 2.6: Orakelpolynomiale Reduktionen für Matroide

Hausmann and Korte (1980) haben dieselbe Fragestellung auch für allgemeine Unabhängigkeitssysteme untersucht, die – wie wir in Abschnitt 2.1 gesehen haben – äquivalent

durch Zirkuitsysteme, Basissysteme oder Rangfunktionen gegeben werden können. Der entsprechende Satz ist bildlich in Abbildung 2.7 veranschaulicht (Interpretation wie bei Abbildung 2.6).

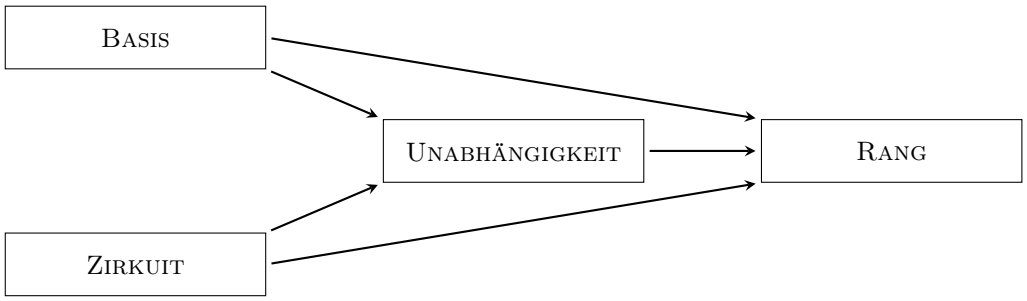


Abbildung 2.7: Orakelpolynomiale Reduktionen für allgemeine Unabhängigkeitssysteme

Insbesondere folgt, dass bei Unabhängigkeitssystemen das Rangorakel das stärkste ist. Durch dieses lassen sich die übrigen Orakel in orakelpolynomialer Zeit simulieren. Jedoch sind hier keine zwei Orakel algorithmisch „äquivalent“ (sie sind natürlich logisch äquivalent).

2.4 Optimierung über Unabhängigkeitssystemen

Wir wollen nun die Frage untersuchen, ob bzw. wie gut ein Optimierungsproblem der Form (2.3)

$$\max\{c(I) \mid I \in \mathcal{I}\},$$

wobei $\mathcal{I}$ ein Unabhängigkeitssystem auf einer Grundmenge E ist, gelöst werden kann. Wir betrachten dazu den folgenden trivialen Algorithmus, vergleiche ADM I, (5.7).

(2.13) Algorithmus Greedy-max für Unabhängigkeitssysteme.

Eingabe: Grundmenge $E = \{1, \dots, n\}$ mit Gewichten $c_i \in \mathbb{R}$ für alle $i \in E$. Ferner ist ein Unabhängigkeitssystem $\mathcal{I} \subseteq 2^E$ durch ein Unabhängigkeitsorakel (siehe (2.11)(a)) gegeben.

Ausgabe: Eine unabhängige Menge $I_g \in \mathcal{I}$.

1. Sortiere die Gewichte in nicht aufsteigender Reihenfolge (d. h. nach Beendigung von Schritt 1 können wir annehmen, dass $c_1 \geq c_2 \geq \cdots \geq c_n$ gilt).
2. Setze $I := \emptyset$.
3. FOR $i = 1$ TO n DO:

   Ist $c_i \leq 0$, gehe zu 4.

   Ist $I \cup \{i\}$ unabhängig (Orakelaufruf), dann setze $I := I \cup \{i\}$.
4. Setze $I_g := I$ und gib I_g aus. △

Der Greedy-Algorithmus durchläuft (nach der Sortierung in Schritt 1) die Elemente von E genau einmal, entweder er nimmt ein Element in die Menge I auf, oder er verwirft es für immer. Die am Ende des Verfahrens gefundene Lösung I_g nennen wir *Greedy-Lösung.*

Für eine Menge $F \subseteq E$ setzen wir

$$r_u(F) := \min\{|B| \mid B \text{ Basis von } F\}$$

und nennen $r_u(F)$ den *unteren Rang von* F. Wir setzen

$$q := \min_{F \subseteq E, r(F)>0} \frac{r_u(F)}{r(F)}$$

und nennen q den *Rangquotient* von $\mathcal{I}$. Ist $(E, \mathcal{I})$ ein Matroid, so gilt nach (I.3″) natürlich $r_u(F) = r(F)$ für alle $F \subseteq E$ und somit $q = 1$. Das folgende Resultat wurde von Jenkyns (1976) bewiesen.

(2.14) Satz. *Sei $\mathcal{I}$ ein Unabhängigkeitssystem auf E, seien I_g eine Greedy-Lösung von* (2.13) *und I_0 eine optimale Lösung von* (2.3)*, dann gilt*

$$q \le \frac{c(I_g)}{c(I_0)} \le 1,$$

und für jedes Unabhängigkeitssystem gibt es Gewichte $c_i \in \{0, 1\}$, $i \in E$, so dass die erste Ungleichung mit Gleichheit angenommen wird. △

Beweis. Wir können o. B. d. A. annehmen, dass $c_i > 0$ für $i = 1, \ldots, n$ und dass $c_1 \ge c_2 \ge \cdots \ge c_n$ gilt. Aus notationstechnischen Gründen führen wir ein Element $n + 1$ ein mit $c_{n+1} = 0$. Wir setzen

$$E_i := \{1, \ldots, i\}, \quad i = 1, \ldots, n.$$

Es gilt dann offenbar:

$$c(I_g) = \sum_{i=1}^{n} |I_g \cap E_i|(c_i - c_{i+1}),$$
$$c(I_0) = \sum_{i=1}^{n} |I_0 \cap E_i|(c_i - c_{i+1}).$$

Da $I_0 \cap E_i \subseteq I_0$, gilt $I_0 \cap E_i \in \mathcal{I}$, und somit $|I_0 \cap E_i| \le r(E_i)$. Die Vorgehensweise des Greedy-Algorithmus impliziert, dass $I_g \cap E_i$ eine Basis von E_i ist, also dass $|I_g \cap E_i| \ge r_u(E_i)$ gilt. Daraus folgt

$$|I_g \cap E_i| \;\ge\; |I_0 \cap E_i| \,\frac{r_u(E_i)}{r(E_i)} \;\ge\; |I_0 \cap E_i|\, q, \quad i = 1, \ldots, n$$

und somit

$$\begin{aligned} c(I_g) &= \sum_{i=1}^{n} |I_g \cap E_i|(c_i - c_{i+1}) \\ &\geq \sum_{i=1}^{n} |I_0 \cap E_i| q(c_i - c_{i+1}) \\ &= q \sum_{i=1}^{n} |I_0 \cap E_i|(c_i - c_{i+1}) \\ &= c(I_0)q. \end{aligned}$$

Die Ungleichung $1 \geq \frac{c(I_g)}{c(I_0)}$ ist trivial. Damit haben wir die Gültigkeit der Ungleichungskette bewiesen.

Sei nun $F \subseteq E$ mit $q = \frac{r_u(F)}{r(F)}$. Wir können annehmen, dass $F = \{1, \ldots, k\}$ gilt und dass $B = \{1, \ldots, p\} \subseteq F$ eine Basis von F ist mit $|B| = r_u(F)$. Wir setzen $c_i = 1$, $i = 1, \ldots, k$, und $c_i = 0$, $i = k+1, \ldots, n$. Dann liefert der Greedy-Algorithmus die Greedy-Lösung $I_g = B$ mit $c(I_g) = r_u(F)$, während für jede Optimallösung I_0 gilt $c(I_0) = r(F)$. Also wird für diese spezielle 0/1-Zielfunktion die linke Ungleichung in der Aussage des Satzes mit Gleichheit angenommen. □

(2.15) Korollar. *Sei $\mathcal{I}$ ein Unabhängigkeitssystem auf E. Dann sind äquivalent:*

(a) *$(E, \mathcal{I})$ ist ein Matroid.*

(b) *Für alle Gewichte $c \in \mathbb{R}^E$ liefert der Greedy-Algorithmus* (2.13) *eine Optimallösung von* (2.3).

(c) *Für alle Gewichte mit* 0/1*-Koeffizienten liefert der Greedy-Algorithmus* (2.13) *eine Optimallösung von* (2.3). △

(2.15) wurde von verschiedenen Autoren unabhängig voneinander bewiesen. Edmonds (1971) zeigte insbesondere, wie man den Greedy-Algorithmus für Matroide als ein Verfahren zur Lösung spezieller linearer Programme deuten kann. Er liefert in der Tat auch duale Lösungen. Wir gehen darauf in Abschnit 2.6 ein.

Da der Algorithmus (5.7) GREEDY-MAX aus ADM I offenbar eine Spezialisierung des Greedy-Algorithmus (2.13) für das graphische Matroid ist, liefert Folgerung (2.15) einen weiteren Beweis von ADM I, Satz (5.8).

Satz (2.14) ist ein Prototyp von Abschätzungssätzen für die Qualität von heuristischen Lösungen, wie wir sie später noch mehrfach kennenlernen werden. Der Greedy-Algorithmus (2.13) ist offenbar orakelpolynomial. Immer dann, wenn wir den Orakelaufruf (Unabhängigkeitstest) in Schritt 3 durch einen polynomialen Algorithmus realisieren können, ist der Greedy-Algorithmus ein polynomialer Algorithmus (im üblichen Sinne). Eine solche polynomiale Realisierung ist z. B. auf triviale Weise für das Cliquenproblem, Stabile-Menge-Problem, Travelling-Salesman-Problem und das azyklische Subdigraphenproblem möglich. Würde der Greedy-Algorithmus auch in diesen Fällen immer Optimallösungen liefern, hätten wir einen Beweis für $\mathcal{P} = \mathcal{NP}$ gefunden, da alle gerade aufgelisteten

Probleme $\mathcal{NP}$-schwer sind. Wir können also nicht erwarten, dass der Greedy-Algorithmus immer optimale Antworten produziert. Die Frage, wie schlecht im schlechtest möglichen Falle eine Greedy-Lösung ist, beantwortet Satz (2.14) auf bestmögliche Weise. Er gibt eine sogenannte *Gütegarantie* an, die besagt, dass der Wert $c(I_g)$ jeder Greedy-Lösung nicht schlechter ist als $qc(I_o)$, wobei q der bereits erwähnte Rangquotient ist. Der Rangquotient liefert also eine Gütegarantie für die Lösungsqualität des Greedy-Algorithmus. Der Rangquotient ist im allgemeinen nicht einfach auszurechnen. Eine Abschätzung wird gegeben durch:

(2.16) Satz. *Sei $\mathcal{I}$ ein Unabhängigkeitssystem auf E, und $(E, \mathcal{I}_i)$, $i = 1, \ldots, k$ sei eine minimale Zahl von Matroiden mit $\mathcal{I} = \bigcap_{i=1}^{k} \mathcal{I}_i$, dann gilt*

$$\min_{F \subseteq E, r(F)>0} \frac{r_u(F)}{r(F)} \geq \frac{1}{k}.$$ △

Daraus folgt zum Beispiel: Ist $\mathcal{I}$ ein Unabhängigkeitssystem, das Durchschnitt von 2 Matroiden ist, dann ist der Wert der Greedy-Lösung mindestens halb so groß wie der Wert der Optimallösung von (2.3). Insbesondere liefert also der Greedy-Algorithmus für Branchingprobleme Lösungen, deren Wert mindestens die Hälfte des Optimums beträgt.

Für das Branching-Problem gibt es einen polynomialen Lösungsalgorithmus. Es gibt also offenbar auch Probleme über Unabhängigkeitssystemen, die keine Matroide sind und für die effiziente Optimierungsverfahren existieren. Ein sehr tiefliegendes Resultat ist der folgende von Edmonds (1979) und Lawler (1975) gefundene Satz.

(2.17) Satz. *Seien $(E, \mathcal{I}_1)$ und $(E, \mathcal{I}_2)$ zwei Matroide gegeben durch Unabhängigkeitsorakel, dann gibt es einen Algorithmus, der für jede beliebige Zielfunktion c das Problem $\max\{c(I) \mid I \in \mathcal{I}_1 \cap \mathcal{I}_2\}$ in orakelpolynomialer Zeit löst.* △

Der Algorithmus, der den Beweis des obigen Satzes liefert, ist relativ kompliziert, und sein Korrektheitsbeweis benötigt Hilfsmittel aus der Matroidtheorie, die uns hier nicht zur Verfügung stehen. Deshalb verzichten wir auf eine Angabe und Analyse dieses (sehr interessanten) Verfahrens.

Man fragt sich nun natürlich sofort, ob auch über dem Durchschnitt von 3 Matroiden in (orakel-)polynomialer Zeit optimiert werden kann. Dies geht (vermutlich) i. A. nicht. Man kann zeigen, dass das Optimierungsproblem für Unabhängigkeitssysteme, die Durchschnitte von 3 Matroiden sind, Probleme enthält, die $\mathcal{NP}$-schwer sind.

Ein Beispiel hierfür ist das asymmetrische Travelling-Salesman-Problem (ATSP). Das zugehörige Unabhängigkeitssystem $\mathcal{H}$ ist wie folgt definiert:

$$\mathcal{H} := \{S \subseteq A_n \mid S \text{ ist Teilmenge eines gerichteten hamiltonschen Kreises in } D_n = (V, A_n)\},$$

wobei D_n der vollständige gerichtete Graph auf n Knoten ist. Wir modifizieren D_n zu dem Graphen $D'_n = (V', A'_n)$, indem wir den Knoten $1 \in V$ in zwei neue Knoten $1, n+1 \in V'$ aufspalten. Alle Bögen aus A_n, die den Knoten 1 aus D_n als Anfangsknoten haben, haben

auch $1 \in V'$ als Anfangsknoten; die Bögen aus A_n, die $1 \in V$ als Endknoten haben, bekommen den neuen Knoten $n+1 \in V'$ als Endknoten zugewiesen. Diese Konstruktion bewirkt, dass jedem gerichteten hamiltonschen Kreis in D_n ein gerichteter hamiltonscher Weg von 1 nach $n+1$ in D'_n entspricht und umgekehrt. Das ATSP-Unabhängigkeitssystem $\mathcal{H}$ ist dann (bis auf Benamung einiger Knoten und Bögen) identisch mit

$$\mathcal{H}' := \{S \subseteq A'_n \mid S \text{ ist Teilmenge eines gerichteten hamiltonschen Wegs von 1 nach } n+1 \text{ in } D'_n\}.$$

Definieren wir auf A'_n die drei folgenden Unabhängigkeitssysteme

$$\begin{aligned} \mathcal{W} &:= \{W \subseteq A'_n \mid W \text{ ist Wald }\}, \\ \mathcal{P}^- &:= \{F \subseteq A'_n \mid |F \cap \delta^-(v)| \leq 1 \; \forall v \in V'\}, \\ \mathcal{P}^+ &:= \{F \subseteq A'_n \mid |F \cap \delta^+(v)| \leq 1 \; \forall v \in V'\}, \end{aligned}$$

so sind $\mathcal{W}$, $\mathcal{P}^-$, $\mathcal{P}^+$ Unabhängigkeitssysteme von Matroiden auf A'_n, und es gilt $\mathcal{H}' = \mathcal{W} \cap \mathcal{P}^- \cap \mathcal{P}^+$.

Wir wollen uns nun noch kurz dem Optimierungsproblem (2.4) über Basissystemen zuwenden. Wie bei Bäumen und Wäldern vorgeführt (siehe ADM I, Abschnitt 5.2), kann man ein Maximierungsproblem des Typs (2.3) mit Hilfe einer polynomialen Transformation in ein Minimierungsproblem (2.4) überführen und umgekehrt. Analog lässt sich auch aus (2.13) ein Minimierungsalgorithmus ableiten.

(2.18) Algorithmus Greedy-min für Basissysteme.
Eingabe: Grundmenge $E = \{1, \ldots, n\}$ mit Gewichten $c_i \in \mathbb{R}$ für alle $i \in E$, ein Unabhängigkeitssystem $\mathcal{I} \subseteq 2^E$ gegeben durch ein Unabhängigkeitsorakel.
Ausgabe: Eine Basis $B_g \in \mathcal{I}$.

1. Sortiere die Gewichte in nicht absteigender Reihenfolge (nach Beendigung von Schritt 1 gelte $c_1 \leq c_2 \leq \cdots \leq c_n$).
2. Setze $I := \emptyset$.
3. FOR $i = 1$ TO n DO:
   Ist $I \cup \{i\} \in \mathcal{I}$ (Orakelaufruf), dann setze $I := I \cup \{i\}$.
4. Setze $B_g := I$ und gib B_g aus. △

Offenbar ist B_g eine Basis der Grundmenge E des Unabhängigkeitssystems $\mathcal{I}$, gehört also zum zugehörigen Basissystem $\mathcal{B}$. Folgerung (2.15) impliziert:

(2.19) Satz. *Sei $\mathcal{I}$ ein Unabhängigkeitssystem auf E und $\mathcal{B}$ das zugehörige Basissystem. Dann sind äquivalent:*

(a) $(E, \mathcal{I})$ *ist ein Matroid.*

(b) *Für alle Gewichte* $c \in \mathbb{R}^E$ *liefert der Greedy-Algorithmus* (2.18) *eine Optimallösung von* $\min\{c(B) \mid B \in \mathcal{B}\}$.

(c) *Für alle Gewichte* $c \in \{0,1\}^E$ *liefert der Greedy-Algorithmus* (2.18) *eine Optimallösung von* $\min\{c(B) \mid B \in \mathcal{B}\}$. △

Sortiert man in Schritt 1 von (2.18) die Gewichte in nicht aufsteigender Reihenfolge, so erhält man offenbar eine Basis maximalen Gewichts. Satz (2.19) gilt analog, wenn man „min“ durch „max“ ersetzt.

Leider kann man die schöne Abschätzung aus Satz (2.14) für die Gütegarantie von (2.13) nicht ohne weiteres auf Algorithmus (2.18) übertragen. In der Tat gibt es keine universelle Konstante, die bezüglich eines Problems die Qualität des Ergebnisses von (2.18) unabhängig von den Zielfunktionskoeffizienten beschränkt. Betrachten wir z. B. das Cliquen-Problem auf dem in Abbildung 2.8 dargestellten Graphen mit der Gewichts-

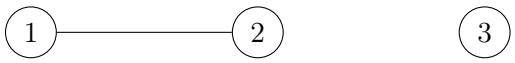


Abbildung 2.8: Graph, auf dem der Greedy-Algorithmus für das Cliquen-Problem beliebig schlechte Lösungen liefert

funktion $c_1 = 1$, $c_2 = M > 2$, $c_3 = 2$. Das zugehörige Unabhängigkeitssystem hat die Basen $\{1,2\}$ und $\{3\}$. Bei Anwendung von (2.18) erhalten wir immer die Greedylösung $B_g = \{1,2\}$ mit Gewicht $1 + M$, während die Optimalbasis B_0 das Gewicht 2 hat. Der Quotient aus $c(B_g)$ und $c(B_0)$ geht also gegen ∞ falls $M \to \infty$. Denken Sie darüber nach, was diese (triviale) Beobachtung über die in Satz (2.14) angegebene Gütegarantie aussagt!

2.5 Ein primal-dualer Greedy-Algorithmus

Die Matroidtheorie ist aufgrund ihres technischen Apparates besonders gut dazu geeignet, eine gemeinsame Verallgemeinerung des primalen und des dualen Greedy-Algorithmus zur Berechnung minimaler aufspannender Bäume (siehe (5.9) und (5.11) im Skript zur ADM I) zu formulieren. Grundlegend hierfür ist der folgende

(2.20) Satz (und Definition). *Sei* M *ein Matroid auf einer Grundmenge* E *mit Basissystem* $\mathcal{B}$. *Setze* $\mathcal{B}^* := \{E \setminus B \mid B \in \mathcal{B}\}$. *Dann ist* $\mathcal{B}^*$ *das Basissystem eines Matroids* M^* *auf* E. M^* *wird das zu* M duale Matroid *genannt.* △

Offensichtlich gilt für ein Basissystem

$$\mathcal{B}^{**} = \mathcal{B},$$

d. h. das zu M^* duale Matroid ist das Matroid M. Wir haben bereits ein Matroid und das dazu duale Matroid kennengelernt. Ist nämlich M das graphische Matroid auf einem

Graphen G (siehe (2.8)(a)), so ist das cographische Matroid auf G (siehe (2.9)(a)) das zu M duale Matroid M^*.

Für Matrix-Matroide (siehe (2.8)(b)) kann man auf einfache Weise das duale Matroid konstruieren. Wir nehmen an, dass M durch die (m,n)-Matrix A (über irgendeinem Körper) definiert ist. O. B. d. A. können wir annehmen, dass A vollen Zeilenrang m hat. Mit Hilfe der Gauß-Elimination bringen wir A in **Standardform**, d. h. wir formen A so um, dass $A = (I, B)$ gilt, wobei I die (m,m)-Einheitsmatrix ist. (Möglichweise sind hierzu Zeilen- und Spaltenvertauschungen erforderlich.) Man kann nun zeigen, dass das zu M duale Matroid M^* durch die Matrix

$$(-B^T, I)$$

definiert ist. Hierbei ist I eine $(n-m, n-m)$-Einheitsmatrix. Das vor Satz (2.10) beschriebene aus der Geometrie stammende Fano-Matroid F_7 hat folglich als duales Matroid das Matroid, das durch die folgende Matrix

$$\begin{array}{c} \\ 4 \\ 5 \\ 6 \\ 7 \end{array} \begin{array}{c} \begin{array}{ccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

über den Körper $GF(2)$ gegeben ist. (Man beachte, dass $-1 = 1$ in $GF(2)$ gilt.) Man nennt es das *duale Fano-Matroid* und bezeichnet es mit F_7^*.

In der Matroidtheorie hat sich die folgende Sprechweise eingebürgert. Ist M ein Matroid und A eine Basis oder ein Zirkuit des dualen Matroids M^*, so nennt man A auch eine *Cobasis* oder ein *Cozirkuit* von M. Analog heißt der Rang von A bezüglich der Rangfunktion von M^* auch der *Corang* von A. Ein *Zyklus* ist die Vereinigung von paarweise disjunkten Zirkuits, analog ist ein *Cozyklus* die Vereinigung paarweise disjunkter Cozirkuits. Die nachfolgenden (einfachen) Beobachtungen sind für die anschließenden Betrachtungen wichtig.

(2.21) Satz.

(a) *Ein System $\mathcal{C}$ von Teilmengen von E ist genau dann die Zirkuitmenge eines Matroids M auf E, wenn $\mathcal{C}$ genau die minimalen, nichtleeren Teilmengen $C \subseteq E$ enthält, so dass gilt: $|C \cap C^*| \neq 1$ für alle Cozirkuits C^* von M.*

(b) *Sei B eine Basis (Cobasis) eines Matroids M, und sei $e \in E \setminus B$, dann enthält $B \cup \{e\}$ genau einen Zirkuit (Cozirkuit) C_e, der e enthält. C_e heißt der durch e und B erzeugte* fundamentale Zirkuit (Cozirkuit). △

(2.22) Satz. *Sei M ein Matroid auf E mit Gewichten c_e für alle $e \in E$. Sei B eine Basis von M. Dann sind die folgenden Aussagen äquivalent:*

(i) *B ist eine gewichtsminimale Basis.*

(ii) *Für jedes Element* $e \in E \setminus B$ *gilt* $c_e \geq c_f$ *für alle* f *aus dem von* e *und* B *erzeugten fundamentalen Zirkuit* K_e.

(iii) *Für jeden Cozirkuit* C *gilt* $\min\{c_e \mid e \in C\} = \min\{c_e \mid e \in B \cap C\}$

(iv) $E \setminus B$ *ist eine gewichtsmaximale Cobasis.*

(v) *Für jedes Element* $e \in B$ *gilt* $c_e \leq c_f$ *für alle* f *aus dem von* e *und* $E \setminus B$ *erzeugten fundamentalen Cozirkuit* C_e.

(vi) *Für jeden Zirkuit* K *gilt* $\max\{c_e \mid e \in K\} = \max\{c_e \mid e \in (E \setminus B) \cap K\}$. △

Beweis. (i) $\iff$ (iv). Trivial, nach Definition.

(i) $\Rightarrow$ (ii). Angenommen, es existieren $e \in E \setminus B$ und $f \in K_e$ mit $c_e < c_f$. Dann ist $(B \setminus \{f\}) \cup \{e\}$ eine Basis mit kleinerem Gewicht als B. Widerspruch!

(ii) $\Rightarrow$ (iii). Angenommen, es existieren ein Cozirkuit C und $e \in C \setminus B$ mit $c_e < \min\{c_f \mid f \in B \cap C\}$. Sei K_e der durch e mit B erzeugte fundamentale Zirkuit. Dann existiert nach (2.21)(a) ein Element $f \in (K_e \setminus \{e\}) \cap C$. Aber dann gilt für $e \in E \setminus B$ und $f \in B$: $c_e < c_f$. Widerspruch!

(iii) $\Rightarrow$ (i). Sei B' eine gewichtsminimale Basis, so dass $|B \cap B'|$ so groß wie möglich ist. Angenommen, es existiert ein Element $f \in B' \setminus B$. Sei C_f der von f mit $E \setminus B'$ erzeugte fundamentale Cozirkuit. Da B eine Basis ist, gilt $B \cap C_f \neq \emptyset$. Sei $g \in B \cap C_f$ mit $c_g = \min\{c_e \mid e \in B \cap C_f\}$. Nach (iii) gilt wegen $f \in C_f$: $c_f \geq c_g$. Nun aber ist $B'' := (B' \setminus \{f\}) \cup \{g\}$ eine Basis von M mit $c(B'') \leq c(B')$ und $|B \cap B''| > |B \cap B'|$. Widerspruch! Daraus folgt $B = B'$, und somit ist B gewichtsminimal.

(iv) $\Rightarrow$ (v) $\Rightarrow$ (vi) $\Rightarrow$ (iv) folgt aus Dualitätsgründen. □

(2.23) Algorithmus Primal-dualer Greedy-Algorithmus.
Eingabe: Grundmenge E mit Gewichten c_e für alle $e \in E$ und ein Matroid $M = (E, \mathcal{I})$.
Ausgabe: Eine gewichtsminimale Basis B von M und eine gewichtsmaximale Cobasis B^* von M.

(Terminologie: Wir sagen, dass eine Menge S durch eine Menge F *überdeckt* ist, falls $F \cap S \neq \emptyset$.)

1. Setze $B := B^* := \emptyset$.
2. Führe einen (beliebigen) der beiden folgenden Schritte 3 oder 4 aus:
3. Primaler Schritt
   3.1. Sind alle Cozirkuits von M durch B überdeckt, STOP.
   (Gib B und $B^* := E \setminus B$ aus.)
   3.2. Wähle einen Cozyklus $C \neq \emptyset$ von M, der nicht durch B überdeckt ist.

3.3. Bestimme $f \in C$ mit $c_f = \min\{c_e \mid e \in C\}$.

3.4. Setze $B := B \cup \{f\}$ und gehe zu 2.

4. Dualer Schritt

4.1 Sind alle Zirkuits von M durch B^* überdeckt, STOP.
(Gib B^* und $B := E \setminus B^*$ aus.)

4.2 Wähle einen Zyklus $Z \neq \emptyset$ von M, der nicht durch B^* überdeckt ist.

4.3 Bestimme $g \in Z$ mit $c_g = \max\{c_e \mid e \in Z\}$.

4.4 Setze $B^* := B^* \cup \{g\}$ und gehe zu 2. △

Will man Algorithmus (2.23) implementieren, so muss entweder für Schritt 3.1 oder für Schritt 4.1 (oder für die beiden Schritte) ein Algorithmus vorhanden sein, der die verlangten Überprüfungen durchführt. Wir gehen hier (in der Theorie) davon aus, dass ein Orakel die Aufgabe für uns erledigt. Ob das Orakel effizient implementierbar ist, hängt natürlich vom Matroidtyp ab, den man behandeln will.

(2.24) Satz. *Der obige Algorithmus* (2.23) *funktioniert.* △

Beweis. Wir beweisen durch Induktion nach $|B| + |B^*|$, dass die jeweils während des Ablaufs des Verfahrens konstruierten Mengen B und B^* in einer gewichtsminimalen Basis bzw. gewichtsmaximalen Cobasis enthalten sind.

Für den Induktionsanfang $B = B^* = \emptyset$ ist die Behauptung trivial.

Wir nehmen an, dass der Satz gilt, wenn $|B| + |B^*| \leq k$, d. h. wenn wir Schritt 2 höchstens k-mal ausgeführt haben. Wir beginnen jetzt mit der $(k+1)$-ten Ausführung von Schritt 2 und entscheiden uns, Schritt 3 durchzuführen. Sei B die bisher konstruierte Menge und B' eine gewichtsminimale Basis mit $B \subseteq B'$. Wir wählen in 3.2 einen Cozyklus C und in 3.3 ein Element $f \in C$. Gilt $f \in B'$, so ist alles klar. Anderenfalls sei $g \in C \cap B'$ ein Element, das mit f auf einem Cozirkuit liegt. Dann muss wegen 3.3 gelten $c_g \geq c_f$. $B'' := (B' \setminus \{g\}) \cup \{f\}$ ist damit eine Basis mit $c(B'') \leq c(B')$ und $\{f\} \cup B \subseteq B''$. Also ist $B \cup \{f\}$ in einer gewichtsminimalen Basis enthalten.

Entscheiden wir uns in 2 für die Durchführung von Schritt 4, so folgt die Behauptung analog.

Stellen wir in 3.1 fest, dass B eine Basis ist, so ist nach Induktion $c(B)$ minimal und $E \setminus B$ eine maximale Cobasis. Analog schließen wir im Falle, dass B^* in 4.1 als Cobasis erkannt wird. □

Algorithmus (2.23) ist aufgrund der vielen Freiheitsgrade ein äußerst flexibel einsetzbares Instrument zur Konstruktion optimaler Basen und Cobasen. Durch geeignete Spezialisierung erhält man alle in Kapitel 5 des Skripts zur ADM I vorgestellten Algorithmen zur Bestimmung minimaler Bäume und einige weitere derartige Verfahren.

2.6 Polyedrische und LP/IP-Aspekte

Wir wenden uns nun LP/IP-Aspekten zu und werden untersuchen, wie man Optimierungsprobleme über Unabhängigkeitssystemen als ganzzahlige Programme formulieren kann. Ein besonderes Highlight wird die vollständige Charakterisierung von Matroid-Polytopen sein.

Wir beginnen mit einem beliebigen Unabhängigkeitssystem $(E, \mathcal{I})$ auf einer endlichen Grundmenge E. Jedem Element $e \in E$ ordnen wir wie üblich eine Variable x_e zu. Nimmt x_e den Wert 1 an, so interpretieren wir dies als Wahl des Elements $e \in E$, falls $x_e = 0$, so entscheiden wir uns gegen die Wahl von E. Für jede Teilmenge $F \subseteq E$ führen wir wie üblich den Inzidenzvektor $\chi^F = (\chi^F_e)_{e \in E} \in \mathbb{K}^E$ ein, dessen Komponenten wie folgt definiert sind: $\chi^F_e = 1$, falls $e \in F$ und $\chi^F_e = 0$, falls $e \notin F$.

Dem Unabhängigkeitssystem $\mathcal{I}$ ordnen wir ein Polytop $\mathrm{IND}(\mathcal{I}) \subseteq \mathbb{K}^E$ wie folgt zu:

$$\mathrm{IND}(\mathcal{I}) := \mathrm{conv}\{\chi^I \in \mathbb{K}^E \mid I \in \mathcal{I}\}. \tag{2.25}$$

Ist $(E, \mathcal{I})$ ein Matroid, so nennt man $\mathrm{IND}(\mathcal{I})$ *Matroid-Polytop*.

Offensichtlich kann man bei gegebener linearer Zielfunktion $c \in \mathbb{K}^E$ das kombinatorische Optimierungsproblem $\max\{c(I) \mid I \in \mathcal{I}\}$ als lineares Programm $\max\{c^T x \mid x \in \mathrm{IND}(\mathcal{I})\}$ lösen. (Jedes LP über dem Polytop $\mathrm{IND}(\mathcal{I})$ hat eine optimale Ecklösung, und diese ist nach Definition der Inzidenzvektor einer unabhängigen Menge $I \in \mathcal{I}$.) Um LP-Methoden anwenden zu können, müssen wir jedoch lineare Ungleichungen finden, die $\mathrm{IND}(\mathcal{I})$ vollständig (oder zumindest partiell) beschreiben.

Zunächst entledigen wir uns einer Trivialität. In einem Unabhängigkeitssystem kann es Elemente $e \in E$ geben, so dass $\{e\}$ keine unabhängige Menge ist. Solche Elemente heißen *Schlingen*, *Schleifen* oder *Loops*. Sie können niemals in einer optimalen unabhängigen Menge enthalten sein. Wir nehmen daher im Folgenden o. B. d. A. an, dass $(E, \mathcal{I})$ keine Schlingen enthält. (In der Literatur nennt man solche Unabhängigkeitssysteme *normal*.)

Die folgende einfache Überlegung führt uns auf die Spur eines wichtigen Ungleichungssystems für $\mathrm{IND}(\mathcal{I})$. Ist $F \subseteq E$, so ist der Rang $r(F)$ von F die größte Kardinalität einer in F enthaltenen unabhängigen Menge (falls $F \in \mathcal{I}$, so gilt natürlich $r(F) = |F|$). Für jede unabhängige Menge $I \in \mathcal{I}$ ist $F \cap I \in \mathcal{I}$, und daher gilt $|F \cap I| \leq r(F)$. Daraus folgt, dass jeder Inzidenzvektor χ^I einer unabhängigen Menge I die Ungleichung (genannt *Rang-Ungleichung*)

$$\sum_{e \in F} x_e = x(F) \leq r(F) \tag{2.26}$$

erfüllt. Dabei ist $x(\{e\}) \leq r(\{e\})$ nichts anderes als die triviale Ungleichung $x_e \leq 1$, für alle $e \in E$.

(2.27) Satz. *Seien $(E, \mathcal{I})$ ein Unabhängigkeitssystem und $c \in \mathbb{K}^E$. Dann gilt:*

(a) $\mathrm{IND}(\mathcal{I}) \subseteq P(\mathcal{I}) := \{x \in \mathbb{K}^E \mid x(F) \leq r(F)\ \forall F \subseteq E,\ x_e \geq 0\ \forall e \in E\}$.

(b) $\max\{c(I) \mid I \in \mathcal{I}\} = \max \quad c^T x$

$$\begin{aligned} \text{s.t.} \quad x(F) &\leq r(F) \quad \forall F \subseteq E \\ x_e &\geq 0 \quad \forall e \in E \\ x_e &\in \{0,1\} \quad \forall e \in E. \end{aligned}$$

△

Beweis. Wie bereits erläutert, erfüllt jeder Vektor χ^I, $I \in \mathcal{I}$, das obige Ungleichungssystem, somit ist jede Ecke von $\text{IND}(\mathcal{I})$ in $P(\mathcal{I})$ enthalten, und daher gilt $\text{IND}(\mathcal{I}) \subseteq P(\mathcal{I})$; (a) ist damit bewiesen.

Um (b) zu zeigen, müssen wir beweisen, dass die Vektoren χ^I, $I \in \mathcal{I}$ die einzigen ganzzahligen Vektoren in $P(\mathcal{I})$ sind. Sei also $y \in P(\mathcal{I})$ ganzzahlig, dann muss y ein 0/1-Vektor und somit Inzidenzvektor einer Teilmenge $F \subseteq E$ sein. Ist $F \notin \mathcal{I}$, so gilt $r(F) < |F| = \sum_{e \in F} y_e$ und somit erfüllt y nicht die Ungleichung $x(F) \leq r(F)$. Also muss F unabhängig sein, und der Beweis ist erledigt. □

Satz (2.27)(b) gibt uns somit die Möglichkeit, beliebige lineare Optimierungsprobleme über Unabhängigkeitssystemen als ganzzahlige Programme zu lösen. Natürlich ist zu bemerken, dass das System aus Satz (2.27) insgesamt $2^m + m$ Ungleichungen enthält ($|E| = m$). Dieses Problem werden wir später behandeln.

Ist das System vollständig, d. h., gilt $\text{IND}(\mathcal{I}) = P(\mathcal{I})$? Wir betrachten dazu das Stabile-Mengen-Problem auf dem Graphen $G = (V, E)$ aus Abbildung 2.9. Das Unabhängigkeits-

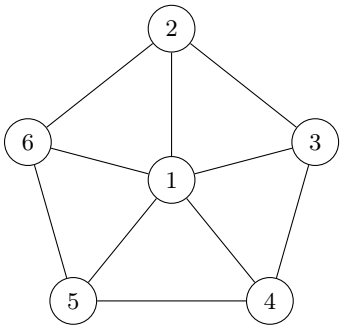


Abbildung 2.9: Graph mit 6 Knoten

system ist auf der Knotenmenge $V = \{1, \ldots, 6\}$ definiert. Die stabilen Knotenmengen bilden das System unabhängiger Mengen $\mathcal{I} \subseteq 2^V$. Offensichtlich sind nur die leere Menge, die Mengen $\{v\}$, $v = 1, \ldots, 6$ und $\{2,4\}, \{2,5\}, \{3,5\}, \{3,6\}, \{4,6\}$ stabile Knotenmengen. Betrachten wir die Zielfunktion $c \in \mathbb{K}^6$ mit $c_1 = 2$, $c_i = 1$, $i = 2, \ldots, 6$, so hat das Optimum von $c^T x$ über $\text{IND}(\mathcal{I})$ offenbar den Wert 2. Der Vektor $x^T = (x_1, \ldots, x_6)$ mit $x_1 = \cdots = x_6 = \frac{1}{3}$ hat den Zielfunktionswert $\frac{7}{3}$ und erfüllt offenbar alle Rangungleichungen, ist also in $P(\mathcal{I})$ enthalten. Daraus folgt $P(\mathcal{I}) \neq \text{IND}(\mathcal{I})$ für dieses Unabhängigkeitssystem.

Wir beweisen nun, dass die Rangungleichungen zur Beschreibung von Matroid-Polytopen ausreichen.

(2.28) Satz. *Sei $(E, \mathcal{I})$ ein Matroid, dann gilt:*

$$P(\mathcal{I}) = \mathrm{IND}(\mathcal{I}). \qquad \triangle$$

Beweis. Nach Annahme ist $(E, \mathcal{I})$ normal. Damit enthält $\mathrm{IND}(\mathcal{I})$ neben dem Nullvektor $\chi^{\emptyset}$ auch alle Einheitsvektoren $\chi^{\{e\}}$, $e \in E$. Mithin ist $\mathrm{IND}(\mathcal{I})$ volldimensional ($\dim \mathrm{IND}(\mathcal{I}) = |E|$) und alle Facetten definierenden Ungleichungen von $\mathrm{IND}(\mathcal{I})$ sind bis auf positive Vielfache eindeutig bestimmt, siehe Korollar (1.40).

Sei nun $a^T x \leq b$ eine Ungleichung, die eine Facette $H := \mathrm{IND}(\mathcal{I}) \cap \{x \in \mathbb{K}^E \mid a^T x = b\}$ von $\mathrm{IND}(\mathcal{I})$ definiert. Wir werden zeigen, dass dann $a^T x \leq b$ ein positives Vielfaches von einer der Ungleichungen $-x_e \leq 0$, $e \in E$, oder $x(F) \leq r(F)$, $F \subseteq E$ ist. Das bedeutet, dass das $P(\mathcal{I})$ definierende System alle Ungleichungen enthält, die Facetten von $\mathrm{IND}(\mathcal{I})$ definieren, und somit gilt $P(\mathcal{I}) = \mathrm{IND}(\mathcal{I})$.

Wir nehmen zunächst an, dass ein Koeffizient von a negativ ist, sagen wir $a_e < 0$, $e \in E$. Gibt es eine unabhängige Menge $I \in \mathcal{I}$ mit $e \in I$ und $a^T \chi^I = b$, dann gilt für die ebenfalls unabhängige Menge $J := I \setminus \{e\}$: $a^T \chi^J = a^T \chi^I - a_e = b - a_e > b$. Dies widerspricht der Gültigkeit von $a^T x \leq b$. Folglich gilt $x_e = 0$ für alle Punkte der Facette H und damit, dass

$$H \;\subseteq\; \mathrm{IND}(\mathcal{I}) \cap \{x \in \mathbb{K}^E \mid x_e = 0\} \;=:\; G.$$

Da H eine maximale Seitenfläche von $\mathrm{IND}(\mathcal{I})$ ist, folgt $H = G$, und aus (1.40) folgt somit, dass $a^T x \leq b$ ein positives Vielfaches von $-x_e \leq 0$ ist.

Nehmen wir nun an, dass $a_e \geq 0$ für alle $e \in E$ gilt. Setze $F := \{e \in E \mid a_e > 0\}$. Wir behaupten nun, dass $a^T x \leq b$ ein positives Vielfaches von $x(F) \leq r(F)$ ist. Um das zu beweisen, zeigen wir, dass, falls $I \in \mathcal{I}$ und $a^T \chi^I = b$ gilt, dann gilt auch $\chi^I(F) = r(F)$, d. h. dass $I \cap F$ eine Basis von F ist. Hieraus folgt wie oben die Behauptung.

Seien also $I \in \mathcal{I}$ und $a^T \chi^I = b$. Angenommen es gibt ein Element $e \in F \setminus I$, sodass $J := (I \cap F) \cup \{e\} \in \mathcal{I}$, dann gilt $a^T \chi^J = a^T \chi^I + a_e > a^T \chi^I = b$. Dies widerspricht der Gültigkeit von $a^T x \leq b$. Da das Hinzufügen eines jeden Elements $e \in F \setminus I$ zu $F \cap I$ eine abhängige Menge liefert, ist $I \cap F$ eine Basis von F und somit erfüllt I die Rangungleichung $x(F) \leq r(F)$ mit Gleichheit. □

Man kann sogar die Facetten von Matroid-Polytopen charakterisieren. Wir benötigen hierzu zwei Begriffe. Eine Teilmenge $F \subseteq E$ heißt *abgeschlossen*, wenn

$$r(F \cup \{e\}) > r(F) \qquad \forall e \in E \setminus F$$

gilt. F heißt *separabel*, wenn es zwei Teilmengen $F_1, F_2 \subseteq F$ mit $F_1 \cup F_2 = F$ und $F_1 \cap F_2 = \emptyset$ gibt, sodass

$$r(F) = r(F_1) + r(F_2),$$

anderenfalls heißt F *inseparabel*.

(2.29) Satz. *Sei $(E, \mathcal{I})$ ein normales Matroid.*

(a) *Eine Rangungleichung $x(F) \leq r(F)$, $F \subseteq E$ definiert eine Facette von* $\mathrm{IND}(\mathcal{I})$ *genau dann, wenn F abgeschlossen und inseparabel ist.*

(b) *Das folgende System von Ungleichungen ist eine vollständige und irredundante Beschreibung des Matroid-Polytops* $\mathrm{IND}(\mathcal{I})$*:*

$$\begin{aligned} x(F) &\leq r(F) && \forall F \subseteq E,\ F \text{ abgeschlossen und inseparabel} \\ x_e &\geq 0 && \forall e \in E. \end{aligned}$$ △

Der Beweis von Satz (2.29) ist umfangreich und kann aus Zeitgründen hier nicht vorgeführt werden.

(2.30) Bemerkung.

(a) Sei $G = (V, E)$ ein schlingenfreier Graph. Eine Teilmenge $F \subseteq E$ ist abgeschlossen und inseparabel im graphischen Matroid auf E genau dann, wenn entweder $F = \{e\}$, $e \in E$ gilt oder wenn F die Kantenmenge eines knoteninduzierten Untergraphen $(W, E(W))$ mit $|W| \geq 3$ ist, der 2-fach knotenzusammenhängend ist.

(b) Sei $(E, \mathcal{I})$ das Partitionsmatroid auf E, das definiert ist durch $E_1, \ldots, E_k \subseteq E$ und $b_1, \ldots, b_k$ mit $1 \leq b_i < |E_i|$ für alle $i = 1, \ldots, k$, siehe (2.9)(c). $F \subseteq E$ ist genau dann abgeschlossen und inseparabel in $\mathcal{I}$, wenn $F = E_i$ für ein $i \in \{1, \ldots, k\}$ gilt. △

Beweis. Übungsaufgabe. □

Eine Anwendung von Satz (2.29) unter der Berücksichtigung der Erkenntnisse in (2.30) ist der folgende Satz.

(2.31) Satz.

(a) *Das Matroid-Polytop eines graphischen Matroids auf einem schlingenfreien Graphen $G = (V, E)$ ist vollständig und irredundant charakterisiert durch das folgende Ungleichungssystem:*

$$\begin{aligned} x(E(W)) &\leq |W| - 1 && \forall\, W \subseteq V, |W| \geq 3, \\ & && \quad (W, E(W)) \text{ 2-fach knotenzusammenhängend} \\ 0 \leq x_e &\leq 1 && \forall\, e \in E. \end{aligned}$$

(b) *Das Matroid-Polytop eines Partitionsmatroids auf einer endlichen Menge E, gegeben durch $E_1, \ldots, E_k$ und $b_1, \ldots, b_k$ mit $1 \leq b_i < |E_i|$ für alle $i = 1, \ldots, k$, ist vollständig und irredundant charakterisiert durch das folgende Ungleichungssystem:*

$$\begin{aligned} x(E_i) &\leq b_i && i = 1, \ldots, k \\ x_e &\leq 1 && \forall\, e \in E_i \text{ mit } b_i \geq 2 \\ x_e &\geq 0 && \forall\, e \in E_i. \end{aligned}$$ △

Satz (2.31) zeigt eine mögliche dramatische Reduktion der Anzahl der zur Beschreibung eines Matroid-Polytops benötigten Ungleichungen. In Satz (2.28) wird bewiesen, dass die in Satz (2.27) angegebenen Ungleichungen zur Beschreibung ausreichen. Die Anzahl dieser Ungleichungen ist $2^m + m$, mit $m = |E|$. Satz (2.31) zeigt, dass diese Zahl bei vollständigen Graphen auf ungefähr 2^n, $n = |V|$ reduziert werden kann. Sie ist jedoch immer noch exponentiell in der Kodierungslänge des Graphen. Bei „sehr dünnen“ Graphen ist die Anzahl der Ungleichungen erheblich kleiner. Bei einem Partitionsmatroid werden statt $2^m + m$ höchstens $k + 2m$ Ungleichungen benötigt.

Es gibt noch eine erstaunliche Verschärfung von Satz (2.29), die wir ebenfalls ohne Beweis angeben.

(2.32) Satz. *Sei E eine endliche Menge. $(E, \mathcal{I}_1)$ und $(E, \mathcal{I}_2)$ seien zwei Matroide auf E. Das Polytop des Durchschnitts dieser beiden Matroide*

$$P(\mathcal{I}_1 \cap \mathcal{I}_2) = \operatorname{conv}\{\chi^I \in \mathbb{K}^E \mid I \in \mathcal{I}_1 \cap \mathcal{I}_2\}$$

wird durch das folgende System von Ungleichungen vollständig beschrieben:

$$\begin{aligned} x(F) &\leq \min\{r_1(F), r_2(F)\} && \forall F \subseteq E \\ x_e &\geq 0 && \forall e \in E, \end{aligned}$$

wobei $r_i(F)$ die Rangfunktion von $(E, \mathcal{I}_i)$, $i = 1, 2$ bezeichnet. △

An dieser Stelle sollen noch einmal Satz (2.14) und der Beweis aufgenommen und mit den „LP-Erkenntnissen“ dieses Abschnitts verbunden werden. Für ein beliebiges Unabhängigkeitssystem $(E, \mathcal{I})$ und eine beliebige Zielfunktion $c \in \mathbb{K}^E$ bezeichnet I_0 eine optimale Lösung des so definierten Optimierungsproblems $\max\{c(I) \mid I \in \mathcal{I}\}$, I_g bezeichnet die Greedy-Lösung, $E_1, \ldots, E_n$ seien die im Beweis von (2.14) definierten Mengen $E_i = \{1, \ldots, i\}$, $i = 1, \ldots, n$ (unter der Annahme $c_1 \geq c_2 \geq \cdots \geq c_n \geq c_{n+1} := 0$) und q bezeichnet den Rangquotienten. Mit $y^* \in \mathbb{K}^{2^E}$, definiert durch

$$y^*_F := \begin{cases} c_i - c_{i+1} & \text{für } F = E_i \\ 0 & \text{sonst} \end{cases}$$

gilt dann die folgende Abschätzung:

$$
\begin{aligned}
& \max \sum_{e\in E} c_e x_e,\ \sum_{e\in F} x_e \le r(F)\ \forall F \subseteq E, x_e \ge 0\ \forall e \in E \\
\ge\ & \max \sum_{e\in E} c_e x_e,\ \sum_{e\in F} x_e \le r(F)\ \forall F \subseteq E, x_e \ge 0\ \forall e \in E,\ x \text{ ganzzahlig} \\
=\ & c(I_0)\ \ge\ c(I_g) \\
=\ & \sum_{i=1}^{n} (c_i - c_{i+1}) |I_g \cap E_i|\ \ge\ \sum_{i=1}^{n} (c_i - c_{i+1})\, r_u(E_i)\ =\ \sum_{i=1}^{n} y^*_{E_i} r_u(E_i) \\
\ge\ & \min \sum_{F\subseteq E} y_F\, r_u(F),\ \sum_{F\ni e} y_F \ge c_e\ \forall e \in E, y_F \ge 0\ \forall F \subseteq E \\
\ge\ & q \min \sum_{F\subseteq E} y_F\, r(F),\ \sum_{F\ni e} y_F \ge c_e\ \forall e \in E, y_F \ge 0\ \forall F \subseteq E \\
=\ & q \max \sum_{e\in E} c_e x_e,\ \sum_{e\in F} x_e \le r(F)\ \forall F \subseteq E, x_e \ge 0\ \forall e \in E \\
\ge\ & q \max \sum_{e\in E} c_e x_e,\ \sum_{e\in F} x_e \le r(F)\ \forall F \subseteq E, x_e \ge 0\ \forall e \in E,\ x \text{ ganzzahlig} \\
=\ & q\, c(I_0)
\end{aligned}
$$

Satz (2.14) beweist

$$c(I_g)\ \ge\ q\, c(I_0).$$

Die obige Abschätzung liefert die Verbesserung

$$c(I_g)\ \ge\ q\ \max\{c^T x \mid x \in P(\mathcal{I})\}, \tag{2.33}$$

d. h. der Wert der Greedy-Lösung ist garantiert so groß wie das q-fache der LP-Relaxierung des kombinatorischen Optimierungsproblems.

Anwendungen dieser sehr allgemeinen Theorie auf einige konkrete kombinatorische Optimierungsprobleme mit praktischer Relevanz werden in den Übungen besprochen.

Literaturverzeichnis

F. Barahona and M. Grötschel. On the cycle polytope of a binary matroid. *Journal of Combinatorial Theory, Series B*, 40:40–62, 1986.

R. E. Bixby and W. H. Cunningham. Matroid optimization and algorithms. In R. L. Graham, M. Grötschel, L. Lovász, editor, *Handbook of Combinatorics, Volume I*, chapter 11, pages 551–609. North-Holland, Amsterdam, 1995.

J. Edmonds. Matroids and the greedy algorithm. *Mathematical Programming*, 1:127–136, 1971.

J. Edmonds. Matroid intersection. *Annals of Discrete Mathematics*, 4:39–49, 1979.

M. Grötschel and K. Truemper. Decomposition and Optimization over Cycles in Binary Matroids. *Journal of Combinatorial Theory, Series B*, 46(3):306–337, 1989.

D. Hausmann and B. Korte. The relative strength of oracles for independence systems. In J. Frehse, D. Pallaschke, and U. Trottenberg, editors, *Special topics of applied mathematics, functional analysis, numerical analysis, and optimization*, pages 195–211. North-Holland, Amsterdam, 1980.

D. Hausmann and B. Korte. Algorithmic versus axiomatic definitions of matroids. *Mathematical Programming Studies*, 14:98–111, 1981.

T. A. Jenkyns. The efficacy of the greedy algorithm. In F. Hoffman, L. Lesniak, and R. Mullin, editors, *Proceedings of the 7th Southeastern Conference on Combinatorics, Graph Theory and Computing*, Baton Rouge, 1976.

E. L. Lawler. Matroid intersection algorithms. *Mathematical Programming*, 9:31–56, 1975.

J. Lee. *A first course in combinatorial optimization.* University Press, Cambridge, 2004.

J. G. Oxley. *Matroid Theory.* Oxford University Press, Oxford, 1992.

A. Schrijver. *Combinatorial Optimization – Polyhedra and Efficiency.* Springer-Verlag, Berlin, 2003.

K. Truemper. *Matroid Decomposition.* Academic Press, Boston, 1992.

D. J. A. Welsh. *Matroid Theory.* Academic Press, London, 1976.

D. J. A. Welsh. Matroids: Fundamental concepts. In R. L. Graham, M. Grötschel, L. Lovász, editor, *Handbook of Combinatorics, Volume I*, chapter 9, pages 481–526. North-Holland, Amsterdam, 1995.

3 Varianten der Simplex-Methode

Der Simplex-Algorithmus ist das „Arbeitspferd“ der Optimierung. Effiziente Implementationen von Varianten dieses Verfahrens sind die Basis vieler akademischer und kommerzieller Codes zur Lösung von Optimierungsproblemen. Insbesondere die kombinatorische und gemischt-ganzzahlige Optimierung kommen bei der Lösung von schwierigen Problemen nicht ohne diese Methode aus.

In Kapitel 9 von ADM I haben wir das Grundprinzip des Simplex-Algorithmus erläutert. Wir setzen hier die Kenntnis dieses Kapitels (Definitionen, Sätze, Notation, etc.) voraus. Im Nachfolgenden wollen wir das Verständnis des Verfahrens vertiefen und insbesondere die Ingredienzien guter Implementierungen erläutern. Die im Kapitel 9 von ADM I eingeführte Tableau-Methode ist in der Praxis unbrauchbar. Bitte nicht implementieren! Wie es besser gemacht werden kann, wird in diesem Kapitel (zumindest teilweise) skizziert, wobei auf numerische Aspekte speziell in den Übungen eingegangen wird.

3.1 Der revidierte Simplexalgorithmus

Betrachtet man die Grundversion des Simplexalgorithmus (9.17) im ADM I Skript, so stellt man fest, dass während einer Iteration i. A. nicht sämtliche Spalten der Matrix $\overline{A}$ benötigt werden. Der revidierte Simplexalgorithmus nutzt diese Tatsache aus, indem er stets nur solche Spalten von $\overline{A}$ berechnet, die im jeweiligen Schritt benötigt werden.

Alle numerischen Implementationen des Simplexverfahrens benutzen als Grundgerüst den revidierten Simplexalgorithmus. Wir wollen deshalb im folgenden den revidierten Simplexalgorithmus in Unterprogramme (Subroutines) zerlegen und später zeigen, wie diese Subroutines bei den jeweiligen numerischen Implementationsvarianten realisiert werden. Wir gehen immer davon aus, dass wir ein LP in Standardform

$$\begin{aligned} \max \quad & c^T x \\ & Ax = b \\ & x \geq 0 \end{aligned} \tag{$*$}$$

vorliegen haben (siehe ADM I, Definition (9.1)). Alle Bezeichnungen werden, wie im ADM I Skript, Kapitel 9 festgelegt, verwendet: $\{1, \ldots, m\}$ ist die Zeilenindexmenge, $\{1, \ldots, n\}$ die Spaltenindexmenge von A, $B = (p_1, \ldots, p_m)$ und $N = (q_1, \ldots, q_{n-m})$ bezeichnen Spaltenindexvektoren von A und werden gleichzeitig als Teilmengen von $\{1, \ldots, n\}$ aufgefasst, und statt $A_{\cdot B}$ bzw. $A_{\cdot N}$ schreiben wir vereinfacht A_B bzw. A_N; vgl. die Konventionen in ADM I, (9.3).

Der nachfolgende Algorithmus beschreibt nur die Phase II des Simplexverfahrens. Wir gehen also davon aus, dass zu Beginn eine zulässige Basis A_B vorliegt.

(3.1) Algorithmus (Revidierter Simplexalgorithmus).
Eingabe: $A \in \mathbb{K}^{(m,n)}$, $b \in \mathbb{K}^m$, $c \in \mathbb{K}^n$, Spaltenindexvektor B, zugehörige Matrix A_B^{-1} und Vektor $\overline{b} = A_B^{-1}b$.
Ausgabe: Eine optimale Lösung x des linearen Programms $(*)$.

1. **BTRAN** (Backward Transformation)
   Berechne $\pi^T := c_B^T A_B^{-1}$. (Wir werden diese Werte später *Schattenpreise* nennen.)

2. **PRICE** (Pivotspaltenauswahl)
   Berechne die reduzierten Kostenkoeffizienten $\overline{c}_j := (c_N^T)_j - \pi^T A_N e_j$, für $j = 1, \ldots, n-m$ und wähle einen Index s mit $\overline{c}_s > 0$.
   Abbruchkriterium (vgl. Schritt (II.1) von ADM I, (9.17)):
   Gilt $\overline{c}_j \leq 0$ für $j = 1, \ldots, n-m$, so ist die gegenwärtige Basislösung optimal. Gib den Vektor x mit $x_B = \overline{b}$, $x_N = 0$ aus. Der Optimalwert von $(*)$ ist $c^T x = c_B^T \overline{b} = c_0$. STOP!

3. **FTRAN** (Forward Transformation)
   Aktualisiere die Pivotspalte $\overline{d} := A_B^{-1} d = A_B^{-1} A_{\cdot q_s}$.

4. **CHUZR** (Pivotzeilenauswahl)
   Berechne $\lambda_0 = \min\{\frac{\overline{b}_i}{\overline{d}_i} \mid \overline{d}_i > 0,\ i = 1, \ldots, m\}$ und wähle einen Index $r \in \{i \in \{1, \ldots, m\} \mid \frac{\overline{b}_i}{\overline{d}_i} = \lambda_0\}$.

5. **WRETA** (Updating der Basis)
   Entferne das r-te Element von B und ersetze es durch q_s. Der neue Spaltenindexvektor heiße B'.
   Berechne $A_{B'}^{-1}$, $\overline{b} = A_{B'}^{-1} b$. △

Wir werden später und in den Übungen auf numerische „Tricks“ zur Implementation der Updates eingehen. Es seien hier jedoch bereits einige Vorteile der revidierten Simplexmethode festgehalten:

- Wesentlich weniger Rechenaufwand, speziell bei Programmen mit erheblich mehr Variablen als Gleichungen.
- Die Ausgangsmatrix A kann auf externem Speicher bzw. bei dünn besetzten Matrizen durch spezielle Speichertechniken gehalten werden. Spalten von A werden je nach Bedarf generiert. (Sparse-Matrix-Techniques)
- Die Kontrolle über die Rechengenauigkeit ist besser. Bei der Tableaumethode akkumulieren die Rundefehler. Beim revidierten Simplexalgorithmus kann bei Bedarf mit Hilfe eines Inversionsunterprogramms A_B^{-1} neu berechnet werden.
- Es gibt besondere Speichertechniken, die eine explizite Speicherung von A_B^{-1} vermeiden. Man merkt sich lediglich die Etavektoren und berechnet dann über die Formel aus ADM I, Satz (9.12) aus der ersten Basisinversen durch Linksmultiplikation mit den Elementarmatrizen die gegenwärtige Basisinverse.

3.2 Die Behandlung oberer Schranken

In linearen Programmen kommen häufig Beschränkungen der folgenden Form vor:

$$0 \leq x \leq u.$$

Da diese strukturell sehr einfach sind, kann man sie algorithmisch besser behandeln als allgemeine Ungleichungen. Normalerweise führen wir Ungleichungen durch Einfügung von Schlupfvariablen wie folgt in Gleichungen und Nichtnegativitätsbedingungen über:

$$x + \overline{x} = u, \quad x \geq 0,\ \overline{x} \geq 0.$$

Ist jedoch der Wert von x (bzw. $\overline{x}$) festgelegt, so ist der Wert der Komplementärvariablen $\overline{x}$ (bzw. x) bereits eindeutig bestimmt. Diesen Vorteil kann man nun so ausnutzen, dass man im Simplexverfahren nur eine der beiden Variablen mitschleppt und die zusätzliche Gleichung völlig weglässt. Für jede Zeile oder Spalte muss jedoch festgehalten werden, ob sie x oder der Komplementärvariablen $\overline{x} = u - x$ entspricht. Die Zeilenauswahlregeln sind ebenfalls etwas komplizierter, da eine neue in die „Basis“ hineinzunehmende Variable nur maximal bis zu ihrer Beschränkung erhöht werden darf und die übrigen Basisvariablen ebenfalls ihre Schranken nicht überschreiten dürfen.

Wir wollen im folgenden die sogenannte *Obere-Schranken-Technik* zur Behandlung von linearen Programmen der Form

$$\begin{aligned} \max \quad & c^T x \\ & Ax = b \\ & 0 \leq x \leq u \end{aligned} \tag{3.2}$$

besprechen. Diese „upper-bound-technique“ behandelt nur explizit nach oben beschränkte Variablen. Sind einige der Variablen nicht explizit nach oben beschränkt, so legen wir, um die weitere Diskussion und die Formeln zu vereinfachen, fest, dass ihre obere Schranke $+\infty$ ist. Das heißt, die Komponenten von u haben Werte aus der Menge $\mathbb{R}_+ \cup \{+\infty\}$.

Weiter benötigen wir eine erweiterte Definition von Basis und Nichtbasis. Wir halten uns im Prinzip an Konvention (9.3) aus dem ADM I Skript, lassen jedoch zu, dass der Vektor der Indizes von Nichtbasisvariablen positive und negative Indizes enthalten kann. Durch das Vorzeichen wollen wir uns merken, ob eine Nichtbasisvariable die obere oder untere Schranke annimmt, und zwar legen wir fest, dass für eine Nichtbasisvariable q_s der Wert x_{q_s} Null ist, falls das Vorzeichen von q_s positiv ist, andernfalls ist der Wert von x_{q_s} die obere Schranke u_{q_s}. Um dies formeltechnisch einfach aufschreiben zu können, treffen wir die folgenden Vereinbarungen. Ist $B = (p_1, \ldots, p_m)$ ein Spaltenindexvektor, so dass A_B eine Basis ist und $N = (q_1, \ldots, q_{n-m})$ wie in ADM I (9.3) definiert, dann sei

$$\overline{N} \ := \ (\overline{q}_1, \ldots, \overline{q}_{n-m}) \quad \text{mit } \overline{q}_i = q_i \text{ oder } \overline{q}_i = -q_i, \tag{3.3}$$

$$\begin{aligned} N^+ &:= (q_i \mid \overline{q}_i > 0), \\ N^- &:= (q_i \mid \overline{q}_i < 0). \end{aligned} \tag{3.4}$$

Die in B, N^+ und N^- enthaltenen Indizes bilden also eine Partition der Spaltenindexmenge $\{1,\ldots,n\}$ von A. Bei einer Rechnerimplementation genügt es natürlich, sich $\overline{N}$ zu merken, denn $N = (|\overline{q}_1|,\ldots,|\overline{q}_{n-m}|)$. Ist A_B regulär, so nennen wir A_B eine *E-Basis* (erweiterte Basis) von A und $\overline{N}$ eine *E-Nichtbasis* von A. Der Vektor $x \in \mathbb{K}^n$ mit

$$\begin{aligned} x_{N^+} &= 0 \\ x_{N^-} &= u_{N^-} \\ x_B &= A_B^{-1}b - A_B^{-1}A_{N^-}u_{N^-} \end{aligned} \tag{3.5}$$

heißt *E-Basislösung* zur E-Basis A_B. Eine $\left\{ \begin{array}{l} \text{E-Basis } A_B \\ \text{E-Basislösung } x \end{array} \right\}$ heißt *zulässig*, wenn $0 \le x_B \le u_B$ und heißt *nicht degeneriert* (*nicht entartet*), falls $0 < x_B < u_B$, andernfalls *degeneriert* (*entartet*). Gibt es keine oberen Schranken, so gilt offenbar $N = \overline{N} = N^+$ und alle Formeln reduzieren sich auf den bereits behandelten Fall.

(3.6) Satz. *Gegeben sei ein Polyeder* $P := \{x \in \mathbb{K}^n \mid Ax = b, 0 \le x \le u\}$ *mit* $\operatorname{rang}(A) = m$. *Ein Vektor* $x \in \mathbb{K}^n$ *ist genau dann eine Ecke von* P, *wenn* x *zulässige E-Basislösung ist.* △

Beweis. Es gilt $P = P(D,d)$ mit

$$D = \begin{pmatrix} A \\ -A \\ I \\ -I \end{pmatrix}, \quad d = \begin{pmatrix} b \\ -b \\ u \\ 0 \end{pmatrix}.$$

Also folgt mit Satz (8.8) aus dem ADM I Skript: x Ecke von $P \iff \operatorname{rang}(D_{\operatorname{eq}(\{x\})}) = n$. Seien $J = \operatorname{eq}(\{x\})$ und J_1, J_2, J_3, J_4 so gewählt, dass

$$D_{J.} = \begin{pmatrix} A_{J_1} & & * \\ -A_{J_2} & & \\ 0 & I_{J_3} & 0 \\ 0 & 0 & -I_{J_4} \end{pmatrix}$$

Gilt $\operatorname{rang}(D_{J.}) = n$, so besitzt $A_{J_1 \cup J_2}$. vollen Rang und mit $K := \{1,\ldots,n\} \setminus (J_3 \cup J_4)$ ist $A_B := A_{J_1 \cup J_2, K}$ regulär. Man verifiziert sofort, dass A_B zulässig ist, wenn $N^- = J_3$, $N^+ = J_4$ gewählt wird. Die Rückrichtung ist nun evident. □

Wir kennzeichnen also im Weiteren Basen, Nichtbasen und Basislösung von (3.2) mit einem E, um sie von den in ADM I, Kapitel 9 eingeführten Begriffen zu unterscheiden.

(3.7) Algorithmus (Upper-Bound-Technique zur Lösung von linearen Programmen der Form (3.2)**).** Wir nehmen an, dass $u \in (\mathbb{R}_+ \cup \{+\infty\})^n$ gilt und dass eine zulässige E-Basis A_B vorliegt. Wir beschreiben lediglich die Phase II.

Eingabe: Eine zulässige E-Basis A_B, A_B^{-1}, $\overline{b} = A_B^{-1}b$, c, $B = (p_1, \ldots, p_m)$ und $\overline{N} = (\overline{q}_1, \ldots, \overline{q}_{n-m})$ (die Mengen N, N^+, N^- können daraus bestimmt werden), $\overline{\overline{b}} = A_B^{-1}b - A_B^{-1}A_{N^-}u_{N^-}$.
Ausgabe: Eine optimale Lösung des linearen Programms (3.2).

1. **BTRAN:**
   Berechne $\pi^T := c_B^T A_B^{-1}$.

2. **PRICE:**
   Berechne $\overline{c}^T := c_N^T - \pi^T A_N$ und wähle ein $s \in \{1, \ldots, n-m\}$ mit

$$\overline{c}_s \begin{cases} > 0 & \text{falls } \overline{q}_s > 0 \\ < 0 & \text{falls } \overline{q}_s < 0 \end{cases}$$

   mittels irgendeiner der Pivotspaltenauswahlregeln. Gibt es keinen solchen Index s, so ist die aktuelle E-Basislösung optimal.
   Begründung: Offenbar gilt $x \in \{y \in \mathbb{K}^n \mid Ay = b, 0 \le y \le u\} \iff x_B = A_B^{-1}b - A_B^{-1}A_{N^+}x_{N^+} - A_B^{-1}A_{N^-}x_{N^-}$ und $0 \le x_B, x_{N^+}, x_{N^-} \le u$. Also ist

$$\begin{aligned} c^T x &= c_B^T x_B + c_{N^+}^T x_{N^+} + c_{N^-}^T x_{N^-} \\ &= c_B^T A_B^{-1} b + (c_{N^+}^T - c_B^T A_B^{-1} A_{N^+}) x_{N^+} + (c_{N^-}^T - c_B^T A_B^{-1} A_{N^-}) x_{N^-} \\ &= \pi^T b + (c_{N^+}^T - \pi^T A_{N^+}) x_{N^+} + (c_{N^-}^T - \pi^T A_{N^-}) x_{N^-}. \end{aligned}$$

   Da für die gegenwärtige E-Basislösung gilt $x_{N^-} = u_{N^-}$, $x_{N^+} = 0$, kann der Zielfunktionswert nur verbessert werden, wenn für eine Nichtbasisvariable q_s entweder gilt $\overline{q}_s > 0$ und $\overline{c}_s = (c_N^T - \pi^T A_N)_s > 0$ oder $\overline{q}_s < 0$ und $\overline{c}_s < 0$.

3. **FTRAN:**
   Berechne $\overline{d} := A_B^{-1} A_{\cdot q_s} = \overline{A}_{\cdot s}$.

4. **CHUZR:**
   Setze $\sigma = \operatorname{sign}(\overline{q}_s)$ und berechne

$$\begin{aligned} \lambda_0 &:= \min\{\frac{\overline{\overline{b}}_i}{\sigma \overline{a}_{is}} \mid \sigma \overline{a}_{is} > 0,\ i = 1, \ldots, m\}, \\ \lambda_1 &:= \min\{\frac{\overline{\overline{b}}_i - u_{p_i}}{\sigma \overline{a}_{is}} \mid \sigma \overline{a}_{is} < 0,\ i = 1, \ldots, m\}, \\ \lambda_2 &:= u_{q_s}. \end{aligned}$$

   (a) Ist keine der Zahlen λ_0, λ_1, λ_2 endlich, so ist das Programm (3.2) unbeschränkt (das kann natürlich nur vorkommen, wenn x_{q_s} nicht explizit beschränkt ist).

   (b) Ist $\lambda_0 = \min\{\lambda_0, \lambda_1, \lambda_2\}$, so wähle einen Index

$$r \in \Big\{ i \in \{1, \ldots, m\} \mid \frac{\overline{\overline{b}}_i}{\sigma \overline{a}_{is}} = \lambda_0,\ \sigma \overline{a}_{is} > 0 \Big\}$$

   mittels irgendeiner Pivotzeilenauswahlregel.

(c) Ist $\lambda_1 = \min\{\lambda_0, \lambda_1, \lambda_2\}$, so wähle

$$r \in \left\{ i \in \{1, \ldots, m\} \mid \frac{\overline{\overline{b}}_i - u_{p_i}}{\sigma \overline{a}_{is}} = \lambda_1,\ \sigma \overline{a}_{is} < 0 \right\}$$

mittels irgendeiner Pivotzeilenauswahlregel.

(d) Ist $\lambda_2 = \min\{\lambda_0, \lambda_1, \lambda_2\}$, so setze $\overline{q}_s := -\overline{q}_s$, berechne $\overline{\overline{b}}$ neu und gehe zurück zur PRICE-Routine 2.

5. **WRETA:**
Setze $B' = (p_1, \ldots, p_{r-1}, q_s, p_{r+1}, \ldots, p_m)$, $\overline{N} = (\overline{q}_1, \ldots, \overline{q}_{s-1}, \overline{q}, \overline{q}_{s+1}, \ldots, \overline{q}_{n-m})$, wobei $\overline{q} = p_r$, falls $\lambda_0 = \min\{\lambda_0, \lambda_1, \lambda_2\}$, bzw. $\overline{q} = -p_r$, falls $\lambda_1 = \min\{\lambda_0, \lambda_1, \lambda_2\}$. Berechne $A_{B'}^{-1}$, $\overline{\overline{b}} = A_{B'}^{-1} b - A_{B'}^{-1} A_{N^-} u_{N^-}$. △

Begründung: Die Pivotzeilenauswahl (CHUZR-Routine) dient dazu, die Zeilenauswahl so zu bestimmen, dass die transformierten Variablen nach erfolgter Pivotoperation wieder zulässig sind. Entsprechend der Transformationsregeln beim Pivotisieren mit dem Pivotelement $\overline{a}_{rs}$ (vergleiche ADM I, Satz (9.12)) muss gewährleistet sein, dass nach Ausführung des Pivotschrittes für die neue E-Basislösung folgendes gilt:

(i) $0 \leq x'_{p_i} = \overline{\overline{b}}_i - \dfrac{\overline{a}_{is}}{\overline{a}_{rs}} \overline{\overline{b}}_r \leq u_{p_i}$, für $i \neq r$,

(ii) $0 \leq x'_{q_s} = \dfrac{1}{\overline{a}_{rs}} \overline{\overline{b}}_r \leq u_{q_s}$,

(iii) $x'_{p_r} \in \{0, u_{p_r}\}$,

(iv) $x'_{q_i} \in \{0, u_{q_i}\}$, für $i \neq s$.

Wollen wir nun den Wert der Variablen x_{q_s} um $\lambda \geq 0$ ändern (d. h. erhöhen, falls $\overline{q}_s > 0$, bzw. um λ erniedrigen, falls $\overline{q}_s < 0$), so können wir das so lange tun bis entweder

- eine Basisvariable den Wert ihrer oberen oder unteren Schranke annimmt oder
- die Nichtbasisvariable q_s den Wert ihrer anderen Schranke annimmt.

Aus letzterem folgt natürlich, dass $\lambda \leq u_{q_s}$ gelten muss, ansonsten würden wir bei Erhöhung ($\overline{q}_s > 0$) die obere Schranke bzw. bei Erniedrigung ($\overline{q}_s < 0$) die untere Schranke für x_{q_s} verletzen. Dies erklärt die Bestimmung von λ_2 in 4.

Ist $\overline{q}_s > 0$, so bewirkt eine Erhöhung von x_{q_s} um den Wert λ, dass $x'_{p_i} = \overline{\overline{b}}_i - \lambda \overline{a}_{is}$ gilt. Aus der Bedingung (i) erhalten wir daher die folgenden Schranken für λ:

$$\lambda \leq \frac{\overline{\overline{b}}_i}{\overline{a}_{is}} \qquad \text{für } \overline{a}_{is} > 0,$$

$$\lambda \leq \frac{\overline{\overline{b}}_i - u_{p_i}}{\overline{a}_{is}} \qquad \text{für } \overline{a}_{is} < 0.$$

Ist $\overline{q}_s < 0$, so bewirkt die Verminderung von x_{q_s} $(= u_{q_s})$ um den Wert $\lambda \geq 0$, dass $x'_{p_i} = \overline{\overline{b}}_i + \lambda \overline{a}_{is}$ gilt. Aus (i) erhalten wir somit:

$$\lambda \leq -\frac{\overline{\overline{b}}_i}{\overline{a}_{is}} \qquad \text{für } \overline{a}_{is} < 0,$$
$$\lambda \leq \frac{u_{p_i} - \overline{\overline{b}}_i}{\overline{a}_{is}} \qquad \text{für } \overline{a}_{is} > 0.$$

Dies begründet die Definition von λ_0 und λ_1. Sei nun $\lambda_{\min} = \min\{\lambda_0, \lambda_1, \lambda_2\}$. Gilt $\lambda_{\min} = \lambda_2$, so wird einfach der Wert einer Nichtbasisvariablen von der gegenwärtigen Schranke auf die entgegengesetzte Schranke umdefiniert. Die Basis ändert sich dadurch nicht, jedoch konnte aufgrund der Auswahl in PRICE eine Zielfunktionsverbesserung erreicht werden. Gilt $\lambda_{\min} = \lambda_1$ oder $\lambda_{\min} = \lambda_2$, so führen wir einen üblichen Pivotschritt durch. Bei $\lambda_{\min} = \lambda_1$ wird die neue Nichtbasisvariable x_{p_r} mit Null festgesetzt, da die untere Schranke von x_{p_r} die stärkste Einschränkung für die Veränderung von x_{q_s} darstellte. Bei $\lambda_{\min} = \lambda_2$, wird analog x_{p_r} eine Nichtbasisvariable, die den Wert ihrer oberen Schranke annimmt.

Wir wollen nun anhand eines Beispiels die Ausführung von Algorithmus (3.7) in Tableautechnik demonstrieren. Dabei werden wir keine neuen Tableau-Update-Formeln definieren, sondern mit den bekannten Formeln für die Pivotschritte rechnen. Wir führen lediglich eine zusätzliche Spalte ein, in der jeweils aus $\overline{b}$ und c_0 der Wert der gegenwärtigen E-Basislösung berechnet wird. Ist also c_0 der gegenwärtige Wert der Basislösung und $x_B = A_B^{-1} b = \overline{b}$ so hat die zusätzliche Spalte in der ersten Komponente den Eintrag $\overline{\overline{c}}_0 := -c_0 - \overline{c}_{N^-} u_{N^-}$, und die restlichen Komponenten berechnen sich durch

$$\overline{\overline{b}} := \overline{b} - \overline{A}_{N^-} u_{N^-}.$$

(3.8) Beispiel. Wir betrachten das folgende LP, dessen Lösungsmenge in Abbildung 3.1 dargestellt ist:

$$\begin{aligned} \max \quad & 2x_1 + x_2 \\ & -x_1 + x_2 \leq \frac{1}{2} \\ & x_1 + x_2 \leq 2 \\ & x_1 \leq \frac{3}{2} \ (= u_1) \\ & x_2 \leq 1 \ (= u_2) \\ & x_1, x_2 \geq 0. \end{aligned}$$

Wir führen 2 Schlupfvariablen x_3, x_4 ein (mit oberen Schranken $+\infty$) und erhalten als Anfangstableau T_0 (mit zusätzlicher Spalte $\binom{\overline{\overline{c}}_0}{\overline{\overline{b}}}$, die mit der Spalte $\binom{-c_0}{\overline{b}}$ identisch ist,

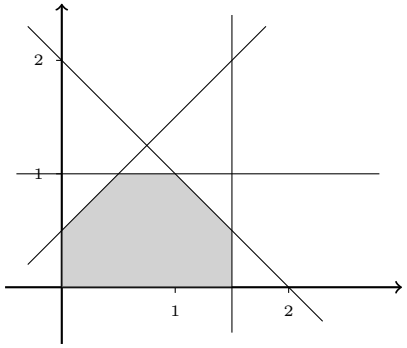


Abbildung 3.1: Lösungsmenge zum LP aus Beispiel (3.8)

da keine Nichtbasisvariable von Null verschieden ist):

$$T_0: \quad \begin{array}{c|cccc|c|c} & 1 & 2 & 3 & 4 & & \\ & 2 & 1 & 0 & 0 & 0 & 0 \\ \hline 3 & -1 & 1 & 1 & 0 & \frac{1}{2} & \frac{1}{2} \\ 4 & 1 & 1 & 0 & 1 & 2 & 2 \end{array} \quad = \quad \begin{array}{cc|c|c} \overline{c} & 0 & -c_0 & \overline{\overline{c}}_0 \\ \hline \overline{A} & I & \overline{b} & \overline{\overline{b}} \end{array} \qquad \begin{array}{l} B = (3,4) \\ \overline{N} = (1,2) \end{array}$$

Wir wählen die erste Spalte als Pivotspalte, d. h. $\overline{q}_1 = 1$. Schritt 4. CHUZR ergibt

$$\lambda_0 = \frac{\overline{\overline{b}}_2}{\overline{a}_{21}} = \frac{2}{1} = 2, \quad \lambda_1 \text{ ist wegen } u_3 = \infty \text{ nicht definiert}, \quad \lambda_2 = u_1 = \frac{3}{2},$$

d. h. $\lambda_{\min} = \lambda_2$. Wir führen also Schritt 4.(d) aus, setzen $\overline{q}_1 := -\overline{q}_1$, d. h. $\overline{q}_1 = -1$, und berechnen $\overline{\overline{b}}$ neu. Die neue letzte Spalte, d. h. die neue E-Basislösung zur Basis A_B erhalten wir aus der (normalen) Basislösung wegen $N^- = (1)$ wie folgt:

$$\overline{\overline{b}} = \overline{b} - \overline{A}_{\cdot 1} u_1 = \begin{pmatrix} \frac{1}{2} \\ 2 \end{pmatrix} - \frac{3}{2} \begin{pmatrix} -1 \\ 1 \end{pmatrix} = \begin{pmatrix} 2 \\ \frac{1}{2} \end{pmatrix},$$

und der Zielfunktionswert erhöht sich um $\overline{c}_{1.} u_1 = 2 \cdot \frac{3}{2} = 3$. Mithin erhalten wir als nächstes Tableau (mit $\overline{q}_1 := -\overline{q}_1$):

$$T_1: \quad \begin{array}{c|cccc|c|c} & -1 & 2 & 3 & 4 & & \\ & 2 & 1 & 0 & 0 & 0 & -3 \\ \hline 3 & -1 & 1 & 1 & 0 & \frac{1}{2} & 2 \\ 4 & 1 & 1 & 0 & 1 & 2 & \frac{1}{2} \end{array} \qquad \begin{array}{l} B = (3,4) \\ \overline{N} = (-1,2) \end{array}$$

Als Pivotspalte kommt nur die 2. Spalte ($\overline{q}_2 = 2$) in Frage. CHUZR ergibt

$$\lambda_0 = \frac{\overline{\overline{b}}_2}{\overline{a}_{22}} = \frac{1}{2}, \quad \lambda_1 \text{ nicht definiert}, \quad \lambda_2 = u_2 = 1.$$

Die Wahl der Austauschzeile in 4.(b) ergibt ein eindeutig bestimmtes $r = 2$. Wir führen dann 5. in Tableautechnik aus. Wir führen also einen Standardpivotschritt auf dem Element $\overline{a}_{22} = 1$ durch und korrigieren anschließend $\overline{b}$ und c_0, um $\overline{\overline{b}}$ und $\overline{\overline{c}}_0$ zu erhalten.

T_2 :

	–1	2	3	4		
	1	0	0	−1	−2	$-\frac{7}{2}$
3	−2	0	0	−1	$-\frac{3}{2}$	$\frac{3}{2}$
2	1	1	0	1	2	$\frac{1}{2}$

$B = (3, 2)$
$\overline{N} = (-1, 4)$

Dieses Tableau ist optimal, mit $x_1 = \frac{3}{2}$, $x_2 = \frac{1}{2}$.

Zur Erläuterung der übrigen Fälle beginnen wir noch einmal mit Tableau T_0, wählen aber die 2. Spalte als Pivotspalte. CHUZR ergibt

$$\lambda_0 = \frac{\overline{\overline{b}}_1}{\overline{a}_{12}} = \frac{1}{2}, \quad \lambda_1 \text{ nicht definiert}, \quad \lambda_2 = 1.$$

In diesem Fall machen wir einen Standardpivotschritt mit dem Pivotelement $\overline{a}_{rs} = \overline{a}_{12} = 1$.

T_3 :

	1	2	3	4		
	3	0	−1	0	$-\frac{1}{2}$	$-\frac{1}{2}$
2	−1	1	1	0	$\frac{1}{2}$	$\frac{1}{2}$
4	2	0	−1	1	$\frac{3}{2}$	$\frac{3}{2}$

$B = (2, 4)$
$\overline{N} = (1, 3)$

Die gegenwärtige Basislösung wird wie üblich in eine E-Basislösung transformiert. Als Pivotspalte kommt nur die erste in Frage. CHUZR ergibt

$$\lambda_0 = \frac{\overline{\overline{b}}_2}{\overline{a}_{21}} = \frac{3}{4}, \quad \lambda_1 = \frac{\overline{\overline{b}}_1 - u_2}{\overline{a}_{11}} = \frac{1}{2}, \quad \lambda_2 = u_1 = \frac{3}{2} \quad (\lambda_{\min} = \lambda_1)$$

Wir führen nun Schritt 5 des Algorithmus (3.7) durch einen Pivotschritt auf $\overline{a}_{rs} = \overline{a}_{11} = -1$ aus und berechnen das gesamte Tableau neu. Anschließend müssen wir die rechteste Spalte des Tableaus korrigieren. Da $\lambda_{\min} = \lambda_1$, wird die neue Nichtbasisvariable x_2 mit ihrer oberen Schranke $u_2 = 1$ belegt (und $\overline{q}_1 = -2$ gesetzt). Wir müssen daher vom neuberechneten $\overline{b} = (-\frac{1}{2}, \frac{5}{2})^T$ noch das u_2-fache der zum Index 2 gehörigen Spalte von $\overline{A}$ (also $\overline{A}_{\cdot 1} = (-1, 2)^T$) subtrahieren. Vom Zielfunktionswert c_0 subtrahieren wir $u_2 \cdot \overline{c}_2 = 1 \cdot 3$. Daraus ergibt sich

T_4 :

	1	–2	3	4		
	0	3	2	0	1	−2
1	1	−1	−1	0	$-\frac{1}{2}$	$\frac{1}{2}$
4	0	2	1	1	$\frac{5}{2}$	$\frac{1}{2}$

$B = (1, 4)$
$\overline{N} = (-2, 3)$

$$\overline{\overline{b}} = \begin{pmatrix} -\frac{1}{2} \\ \frac{5}{2} \end{pmatrix} - 1 \begin{pmatrix} -1 \\ 2 \end{pmatrix}.$$

Als Pivotspalte müssen wir nun die 3. Spalte von T_4, also $\overline{A}_{\cdot 2}$, wählen.

$$\lambda_0 = \frac{\overline{\overline{b}}_2}{\overline{a}_{22}} = \frac{1}{2} \quad \lambda_1 = \frac{\overline{\overline{b}}_1 - u_1}{\overline{a}_{12}} = 1, \quad \lambda_2 = u_1 = \frac{3}{2}.$$

Somit führen wir einen Standardpivotschritt auf $\overline{a}_{22} = 1$ aus

T_5 :

		1	*−2*	*3*	*4*		
		0	−1	0	−2	−4	−3
1	1	1	0	1		2	1
3	0	2	1	1		$\frac{5}{2}$	$\frac{1}{2}$

$B = (1, 3)$
$\overline{N} = (-2, 4)$

Als Pivotspalte kommt nur $\overline{A}_{\cdot 1}$, also die 2. Spalte von T_5, in Frage, d. h. $q_s = q_1 = 2$, $\overline{q}_1 = -2$. Wir erhalten durch CHUZR

$$\lambda_0 \text{ undefiniert}, \quad \lambda_1 = \frac{\overline{\overline{b}}_1 - u_1}{\sigma \overline{a}_{11}} = \frac{1}{2}, \quad \lambda_2 = u_2 = 1.$$

Unser Pivotelement ist somit $\overline{a}_{11} = 1$. Wie üblich führen wir einen Standardpivotschritt durch. Da wir die neue Nichtbasisvariable x_1 mit ihrer oberen Schranke $u_1 = \frac{3}{2}$ belegen, ziehen wir wieder vom neuberechneten $\overline{b}$ das u_1-fache der zu 1 gehörigen Spalte von $\overline{A}$ (das ist $\overline{A}_{\cdot 1} = (1, -2)^T$) ab, um $\overline{\overline{b}}$ zu erhalten.

T_6 :

		−1	*2*	*3*	*4*		
		1	0	0	−1	−2	$-\frac{7}{2}$
2	1	1	1	0	1	2	$\frac{1}{2}$
3	−2	0	1	−1		$-\frac{3}{2}$	$\frac{3}{2}$

$B = (2, 3)$
$\overline{N} = (-1, 4)$

Wir haben wieder die Optimallösung erreicht. △

Analog zu oberen Schranken können natürlich auch untere Schranken behandelt werden. Ferner kann man auch allgemeine obere Schranken (generalized upper bounds, GUB), dies sind Restriktionen der Form $\sum x_i \leq a$, auf ähnliche Weise berücksichtigen. Bei solchen Restriktionen können ebenfalls Pivotschemata entworfen werden, die wesentlich weniger Speicher- und Rechenaufwand erfordern als die Standardmethode. Weiterhin gibt es effiziente Techniken zur Behandlung von sogenannten variable upper bounds (VUB) der Form $0 \leq x \leq y$.

3.3 Das duale Simplexverfahren

Wir wollen nun eine Variante des Simplexverfahrens darstellen, die – wie wir noch sehen werden – bei bestimmten Problemstellungen von Vorteil ist. Sei A_B eine Basis von A, und betrachte das Paar dualer linearer Programme:

$$\text{(P)} \quad \begin{aligned} \max \quad & c^T x \\ & Ax = b \\ & x \geq 0 \end{aligned} \qquad \text{und} \qquad \text{(D)} \quad \begin{aligned} \min \quad & u^T b \\ & u^T A \geq c^T \end{aligned}$$

bzw.

$$\begin{array}{lll} \max & c^T x & \min \quad u^T b \\ & x_B = A_B^{-1} b - A_B^{-1} A_N x_N \quad \text{und} & u^T A_B \geq c_B^T \\ & x_N, x_B \geq 0 & u^T A_N \geq c_N^T \end{array}$$

(3.9) Definition. *Die Basis A_B von A heißt* primal zulässig, *falls $A_B^{-1} b \geq 0$ und* dual zulässig, *falls $\overline{c}^T = c_N^T - c_B^T A_B^{-1} A_N \leq 0$. Die zugehörigen Basislösungen x bzw. u mit $x_B = A_B^{-1} b$, $x_N = 0$ bzw. $u^T = c_B^T A_B^{-1}$ heißen dann* primal *bzw.* dual zulässig. △

(3.10) Satz. *Sei $P = \{u \in \mathbb{K}^m \mid u^T A \geq c^T\}$. Der Vektor $\overline{u}$ ist genau dann eine Ecke von P, wenn $\overline{u}$ eine dual zulässige Basislösung ist.* △

Beweis. Sei $\overline{u}$ Ecke von $P = P(-A^T, -c)$ und $I = \text{eq}(\{\overline{u}\})$. Mit Satz (8.8) aus dem ADM I Skript folgt $\text{rang}((-A^T)_{I.}) = m$, d. h. es existiert $B \subseteq I$ mit A_B Basis von A, und es gilt $\overline{u}^T A_B = c_B^T$, $\overline{u}^T A_N \geq c_N^T$. Also ist $\overline{u}^T = c_B^T A_B^{-1}$ und $c_B^T A_B^{-1} A_N \geq c_N^T$, d. h. A_B ist dual zulässig. Ist umgekehrt A_B dual zulässig, so ist $\overline{u}^T := c_B^T A_B^{-1}$ aufgrund von ADM I, Satz (8.8) eine Ecke von P. □

(3.11) Bemerkung. Ist A_B eine Basis von A, und sind x bzw. u die zu A_B gehörenden primalen bzw. dualen (aber nicht notwendigerweise zulässigen) Basislösungen, so gilt: $c^T x = u^T b$. △

Beweis. $u^T b = c_B^T A_B^{-1} b = c_B^T x_B = c^T x$. □

(3.12) Korollar. *Ist A_B eine Basis von A, so ist A_B optimal genau dann, wenn A_B primal und dual zulässig ist.* △

Beweis. Dualitätssatz. □

(3.13) Korollar. *Ist A_B eine optimale Basis für das Programm (P), dann ist $c_B^T A_B^{-1}$ eine optimale Lösung des zu (P) dualen Programms (D).* △

Der Vektor $\pi := c_B^T A_B^{-1}$ (mit A_B dual zulässig) heißt der Vektor der *Schattenpreise* (vgl. ökonomische Interpretation der Dualität und Schritt 1 in (3.1)).

Wir wollen nun die dem dualen Simplexverfahren zugrunde liegende Idee entwickeln und bemerken, dass – im Prinzip – das duale Simplexverfahren das primale Simplexverfahren angewendet auf das duale Problem ist. Sei A_B eine Basis von A, so lässt sich (P) umformen in:

$$\begin{array}{ll} \max & c_B^T A_B^{-1} b + \overline{c}^T x_N \\ & \overline{A} x_N + I x_B = A_B^{-1} b \; (= \overline{b}) \\ & x_B, x_N \geq 0 \end{array} \tag{3.14}$$

oder

$$\begin{array}{lll} c_B^T A_B^{-1} b \; + & \max & \overline{c}^T x_N \\ & & \overline{A} x_N \leq \overline{b} \\ & & x_N \geq 0 \end{array} \tag{3.15}$$

Das zu (3.15) duale Programm lautet (bei Weglassung des konstanten Terms $c_B^T A_B^{-1} b$ der Zielfunktion):

$$\begin{aligned} \min \quad & u^T \overline{b} \\ & \overline{A}^T u \geq \overline{c} \\ & u \geq 0 \end{aligned} \tag{3.16}$$

Die Einführung von Schlupfvariablen y_N (und Variablenumbenennung $y_B := u$) ergibt:

$$\begin{aligned} -\max \quad & -\overline{b}^T y_B \\ & -\overline{A}^T y_B + I y_N = -\overline{c} \\ & y \geq 0 \end{aligned} \tag{3.17}$$

Durch eine lange Kette von Umformungen ist es uns also gelungen, ein LP in Standardform (P) in ein anderes LP in Standardform (3.17) zu transformieren. Was haben wir gewonnen? Nicht viel, es sei denn, die Basis A_B ist *dual zulässig*. Denn in diesem Falle gilt, dass die rechte Seite von (3.17), also $-\overline{c}$, nichtnegativ ist. Also ist die Matrix I eine zulässige (Ausgangs-)basis für (3.17), und wir können auf (3.17) den Simplexalgorithmus (direkt mit Phase II beginnend) anwenden.

Eine besonders interessante Anwendung ergibt sich, wenn das ursprüngliche Problem die Form

$$\begin{aligned} \max \quad & c^T x \\ & Ax \leq b \\ & x \geq 0 \end{aligned}$$

hat, wobei $c \leq 0$ und $b \not\geq 0$ ist. Hier ist eine dual zulässige Basis direkt gegeben, während eine primal zulässige Basis erst mittels Phase I gefunden werden müsste.

(3.18) Algorithmus (Duale Simplexmethode).
Eingabe: $A' \in \mathbb{K}^{(m,n)}$, $b' \in \mathbb{K}^m$, $c \in \mathbb{K}^n$.
Ausgabe: Optimale Lösung x des LP $\max\{c^T x \mid A'x = b',\, x \geq 0\}$.

Phase I: Bestimmung eines Subsystems $Ax = b$, $x \geq 0$ mit $P^=(A, b) = P^=(A', b')$, welches (9.2) erfüllt (falls möglich) und Bestimmung einer dual zulässigen Basis A_B von A. Berechne: $\overline{A} = A_B^{-1} A_N$, $\overline{b} = A_B^{-1} b$, $\overline{c}^T = c_N^T - c_B^T A_B^{-1} A_N$. (Dieser Teil des Algorithmus wird analog zur Phase I (9.24) der Grundversion des Simplexalgorithmus ausgeführt.)

Phase II: Optimierung

(II.1) (Optimalitätsprüfung)
Gilt $\overline{b}_i \geq 0$ $(i = 1, \ldots, m)$, so ist die gegenwärtige Basislösung optimal (A_B ist primal und dual zulässig). Setze $x_B = \overline{b}$ und $x_N = 0$, andernfalls gehe zu (II.2).

(II.2) (Bestimmung der Pivotzeile)
Wähle einen Index r mit $\overline{b}_r < 0$.

(II.3) (Prüfung auf Beschränktheit des Optimums)
Gilt $\overline{A}_{r\cdot} \geq 0$, so hat das duale Programm keine endliche Lösung, also ist $P^{=}(A, b) = \emptyset$. STOP!

(II.4) Berechne $\lambda_0 := \min\left\{ \frac{\overline{c}_j}{\overline{a}_{rj}} \mid \overline{a}_{rj} < 0,\ j = 1, \ldots, n \right\}$.

(II.5) (Bestimmung der Pivotspalte)
Wähle Index $s \in \left\{ j \in \{1, \ldots, n-m\} \mid \frac{\overline{c}_j}{\overline{a}_{rj}} = \lambda_0 \right\}$.

(II.6) (Pivotoperation)
Setze $A_{B'}^{-1} := E A_B^{-1}$ mit E aus ADM I, Satz (9.12), und berechne alle notwendigen Parameter neu. Gehe zu (II.1). △

Der duale Simplexalgorithmus wurde 1954 von Lemke entwickelt. Wie wir gerade gezeigt haben, ist er (nach Transformation) jedoch nichts anderes als die Anwendung des primalen Simplexalgorithmus auf das duale Problem. Aus diesem Grunde hat sich lange Zeit niemand die Mühe gemacht, eine „echte“ Implementation des dualen Verfahrens vorzunehmen. Als in den 90er Jahren die Anwendung der ganzzahligen Optimierung immer wichtiger und Schnittebenenverfahren (die wir später erklären werden) als die zentralen Methoden zur Lösung ganzzahliger Optimierungsprobleme erkannt wurden, sind erstmals duale Methoden programmiert worden. Zur Überraschung vieler erwiesen sich diese dualen Codes als (in der Praxis) schneller als die primalen, so dass sie heute bei praktischen Rechnungen dominieren. Ein Grund dafür ist, dass Goldfarbs “steepest edge” Regel beim dualen Simplexalgorithmus sehr gut funktioniert, siehe Goldfarb and Reid (1977). In Bixby (2002) finden sich ein experimenteller Nachweis und heuristische Begründungen für die bessere Performanz des dualen Simplexverfahrens.

(3.19) Bemerkung (Tableauform des dualen Simplexalgorithmus). Wollen wir das duale Simplexverfahren auf das Problem (3.14) (mit dual zulässiger Basis A_B) anwenden, so können wir das verkürzte Tableau

$$VT := \begin{array}{|c|c|} \hline -\overline{b}^T & -z_0 \\ \hline -\overline{A}^T & -\overline{c} \\ \hline \end{array}$$

verwenden und auf dieses Tableau die Update Formeln des verkürzten Simplextableaus (diese sind in Schritt (II.6) von (9.17) im ADM I Skript angegeben) anwenden. Jedoch zeigt eine einfache Überlegung, dass wir bei der Auswahl der Indizes r und s entsprechend (II.2) bzw. (II.5) die Updateformeln (II.6) aus ADM I, (9.17) auch direkt auf die Matrix $\overline{A}$ bzw. $\overline{b}$ und $\overline{c}$ anwenden können und zwar genau in der Form wie sie in ADM I, (9.17)(II.6) aufgeschrieben wurden (um das neue Tableau VT' zu erhalten).

$$VT' := \begin{array}{|c|c|} \hline -(\overline{b}')^T & -z_0' \\ \hline -(\overline{A}')^T & -\overline{c}' \\ \hline \end{array}$$

Wir führen also die Transponierung bzw. den Vorzeichenwechsel nicht explizit durch, sondern beachten den Vorzeichenwechsel bzw. die Transponierung implizit bei der Bestimmung der Pivotzeile bzw. Pivotspalte.

Dann könnten wir aber auch wieder das primale (verkürzte) Tableau verwenden und anhand dieses Tableaus sowohl den primalen als auch den dualen Simplexalgorithmus durchführen. △

(3.20) Bemerkung (Das Tucker-Tableau). Für die nachfolgende spezielle Form linearer Programme erhalten wir primale und duale Optimallösungen direkt aus der Tableaurechnung. Wir betrachten:

$$\text{(P)} \quad \begin{aligned} \max \quad & c^T x \\ & Ax \le b \\ & x \ge 0 \end{aligned} \qquad \text{und} \qquad \text{(D)} \quad \begin{aligned} \min \quad & u^T b \\ & u^T A \ge c^T \\ & u \ge 0 \end{aligned}$$

oder mit Schlupfvariablen

$$\begin{aligned} \max \quad & z_0 \\ & c^T x - z_0 = 0 \\ & Ax - b = -u \\ & x, u \ge 0 \end{aligned} \qquad \text{und} \qquad \begin{aligned} \max \quad & z_0 \\ & u^T b + z_0 = 0 \\ & -c^T + u^T A = x^T \\ & u, x \ge 0 \end{aligned}$$

oder in Tableauschreibweise

$c_1, c_2, \ldots, c_n$	z_0
A	b_1 $\vdots$ b_m

$$\begin{aligned} N &= \text{Nichtbasisvariable des primalen Programms} \\ &= \text{Basisvariable des dualen Programms} \\ B &= \text{Basisvariable des primalen Programms} \\ &= \text{Nichtbasisvariable des dualen Programms} \end{aligned}$$

Ein solches Tucker-Tableau heißt *primal zulässig*, wenn $b \ge 0$, und *dual zulässig*, wenn $c \le 0$.

Führen wir den primalen oder dualen Simplexalgorithmus bis zum Optimaltableau durch, so sind die jeweiligen Optimallösungen gegeben durch:

$$\left.\begin{aligned} x_{B(i)} &= \overline{b}_i \quad i = 1, \ldots, m \\ x_{N(i)} &= 0 \quad i = 1, \ldots, n \end{aligned}\right\} \quad \text{optimale Lösung des primalen Programms,}$$

$$\left.\begin{aligned} u_{N(i)} &= -\overline{c}_i \quad i = 1, \ldots, n \\ u_{B(i)} &= 0 \quad i = 1, \ldots, m \end{aligned}\right\} \quad \text{optimale Lösung des dualen Programms.}$$

△

Hat man eine Optimallösung von einem LP in Standardform ausgehend berechnet, so lässt sich die Optimallösung des dualen Programms nicht ohne Weiteres aus dem Tableau ablesen.

(3.21) Beispiel. Wir betrachten die folgenden zueinander dualen Programme (P) und (D). Die Lösungsmenge von (P) sei mit P bezeichnet, die von (D) mit D. P und D sind in Abbildung 3.2 dargestellt.

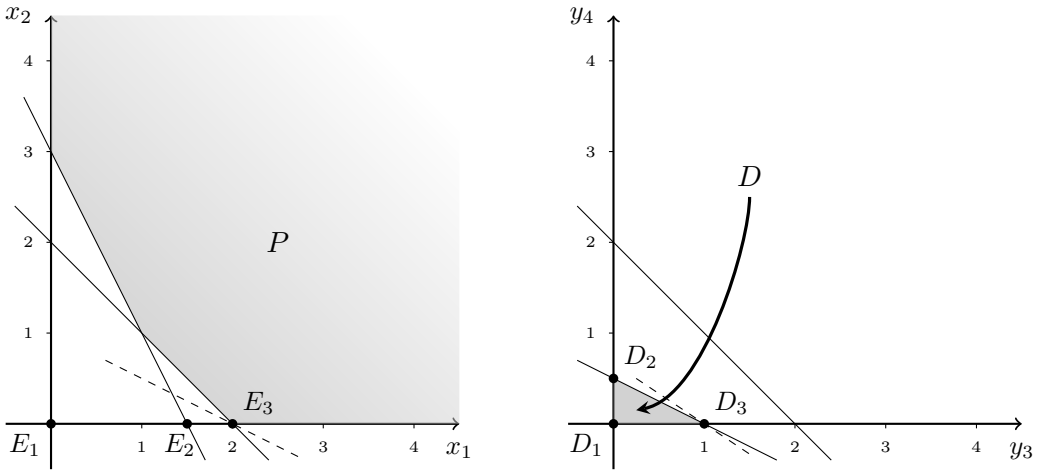


Abbildung 3.2: Primale und duale Lösungsmengen aus Beispiel (3.21)

$$\text{(P)} \quad \begin{aligned} \max \quad & -x_1 - 2x_2 & \\ & -x_1 - x_2 \leq -2 & (x_3) \\ & -2x_1 - x_2 \leq -3 & (x_4) \\ & x_1, x_2 \geq 0 & \end{aligned} \qquad \text{(D)} \quad \begin{aligned} \min \quad & -2y_3 - 3y_4 & \\ & -y_3 - 2y_4 \geq -1 & (-y_1) \\ & -y_3 - y_4 \geq -2 & (-y_2) \\ & y_3, y_4 \geq 0 & \end{aligned}$$

Wir schreiben das Tucker-Tableau zu diesem Problem auf.

	1	*2*	
	-1	-2	0
3	-1	-1	-2
4	-2	-1	-3

$N = (1, 2)$
$B = (3, 4)$

$$\begin{aligned} x_1 &= 0 \\ x_2 &= 0 \\ x_3 &= -2 \\ x_4 &= -3 \end{aligned} \qquad \begin{aligned} y_3 &= 0 \\ y_4 &= 0 \\ y_1 &= 1 \\ y_2 &= 2 \end{aligned}$$

Die gegenwärtige Basis ist dual aber nicht primal zulässig. Wir machen einen Update mit dem Pivotelement $\overline{a}_{rs} = \overline{q}_{21} = -2$. Das neue Tableau hat folgende Form.

	4	*2*	
	$-\frac{1}{2}$	$-\frac{3}{2}$	$-\frac{3}{2}$
3	$-\frac{1}{2}$	$-\frac{1}{2}$	$-\frac{1}{2}$
1	$-\frac{1}{2}$	$\frac{1}{2}$	$\frac{3}{2}$

$N = (4, 2)$
$B = (3, 1)$

$$\left.\begin{aligned} x_1 &= \tfrac{3}{2} \\ x_2 &= 0 \end{aligned}\right\} = E_2 \qquad \left.\begin{aligned} y_3 &= 0 \\ y_4 &= \tfrac{1}{2} \end{aligned}\right\} = D_2$$

Es folgt ein Pivotschritt mit $\overline{a}_{rs} = \overline{a}_{11} = -\frac{1}{2}$.

	3	*2*	
	-1	-1	-2
4	-2	1	1
1	-1	1	2

$N = (3, 2)$
$B = (4, 1)$

$$\left.\begin{aligned} x_1 &= 2 \\ x_2 &= 0 \end{aligned}\right\} = E_3 \qquad \left.\begin{aligned} y_3 &= 1 \\ y_4 &= 0 \end{aligned}\right\} = D_3$$

Dieses Tableau ist optimal. △

3.4 Postoptimierung und parametrische Programme

Wir haben bisher die Theorie (und Praxis) des Simplexverfahrens entwickelt unter der Annahme, dass die Daten, A, b, und c fest vorgegeben sind. Bei praktischen Problemen können jedoch häufig nicht alle der benötigten Zahlen mit Sicherheit angegeben werden, d. h. es kann Unsicherheit über die Beschränkungen b oder den zu erwartenden Gewinn $c^T x$ geben. Man muss also die Tatsache mit in Betracht ziehen, dass die Wirklichkeit nicht exakt in das lineare Modell abgebildet wurde bzw. werden konnte. Darüber hinaus können im Nachhinein zusätzliche Variablen oder Restriktionen auftreten, die bei Aufstellung des Modells übersehen wurden oder nicht berücksichtigt werden konnten.

Wir werden uns nun überlegen, wie wir gewisse auftretende Änderungen behandeln können, wenn wir z. B. die optimale Lösung des ursprünglichen Programms gefunden haben. Wir wollen uns auf die folgenden Fälle beschränken:

1. Variation der rechten Seite b.
2. Variation der Zielfunktion c.
3. Änderung einer Nichtbasisspalte.
4. Hinzufügung einer neuen Variablen.
5. Hinzufügung einer neuen Restriktion.

Im Prinzip könnten wir natürlich versuchen, eine Theorie zu entwerfen, bei der Schwankungen aller Daten berücksichtigt werden. Das ist jedoch außerordentlich kompliziert, und es gibt darüber kaum rechnerisch verwertbare Aussagen.

Wir gehen im weiteren davon aus, dass ein LP in Standardform (siehe (9.1) im ADM I Skript)

$$\begin{aligned} \max \quad & c^T x \\ & Ax = b \\ & x \geq 0 \end{aligned} \tag{3.22}$$

gegeben ist und dass wir eine optimale Basis A_B von A kennen.

Änderung der rechten Seite b

Wir wollen uns zunächst überlegen, welchen Einfluss eine Änderung der rechten Seite auf die Optimallösung bzw. den Wert der Optimallösung hat.

(3.23) Bemerkung. Gegeben seien ein LP in Standardform (3.22) und eine optimale Basis A_B von A. Die neue rechte Seite des LP sei $b' := b + \Delta$. Wir berechnen die neue Basislösung zur Basis A_B. Diese ergibt sich aus:

$$x'_B = A_B^{-1}(b + \Delta) = x_B + A_B^{-1}\Delta, \quad x'_N = 0.$$

Gilt $x'_B \geq 0$, so ist die neue Basislösung optimal, da sich ja an den reduzierten Kosten $\overline{c}^T = c_N^T - c_B^T A_B^{-1} A_N$ nichts geändert hat, d. h. es gilt weiterhin $\overline{c} \leq 0$.

Gilt $x'_B \not\geq 0$, so ist die neue Basislösung primal nicht zulässig. Die Optimalitätsbedingung $\overline{c} \leq 0$ ist jedoch weiterhin erfüllt. Das aber heißt nach (3.9) dass die Basis dual zulässig ist. Folglich haben wir mit A_B eine zulässige Basis für die duale Simplexmethode (3.18), und wir können direkt mit Phase II von (3.18) mit der Neuberechnung der Optimallösung beginnen. △

Exkurs. In diesem Zusammenhang sei auf eine wichtige Funktion, die die Änderung des Zielfunktionswertes eines LP bei Variationen der rechten Seite angibt, hingewiesen. Diese Funktion hat interessante Eigenschaften, von denen wir nachfolgend einige aufführen wollen. Die Funktion

$$L : \{b \in \mathbb{K}^m \mid P^=(A, b) \neq \emptyset\} \longrightarrow \mathbb{K} \cup \{\infty\}$$

definiert durch

$$L(b) := \sup\{c^T x \mid Ax = b,\, x \geq 0\} \tag{3.24}$$

heißt *Perturbationsfunktion* bzgl. des LPs

$$\begin{aligned} \max \; & c^T x \\ & Ax = b \\ & x \geq 0. \end{aligned}$$

Es bezeichne

$$\mathrm{RP}(A) := \{b \in \mathbb{K}^m \mid P^=(A,b) \neq \emptyset\}$$

den Definitionsbereich von L (wegen $0 \in \mathrm{RP}(A)$ ist $\mathrm{RP}(A) \neq \emptyset$) und

$$\mathrm{epi}(L) := \left\{ \begin{pmatrix} v \\ z \end{pmatrix} \in \mathbb{K}^{m+1} \mid v \in \mathrm{RP}(A),\, z \in \mathbb{K},\, L(v) \geq z \right\}$$

den Epigraphen von L.

(3.25) Satz. *Die Menge* $\mathrm{epi}(L)$ *ist ein polyedrischer Kegel.* △

Beweis. Offenbar ist $\mathrm{epi}(L)$ eine Projektion des polyedrischen Kegels $\{(x^T, v^T, z)^T \mid Ax - v = 0,\, x \geq 0,\, c^T x - z \geq 0\}$ auf die (v,z)-Koordinaten. Also ist nach (1.1) $\mathrm{epi}(L)$ ein Polyeder, das trivialerweise ein Kegel ist. □

(3.26) Bemerkung.

$$\exists b \in \mathrm{RP}(A) \text{ mit } L(b) < \infty \iff \forall b \in \mathrm{RP}(A) \text{ gilt } L(b) < \infty. \qquad \triangle$$

Beweis. Sei $V := \{x \in \mathbb{K}^n \mid Ax = 0\}$, dann gibt es zu jedem $b \in \mathbb{K}^m$ einen Vektor $x(b) \in \mathbb{K}^n$ mit $P^=(A,b) = (V + x(b)) \cap \mathbb{K}^n_+$. Folglich gilt für alle $b \in \mathrm{RP}(A)$: $L(b) = \infty \iff L(0) = \infty$. □

(3.27) Satz. *Ist* $L \not\equiv \infty$*, so gibt es Vektoren* $g^i \in \mathbb{K}^m$*,* $i = 1, \ldots, k$*, so dass*

$$L(v) = \min\{v^T g^i \mid i = 1, \ldots, k\}$$

für alle $v \in \mathrm{RP}(A)$ *gilt.* △

Beweis. Mit Satz (3.25) ist $K := \mathrm{epi}(L)$ ein polyedrischer Kegel. Also gibt es nach Bemerkung (2.9) aus dem ADM I Skript eine $(r, m+1)$-Matrix B, so dass $K = P(B, 0)$ gilt. Da $L(0) \geq 0$, ist $\binom{0}{z} \notin K$ für alle $z > 0$. Also kann die letzte Spalte von B kein Nullvektor sein, und wir können o. B. d. A. annehmen, dass B die Form

$$B = \begin{pmatrix} H & 0 \\ -G & h \end{pmatrix}$$

mit $h \neq 0$ besitzt. Da $L \not\equiv \infty$, folgt mit (3.26) $L(0) < \infty$, und daraus folgt direkt $L(0) = 0$. Also wissen wir, dass $\binom{0}{-1} \in K$. Dies impliziert $h > 0$. Sei o. B. d. A. $h = \mathbb{1}$. Dann gilt: $\mathrm{epi}(L) = \{(v^T, z)^T \mid Hv \leq 0,\, Gv \geq z\}$, und es ist $L(v) = z \iff G_{i\cdot} v = z$ für ein i. Bezeichnen g^i, $i = 1, \ldots, k$ die Zeilen von G, so gilt

$$L(v) = \min\{v^T g^i \mid i = 1, \ldots, k\}. \qquad \square$$

(3.28) Korollar. *Die Perturbationsfunktion* L *ist stückweise linear und konkav.* △

Speziell folgt daraus, dass sich der Wert der Zielfunktion eines linearen Programms stetig mit einer Variation der rechten Seite des LP ändert. Die stetige Änderung lässt sich durch (3.27) explizit angeben. Sie ist „fast überall“ linear, bis auf einige „Knicke“.

Änderungen der Zielfunktion c

Wir gehen wieder davon aus, dass wir eine Optimalbasis A_B von (3.22) kennen und dass sich die Zielfunktion c um Δ ändert. Da wir eine Basis A_B gegeben haben, können wir die neuen Kosten der Basislösung ausrechnen. Es gilt

$$\begin{aligned}(c^T + \Delta^T)x &= (c_B^T + \Delta_B^T)A_B^{-1}b + (c_N^T + \Delta_N^T - (c_B^T + \Delta_B^T)A_B^{-1}A_N)x_N \\ &= c_B^T\overline{b} + \overline{c}x_N + \Delta_B^T\overline{b} + \underbrace{(\Delta_N^T - \Delta_B^T\overline{A})}_{=:\overline{\Delta}}x_N.\end{aligned}$$

(a) Wird keiner der Koeffizienten von c_B^T geändert (ist also $\Delta_B = 0$), ändert sich wegen $x_N = 0$ der Zielfunktionswert der Basislösung zur Basis A_B nicht.

(b) Sind die neuen reduzierten Kosten $\overline{c} + \overline{\Delta}$ nicht positiv, so ist das Optimalitätskriterium (9.15)(b) aus dem ADM I Skript weiterhin erfüllt, d. h. A_B bleibt die optimale Basis, jedoch ändert sich u. U. der Wert der zugehörigen Basislösung wenn $\Delta_B \neq 0$.

(c) Ist einer der Koeffizienten $\overline{c}_j + \overline{\Delta}_j$ positiv, so wenden wir Phase II des primalen Simplexalgorithmus mit (zulässiger) Anfangsbasislösung A_B auf das neue Problem an.

(Zur Berechnung von $\overline{\Delta}$ genügt die Kenntnis von $\overline{A}$ oder A_B^{-1} und A_N.)

An den obigen Berechnungen kann man sehen, wie man eine Schar von linearen Programmen, bei denen entweder nur b oder nur c einer Änderung unterworfen wird, lösen kann.

Änderungen des Spaltenvektors $\mathbf{A_{\cdot j}}$, $\mathbf{j = N(s)}$

Wollen wir die s-te Nichtbasisspalte (d. h. die j-te Spalte von A) um Δ ändern und kennen wir die Basisinverse A_B^{-1}, so können wir die neue s-te Spalte von $\overline{A}$ berechnen durch

$$\overline{A}'_{\cdot s} = A_B^{-1}(A_{\cdot j} + \Delta) = \overline{A}_{\cdot s} + A_B^{-1}\Delta.$$

Die Basislösung $\overline{x}$ zu A_B bleibt weiterhin primal zulässig, da diese Änderung keinen Einfluss auf $\overline{x}_B = \overline{b}$, $\overline{x}_N = 0$ hat. Jedoch bleibt $\overline{x}$ i. A. nicht optimal, da sich der s-te Koeffizient der reduzierten Kosten ($j = N(s)$) wie folgt ändert

$$\begin{aligned}\overline{c}'_s &= c_j - c_B^T(\overline{A}_{\cdot s} + A_B^{-1}\Delta) \\ &= c_j - c_B^T\overline{A}_{\cdot s} - c_B^T A_B^{-1}\Delta \\ &= \overline{c}_s - c_B^T A_B^{-1}\Delta = \overline{c}_s - u^T\Delta,\end{aligned}$$

wobei u eine optimale duale Lösung ist.

Die Optimalität der gegenwärtigen Basislösung $\overline{x}$ bleibt genau dann erhalten, wenn $\overline{c}_s \leq u^T\Delta$ gilt. Andernfalls können wir die neue Optimallösung mit Phase II des primalen Simplexalgorithmus berechnen, wobei wir von der zulässigen Basis A_B ausgehen (die revidierte Methode ist natürlich von Vorteil).

(Bei Änderung einer Basisspalte kann man kaum Vorhersagen machen. Es muss neu invertiert werden. Dabei kann sowohl primale als auch duale Zulässigkeit verloren gehen, und es kann passieren, dass A_B nach Änderung einer Basisspalte singulär wird und damit keine Basis mehr ist.)

(3.29) Beispiel. Wir betrachten nochmals unser Beispiel (3.21)

$$\begin{aligned} \max \quad & -x_1 - 2x_2 \\ & -x_1 - x_2 \leq -2 \\ & -2x_1 - x_2 \leq -3 \\ & x_1, x_2 \geq 0 \end{aligned}$$

Das optimale (Tucker) Tableau ist:

-1	-1	-2
-2	1	1
-1	1	2

duale Lösung $y_3 = 1$, $y_4 = 0$, primale Lösung $x_1 = 2$, $x_2 = 0$.

Die Spalte $A_{\cdot 2}$ ist eine Nichtbasisspalte. Wir untersuchen, um wieviel diese Spalte geändert werden kann, ohne Optimalität zu verlieren.

$$A_{\cdot 2} = \begin{pmatrix} -1 \\ -1 \end{pmatrix}, \quad \Delta = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}, \quad \overline{c}_2 = -1, \quad u = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

also $\overline{c}_2' = -1 - (1,0)\binom{\alpha}{\beta} = -1 - \alpha$,

$$\overline{c}_2' \leq 0 \iff -1 - \alpha \leq 0 \iff \alpha \geq -1.$$

Die obige Basislösung ist also für alle linearen Programme der Form

$$\begin{aligned} \max \quad & -x_1 - 2x_2 \\ & -x_1 + (\alpha - 1)x_2 \leq -2 \\ & -2x_1 + (\beta - 1)x_2 \leq -3 \\ & x_1, x_2 \geq 0 \end{aligned}$$

optimal, wenn nur $\alpha \geq -1$ gilt. Zur geometrischen Interpretation siehe Abbildung 3.3. △

Hinzufügung einer neuen Variablen

Sei $A_{\cdot n+1}$ die neue Spalte von A und c_{n+1} der Zielfunktionswert der neuen Variablen x_{n+1}. Wir verlängern den Vektor N der Nichtbasisindizes um eine Komponente und setzen $N(n-m+1) := n+1$. Wir berechnen

$$\overline{c}_{n-m+1} := c_{n+1} - \underbrace{c_B^T A_B^{-1}}_{u^T} A_{\cdot n+1}.$$

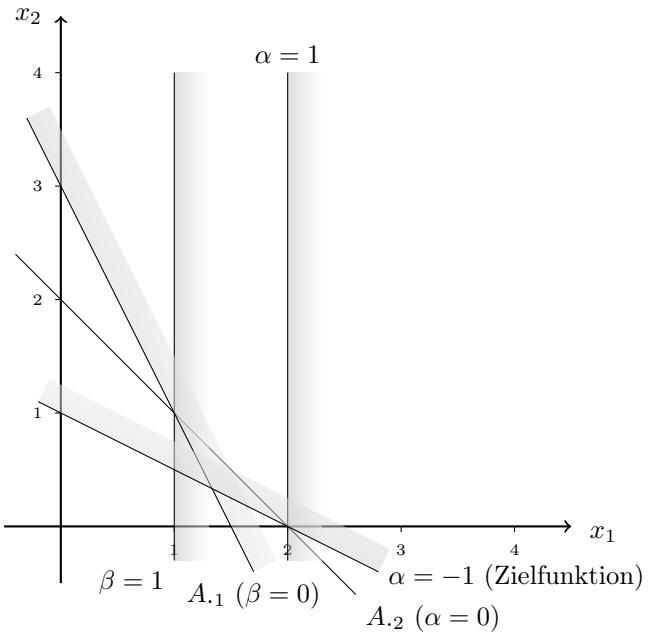


Abbildung 3.3: Änderung der Nichtbasisspalte $A_{\cdot 2}$

Gilt $\overline{c}_{n-m+1} \leq 0$, so bleibt aufgrund von ADM I, (9.15)(b) die alte Lösung optimal. Andernfalls führen wir mit der Anfangsbasis A_B den primalen Simplexalgorithmus aus. Dabei ist

$$\overline{A}_{\cdot n-m+1} = A_B^{-1} A_{\cdot n+1}.$$

Im ersten Schritt wird die neue Variable in die Basis aufgenommen.

Hinzufügung einer neuen Restriktion

Zu den bestehenden Nebenbedingungen fügen wir die neue Restriktion

$$A_{m+1 \cdot} x = \sum_{i=1}^{n} a_{m+1,i} x_i = b_{m+1}$$

hinzu.

1. Fall: Die Basislösung $\overline{x}$ zu A_B erfüllt $A_{m+1 \cdot} \overline{x} = b_{m+1}$.
In diesem Fall ist $\overline{x}$ auch die optimale Lösung des erweiterten Problems. Ist die neue Zeile linear abhängig von den übrigen Zeilen von A, so ist die neu hinzugefügte Gleichung irrelevant für das Problem und kann gestrichen werden. Ist die neue Zeile linear unabhängig von den übrigen, so ist $\overline{x}$ eine entartete Basislösung. Eine der Nichtbasisvariablen wird mit Wert 0 in die Basis aufgenommen und die Basis entsprechend erweitert.

2. Fall: Die Basislösung $\overline{x}$ erfüllt die neue Gleichung nicht.
Wir führen eine neue Schlupfvariable $x_{n+1} \geq 0$ (mit dem Zielfunktionswert $c_{n+1} = 0$) ein und verändern die neue Gleichung in

$$\sum_{i=1}^{n} a_{m+1,i} x_i \pm x_{n+1} = b_{m+1},$$

wobei wir ein + Zeichen schreiben, falls $A_{m+1.}\overline{x} > b_{m+1}$ gilt, andernfalls ziehen wir x_{n+1} ab. Wir verlängern B um eine Komponente und setzen $B(m+1) := n+1$, d. h. wir nehmen x_{n+1} mit dem Wert $\overline{b}_{m+1} = -|b_{m+1} - A_{m+1.}\overline{x}|$ in die Basis auf. Da $\overline{b}_{m+1}$ negativ ist, ist die neue Basis primal nicht zulässig; wir haben jedoch keine Änderung an der Zielfunktion vorgenommen, d. h. die reduzierten Kosten sind weiterhin nicht positiv. Also ist die gegenwärtige neue Basis dual zulässig. Die neue Basis hat die Form

$$\begin{pmatrix} A_B & 0 \\ A_{m+1,B} & 1 \end{pmatrix}.$$

Die Basisinverse lässt sich wie folgt schreiben:

$$\begin{pmatrix} A_B^{-1} & 0 \\ -A_{m+1,B}A_B^{-1} & 1 \end{pmatrix}.$$

Die neue $(m+1)$-te Zeile von $\overline{A}$ lautet:

$$-A_{m+1,B}\overline{A} + A_{m+1,N} = \overline{A}_{m+1}..$$

Wir führen nun einen dualen Pivotschritt auf der $(m+1)$-ten Zeile von $\overline{A}$ durch, wobei wir versuchen, die Variable x_{n+1} aus der Basis zu entfernen.
Ergibt der duale Beschränktheitstest ($\overline{A}_{m+1.} \geq 0$, (3.18)(II.3)), dass das duale Problem unbeschränkt ist, so ist das neue primale Problem unzulässig, d. h. die Hyperebene $A_{m+1.}x = b_{m+1}$ hat einen leeren Schnitt mit der Menge der zulässigen Lösungen des alten Problems, und wir können den Algorithmus beenden.
Andernfalls führen wir Phase II des dualen Simplexalgorithmus (3.18) aus. Endet der Algorithmus mit einer primal und dual zulässigen Lösung, so gibt es eine optimale primale Lösung x^* mit $x^*_{n+1} = 0$. Diese kann wie folgt konstruiert werden, falls x^*_{n+1} nicht bereits Null ist:
Sei x' die primale zulässige Basislösung nach Beendigung des dualen Verfahrens und $\overline{x}$ die Anfangsbasislösung bei Beginn des dualen Programms, d. h. $\overline{x}_{n+1} = \overline{b}_{m+1} < 0$. Setzen wir

$$\lambda := \frac{x'_{n+1}}{x'_{n+1} - \overline{x}_{n+1}},$$

so gilt $\lambda > 0$, da $x'_{n+1} > 0$ und $\overline{x}_{n+1} < 0$. Sei $x^* := \lambda\overline{x} + (1-\lambda)x'$, dann gilt

$$\begin{aligned} x_i^* &\geq 0 \quad \text{für } i = 1, \ldots, n, \text{ da } \overline{x}_i \geq 0,\, x' \geq 0, \\ x_{n+1}^* &= \frac{x'_{n+1}}{x'_{n+1} - \overline{x}_{n+1}}\overline{x}_{n+1} + x'_{n+1} - \frac{x'_{n+1}}{x'_{n+1} - \overline{x}_{n+1}}x'_{n+1} \\ &= \frac{1}{x'_{n+1} - \overline{x}_{n+1}}(x'_{n+1}\overline{x}_{n+1} - x'_{n+1}x'_{n+1} + x'_{n+1}x'_{n+1} - \overline{x}_{n+1}x'_{n+1}) \\ &= 0. \end{aligned}$$

Also gilt $x^* \geq 0$ und ferner

$$c^T x^* \;=\; \lambda \underbrace{c^T\overline{x}}_{\geq c^T x'} + (1-\lambda)c^T x' \;\geq\; c^T x'.$$

Daraus folgen die Zulässigkeit und die Optimalität von x^*.

3.5 Zur Numerik des Simplexverfahrens

Dieser Abschnitt ist nur äußerst kursorisch und oberflächlich ausgearbeitet. Eine sorgfältige Behandlung des Themas würde eine gesamte Spezialvorlesung erfordern. Wir empfehlen dem Leser die Bemerkungen zur Numerik in Chvátal (1983) oder das Buch von Bastian (1980) zu lesen, das einem Spezialaspekt dieses Themas gewidmet ist und diesen einigermaßen erschöpfend behandelt. Das Buch von Murtagh (1981) ist ganz Rechen- und Implementierungstechniken der linearen Optimierung gewidmet. Generelle Methoden zur Behandlung spezieller (z. B. dünn besetzter oder triangulierter oder symmetrischer) Matrizen im Rahmen numerischer Verfahren (etwa Gauß-Elimination, Matrixinvertierung) finden sich in Pissanetsky (1984).

Generell wird bei Implementationen die revidierte Simplexmethode verwendet. Es gibt zwei grundsätzliche Varianten für die Reinversion: **Produktform der Inversen (PFI)**, **Eliminationsform der Inversen (EFI)**.

Produktform der Inversen

B^{-1} liegt in Produktform vor: $B^{-1} = E_k \cdot E_{k-1} \cdot \cdots \cdot E_1$ mit E_i aus Satz (9.12) des ADM I Skripts.
Prinzip: Gauß-Jordan Verfahren.
Freiheitsgrade:

a) Positionen der Pivotelemente,

b) Reihenfolge der Pivots.
   a) hat Einfluss auf die numerische Stabilität
   b) hat Einfluss auf die Anzahl NNE (nicht Null Elemente) im Etafile (Speicherung der Spalten η von E_i).

(3.30) Beispiel.

$$B = \begin{pmatrix} 1 & 1 & 1 & 1 \\ -1 & 1 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ -1 & 0 & 0 & 1 \end{pmatrix}$$

besitzt folgende Produktform-Darstellungen der Inversen:

a) Pivotisieren auf der Hauptdiagonalen von „links oben“ nach „rechts unten“ ergibt

$$B^{-1} = \begin{pmatrix} 1 & & -\frac{1}{4} \\ & 1 & -\frac{1}{4} \\ & & -\frac{1}{4} \\ & & \frac{3}{4} \end{pmatrix} \begin{pmatrix} 1 & & -\frac{1}{3} & \\ & 1 & -\frac{1}{3} & \\ & & \frac{2}{3} & \\ & & -\frac{1}{3} & 1 \end{pmatrix} \begin{pmatrix} 1 & -\frac{1}{2} & & \\ & \frac{1}{2} & & \\ & -\frac{1}{2} & 1 & \\ & -\frac{1}{2} & & 1 \end{pmatrix} \begin{pmatrix} 1 & & & \\ 1 & 1 & & \\ 1 & & 1 & \\ 1 & & & 1 \end{pmatrix}$$

Zu speichern sind 16 Werte (+Positionen).

b) Pivotisieren auf der Hauptdiagonalen von „rechts unten“ nach „links oben“ ergibt:

$$B^{-1} = \begin{pmatrix} \frac{1}{4} & & & \\ \frac{1}{4} & 1 & & \\ \frac{1}{4} & & 1 & \\ \frac{1}{4} & & & 1 \end{pmatrix} \begin{pmatrix} 1 & -1 & & \\ & 1 & & \\ & 0 & 1 & \\ & 0 & & 1 \end{pmatrix} \begin{pmatrix} 1 & & -1 & \\ & 1 & 0 & \\ & & 1 & \\ & & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & & & -1 \\ & 1 & & 0 \\ & & 1 & 0 \\ & & & 1 \end{pmatrix}$$

Zu speichern sind 10 Werte (+Positionen) △

Folgende Gegebenheiten sollten bei der Pivotauswahl berücksichtigt werden:

(3.31) Bemerkung. Es ist günstig, in einer Spalte mit wenigen NNE zu pivotisieren, da dann die Etavektoren dünn besetzt sind. △

(3.32) Bemerkung. Es ist günstig, in einer Zeile mit wenigen NNE zu pivotisieren, da dann in anderen Zeilen der umgeformten Restmatrix potentiell weniger neue NNE entstehen. △

(3.33) Bemerkung. Es ist aus Gründen der Stabilität nicht günstig, ein Pivotelement zu wählen, dessen Betrag sehr klein ist. △

In den Inversionsroutinen, die bei den ersten Implementationen des Simplexverfahrens benutzt wurden, beachtete man nur die numerische Stabilität, nahm sich die Spalten der Basismatrix in der Reihenfolge, in der sie abgespeichert waren, multiplizierte sie mit den bereits bestimmten Elementarmatrizen und pivotisierte auf der Komponente mit dem größten Absolutbetrag. Später nahm man eine Vorsortierung der Basisspalten in der Weise vor, so dass die dünn-besetzten zuerst bearbeitet wurden: dies war ohne großen Aufwand möglich, da aufgrund der spaltenweisen Speicherung der NNE die „column counts“ (Anzahl NNE einer bestimmten Spalte) bekannt waren.

Heutzutage wird die Inversion üblicherweise in 2 Phasen zerlegt:

(3.34) Bemerkung (Boolesche Phase). Entscheidung über Pivotposition! △

(3.35) Bemerkung (Numerische Phase). Rechentechnisch günstige Ausführung des Pivotschrittes! △

Die folgende Beobachtung über Dreiecksmatrizen ist für Implementationen nützlich.

(3.36) Bemerkung. Sei B eine untere (m,m)-Dreiecksmatrix mit $b_{ii} \neq 0$, $i = 1, \ldots, m$. Dann ist durch

$$B^{-1} = E_m \cdot E_{m-1} \cdot \cdots \cdot E_2 \cdot E_1$$

und

$$E_j = \begin{pmatrix} 1 & & & 0 & & \\ & \ddots & & & & \\ & & 1 & & & \\ & & & \frac{1}{b_{jj}} & & \\ & & & -d_{j+1,j} & 1 & \\ & & & \vdots & & \ddots \\ & & & -d_{mj} & & 1 \end{pmatrix}, \qquad d_{ij} = \frac{b_{ij}}{b_{jj}}, \quad i > j$$

eine Darstellung von B^{-1} in Produktform gegeben. △

Einige Inversionsverfahren versuchen, diese schöne Eigenschaft von L-Matrizen (lower triangular) auch bei der Inversion anderer dünn besetzter Matrizen auszunutzen. Die gegebenen Basismatrizen werden dabei durch implizites Vertauschen von Zeilen und Spalten so umgeformt, dass sie L-Matrizen möglichst ähnlich werden und ein Aussehen wie z. B. in Abbildung 3.4 haben. Der Bereich, der nicht in Dreiecksform gebracht werden kann,

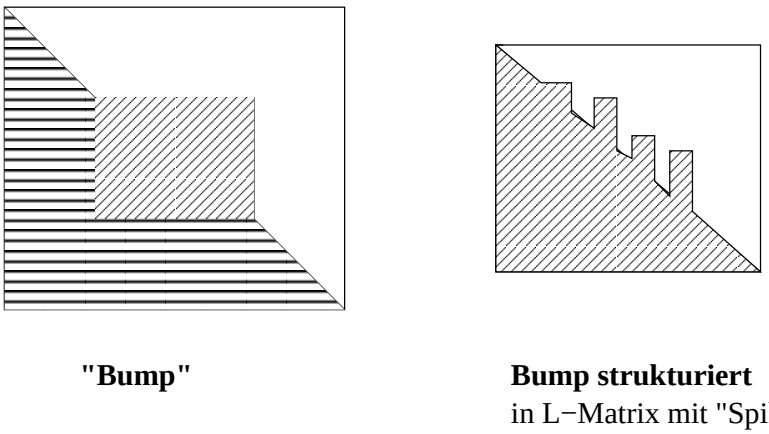


Abbildung 3.4: Matrizen mit Bumps

wird *Bump* genannt.

Diese Methode wird z. B. verwendet in der Preassigned Pivot Procedure (Hellerman und Rarick) und ist implementiert in OPTIMA (CDC) und MPS/III (IBM).

Eliminationsform der Inversen (EFI)

Die Grundlage dieser Methode bildet die folgende Beobachtung:

(3.37) Satz (LU-Zerlegung). *Ist $\overline{B}$ eine reguläre (m,m)-Matrix, so gibt es eine Matrix B, die durch Spaltenvertauschungen aus $\overline{B}$ hervorgeht und sich in der Form $B = LU$ darstellen lässt, wobei L eine untere und U eine obere Dreiecksmatrix ist. Es gilt $B^{-1} = U^{-1}L^{-1}$.* △

Prinzipielles Vorgehen: Gauß'sches Eliminationsverfahren.

Die Festlegung der Pivotpositionen und ihrer Reihenfolge erfolgt wie bei der PFI in einer vorgeschalteten Booleschen Phase. Da $U^{-1} = U_2 \cdot U_3 \cdot \cdots \cdot U_m$ mit

$$U_i = \begin{pmatrix} 1 & & -u_{1i} & & \\ & \ddots & -u_{i-1,i} & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & 1 \end{pmatrix}$$

gilt, ist der Rechenaufwand bei der Inversion geringer als bei der PFI, was sofort aus der Tatsache folgt, dass die U_i bereits durch die Spalten von U gegeben sind und die L_i sich durch weniger Rechenoperationen ergeben als die entsprechenden Elementarmatrizen bei der PFI.

Die nachfolgende Beschreibung der Inversionsroutine des LP-Codes MPSX/370 der Firma IBM ist dem oben genannten Buch von Bastian (1980) (Seite 31ff) entnommen.

(3.38) Bemerkung (Die Inversionsroutine von MPSX/370). Die Routine INVERT von MPSX/370 beruht auf der im vorhergehenden besprochenen LU-Zerlegung der Basismatrix. Sie ist zur Inversion großer dünn-besetzter Matrizen entwickelt worden; es wird daher versucht, durch eine sehr weitgehende Listenverarbeitung den Aufwand proportional zur Anzahl der von Null verschiedenen Elemente (NNE) in der Darstellung der Inversen zu halten.

(a) **Die Boolesche Phase**

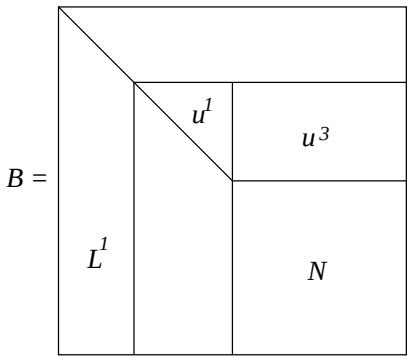
In der Booleschen Phase dieser Inversionsroutine werden die Pivotpositionen so vorbestimmt, dass die Anzahl der NNE in der Darstellung der Inversen möglichst gering wird (vgl. Bénichou et al. (1977) u. a.).

Zunächst wird die Besetzungsstruktur der Ausgangsmatrix in die Form zweier Listen übertragen: die eine Liste enthält spaltenweise die Zeilenindices der NNE, und in der anderen werden entsprechend zeilenweise die Spaltenindices gespeichert. Darüber hinaus werden noch column und row counts (Anzahl der NNE in einer Spalte bzw. Zeile) festgehalten. Auf diese Weise wird ein rascher Zugriff auf Spalten oder Zeilen nach dem Kriterium der Besetzungsdichte ermöglicht.

Die Bestimmung der Pivotpositionen erfolgt in drei Schritten, wobei die beiden ersten Schritte der Identifizierung des „Bumps“ in den oben besprochenen Triangularisierungsverfahren entsprechen:

1. Wähle Spalten mit row count 1; dadurch sind die zugehörigen Pivotzeilen ebenfalls eindeutig festgelegt.
   Die Spalten- und Zeilenindices der NNE dieser Spalten werden aus den beiden Listen entfernt und die row und column counts angepasst. Dadurch entstehen möglicherweise wieder Zeilen mit row count 1, die anschließend ausgewählt werden.

2. Entsprechend wie im Schritt 1 werden dann Spalten mit column count 1 ausgewählt, Pivotspalten und -zeilen aus den Indices-Listen gestrichen und row sowie column counts angepasst.
   Das Ergebnis der beiden ersten Schritte ist die Umordnung von Spalten der Ausgangsmatrix so, dass die vorbestimmten Pivotelemente auf der Hauptdiagonale liegen. Durch symmetrisches Vertauschen der Zeilen und Spalten gemäß der vorgesehenen Pivotfolge erhält man eine Matrix B der folgenden Gestalt:



   Der Nukleus N entspricht dem Bump beim Verfahren von Hellerman und Rarick und in der Tat könnte dies Verfahren auch zur Zerlegung des Nukleus benutzt werden.

3. Stattdessen beruht der Schritt 3 der Reinversionsroutine von MPSX/370 auf einer LU-Zerlegung des Nukleus in der Booleschen Matrix, wobei die folgenden einfachen Auswahlregeln angewandt werden:

   - wähle unter den Spalten mit minimalem column count eine solche als Pivotspalte, die ein NNE in einer Zeile mit möglichst kleinem row count besitzt;
   - wähle als Pivotzeile eine Zeile mit minimalem row count unter den Zeilen mit NNE in der Pivotspalte.

   Ist eine Pivotposition bestimmt worden, so werden die beiden Indices-Listen aktualisiert; die Anpassung der row und column counts erfolgt entsprechend.
   Durch besondere Maßnahmen wird versucht, das Durchsuchen von Listen sowie die Suche nach NNE in Vektoren einzuschränken (eine detaillierte Beschreibung der eingesetzten Methoden findet man bei Gustavson (1972). So wird die Pivotwahl z. B. dadurch vereinfacht, dass zu jeder Spalte j, in der noch nicht pivotisiert wurde, nicht nur der column count sondern auch der minimale row count unter den Zeilen mit NNE in Spalte j mitgeführt wird. Spalten mit gleichem column count und minimalem row count werden verkettet. Zusätzlich werden die Listen

der Zeilen- und Spaltenindices von NNE zu der vorbestimmten Pivotposition für die numerische Phase aufgehoben:

- die Liste der Spaltenindices in der Pivotzeile gibt die Spalten an, die aktualisiert werden müssen (durch Umspeicherung erhält man zu jeder Spalte die Adressen derjenigen Etavektoren, die zur Aktualisierung dieser Spalte herangezogen werden müssen);
- die Liste der Zeilenindices in der Pivotspalte entspricht der Besetzung mit NNE in der aktualisierten Pivotspalte, und ihre Speicherung verhindert, dass die gesamte Spalte nach NNE durchsucht werden muss.

Werden die Listen im Verlaufe zu umfangreich und damit auch die Aktualisierung zu aufwendig, so wird die Boolesche Matrix explizit als Matrix umgespeichert (ein Byte pro Element). Ist diese Matrix voll-besetzt oder erreicht sie eine vom Benutzer zu spezifizierende Dichte, so wird die Boolesche Phase abgebrochen.

(b) **Die Numerische Phase**
In der Numerischen Phase werden die Pivotspalten in der Reihenfolge, die in der Booleschen Phase vorbestimmt wurde, aktualisiert und die Etavektoren gebildet. Die Etavektoren von L^{-1} und U^{-1} werden dabei auf getrennten Files, dem L-File und dem U-File, abgespeichert. Die Etavektoren zu L^1 und zu U^1 ergeben sich direkt aus den Ausgangsdaten und werden sofort extern gespeichert. Bei der anschließenden Zerlegung der Nukleusspalten hält man die neu entstehenden Etaspalten des L-Files zunächst im Hauptspeicher, sofern genug Speicherplatz vorhanden ist.
Sei d diejenige Basisspalte, aus der die Elementarmatrizen L_k und U_k berechnet werden sollen. Ausgangspunkt ist die Spalte

$$\hat{d} := L_{k-1} \cdot \cdots \cdot L_2 \cdot L_1 \cdot d,$$

die man sich in einem Arbeitsbereich erzeugt (man beachte, dass die Etavektoren von $L_1, \ldots, L_j$ nicht benötigt werden, wenn L^1 aus j Spalten besteht).
Diejenigen Komponenten von $\hat{d}$, die in Zeilen liegen, in denen bereits pivotisiert wurde, liefern nun den Etavektor von U_k; aus den anderen ergibt sich der Etavektor von L_k.
Wegen der in der Booleschen Phase geleisteten Vorarbeiten sind hierbei nur sehr wenige Suchvorgänge erforderlich:

- die Pivotzeile ist bereits bekannt;
- diejenigen vorausgegangenen Pivots, die für die Berechnung von $\hat{d}$ relevant sind, sind bekannt; das L-File muss also nicht durchsucht werden, sondern auf die benötigten L_j kann direkt zugegriffen werden;
- die Indices der NNE in der aktualisierten Spalte sind von vornherein bekannt; dies erleichtert nicht nur die Abspeicherung der Etavektoren sondern auch das Löschen des Arbeitsbereichs für die nachfolgenden Operationen.

Ist ein vorbestimmtes Pivotelement dem Betrag nach zu klein bzw. durch Differenzbildung tatsächlich gleich Null, so wird die betreffende Spalte zunächst zurückgestellt

bis alle anderen vorbestimmten Pivots durchgeführt worden sind. Die Verarbeitung dieser Spalten ist wesentlich aufwendiger, da keine Informationen aus der Booleschen Phase verwendet werden können. Die Pivot-Wahl in den zurückgestellten Spalten sowie in denjenigen Spalten, für die in der Booleschen Phase kein Pivotelement bestimmt wurde, erfolgt nach dem Gesichtspunkt der numerischen Stabilität: man entscheidet sich für die Komponente mit dem größten Absolutbetrag in einer Zeile, in der noch nicht pivotisiert worden ist (Pivoting for Size). △

Spezialliteratur zum Simplexalgorithmus

Das folgende Literaturverzeichnis enthält eine (unvollständige) Sammlung einiger wichtiger Paper zu den in diesem Kapitel behandelten Themen.

Literaturverzeichnis

D. Avis and V. Chvátal. Notes on Bland's pivoting rule. *Mathematical Programming Studies*, 8:24–34, 1978.

R. H. Bartels. A stabilization of the simplex method. *Numerische Mathematik*, 16: 414–434, 1971.

R. H. Bartels and G. H. Golub. The simplex method of linear programming using LU decomposition. *Communications of the Association for Computing Machinery*, 12: 266–268, 275–278, 1969.

M. Bastian. *Lineare Optimierung großer Systeme.* Athenäum/Hain/Skriptor/Hanstein, Königstein, 1980.

E. M. L. Beale and J. A. Tomlin. Special facilities in a general mathematical programming system for non-convex problems using sets of variables. In J. Lawrence, editor, *Proceedings of the fifth IFORS Conference*, pages 447–454, Tavistock, London, 1970.

M. Bénichou, J. M. Gauthier, P. Girodet, G. Hentgès, G. Ribière, and O. Vincent. Experiments in mixed integer linear programming. *Mathematical Programming*, 1:76–94, 1971.

M. Bénichou, J. M. Gauthier, G. Hentgès, and G. Ribière. The efficient solution of large-scale linear programming problems – some algorithmic techniques and computational results. *Mathematical Programming*, 13:280–322, 1977.

R. E. Bixby. Solving real-world linear programs: A decade and more of progress. *Operations Research*, 50:3–15, 2002.

R. G. Bland. New finite pivoting rules for the simplex method. *Mathematics of Operations Research*, 2:103–107, 1977.

R. K. Brayton, F. G. Gustavson, and R. A. Willoughby. Some results on sparse matrices. *Mathematics of Computation*, 24:937–954, 1970.

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

H. Crowder and J. M. Hattingh. *Mathematical Programming Studies*, 4:12–25, 1975.

A. R. Curtis and J. K. Reid. On the automatic scaling of matrices for Gaussian elimination. *Journal of the Institute of Mathematics and its Applications*, 10:118–124, 1972.

G. B. Dantzig, R. P. Harvey, R. D. McKnight, and S. S. Smith. Sparse matrix techniques in two mathematical programming codes. In R. A. Willoughby, editor, *Sparse Matrix Proceedings RA-1*, pages 85–99. IBM Research Center, Yorktown Heights, New York, 1969.

R. Fletcher and S. P. J. Matthews. Stable modification of explicit LU factors for simplex updates. *Mathematical Programming*, 30:267–284, 1984.

J. J. H. Forrest and J. A. Tomlin. Updated triangular factors of the basis to maintain sparsity in the product form simplex method. *Mathematical Programming*, 2:263–278, 1972.

A. M. Geoffrion and R. E. Marsten. Integer programming algorithms: a framework and state-of-the-art survey. *Management Science*, 18:465–491, 1972.

D. Goldfarb. Using the steepest edge simplex algorithm to solve sparse linear programs. In J. Bunch and D. Rose, editors, *Sparse Matrix Computations*, pages 227–240. Academic Press, New York, 1976.

D. Goldfarb and J. K. Reid. A practicable steepest edge simplex algorithm. *Mathematical Programming*, 12:361–371, 1977.

F. G. Gustavson. Some basic techniques for solving sparse systems of linear equations. In D. J. Rose and R. A. Willoughby, editors, *Sparse Matrices and Their Applications*, pages 41–52. Plenum Press, New York, 1972.

P. M. J. Harris. Pivot selection methods of the Devex LP code. *Mathematical Programming Studies*, 4:30–57, 1975.

R. G. Jeroslow. The simplex algorithm with the pivot rule of maximizing improvement criterion. *Discrete Mathematics*, 4:367–377, 1973.

V. Klee and G. J. Minty. How good is the simplex algorithm? In O. Shisha, editor, *Inequalities – III*, pages 159–175. Academic Press, New York, 1972.

H. Kuhn and R. E. Quandt. An experimental study of the simplex method. In N. C. Metropolis, editor, *"Experimental Arithmetic, High-Speed Computing and Mathematics", Proceedings of Symposia on Applied Mathematics XV*, pages 107–124, American Mathematical Society, Providence, RI, 1963.

B. A. Murtagh. *Advanced Linear Programming: Computation and Practice*. McGraw-Hill, New York, 1981.

S. Pissanetsky. *Sparse Matrix Technology*. Academic Press, London, 1984.

M. A. Saunders. The complexity of LU updating in the simplex method. In R. S. Anderssen and R. P. Brent, editors, *The Complexity of Computational Problem Solving*, pages 214–230. Queensland University Press, Queensland, 1976a.

M. A. Saunders. A fast, stable implementation of the simplex method using bartels-golub updating. In J. Bunch and D. Rose, editors, *Sparse Matrix Computations*, pages 213–226. Academic Press, New York, 1976b.

M. J. Todd. Modifying the Forrest-Tomlin and Saunders updates for linear programming problems with variable upper bounds. Technical report, Cornell University, School of OR/IE, 1984. URL `http://hdl.handle.net/1813/8502`. Technical report 617.

J. A. Tomlin. On scaling linear programming problems. *Mathematical Programming Studies*, 4:146–166, 1975.

4 Die Ellipsoidmethode

In (9.37) im ADM I Skript haben wir angemerkt, dass man Beispiele von Polyedern und Zielfunktionen konstruieren kann, so dass der Simplexalgorithmus alle Ecken der Polyeder durchläuft. Die in den Übungen besprochene Klasse von Klee & Minty-Beispielen, bei denen dieser Fall auftritt, ist definiert durch n Variablen und $2n$ Ungleichungen. Die zugehörigen Polyeder haben 2^n Ecken. Es ist offensichtlich, dass 2^n Iterationen des Simplexalgorithmus zu nicht tolerierbaren Rechenzeiten führen. Man kann zwar zeigen (siehe Borgwardt (1982)), dass das Laufzeitverhalten des Simplexalgorithmus im Durchschnitt gut ist, aber dennoch bleibt die Frage, ob es nicht möglich ist, eine generelle „niedrige“ Schranke für die Anzahl der Iterationsschritte des Simplexalgorithmus (mit einer speziellen Zeilen- und Spaltenauswahlregel) zu finden. Dieses Problem ist bis heute ungelöst!

Im Jahre 1979 hat L. G. Khachiyan (siehe Khachiyan (1979)) gezeigt, dass lineare Programme in „polynomialer Zeit“ gelöst werden können. Das Verfahren, das er dazu angegeben hat, die sogenannte Ellipsoidmethode, arbeitet allerdings völlig anders als der Simplexalgorithmus und scheint in der Praxis im Laufzeitverhalten dem Simplexalgorithmus „im Durchschnitt“ weit unterlegen zu sein. Theoretisch beweisbar ist jedoch, dass es keine Beispiele linearer Programme (z. B. vom Klee & Minty-Typ) gibt, die die Ellipsoidmethode zu exorbitanten Laufzeiten zwingen.

Die dieser Methode zugrundeliegenden Ideen benutzen ganz andere geometrische Überlegungen, als wir sie bisher gemacht haben. Außerdem sind zu einer korrekten Implementation so viele numerische Details zu klären, dass der zeitliche Rahmen dieser Vorlesung durch eine vollständige Diskussion aller Probleme gesprengt würde. Wir wollen daher nur einen Überblick über die Methode geben und nur einige Beweise ausführen.

4.1 Polynomiale Reduktionen

In ADM I haben wir in Kapitel 4 bereits einige Grundbegriffe kennengelernt, die nötig sind, das Laufzeitverhalten eines Algorithmus exakt zu beschreiben. Hier wollen wir die wichtigsten Konzepte dieser Theorie noch einmal für den Spezialfall der linearen Programmierung darstellen.

Zunächst müssen wir unser Konzept, dass Zahlen aus dem Körper $\mathbb{K}$ gewählt werden können, aufgeben. Jede Zahl, die wir in unserem Berechnungsmodell betrachten, muss endlich kodierbar sein. Es gibt aber kein Kodierungsschema, so dass z. B. alle reellen Zahlen durch endlich viele Symbole beschrieben werden könnten. Wir beschränken uns daher auf rationale Zahlen. Haben wir eine Ungleichung

$$a^T x \leq \alpha$$

mit $a \in \mathbb{Q}^n$, $\alpha \in \mathbb{Q}$, so können wir auf einfache Weise das kleinste gemeinsame Vielfache p der Nenner der Komponenten von a und des Nenners von α bestimmen. Die Ungleichung

$$pa^T x \leq p\alpha$$

hat offenbar die gleiche Lösungsmenge wie $a^T x \leq \alpha$, während alle Koeffizienten dieser Ungleichung ganzzahlig sind. Der Fall rationaler Daten lässt sich also direkt auf den Fall ganzzahliger Daten reduzieren. Es ist zwar häufig einfacher unter der Voraussetzung ganzzahliger Daten zu rechnen, und fast alle Aufsätze zur Ellipsoidmethode machen diese Annahme, wir wollen aber dennoch für dieses Kapitel voraussetzen:

(4.1) Annahme. *Alle Daten linearer Programme sind rational, d. h. für jedes LP der Form* $\max c^T x$, $Ax \leq b$ *gilt* $c \in \mathbb{Q}^n$, $A \in \mathbb{Q}^{(m,n)}$, $b \in \mathbb{Q}^m$. △

Wir erinnern noch einmal an die Definition der *Kodierungslänge* aus Abschnitt 4.1 des ADM I Skripts. Für eine ganze Zahl n ist die Kodierungslänge gegeben durch

$$\langle n \rangle := \lceil \log_2(|n| + 1) \rceil + 1$$

und für eine rationale Zahl $r = \frac{p}{q}$ mit p und q teilerfremd und $q > 0$ durch

$$\langle r \rangle := \langle p \rangle + \langle q \rangle.$$

Die Kodierungslänge einer Matrix $A = (a_{ij}) \in \mathbb{Q}^{(m,n)}$ (oder analog eines Vektors) ist

$$\langle A \rangle := \sum_{i=1}^{m} \sum_{j=1}^{n} \langle a_{ij} \rangle.$$

Daraus folgt, dass die Kodierungslänge eines linearen Programms der Form $\max c^T x$, $Ax \leq b$ gegeben ist durch

$$\langle c \rangle + \langle A \rangle + \langle b \rangle.$$

In Abschnitt 4.1 des ADM I Skripts haben wir auch die *Laufzeit* eines Algorithmus' $\mathcal{A}$ zur Lösung eines Problems Π (kurz $L_{\mathcal{A}}(\Pi)$) definiert als die Anzahl der elementaren Rechenschritte, die während der Ausführung des Algorithmus durchgeführt wurden, multipliziert mit der Kodierungslänge der (bezüglich ihrer Kodierungslänge) größten Zahl, die während der Ausführung des Algorithmus aufgetreten ist. Der Algorithmus $\mathcal{A}$ hat dann *polynomiale Laufzeit*, wenn die Laufzeitfunktion nach oben durch ein Polynom $p : \mathbb{N} \to \mathbb{N}$ beschränkt werden kann:

$$f_{\mathcal{A}}(n) \leq p(n) \qquad \forall n \in \mathbb{N},$$

siehe ADM I, Definition (4.2).

Für die Klasse der linearen Programme der Form $\max c^T x$, $Ax \leq b$ muss also die Laufzeit eines Algorithmus durch ein Polynom in $\langle A \rangle + \langle b \rangle + \langle c \rangle$ beschränkt werden können, wenn er polynomial sein soll. Wie die Klee & Minty-Beispiele zeigen, ist der Simplexalgorithmus kein polynomialer Algorithmus zur Lösung linearer Programme!

Die Ellipsoidmethode ist kein Optimierungsverfahren, sondern lediglich eine Methode, die in einem gegebenen Polyeder einen Punkt findet, falls ein solcher existiert. Wir müssen daher das allgemeine lineare Optimierungsproblem auf diesen Fall zurückführen. Aus ADM I, Abschnitt 2.3 (allgemeine Transformationsregeln) wissen wir, dass jedes lineare Programm in der Form

$$\begin{aligned} \max\ & c^T x \\ & Ax \leq b \\ & x \geq 0 \end{aligned} \tag{4.2}$$

geschrieben werden kann. Das zu (4.2) duale Programm ist

$$\begin{aligned} \min\ & b^T y \\ & A^T y \geq c \\ & y \geq 0. \end{aligned} \tag{4.3}$$

Aus dem Dualitätssatz (11.24) im ADM I Skript wissen wir, dass (4.2) und (4.3) genau dann optimale Lösungen haben, deren Werte gleich sind, wenn beide Programme zulässige Lösungen haben. Daraus folgt, dass jedes Element $\binom{x}{y}$ des Polyeders P, definiert durch

$$\begin{pmatrix} -c^T & b^T \\ A & 0 \\ 0 & -A^T \\ -I & 0 \\ 0 & -I \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \leq \begin{pmatrix} 0 \\ b \\ -c \\ 0 \\ 0 \end{pmatrix} \tag{4.4}$$

eine Optimallösung x von (4.2) und eine Optimallösung y von (4.3) bestimmt. Dies impliziert, dass es zur Lösung von (4.2) genügt, einen Punkt in (4.4) zu finden.

Der obige „Trick“, das Primal- und das Dualprogramm zusammenzufassen, bläht natürlich das zu bearbeitende System ungeheuer auf. Eine andere Methode der Reduktion ist die binäre Suche, die wir nun schildern. Zur korrekten Darstellung benötigen wir jedoch noch einige (auch für andere Zwecke) wichtige Hilfssätze. Zunächst wollen wir einige Parameter durch die Kodierungslänge abschätzen. Um den ersten Hilfssatzes zu beweisen, benötigen wir die bekannte Hadamard-Ungleichung, die besagt, dass das Volumen eines Parallelepipeds in $\mathbb{R}^n$ mit Kantenlängen $d_1, \ldots, d_n$ nicht größer ist als das Volumen des Würfels mit Kantenlängen $d_1, \ldots, d_n$. Bekanntlich ist das Volumen eines Parallelepipeds, das durch die Vektoren $a_1, \ldots, a_n$ aufgespannt wird, nichts anderes als der Absolutbetrag der Determinante der Matrix A mit Spalten $A_{\cdot 1} = a_1, \ldots, A_{\cdot n} = a_n$. Die *Hadamard-Ungleichung* lautet also

$$|\det A| \leq \prod_{j=1}^{n} \|A_{\cdot j}\|_2 \tag{4.5}$$

(4.6) Lemma.

(a) *Für jede Zahl $r \in \mathbb{Q}$ gilt:* $|r| \leq 2^{\langle r \rangle - 1} - 1$.

(b) *Für je zwei rationale Zahlen r, s gilt:* $\langle rs\rangle \le \langle r\rangle + \langle s\rangle$.

(c) *Für jeden Vektor $x \in \mathbb{Q}^n$ gilt:* $\|x\|_2 \le \|x\|_1 \le 2^{\langle x\rangle - n} - 1$.

(d) *Für jede Matrix $A \in \mathbb{Q}^{(n,n)}$ gilt:* $|\det(A)| \le 2^{\langle A\rangle - n^2} - 1$. △

Beweis. (a) und (b) folgen direkt aus der Definition.
(c) Sei $x = (x_1, \ldots, x_n)^T \in \mathbb{Q}^n$, dann gilt nach (a)

$$1 + \|x\|_1 \;=\; 1 + \sum_{i=1}^{n} |x_i| \;\le\; \prod_{i=1}^{n} (1 + |x_i|) \;\le\; \prod_{i=1}^{n} 2^{\langle x_i\rangle - 1} \;=\; 2^{\langle x\rangle - n}.$$

Die Ungleichung $\|x\|_2 = \sqrt{\sum_{i=1}^n x_i^2} \le \sum_{i=1}^n |x_i| = \|x\|_1$ ist trivial.
(d) Aus der Hadamard-Ungleichung (4.5) und (c) folgt

$$1 + |\det(A)| \;\le\; 1 + \prod_{j=1}^{n} \|A_{\cdot j}\|_2 \;\le\; \prod_{j=1}^{n} (1 + \|A_{\cdot j}\|_2) \;\le\; \prod_{j=1}^{n} 2^{\langle A_{\cdot j}\rangle - n} \;=\; 2^{\langle A\rangle - n^2}.$$

□

Diesen Hilfssatz können wir zur Abschätzung der „Größe“ der Ecken von Polyedern wie folgt benutzen.

(4.7) Satz. *Für jede Ecke $v = (v_1, \ldots, v_n)^T$ eines Polyeders P der Form $P(A, b)$, $P^{=}(A, b)$ oder $P = \{x \in \mathbb{R}^n \mid Ax \le b, x \ge 0\}$, $A \in \mathbb{Q}^{(m,n)}$, $b \in \mathbb{Q}^m$, gilt*

(a) *Der Absolutbetrag des Zählers von v_i ist höchstens $2^{\langle A\rangle + \langle b\rangle - n^2}$, der Absolutbetrag des Nenners von v_i ist höchstens $2^{\langle A\rangle - n^2}$, $i = 1, \ldots, n$.*

(b) $|v_i| \le 2^{2\langle A\rangle + \langle b\rangle - 2n^2}$, $i = 1, \ldots, n$.

(c) *Falls $A \in \mathbb{Z}^{(m,n)}$, so gilt $|v_i| \le 2^{\langle A\rangle + \langle b\rangle - n^2}$, $i = 1, \ldots, n$.* △

Beweis. Ist v Ecke von P, so gibt es nach Satz (8.8) aus dem ADM I Skript eine reguläre Untermatrix und einen entsprechenden Untervektor des Restriktionensystems von P, so dass v die eindeutig bestimmte Lösung des hierdurch bestimmten Gleichungssystems ist. Wir führen den Fall $P = \{x \mid Ax \le b, x \ge 0\}$ vor. Die beiden anderen Fälle verlaufen analog.

Es gibt also eine (n, n)-Matrix D, deren Zeilen Zeilen von A oder negative Einheitsvektoren sind und einen entsprechenden Vektor d, dessen Komponenten Elemente von b oder Nullen sind, so dass v die eindeutige Lösung von $Dx = d$ ist. Nach der Cramerschen Regel gilt dann für $i = 1, \ldots, n$

$$v_i = \frac{\det D_i}{\det D},$$

wobei D_i aus D dadurch entsteht, dass die i-te Spalte von D durch den Vektor d ersetzt wird. Hat D Zeilen, die negative Einheitsvektoren sind, so bezeichnen wir mit $\bar{D}$ die

Matrix, die durch Streichen dieser Zeilen und der Spalten, in denen sich die Elemente -1 befinden, entsteht. Aufgrund des Determinantenentwicklungssatzes gilt $|\det D| = |\det \bar{D}|$, und ferner ist $\bar{D}$ eine Untermatrix von A. Daraus folgt mit (4.6)(d)

$$|\det D| = |\det \bar{D}| \leq 2^{\langle \bar{D} \rangle - n^2} \leq 2^{\langle A \rangle - n^2}.$$

Analog folgt

$$|\det D_i| \leq 2^{\langle A \rangle + \langle b \rangle - n^2}.$$

Damit ist (a) bewiesen.

Ist $|\det D| \geq 1$, dies ist z. B. dann der Fall, wenn $A \in \mathbb{Z}^{(m,n)}$ gilt, so erhalten wir mit (a)

$$|v_i| \leq |\det D_i| \leq 2^{\langle A \rangle + \langle b \rangle - n^2}.$$

Hieraus folgt (c).

Ist $|\det D| < 1$, so müssen wir $|\det D|$ nach unten abschätzen. Sei $q = \prod_{i,j} q_{ij}$ das Produkt der Nenner der Elemente $d_{ij} = \frac{p_{ij}}{q_{ij}}$ von D. Dann gilt $|\det D| = \frac{q|\det D|}{q}$, und sowohl q als auch $q|\det D|$ sind ganze Zahlen. Aus (4.6)(a), (b) folgt

$$q = \prod_{i,j} q_{ij} \leq 2^{\langle \prod_{i,j} q_{ij} \rangle} \leq 2^{\sum_{i,j} \langle q_{ij} \rangle} \leq 2^{\langle D \rangle - n^2} \leq 2^{\langle A \rangle - n^2}.$$

Daraus ergibt sich

$$|v_i| \leq \frac{|\det D_i|}{|\det D|} = \frac{|\det D_i| q}{q|\det D|} \leq |\det D_i| q \leq 2^{2\langle A \rangle + \langle b \rangle - 2n^2}.$$

Damit ist auch (b) bewiesen. □

Da nach ADM I, Satz (8.11) alle Polyeder P der Form $P = \{x \mid Ax \leq b, x \geq 0\}$ spitz sind und nach ADM I, Folgerung (8.14) lineare Programme über spitzen Polyedern optimale Ecklösungen haben (falls sie überhaupt Optimallösungen haben) folgt:

(4.8) Satz. *Das lineare Programm* (4.2) *hat eine Optimallösung genau dann, wenn die beiden linearen Programme*

$$\begin{aligned} \max \quad & c^T x \\ & Ax \leq b \\ & x \geq 0 \\ & x_i \leq 2^{2\langle A \rangle + \langle b \rangle - 2n^2} \qquad i = 1, \ldots, n \end{aligned} \tag{4.9}$$

$$\begin{aligned} \max \quad & c^T x \\ & Ax \leq b \\ & x \geq 0 \\ & x_i \leq 2^{2\langle A \rangle + \langle b \rangle - 2n^2} + 1 \qquad i = 1, \ldots, n \end{aligned} \tag{4.10}$$

eine Optimallösung haben und die Werte der Optimallösungen übereinstimmen. Die optimalen Zielfunktionswerte von (4.9) *und* (4.10) *stimmen genau dann nicht überein, wenn* (4.2) *unbeschränkt ist.* (4.2) *ist genau dann nicht lösbar, wenn* (4.9) *oder* (4.10) *nicht lösbar ist.* △

Wir haben damit das lineare Programmierungsproblem (4.2) über einem (allgemeinen) Polyeder auf die Lösung zweier linearer Programme über Polytopen reduziert. Wir müssen also im Prinzip nur zeigen, wie man LPs über Polytopen löst.

(4.11) Satz. *Ist $P \neq \emptyset$ ein Polytop der Form $P(A,b)$, $P^{=}(A,b)$ oder $P = \{x \mid Ax \leq b, x \geq 0\}$ mit $A \in \mathbb{Q}^{(m,n)}$, $b \in \mathbb{Q}^m$, so kann für $c \in \mathbb{Q}^n$ das lineare Programm $\max c^T x$, $x \in P$ nur Optimalwerte in der endlichen Menge*

$$S := \{\frac{p}{q} \in \mathbb{Q} \mid |p| \leq n \cdot 2^{\langle A\rangle+\langle b\rangle+2\langle c\rangle-n^2-n},\ 1 \leq q \leq 2^{\langle A\rangle+\langle c\rangle-n^2-n}\}$$

annehmen. △

Beweis. Da alle Optimalwerte Zielfunktionswerte von Ecken von P sind, brauchen wir nur die Werte $c^T v$, v Ecke von P, abzuschätzen. Sei also $v = (v_1, \ldots, v_n)^T$ eine Ecke von P. Wie im Beweis von (4.7) können wir feststellen, dass die Komponenten v_i von v eine Darstellung der Form

$$v_i = \frac{\det D_i}{\det D} =: \frac{p_i}{d}$$

haben, wobei D eine reguläre Untermatrix von A bzw. $\binom{A}{I}$ ist und D_i aus D dadurch hervorgeht, dass die i-te Spalte von D durch einen Untervektor der rechten Seite ersetzt wird. Satz (4.7) zeigt $d \leq 2^{\langle A\rangle-n^2}$, $p_i \leq 2^{\langle A\rangle+\langle b\rangle-n^2}$. Sei nun $c = (\frac{s_1}{t_1}, \ldots, \frac{s_n}{t_n})^T \in \mathbb{Q}^n$ eine teilerfremde Darstellung der Zielfunktion, so gilt

$$c^T v = \sum_{i=1}^{n} \frac{s_i p_i}{t_i d} = \frac{1}{dt} \sum_{i=1}^{n} s_i p_i \bar{t}_i \quad \text{mit} \quad t := t_1 \cdot \cdots \cdot t_n,\ \bar{t}_i := \frac{t}{t_i}.$$

Aus (4.6)(a) folgt $t = t_1 \cdot \cdots \cdot t_n \leq 2^{\langle t_1\rangle-1} \cdot \cdots \cdot 2^{\langle t_n\rangle-1} = 2^{\sum_i(\langle t_i\rangle-1)} \leq 2^{\langle c\rangle-n}$ und somit

$$q := dt \leq 2^{\langle A\rangle+\langle c\rangle-n^2-n}.$$

Analog folgt

$$p := \sum_{i=1}^{n} s_i p_i \bar{t}_i \leq \sum_{i=1}^{n} 2^{\langle s_i\rangle-1} 2^{\langle A\rangle+\langle b\rangle-n^2} 2^{\langle c\rangle-n} \leq n \cdot 2^{\langle A\rangle+\langle b\rangle+2\langle c\rangle-n^2-n}.$$

□

Satz (4.11) gibt uns nun die Möglichkeit, das Verfahren der binären Suche zur Lösung von $\max c^T x$, $x \in P$ anzuwenden. Diese Methode funktioniert bezüglich der in Satz (4.11) definierten Menge S wie folgt.

(4.12) Algorithmus Binäre Suche.

1. Wähle ein Element $s \in S$, so dass für $S' := \{t \in S \mid t < s\}$ und $S'' := \{t \in S \mid t \geq s\}$ gilt
$$|S'| \leq |S''| \leq |S'| + 1.$$

2. Überprüfe, ob das Polyeder
$$P_s = \{x \mid Ax \leq b, x \geq 0, c^T x \geq s\}$$
nicht leer ist.

3. Ist P_s leer, so setze $S := S'$, andernfalls setze $S := S''$.

4. Ist $|S| = 1$, so gilt für $s \in S$: $s = \max\{c^T x \mid Ax \leq b,\ x \geq 0\}$, und jeder Punkt in P_s ist eine Optimallösung von $\max c^T x$, $x \in P$. Andernfalls gehe zu 1. △

Die Korrektheit dieses Verfahrens ist offensichtlich. Ist die Binärsuche auch effizient? Da in jedem Schritt die Kardinalität $|S|$ von S (fast) halbiert wird, ist klar, dass höchstens

$$\overline{\overline{N}} := \lceil \log_2(|S|) \rceil \tag{4.13}$$

Aufspaltungen von S in zwei Teile notwendig sind, um eine einelementige Menge zu erhalten. Also muss Schritt 2 von (4.12) $\overline{\overline{N}}$ mal ausgeführt werden. Nach (4.11) gilt

$$|S| \leq 2n \cdot 2^{2\langle A\rangle + \langle b\rangle + 3\langle c\rangle - 2n^2 - 2n} + 1,$$

und daraus folgt, dass Test 2 von (4.12) höchstens

$$\overline{N} := 2\langle A\rangle + \langle b\rangle + 3\langle c\rangle - 2n^2 - 2n + \log_2 n + 2 \tag{4.14}$$

mal durchgeführt wurde. Ist Test 2 in einer Zeit ausführbar, die polynomial in $\langle A\rangle + \langle b\rangle + \langle c\rangle$ ist, so ist auch die binäre Suche ein polynomialer Algorithmus, da ein polynomialer Algorithmus für (4.12). nur $\overline{N}$ mal, also nur polynomial oft ausgeführt werden muss.

(4.15) Korollar. *Es gibt einen polynomialen Algorithmus zur Lösung linearer Programme genau dann, wenn es einen Algorithmus gibt, der in polynomialer Zeit entscheidet, ob ein Polytop P leer ist oder nicht und der, falls $P \neq \emptyset$, einen Punkt in P findet.* △

Damit haben wir das lineare Programmierungsproblem reduziert auf die Frage der Lösbarkeit von Ungleichungssystemen, deren Lösungsmenge beschränkt ist.

4.2 Beschreibung der Ellipsoidmethode

In diesem Abschnitt setzen wir $n \geq 2$ voraus. Wir wollen zunächst einige Eigenschaften von Ellipsoiden beschreiben. Wir erinnern daran, dass Ellipsoide volldimensionale konvexe Körper im $\mathbb{R}^n$ sind, die eine besonders einfache Darstellung haben. Und zwar ist eine Menge $E \subseteq \mathbb{R}^n$ ein *Ellipsoid (mit Zentrum a)* genau dann, wenn es einen Vektor $a \in \mathbb{R}^n$ und eine (symmetrische) positiv definite Matrix A gibt, so dass gilt

$$E = E(A, a) := \{x \in \mathbb{R}^n \mid (x - a)^T A^{-1}(x - a) \leq 1\} \tag{4.16}$$

Aus der linearen Algebra wissen wir, dass für symmetrische Matrizen $A \in \mathbb{R}^{(n,n)}$ die folgenden Aussagen äquivalent sind:

(i) A ist positiv definit.

(ii) A^{-1} ist positiv definit.

(iii) Alle Eigenwerte von A sind positive reelle Zahlen.

(iv) $\det(A_{II}) > 0$ für alle Mengen $I = \{1, \ldots, i\}$, $i = 1, \ldots, n$.

(v) $A = B^T B$ für eine reguläre Matrix $B \in \mathbb{R}^{(n,n)}$.

Das Ellipsoid $E(A, a)$ ist durch die (ebenfalls positiv definite) Inverse A^{-1} von A definiert. Das erscheint zunächst seltsam, liegt aber daran, dass sich viele Eigenschaften von $E(A, a)$ aus algebraischen Eigenschaften von A ableiten lassen. Z. B. ist der Durchmesser (d. h. die Länge der längsten Achse) von $E(A, a)$ gleich $2\sqrt{\Lambda}$, wobei Λ der größte Eigenwert von A ist. Die längsten Achsen sind durch die Eigenvektoren, die zu Λ gehören, gegeben. Die Symmetrieachsen von $E(A, a)$ entsprechen ebenfalls den Eigenvektoren von A. In Abbildung 4.1 ist das Ellipsoid $E(A, 0)$ dargestellt mit

$$A = \begin{pmatrix} 16 & 0 \\ 0 & 4 \end{pmatrix}.$$

Die Eigenwerte von A sind $\Lambda = 16$ und $\lambda = 4$ mit den zugehörigen Eigenvektoren $e_1 = (1, 0)^T$ und $e_2 = (0, 1)^T$. Der Durchmesser von $E(A, 0)$ ist $2\sqrt{\Lambda} = 8$.

Zu jeder positiv definiten Matrix A gibt es eine eindeutig bestimmte positive definite Matrix, die mit $A^{\frac{1}{2}}$ bezeichnet und *Wurzel von A* genannt wird, mit

$$A = A^{\frac{1}{2}} A^{\frac{1}{2}}.$$

Die Einheitskugel $S(0, 1) \subseteq \mathbb{R}^n$ (um den Nullpunkt mit Radius 1) ist das Ellipsoid $E(I, 0)$. Man rechnet leicht nach, dass gilt:

$$E(A, a) = A^{\frac{1}{2}} S(0, 1) + a.$$

Damit sei die Aufzählung von Eigenschaften von Ellipsoiden beendet.

Wir wollen nun die geometrische Idee, die hinter der Ellipsoidmethode steckt, erläutern.

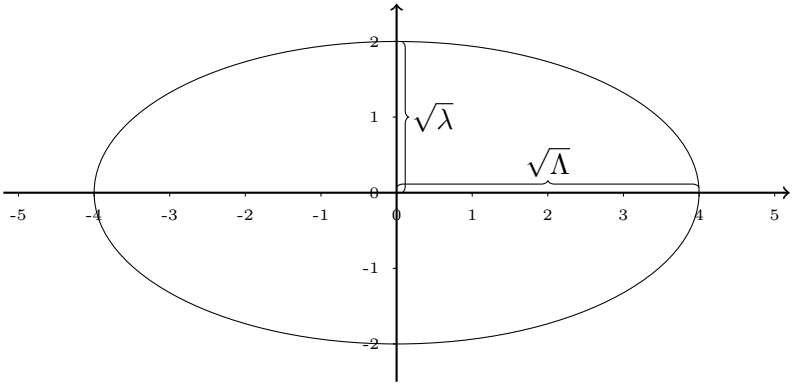


Abbildung 4.1: Ellipsoid mit Symmetrieachsen der Längen $2\sqrt{\Lambda} = 8$ und $2\sqrt{\lambda} = 4$

(4.17) Algorithmus (Geometrische Beschreibung der Ellipsoidmethode). Gegeben sei ein Polytop P. Wir wollen einen Punkt in P finden oder beweisen, dass P leer ist.

1. Konstruiere ein Ellipsoid $E_0 = E(A_0, a_0)$, das P enthält. Setze $k := 0$.
2. Ist das gegenwärtige Ellipsoid E_k „zu klein“, so brich ab mit der Antwort „P ist leer“. STOP.
3. Teste ob der Mittelpunkt a_k von E_k in P enthalten ist.
4. Gilt $a_k \in P$, dann haben wir unser Ziel erreicht, STOP.
5. Gilt $a_k \notin P$, dann gibt es eine P definierende Ungleichung, sagen wir

   $$c^T x \leq \gamma,$$

   die von a_k verletzt wird, d. h. $c^T a_k > \gamma$. Mit

   $$E'_k := E_k \cap \{x \mid c^T x \leq c^T a_k\}$$

   bezeichnen wir das Halbellipsoid von E_k, das P enthält. Wir konstruieren nun das Ellipsoid kleinsten Volumens, das E'_k enthält und nennen es E_{k+1}.
6. Setze $k := k + 1$ und gehe zu 2. △

Das Prinzip des Verfahrens ist klar. Man zäunt das Polytop P durch ein Ellipsoid E_k ein. Ist das Zentrum von E_k nicht in P, so sucht man sich ein kleineres Ellipsoid E_{k+1}, das P enthält und fährt so fort. Die Probleme, die zu klären sind, sind die folgenden:

- Wie findet man ein Ellipsoid, das P enthält?
- Wie kann man E_{k+1} aus E_k konstruieren?

- Wann kann man abbrechen, d. h. was heißt „zu klein“?
- Wieviele Iterationen des Verfahrens sind durchzuführen?

Die nachfolgenden Sätze beantworten diese Fragen, wobei wir nur einige der Beweise, die zum Nachweis der Polynomialität des Verfahrens notwendig sind, angeben wollen.

Unser Anfangsellipsoid soll eine Kugel mit dem Nullpunkt als Zentrum sein. Enthalten die Restriktionen, die das Polytop P definieren, explizite obere und untere Schranken für die Variablen, sagen wir

$$l_i \leq x_i \leq u_i \qquad i = 1, \ldots, n$$

so sei

$$R := \sqrt{\sum_{i=1}^{n} (\max\{|u_i|, |l_i|\})^2}. \tag{4.18}$$

Dann gilt trivialerweise $P \subseteq S(0, R) = E(R^2 I, 0)$. Andernfalls kann man zeigen

(4.19) Lemma. *Sei P ein Polyeder der Form $P(A, b)$, $P^{=}(A, b)$ oder $\{x \in \mathbb{R}^n \mid Ax \leq b,\ x \geq 0\}$ mit $A \in \mathbb{Q}^{(m,n)}$, $b \in \mathbb{Q}^m$, dann gilt*

(a) *Alle Ecken von P sind in der Kugel $S(0, R)$ enthalten mit*

$$R := \sqrt{n} 2^{2\langle A \rangle + \langle b \rangle - 2n^2}.$$

(b) *Ist P ein Polytop, so gilt $P \subseteq S(0, R) = E(R^2 I, 0)$.* △

Beweis. Nach Satz (4.7) gilt für jede Ecke $v^T = (v_1, \ldots, v_n)$ von P

$$|v_i| \leq 2^{2\langle A \rangle + \langle b \rangle - 2n^2} \qquad (i = 1, \ldots, n),$$

und daraus folgt für die euklidische Norm von v:

$$\|v\| \;=\; \sqrt{\sum_{i=1}^{n} v_i^2} \;\leq\; \sqrt{n \max\{v_i^2\}} \;\leq\; \sqrt{n}\, 2^{2\langle A \rangle + \langle b \rangle - 2n^2}.$$

Also ist jede Ecke von P in $S(0, R)$ enthalten. Ist insbesondere P ein Polytop, so folgt daraus $P \subseteq S(0, R)$. □

Damit haben wir durch (4.18) oder (4.19) ein Anfangsellipsoid E_0 gefunden, mit dem wir die Ellipsoidmethode beginnen können. Die Konstruktion von E_{k+1} aus E_k geschieht wie folgt.

(4.20) Satz. *Sei $E_k = E(A_k, a_k) \subseteq \mathbb{R}^n$ ein Ellipsoid, $c \in \mathbb{R}^n \setminus \{0\}$ und $E'_k := \{x \in \mathbb{R}^n \mid c^T x \leq c^T a_k\} \cap E_k$. Setze*

$$d := \frac{1}{\sqrt{c^T A_k c}} A_k c, \tag{4.21}$$

$$a_{k+1} := a_k - \frac{1}{n+1} d, \tag{4.22}$$

$$A_{k+1} := \frac{n^2}{n^2-1}(A_k - \frac{2}{n+1} dd^T), \tag{4.23}$$

dann ist A_{k+1} positiv definit und $E_{k+1} := E(A_{k+1}, a_{k+1})$ ist das eindeutig bestimmte Ellipsoid minimalen Volumens, das E'_k enthält. △

Beweis. Siehe Bland et al. (1981) oder Grötschel et al. (1988). □

Wir wollen kurz den geometrischen Gehalt der Schritte in Satz (4.20) erläutern.

Ist $c \in \mathbb{R}^n$, und wollen wir das Maximum oder Minimum von $c^T x$ über E_k finden, so kann man dies explizit wie folgt angeben. Für den in (4.21) definierten Vektor d und

$$z_{\max} := a_k + d, \qquad z_{\min} := a_k - d$$

gilt

$$c^T z_{\max} = \max\{c^T x \mid x \in E_k\} = c^T a_k + \sqrt{c^T A_k c},$$
$$c^T z_{\min} = \min\{c^T x \mid x \in E_k\} = c^T a_k - \sqrt{c^T A_k c}.$$

Diese Beziehung kann man leicht – z. B. aus der Cauchy-Schwarz-Ungleichung – ableiten. Daraus folgt, dass der Mittelpunkt a_{k+1} des neuen Ellipsoids E_{k+1} auf dem Geradenstück zwischen a_k und $z_{\min}$ liegt. Die Länge dieses Geradenstücks ist $\|d\|$, und a_{k+1} erreicht man von a_k aus, indem man einen Schritt der Länge $\frac{1}{n+1}\|d\|$ in Richtung $-d$ macht. Der Durchschnitt des Randes des Ellipsoids E_{k+1} mit dem Rand von E'_k wird gebildet durch den Punkt $z_{\min}$ und $E''_k := \{x \mid (x-a_k)^T A_k (x-a_k) = 1\} \cap \{x \mid c^T x = c^T a_k\}$. E''_k ist der Rand eines „$(n-1)$-dimensionalen Ellipsoids“ im $\mathbb{R}^n$.

In Abbildung 4.2 sind das Ellipsoid E_k aus Abbildung 4.1 und das Ellipsoid E_{k+1}, definiert durch Satz (4.20) bezüglich des Vektors $c^T = (-1, -2)$ dargestellt. E'_k ist grau eingezeichnet. Als Anfangsdaten haben wir daher: Die verletzte Ungleichung ist $-x_1 - 2x_2 \leq -4$. Die zugehörige Gerade $\{x \mid -x_1 - 2x_2 = -4\}$ ist ebenfalls gezeichnet.

$$a_k = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad A_k = \begin{pmatrix} 16 & 0 \\ 0 & 4 \end{pmatrix}, \quad c = \begin{pmatrix} -1 \\ -2 \end{pmatrix}.$$

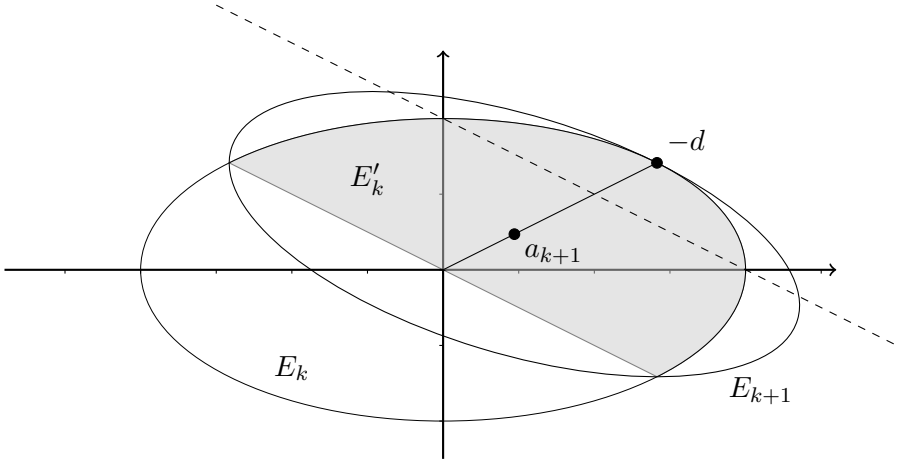


Abbildung 4.2: Ellipsoid E_k aus Abbildung 4.1 und daraus konstruiertes Ellipsoid E_{k+1}

Die Formeln (4.21), (4.22), (4.23) ergeben:

$$\begin{aligned} d &= \frac{-1}{\sqrt{2}}\begin{pmatrix}4\\2\end{pmatrix}, \\ a_{k+1} &= a_k - \frac{1}{3}d = \frac{1}{3\sqrt{2}}\begin{pmatrix}4\\2\end{pmatrix} \approx \begin{pmatrix}0.9428\\0.4714\end{pmatrix}, \\ A_{k+1} &= \frac{n^2}{n^2-1}(A_k - \frac{2}{n+1}dd^T) = \frac{4}{9}\begin{pmatrix}32 & -8\\-8 & 8\end{pmatrix}, \\ E_{k+1} &= E(A_{k+1}, a_{k+1}). \end{aligned}$$

Das Stopkriterium der Ellipsoidmethode beruht auf einem Volumenargument. Nach Konstruktion ist klar, dass das Volumen von E_{k+1} (bezeichnet mit $\mathrm{vol}(E_{k+1})$) kleiner ist als das von E_k. Man kann den Volumenschrumpfungsfaktor explizit berechnen. Er hängt nur von der Dimension n des Raumes $\mathbb{R}^n$ und nicht etwa von Update-Vektor c ab.

(4.24) Lemma.

$$\frac{\mathrm{vol}(E_{k+1})}{\mathrm{vol}(E_k)} \;=\; \left(\left(\frac{n}{n+1}\right)^{n+1}\left(\frac{n}{n-1}\right)^{n-1}\right)^{\frac{1}{2}} \;\leq\; e^{-\frac{1}{2n}} \;<\; 1. \qquad \triangle$$

Beweis. Siehe Grötschel et al. (1988), Lemma (3.1.34). □

Wir sehen also, dass mit den Formeln aus Satz (4.20) eine Folge von Ellipsoiden konstruiert werden kann, so dass jedes Ellipsoid E_{k+1} das Halbellipsoid E'_k und somit das Polytop P enthält und dass die Volumina der Ellipsoide schrumpfen. Die Folge der Volumina konvergiert gegen Null, muss also einmal das Volumen von P, falls P ein positives Volumen hat, unterschreiten. Daher muss nach endlich vielen Schritten der Mittelpunkt eines der Ellipsoide in P sein, falls $P \neq \emptyset$. Wir wollen nun ausrechnen, nach wievielen Schritten dies der Fall ist.

(4.25) Lemma. *Seien* $P = P(A,b) \subseteq \mathbb{R}^n$, $\mathring{P} = \{x \in \mathbb{R}^n \mid Ax < b\}$, *und* $R := \sqrt{n}2^{2\langle A\rangle+\langle b\rangle-2n^2}$.

(a) *Entweder gilt* $\mathring{P} = \emptyset$ *oder*

$$\mathrm{vol}(\mathring{P} \cap S(0,R)) \geq 2^{-(n+1)(\langle A\rangle+\langle b\rangle-n^2)}.$$

(b) *Ist* P *volldimensional (d. h.* $\dim P = n$*), dann gilt*

$$\mathrm{vol}(P \cap S(0,R)) = \mathrm{vol}(\mathring{P} \cap S(0,R)) > 0. \qquad \triangle$$

Lemma (4.25) zusammen mit Lemma (4.19) sind geometrisch und algorithmisch interessant. Die beiden Hilfssätze implizieren folgendes.

- Wenn ein Polyeder P nicht leer ist, dann gibt es Elemente des Polyeders, die „nah“ beim Nullvektor liegen, d. h. in $S(0,R)$ enthalten sind.
- Es reicht aus zu überprüfen, ob $P \cap S(0,R)$ leer ist oder nicht. Daraus kann man schließen, dass P leer ist oder nicht.
- Will man einen Konvergenzbeweis über Volumen führen, so sollte man strikte statt normale Ungleichungssysteme betrachten. Denn ein striktes Ungleichungssystem ist entweder unlösbar oder seine Lösungsmenge hat ein positives (relativ großes) Volumen.

Damit können wir die Ellipsoidmethode formulieren.

(4.26) Algorithmus (Ellipsoidmethode).
Eingabe: $A \in \mathbb{Q}^{(m,n)}$, $b \in \mathbb{Q}^m$.
Ausgabe: Ein Vektor $x \in \mathbb{Q}^n$ mit $Ax \leq b$ oder die Feststellung, dass $\{x \in \mathbb{R}^n \mid Ax < b\}$ leer ist.

1. **Initialisierung:** Setze

   $A_0 := R^2 I$, mit $R = \sqrt{n} \cdot 2^{2\langle A\rangle+\langle b\rangle-2n^2}$ (oder R durch andere Vorinformationen kleiner gewählt),

   $a_0 := 0$,

   $k := 0$,

   $N := 2n((3n+1)\langle A\rangle + (2n+1)\langle b\rangle - n^3)$.

   (Das Anfangsellipsoid ist $E_0 := E(A_0, a_0)$.)

2. **Abbruchkriterium:**

   (2.a) Gilt $k = N$, dann hat $Ax < b$ keine Lösung, STOP!

   (2.b) Gilt $Aa_k \leq b$, dann ist eine Lösung gefunden, STOP!

(2.c) Andernfalls sei c^T irgendeine Zeile von A derart, dass der Mittelpunkt a_k von E_k die entsprechende Ungleichung verletzt.

3. **Update:** Setze

(3.a) $a_{k+1} := a_k - \frac{1}{n+1}\frac{1}{\sqrt{c^T A_k c}} A_k c$,

(3.b) $A_{k+1} := \frac{n^2}{n^2-1}(A_k - \frac{2}{n+1}\frac{1}{c^T A_k c} A_k c c^T A_k^T)$ ($E_{k+1} := E(A_{k+1}, a_{k+1})$ ist das neue Ellipsoid), $k := k+1$.

4. Gehe zu 2. △

(4.27) Satz. *Die Ellipsoidmethode arbeitet korrekt.* △

Beweis. Gibt es ein $k \leq N$, so dass $Aa_k \leq b$, so ist offenbar ein Vektor aus $P(A,b)$ gefunden. Bricht die Ellipsoidmethode in Schritt (2.a) ab, so müssen wir zeigen, dass $\mathring{P} := \{x \mid Ax < b\}$ kein Element besitzt.

Angenommen $\mathring{P} \neq \emptyset$. Sei P' der Durchschnitt von $\mathring{P}$ mit E_0. Dann gilt nach Lemma (4.25), dass das Volumen von P' mindestens $2^{-(n+1)(\langle A\rangle+\langle b\rangle-n^2)}$ beträgt. Ist a_k nicht in $P(A,b)$, $0 \leq k < N$, so wird in (2.c) eine verletzte Ungleichung gefunden, sagen wir $c^T x \leq \gamma$. Wegen $\gamma < c^T a_k$ enthält das Halbellipsoid

$$E'_k := E_k \cap \{x \mid c^T x \leq c^T a_k\}$$

die Menge P'. Das durch die Formeln (3.a), (3.b) konstruierte neue Ellipsoid E_{k+1} ist nach Satz (4.20) das volumenmäßig kleinste Ellipsoid, das E'_k enthält. Wegen $P' \subseteq E'_k$ gilt natürlich $P' \subseteq E_{k+1}$. Daraus folgt

$$P' \subseteq E_k, \quad 0 \leq k \leq N.$$

Das Volumen des Anfangsellipsoids E_0 kann man wie folgt berechnen:

$$\mathrm{vol}(E_0) = \sqrt{\det(R^2 I)} \cdot V_n = R^n V_n,$$

wobei V_n das Volumen der Einheitskugel im $\mathbb{R}^n$ ist. Wir machen nun eine sehr grobe Abschätzung. Die Einheitskugel ist im Würfel $W = \{x \in \mathbb{R}^n \mid |x_i| \leq 1, i = 1, \ldots, n\}$ enthalten. Das Volumen von W ist offensichtlich 2^n. Daraus folgt

$$\mathrm{vol}(E_0) \;<\; R^n 2^n \;=\; 2^{n(2\langle A\rangle+\langle b\rangle-2n^2+\log\sqrt{n}+1)} \;<\; 2^{n(2\langle A\rangle+\langle b\rangle-n^2)}.$$

In jedem Schritt der Ellipsoidmethode schrumpft nach (4.24) das Volumen um mindestens den Faktor $e^{-\frac{1}{2n}}$. Aus der Abschätzung von $\mathrm{vol}(E_0)$ und der in 1. angegebenen Formel für N erhalten wir somit

$$\mathrm{vol}(E_N) \;\leq\; e^{-\frac{N}{2n}}\,\mathrm{vol}(E_0) \;<\; 2^{-(n+1)(\langle A\rangle+\langle b\rangle-n^2)}.$$

Also gilt $\mathrm{vol}(E_N) < \mathrm{vol}(P')$ und $P' \subseteq E_N$. Dies ist ein Widerspruch. Hieraus folgt, dass P' und somit $\{x \mid Ax < b\}$ leer sind, wenn die Ellipsoidmethode in (2.a) abbricht. □

Aus (4.25)(b) folgt nun unmittelbar

(4.28) Korollar. *Ist $P = P(A,b) \subseteq \mathbb{R}^n$ ein Polyeder, von dem wir wissen, dass es entweder volldimensional oder leer ist, dann findet die Ellipsoidmethode entweder einen Punkt in P oder beweist, dass P leer ist.* △

Nun kann man natürlich einem durch ein Ungleichungssystem gegebenen Polyeder nicht unmittelbar ansehen, ob es volldimensional ist oder nicht. Ferner können gerade diejenigen Polyeder, die durch Transformation linearer Programme entstehen (siehe (4.4), hier gilt immer $c^T x = b^T y$), nicht volldimensional sein, so dass die Ellipsoidmethode zu keiner befriedigenden Antwort führt. Diesen Defekt kann man jedoch durch die folgende Beobachtung reparieren.

(4.29) Satz. *Seien $A \in \mathbb{Q}^{(m,n)}$ und $b \in \mathbb{Q}^m$, dann hat das Ungleichungssystem*

$$Ax \leq b$$

genau dann eine Lösung, wenn das strikte Ungleichungssystem

$$Ax < b + 2^{-2\langle A\rangle - \langle b\rangle}\mathbb{1}$$

eine Lösung hat. Ferner kann man aus einer Lösung des strikten Ungleichungssystems in polynomialer Zeit eine Lösung von $Ax \leq b$ konstruieren. △

Damit ist die Beschreibung der Ellipsoidmethode (bis auf die Abschätzung der Rechenzeit) vollständig. Wollen wir entscheiden, ob ein Polyeder $P(A,b)$ einen Punkt enthält, können wir die Methode (4.26) auf das Ungleichungssystem $Ax \leq b$ anwenden. Finden wir einen Punkt in $P(A,b)$, dann sind wir fertig. Andernfalls wissen wir, dass $P(A,b)$ nicht volldimensional ist. In diesem Falle können wir auf Satz (4.29) zurückgreifen und starten die Ellipsoidmethode neu, und zwar z. B. mit dem ganzzahligen Ungleichungssystem

$$2^{2\langle A\rangle + \langle b\rangle} Ax \leq 2^{2\langle A\rangle + \langle b\rangle} b + \mathbb{1}. \tag{4.30}$$

Entscheidet die Ellipsoidmethode, dass das zu (4.30) gehörige strikte Ungleichungssystem keine Lösung hat (Abbruch in Schritt (2.a)), so können wir aus (4.29) folgern, dass $P(A,b)$ leer ist. Andernfalls findet die Ellipsoidmethode einen Vektor x', der (4.30) erfüllt. Gilt $x' \in P(A,b)$, haben wir das gewünschte gefunden, falls nicht, kann man (mit einfachen Methoden der linearen Algebra) aus x' einen Punkt $x \in P(A,b)$ konstruieren, siehe (4.29).

Zur Lösung linearer Programme kann man die in Abschnitt 4.1 besprochenen Reduktionen benutzen. Entweder man fasst das lineare Programm (4.2) und das dazu duale (4.3) zu (4.4) zusammen und sucht wie oben angegeben im nicht volldimensionalen Polyeder (4.4) einen Punkt, oder man wendet $\overline{N}$-mal, siehe (4.14), die Ellipsoidmethode für niederdimensionale Polyeder des Typs P_s aus (4.12)(2) im Rahmen eines binären Suchverfahrens (4.12) an.

In jedem Falle ist das Gesamtverfahren polynomial, wenn die Ellipsoidmethode (4.26) polynomial ist.

4.3 Laufzeit der Ellipsoidmethode

Wir wollen nun die Laufzeit der Ellipsoidmethode untersuchen und auf einige bisher verschwiegene Probleme bei der Ausführung von (4.26) aufmerksam machen. Offenbar ist die maximale Iterationszahl

$$N = 2n((3n+1)\langle A\rangle + (2n+1)\langle b\rangle - n^3)$$

polynomial in der Kodierungslänge von A und b. Also ist das Verfahren (4.26) genau dann polynomial, wenn jede Iteration in polynomialer Zeit ausgeführt werden kann.

Bei der Initialisierung 1 besteht kein Problem. Test (2.a) ist trivial, und die Schritte (2.b) und (2.c) führen wir dadurch aus, dass wir das Zentrum a_k in die Ungleichungen einsetzen und überprüfen, ob die Ungleichungen erfüllt sind oder nicht. Die Anzahl der hierzu benötigten elementaren Rechenschritte ist linear in $\langle A\rangle$ und $\langle b\rangle$. Sie ist also polynomial, wenn die Kodierungslänge des Vektors a_{k+1} polynomial ist.

An dieser Stelle beginnen die Schwierigkeiten. In der Update-Formel (3.a) muss eine Wurzel berechnet werden. I. A. werden also hier irrationale Zahlen auftreten, die natürlich nicht exakt berechnet werden können. Die (möglicherweise) irrationale Zahl $\sqrt{c^T A_k c}$ muss daher zu einer rationalen Zahl gerundet werden. Dadurch wird geometrisch bewirkt, dass der Mittelpunkt des Ellipsoids E_{k+1} ein wenig verschoben wird. Mit Sicherheit enthält das so verschobene Ellipsoid nicht mehr die Menge E'_k (siehe Satz (4.20)) und möglicherweise ist auch $P(A,b)$ nicht mehr in diesem Ellipsoid enthalten. Also bricht unser gesamter Beweis der Korrektheit des Verfahrens zusammen.

Ferner wird beim Update (3.b) durch möglicherweise große Zahlen geteilt, und es ist nicht a priori klar, dass die Kodierungslänge der Elemente von A_{k+1} bei wiederholter Anwendung von (3.b) polynomial in $\langle A\rangle + \langle b\rangle$ bleibt. Also müssen auch die Einträge in A_{k+1} gerundet werden. Dies kann zu folgenden Problemen führen. Die gerundete Matrix, sagen wir A^*_{k+1}, ist nicht mehr positiv definit und das Verfahren wird sinnlos. Oder A^*_{k+1} bleibt positiv definit, aber durch die Rundung hat sich die Form des zugehörigen Ellipsoids, sagen wir E^*_{k+1} so geändert, dass E^*_{k+1} das Polyeder $P(A,b)$ nicht mehr enthält.

Alle diese Klippen kann man mit einem einfachen Trick umschiffen, dessen Korrektheitsbeweis allerdings recht aufwendig ist. Die geometrische Idee hinter diesem Trick ist die folgende. Man nehme die binäre Darstellung der Komponenten des in (3.a) berechneten Vektors und der in (3.b) berechneten Matrix und runde nach p Stellen hinter dem Binärkomma. Dadurch ändert man die Lage des Mittelpunkts und die Form des Ellipsoids ein wenig. Nun bläst man das Ellipsoid ein bisschen auf, d. h. man multipliziert A_{k+1} mit einem Faktor $\zeta > 1$ und zwar so, dass die Menge E'_k beweisbar in dem aufgeblasenen Ellipsoid enthalten ist. Durch die Vergrößerung des Ellipsoids wird natürlich die in (4.24) bestimmte Schrumpfungsrate verschlechtert, was bedeutet, dass man insgesamt mehr Iterationen, sagen wir N', durchführen muss. Daraus folgt, dass der Rundungsparameter p und der Aufblasparameter ζ aufeinander so abgestimmt sein müssen, dass alle gerundeten und mit ζ multiplizierten Matrizen A_k, $1 \le k \le N'$ und alle gerundeten Mittelpunkte a_k, $1 \le k \le N'$ polynomial in $\langle A\rangle + \langle b\rangle$ berechnet werden können, so dass alle A_k positiv definit sind, $P \cap S(0,R)$ in allen Ellipsoiden E_k enthalten ist und die Iterationszahl N' ebenfalls polynomial in $\langle A\rangle + \langle b\rangle$ ist. Dies kann in der Tat durch

Anwendung von Abschätzungstechniken aus der Analysis und linearen Algebra realisiert werden, siehe Grötschel et al. (1988).

Die Ellipsoidmethode (mit Rundungsmodifikation) ist also ein polynomialer Algorithmus, der entscheidet, ob ein Polyeder leer ist oder nicht. Mit Hilfe der im Vorhergehenden beschriebenen Reduktion kann sie dazu benutzt werden, lineare Programme in polynomialer Zeit zu lösen.

Wie sich der Leser denken kann, gibt es eine ganze Reihe von Varianten der Ellipsoidmethode, die dazu dienen sollen, schnellere Konvergenz zu erzwingen. Derartige Varianten sind z. B. in Bland et al. (1981), Grötschel et al. (1988) und Akgül (1984) beschrieben. Aus Platz- und Zeitgründen können wir hier nicht weiter darauf eingehen.

4.4 Ein Beispiel

Wir wollen hier den Ablauf der Ellipsoidmethode anhand eines Beispiels im $\mathbb{R}^2$ geometrisch veranschaulichen. Wir starten mit dem folgenden Polyeder $P(A, b) \subseteq \mathbb{R}^2$ definiert durch

$$\begin{aligned} -x_1 - x_2 &\leq -2 \\ 3x_1 &\leq 4 \\ -2x_1 + 2x_2 &\leq 3 \end{aligned}$$

Dieses Polyeder P ist in der Kugel (Kreis) um den Nullpunkt mit Radius 7 enthalten. Diese Kugel soll unser Anfangsellipsoid $E_0 = E(A_0, 0)$ sein. Wir führen mit dieser Initialisierung die Schritte 2 und 3 des Ellipsoidverfahrens (4.26) durch. Wir rechnen natürlich nicht mit der eigentlich erforderlichen Genauigkeit sondern benutzen die vorhandene Maschinenpräzision. In unserem Fall ist das Programm in PASCAL und die Rechnungen werden in `REAL`-Arithmetik durchgeführt, d. h. wir benutzen eine Mantisse von 3 byte und einen Exponenten von 1 byte zur Zahlendarstellung. Das Verfahren führt insgesamt 7 Iterationen durch und endet mit einem zulässigen Punkt. In den nachfolgenden Abbildungen 4.3 bis 4.9 sind (im Maßstab 1 : 2) jeweils die Ellipsoide E_k und E_{k+1} mit ihren Mittelpunkten a_k und a_{k+1}, $k = 0, \ldots, 6$ aufgezeichnet. Man sieht also, wie sich die Form und Lage eines Ellipsoids in einem Iterationsschritt ändert. Das Startellipsoid ist gegeben durch

$$a_0^T = (0, 0), \qquad A = \begin{pmatrix} 49 & 0 \\ 0 & 49 \end{pmatrix}.$$

Neben jeder Abbildung sind das neue Zentrum a_{k+1} und die neue Matrix A_{k+1} mit $E_{k+1} = E(A_{k+1}, a_{k+1})$ angegeben. Jede Abbildung enthält die Begrenzungsgeraden des Polytops P, so dass man auf einfache Weise sehen kann, welche Ungleichungen durch den neuen Mittelpunkt verletzt sind.

Literaturverzeichnis

M. Akgül. *Topics in relaxation and ellipsoidal methods*, volume 97 of *Research Notes in Mathematics*. Pitman Advanced Publishing Program. VII, Boston, 1984.

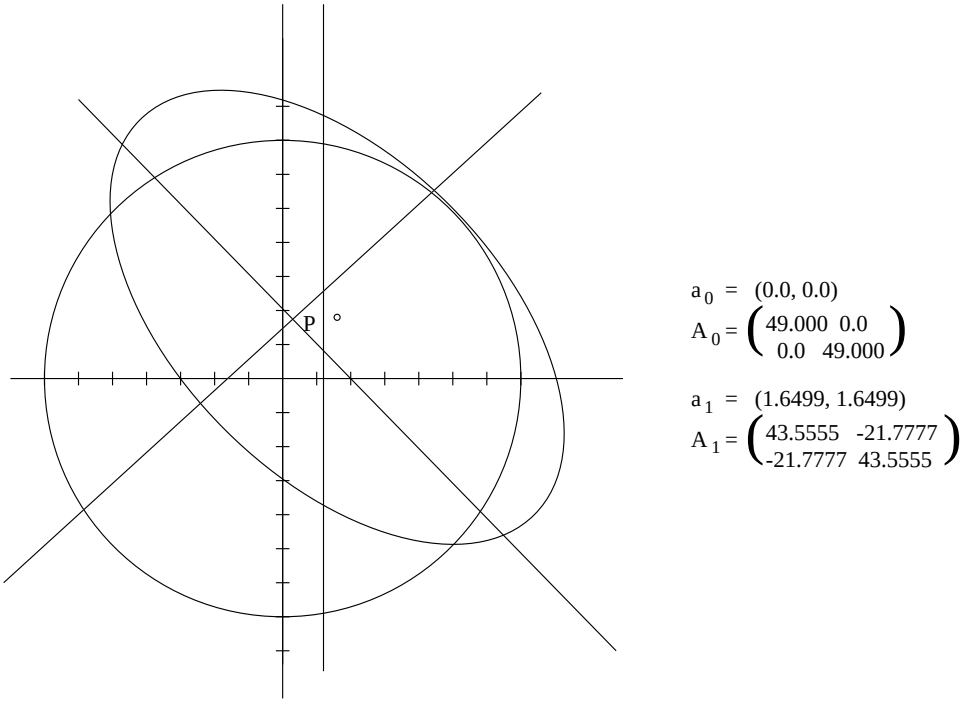


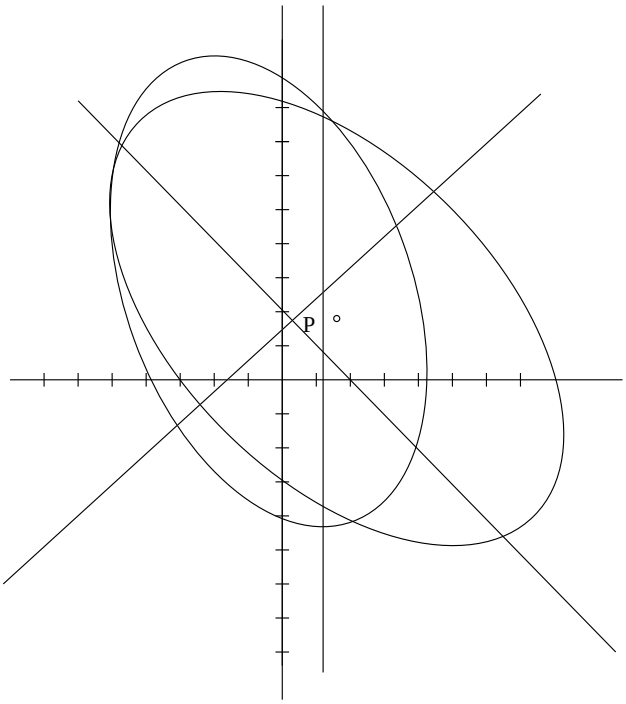
Abbildung 4.3: Iteration 1

R. G. Bland, D. Goldfarb, and M. J. Todd. The ellipsoid method: A survey. *Operations Research*, 29:1039–1091, 1981.

K.-H. Borgwardt. The average number of pivot steps required by the simplex-method is polynomial. *Zeitschrift für Operations Research, Serie A*, 26:157–177, 1982.

M. Grötschel, L. Lovász, and A. Schrijver. *Geometric algorithms and combinatorial optimization*. Algorithms and combinatorics. Springer, 1988.

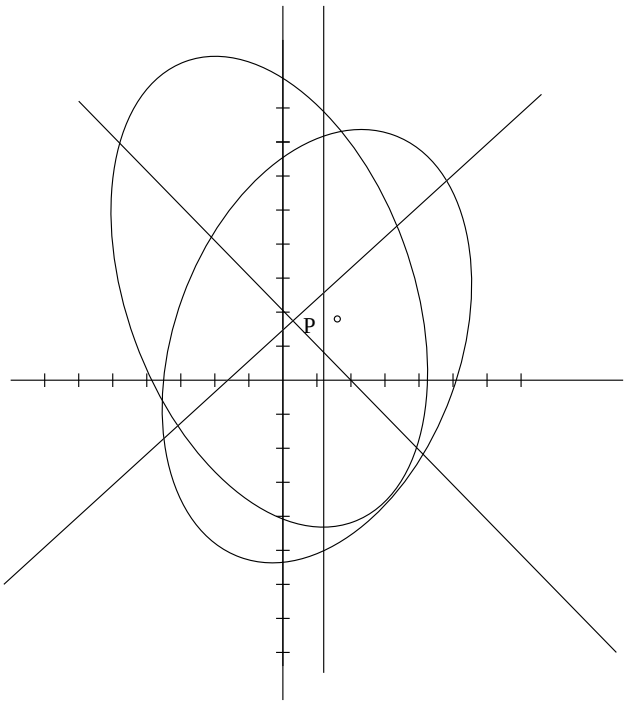
L. G. Khachiyan. A polynomial algorithm in linear programming. *Doklady Akademia Nauk SSSR*, 244:1093–1096, 1979. Englische Übersetzung in Soviet Mathematics Doklady, 20:191–194, 1979.



$a_2 =$ (-0.5499, 2.7499)

$$A_2 = \begin{pmatrix} 19.3580 & -9.6790 \\ -9.6790 & 48.3951 \end{pmatrix}$$

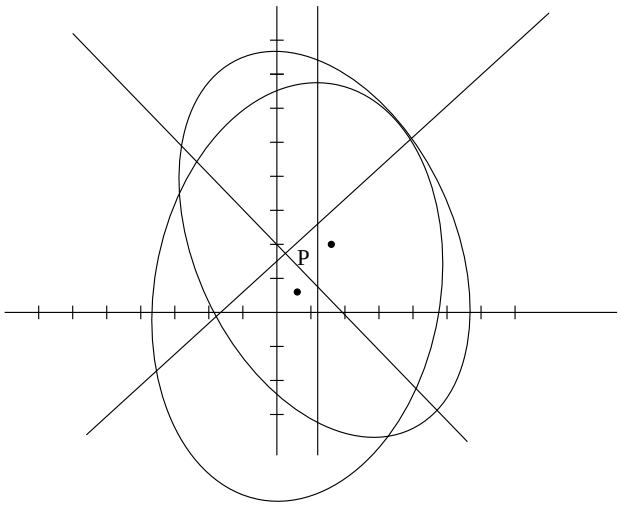
Abbildung 4.4: Iteration 2



$a_3 =$ (0.4871, 0.6758)

$$A_3 \begin{pmatrix} 17.2071 & 4.3018 \\ 4.3018 & 30.1125 \end{pmatrix}$$

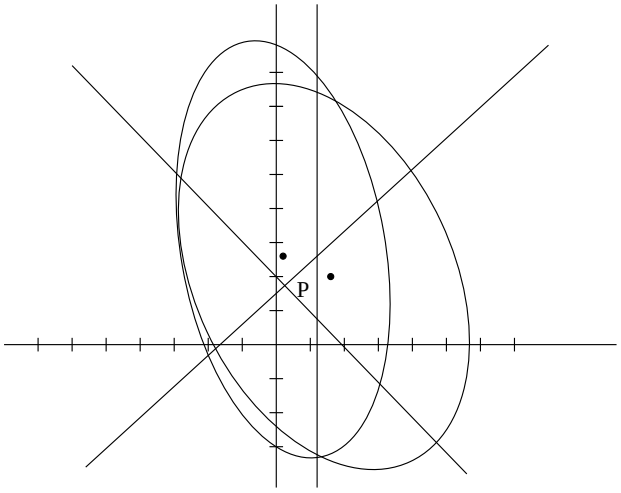
Abbildung 4.5: Iteration 3



$a_4 = (1.4458, 2.2098)$

$$A_4 = \begin{pmatrix} 15.5894 & -6.0299 \\ -6.0299 & 21.351 \end{pmatrix}$$

Abbildung 4.6: Iteration 4



$a_5 = (0.1297, 2.7188)$

$$A_5 = \begin{pmatrix} 6.9286 & -2.6799 \\ -2.6799 & 26.3603 \end{pmatrix}$$

Abbildung 4.7: Iteration 5

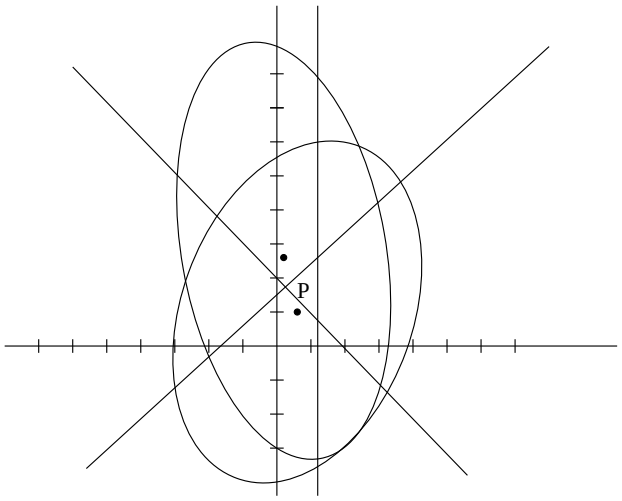


Abbildung 4.8: Iteration 6

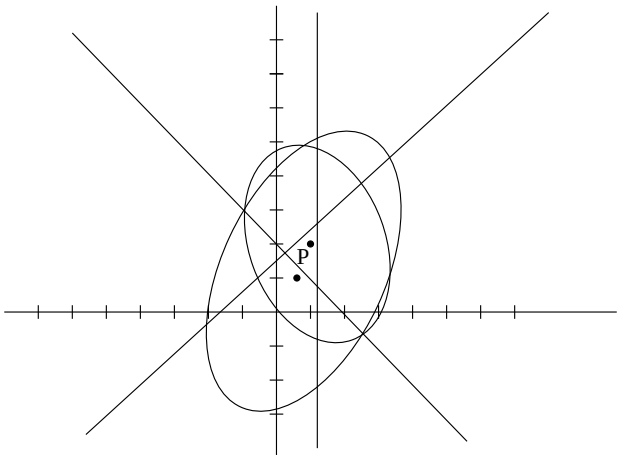


Abbildung 4.9: Iteration 7 (mit zulässigem Punkt a_7)

5 Innere-Punkte-Verfahren

Die Ellipsoidmethode war ein theoretischer Durchbruch aber kein praktischer Erfolg. Fünf Jahre nach Khachiyans Publikation kündigte sich dann eine weitere Sensation an. Eine Idee, die bereits in den 1950er und 1960er Jahren untersucht, aber nie zu einem ernsthaften Konkurrenten des Simplexverfahrens entwickelt wurde, sollte nun sowohl theoretisch als auch praktisch zur Lösung von linearen Optimierungsaufgaben geeignet sein. Der Vorschlag ist, zunächst einen Punkt im Inneren des Polyeders zu konstruieren und dann durch das Innere des Polyeders entlang eines Pfades (den man auf verschiedene Weisen definieren kann) auf eine Optimallösung zuzusteuern. Narendra Karmarkar löste durch seinen Artikel und seine Vorträge im Jahr 1984 eine neue Welle von Algorithmenentwicklungen aus, die dann tatsächlich zu Verfahren geführt hat, die in vielen Fällen, insbesondere bei sehr großen LPs, schneller als der Simplexalgorithmus arbeiten. Die Entstehungsgeschichte der inzwischen Innere-Punkte- oder Barriere-Verfahren genannten Algorithmen ist sehr spannend und berührt auch das Thema „Patentierbarkeit von Algorithmen“. Wir verweisen hierzu auf den Artikel *Who Invented the Interior-Point Method?*, siehe Shanno (2012).

In diesem Kapitel stellen wir zunächst ausführlich die Idee des Karmarkar-Algorithmus dar, danach erläutern wir primale sowie primal-duale Pfadverfolgungsalgorithmen. Numerische Details (die gute Kenntnisse der nichtlinearen Optimierung voraussetzen) können hier – auch aus Zeitgründen – nicht dargestellt werden.

5.1 Der Karmarkar-Algorithmus

Im Jahre 1984 hat N. Karmarkar (AT&T Bell Laboratories) einen neuen Algorithmus zur Lösung linearer Programme entwickelt und in Karmarkar (1984) veröffentlicht. Im Herbst 1984 hat Karmarkar auf Vorträgen mitgeteilt, dass sein Verfahren nicht nur theoretisch polynomial ist (wie auch die in Kapitel 4 beschriebene Ellipsoidmethode), sondern in der Praxis (Implementation bei AT&T Bell Laboratories) erheblich rascher als das Simplexverfahren arbeitet. Karmarkar behauptete, dass die Implementation seines Algorithmus etwa 50-mal schneller ist als MPSX, ein von der Firma IBM angebotenes und auf dem Simplexalgorithmus basierendes Softwarepaket zur Lösung linearer Programme, das zur damaligen Zeit als eines der besten LP-Pakete galt.

Da bei großen Industriefirmen sehr große LPs täglich in nicht unerheblicher Zahl gelöst werden, war die Ankündigung Karmarkars auf sehr großes Interesse bei Praktikern gestoßen. Ja, sein Verfahren hat sogar Aufsehen in der Presse verursacht. Zum Beispiel haben New York Times, Science, Time, Der Spiegel (falsch – wie bei Artikeln über Mathematik üblich), Die Zeit, Süddeutsche Zeitung (sehr ordentlich) über Karmarkars Verfahren berichtet, wie immer natürlich unter dem Hauptaspekt, dass dadurch Millionen von Dollars

gespart werden können. Im Nachfolgenden wollen wir die Grundidee des Algorithmus von Karmarkar darstellen.

Reduktionen

Wie der Simplexalgorithmus und die Ellipsoidmethode, so ist auch der Karmarkar-Algorithmus auf einen speziellen Problemtyp zugeschneidert und nicht auf allgemeine lineare Programme anwendbar. Wie üblich müssen wir daher zunächst zeigen, dass ein allgemeines LP so transformiert werden kann, dass es mit Karmarkars Algorithmus gelöst werden kann. Die Grundversion des Karmarkar-Algorithmus (und nur die wollen wir hier beschreiben) ist ein Verfahren zur Entscheidung, ob ein Polyeder einer recht speziellen Form einen Punkt enthält oder nicht. Das Problem, das Karmarkar betrachtet, ist das folgende.

(5.1) Problem. Gegeben seien eine Matrix $A \in \mathbb{Q}^{(m,n)}$ und ein Vektor $c \in \mathbb{Q}^n$. Entscheide, ob das System

$$Ax = 0, \quad \mathbb{1}^T x = 1, \quad x \geq 0, \quad c^T x \leq 0$$

eine Lösung hat, und falls das so ist, finde eine. △

Um den Karmarkar-Algorithmus starten zu können, ist die Kenntnis eines zulässigen Startvektors im relativ Inneren von $\{x \mid Ax = 0, \mathbb{1}^T x = 1, x \geq 0\}$ notwendig. Wir wollen sogar noch mehr fordern, und zwar soll gelten

$$\overline{x} := \frac{1}{n}\mathbb{1} \quad \text{erfüllt} \quad A\overline{x} = 0. \tag{5.2}$$

Trivialerweise erfüllt $\overline{x}$ die Bedingungen $\mathbb{1}^T\overline{x} = 1$, $\overline{x} \geq 0$. Gilt $c^T\overline{x} \leq 0$, ist man natürlich fertig. Die eigentliche Aufgabe besteht also darin, aus $\overline{x}$ einen Punkt zu konstruieren, der alle Bedingungen von (5.1) erfüllt.

Wir werden nun, wie bei der Darstellung der Ellipsoidmethode, ein allgemeines LP in ein (bzw. mehrere) Probleme des Typs (5.1) umformen.

Wir wissen bereits, dass jedes LP in ein LP in Standardform (siehe ADM I, Definition (9.1)) transformiert werden kann. Gegeben sei also das folgende Problem

$$\begin{aligned} \max \quad & w^T x \\ & Bx = b \\ & x \geq 0 \end{aligned} \tag{5.3}$$

mit $w \in \mathbb{Q}^n$, $B \in \mathbb{Q}^{(m,n)}$, $b \in \mathbb{Q}^m$. Wie in Abschnitt 4.1 gezeigt gibt es zwei prinzipielle Reduktionsmöglichkeiten. Man fasst (5.3) und das dazu duale Problem wie in (4.4) zusammen, oder man führt die in (4.12) beschriebene Binärsuche durch. Wir stellen hier kurz die Reduktion über Binärsuche dar. Daraus ergibt sich auch, wie man die Kombination des primalen mit dem dualen Problem behandeln kann.

Genau wie in (4.12) angegeben (und mit den in den davor angegebenen Sätzen vorgelegten Begründungen) führen wir eine binäre Suche durch über der Menge

$$S := \left\{ \frac{p}{q} \in Q \mid |p| \leq n \cdot 2^{\langle B\rangle+\langle b\rangle+2\langle w\rangle-n^2-n}, 1 \leq q \leq 2^{\langle B\rangle+\langle w\rangle-n^2-n} \right\}$$

der möglichen optimalen Zielfunktionswerte (falls (5.3) eine endliche Optimallösung hat). Zur Bestimmung einer optimalen Lösung von (5.3) sind nach (4.14) höchstens

$$\overline{N} := 2\langle B\rangle + \langle b\rangle + 3\langle w\rangle - 2n^2 - 2n + \log_2 n + 2$$

Aufrufe eines Algorithmus nötig, der für ein $s \in S$ entscheidet, ob

$$\begin{aligned} w^T x &\leq s \\ Bx &= b \\ x &\geq 0 \end{aligned} \tag{5.4}$$

eine Lösung hat. Genau dann, wenn wir zeigen können, dass (5.4) in polynomialer Zeit gelöst werden kann, haben wir also ein polynomiales Verfahren zur Lösung von (5.3). Wir wissen aus Satz (4.7), dass wir alle Variablen durch $2^{2\langle B\rangle+\langle b\rangle-2n^2}$ beschränken können. Führen wir für die Ungleichung in (5.4) eine Schlupfvariable x_{n+1} ein, so gilt $0 \leq x_{n+1} \leq \max S = n \cdot 2^{\langle B\rangle+\langle b\rangle+2\langle w\rangle-n^2-n}$. Setzen wir also

$$M := n(2^{2\langle B\rangle+\langle b\rangle-2n^2} + 2^{\langle B\rangle+\langle b\rangle+2\langle w\rangle-n^2-n}),$$

so ist (5.4) genau dann lösbar, wenn

$$\begin{aligned} \begin{pmatrix} w^T & 1 \\ B & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_{n+1} \end{pmatrix} &= \begin{pmatrix} s \\ b \end{pmatrix} \\ \sum_{i=1}^{n+1} x_i &\leq M \\ x_i &\geq 0, \quad i = 1, \ldots, n+1 \end{aligned} \tag{5.5}$$

lösbar ist. Führen wir eine weitere Schlupfvariable x_{n+2} ein (d. h. es gilt nun $x \in \mathbb{R}^{n+2}$) und skalieren wir die Variablen um, so ist (5.5) genau dann lösbar, wenn

$$\begin{aligned} Cx := \quad \begin{pmatrix} w^T & 1 & 0 \\ B & 0 & 0 \end{pmatrix} x = \frac{1}{M} \begin{pmatrix} s \\ b \end{pmatrix} \quad =: \begin{pmatrix} \overline{c}_0 \\ \vdots \\ \overline{c}_m \end{pmatrix} \\ \mathbb{1}^T x = 1 \\ x_i \geq 0, \quad i = 1, \ldots, n+2 \end{aligned} \tag{5.6}$$

lösbar ist. Von der i-ten Zeile der Matrix C ($i = 0, \ldots, m$) ziehen wir nun das $\overline{c}_i$-fache der letzten Gleichung $\mathbb{1}^T x = 1$ ab. Dadurch machen wir die ersten $m+1$ Gleichungen

homogen und erreichen, dass aus dem Ungleichungs- und Gleichungssystem (5.6) ein äquivalentes System der folgenden Form wird:

$$\begin{aligned} Dx &= 0 \\ \mathbb{1}^T x &= 1 \\ x &\geq 0 \end{aligned} \tag{5.7}$$

Um die Voraussetzung (5.2) herzustellen, führen wir eine weitere Schlupfvariable x_{n+3} wie folgt ein. Wir betrachten

$$\begin{aligned} Dx - D\mathbb{1}x_{n+3} &= 0 \\ \mathbb{1}^T x + x_{n+3} &= 1 \\ x_i \geq 0, \quad i &= 1, \ldots, n+3. \end{aligned} \tag{5.8}$$

Offenbar hat (5.7) genau dann eine Lösung, wenn (5.8) eine Lösung mit $x_{n+3} = 0$ hat. Setzen wir

$$A := (D, -D\mathbb{1}) \in \mathbb{Q}^{(m+1,n+3)}, \qquad c := (0, \ldots, 0, 1)^T \in \mathbb{Q}^{n+3}$$

und betrachten wir x als Vektor im $\mathbb{R}^{n+3}$, so hat also (5.7) genau dann eine Lösung, wenn

$$Ax = 0, \quad \mathbb{1}^T x = 1, \quad x \geq 0, \quad c^T x \leq 0 \tag{5.9}$$

eine Lösung hat. Ferner erfüllt nach Konstruktion der Vektor

$$\overline{x}^T := \left(\frac{1}{n+3}, \ldots, \frac{1}{n+3} \right) \in \mathbb{Q}^{n+3}$$

bezüglich des Systems (5.9) die Forderung (5.2).

Folglich kann jedes lineare Program durch polynomial viele Aufrufe eines Algorithmus zur Lösung von (5.1) (mit Zusatzvoraussetzung (5.2)) gelöst werden.

Die Grundversion des Karmarkar-Algorithmus

Wir wollen nun das Karmarkar-Verfahren zur Lösung von (5.1) unter der Zusatzvoraussetzung (5.2) beschreiben. Wie wir am Ende des letzten Abschnittes gesehen haben, können wir jedes LP in eine Folge von Problemen der Form (5.1) transformieren, und wir können sogar das Zentrum des Simplex $\{x \in \mathbb{R}^n \mid \mathbb{1}^T x = 1, x \geq 0\}$ als Startpunkt wählen. Im weiteren betrachten wir also das folgende

(5.10) Problem. Gegeben seien eine Matrix $A \in \mathbb{Q}^{(m,n)}$ und ein Vektor $c \in \mathbb{Q}^n$. Es seien

$$\begin{aligned} E &:= \{x \in \mathbb{R}^n \mid \mathbb{1}^T x = 1\} && \text{(Hyperebene)} \\ \Sigma &:= E \cap \mathbb{R}^n_+ && \text{(Simplex)} \\ \Omega &:= \{x \in \mathbb{R}^n \mid Ax = 0\} && \text{(linearer Raum).} \end{aligned}$$

Ferner wird vorausgesetzt, dass der Vektor $\overline{x} := \frac{1}{n}\mathbb{1}$ in $\Sigma \cap \Omega$ enthalten ist. Gesucht ist ein Vektor $x \in \mathbb{Q}^n$ mit

$$x \in P := \Sigma \cap \Omega, \quad c^T x \leq 0. \qquad \triangle$$

Problem (5.10) kann sicherlich dadurch gelöst werden, daß eine Optimallösung des folgenden Programms bestimmt wird.

$$\begin{array}{rl} \min & c^T x \\ & Ax = 0 \\ & \mathbb{1}^T x = 1 \\ & x \geq 0 \end{array} \quad \text{bzw.} \quad \begin{array}{l} \min c^T x \\ x \in \Omega \cap E \cap \mathbb{R}^n_+ \end{array} \tag{5.11}$$

Offenbar ist der optimale Zielfunktionswert von (5.11) genau dann größer als Null, wenn (5.10) keine Lösung hat. Unser Ziel wird nun sein (5.11) statt (5.10) zu lösen, wobei wir voraussetzen, dass $\frac{1}{n}\mathbb{1}$ für (5.11) zulässig ist.

Es erscheint natürlich reichlich umständlich, ein lineares Programm, sagen wir der Form (5.3), durch Binärsuche auf eine Folge von Zulässigkeitsproblemen (5.10) zu reduzieren, die dann wieder durch spezielle lineare Programme der Form (5.11) gelöst werden. Diesen Umweg kann man durch die sogenannte „Sliding Objective Function Technique“ begradigen. Die dabei notwendigen zusätzlichen Überlegungen tragen jedoch nicht zu einem besseren Verständnis der Grundversion des Verfahrens bei, um die es hier geht. Ziel dieses Kapitels ist nicht die Darstellung einer besonders effizienten Version des Karmarkar-Algorithmus sondern dessen prinzipielle Idee.

Geometrische Beschreibung eines Iterationsschrittes

Zur Lösung von (5.11) hat Karmarkar ein Verfahren entworfen, das wie fast alle Optimierungsverfahren (insbesondere die Methoden der nichtlinearen Optimierung) auf der folgenden simplen Idee beruht. Angenommen man befinde sich an einem zulässigen Punkt, sagen wir x^k, dann sucht man eine Richtung (also einen Vektor $d \in \mathbb{R}^n$), bezüglich der die Zielfunktion verbessert werden kann. Daraufhin bestimmt man eine Schrittlänge ρ, so dass man von x^k zum nächsten Punkt $x^{k+1} := x^k + \rho d$ gelangt. Der nächste Punkt x^{k+1} soll natürlich auch zulässig sein und einen „wesentlich“ besseren Zielfunktionswert haben.

Die Essenz eines jeden solchen Verfahrens steckt natürlich in der Wahl der Richtung und der Schrittlänge. Bei derartigen Verfahren tritt häufig die folgende Situation ein. Man ist in der Lage, eine sehr gute Richtung zu bestimmen (d. h. die Zielfunktion wird in Richtung d stark verbessert), aber man kann in Richtung d nur einen sehr kleinen Schritt ausführen, wenn man die zulässige Menge nicht verlassen will. Trotz guter Richtung kommt man also im Bezug auf eine tatsächliche Verbesserung kaum vorwärts und erhält u. U. global schlechtes Konvergenzverhalten. Man muss sich also bemühen, einen guten Kompromiss zwischen „Qualität der Richtung“ und „mögliche Schrittlänge“ zu finden, um insgesamt gute Fortschritte zu machen.

Ist man – wie im vorliegenden Fall – im relativen Inneren der Menge P, aber nah am Rand und geht man z. B. in Richtung des Normalenvektors der Zielfunktion, so kann man sehr schnell an den Rand von P gelangen, ohne wirklich weiter gekommen zu sein, siehe Abbildung 5.1. Karmarkars Idee zur Lösung bzw. Umgehung dieser Schwierigkeit

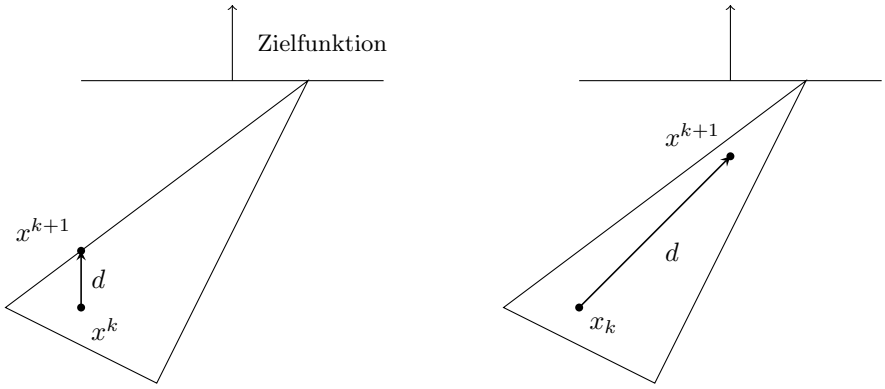


Abbildung 5.1: Gute Richtung, geringer Fortschritt (links), schlechtere Richtung, aber großer Fortschritt (rechts)

ist die folgende. Er führt eine projektive Transformation aus, die den Simplex Σ auf sich selbst, den affinen Teilraum Ω auf einen anderen affinen Teilraum Ω' abbildet und den relativ inneren Punkt x^k auf das Zentrum $\frac{1}{n}\mathbb{1}$ von Σ wirft. P wird dabei auf ein neues Polyeder P_k abgebildet. Offenbar kann man von $\frac{1}{n}\mathbb{1}$ aus recht große Schritte in alle zulässigen Richtungen machen, ohne sofort den zulässigen Bereich P_k zu verlassen. Man bestimmt so durch Festlegung einer Richtung und einer Schrittlänge von $\frac{1}{n}\mathbb{1}$ ausgehend einen Punkt $y^{k+1} \in P_k$ und transformiert diesen zurück, um den nächsten (zulässigen) Iterationspunkt $x^{k+1} \in P$ zu erhalten.

Zur Bestimmung der Richtung macht Karmarkar folgende Überlegung. Ideal wäre es natürlich, direkt das Optimum des transformierten Problems zu bestimmen. Aber die lineare Zielfunktion des ursprünglichen Problems wird durch die projektive Transformation in eine nichtlineare Abbildung übergeführt, so dass dies nicht so einfach zu bewerkstelligen ist. Dennoch kann man diese transformierte Zielfunktion auf natürliche Weise linearisieren. Mit $\overline{c}^T x$ sei die neue (linearisierte) Zielfunktion bezeichnet. Man hat nun ein neues Problem: Es soll eine lineare Zielfunktion über dem Durchschnitt eines Simplex mit einem affinen Raum optimiert werden, wir erhalten

$$\begin{aligned} &\min \overline{c}^T x \\ &\quad x \in \Omega' \cap E \cap \mathbb{R}^n_+ \end{aligned} \tag{5.12}$$

Dieses Problem ist offenbar wiederum vom Typ unseres Ausgangsproblems (5.11). Aber diese neue Aufgabe ist ja nur ein Hilfsproblem! Möglicherweise ist daher eine gute Approximation der Optimallösung von (5.12) ausreichend für das, was wir bezwecken. Durch die Lösung einer Vereinfachung dieses Problems (5.12) könnten u. U. ein Richtungsvektor und eine Schrittlänge gefunden werden, die von hinreichend guter Qualität bezüglich

globaler Konvergenz des Verfahrens sind. Die Vereinfachung, die wir betrachten wollen, geschieht dadurch, dass die bei der Durchschnittsbildung beteiligte Menge $\mathbb{R}^n_+$ durch die größte Kugel, sagen wir K, mit Zentrum $\frac{1}{n}\mathbb{1}$, die im Simplex Σ enthalten ist, ersetzt wird. Statt (5.12) wird also die Aufgabe

$$\begin{aligned} &\min \overline{c}^T x \\ &\quad x \in \Omega' \cap E \cap K \end{aligned} \tag{5.13}$$

betrachtet. Man optimiere eine lineare Zielfunktion über dem Durchschnitt einer Kugel K mit einem affinen Raum $\Omega' \cap E$. Dieses Problem ist ohne Einschaltung eines Algorithmus durch eine explizite Formel lösbar. Durch die Optimallösung von (5.13) werden die gesuchte Richtung und die Schrittlänge explizit geliefert.

Eine Modifikation ist jedoch noch nötig. Das Optimum über $\Omega' \cap E \cap K$ ist i. A. irrational und muss gerundet werden. Dadurch ist es möglich, dass y^{k+1} nicht mehr im relativen Inneren von P_k bzw. der nächste (gerundete) Iterationspunkt x^{k+1} nicht mehr im relativ Inneren von P liegt. Statt K wählt man daher eine kleinere Kugel K' mit dem gleichen Zentrum, so dass auch nach Rundung und Rücktransformation garantiert ist, dass der nächste Iterationspunkt ein relativ innerer Punkt ist.

Analytische Beschreibung eines Iterationsschrittes

Die oben gegebene anschauliche Darstellung wollen wir nun explizit vorführen. Wir starten also mit Problem (5.11). Wir setzen (wie beim Simplexverfahren) voraus, dass A vollen Zeilenrang hat. Außerdem kennen wir mit x^k einen relativ inneren Punkt von P. Ist $D = \operatorname{diag}(x^k)$ die (n,n)-Diagonalmatrix, die die Komponenten $x^k_1, \ldots, x^k_n$ von x^k auf der Hauptdiagonalen enthält, so ist durch

$$T_k(x) \;:=\; \frac{1}{\mathbb{1}^T D^{-1} x}\, D^{-1} x \tag{5.14}$$

eine projektive Transformation definiert. ($T_k(x)$ ist natürlich nur dann endlich, wenn $x \notin \overline{H} = \{y \in \mathbb{R}^n \mid \mathbb{1}^T D^{-1} y = 0\}$ gilt. Wir werden beim Rechnen mit T_k diese Einschränkung nicht weiter erwähnen und gehen davon aus, dass klar ist, wie die Formeln, die T_k enthalten, zu interpretieren sind.) Die projektive Transformation T_k hat, wie leicht zu sehen ist, folgende Eingeschaften:

(5.15) Satz (Eigenschaften von T_k).

(a) $x \geq 0 \implies T_k(x) \geq 0$.

(b) $\mathbb{1}^T x = 1 \implies \mathbb{1}^T T_k(x) = 1$.

(c) $T_k^{-1}(y) = \frac{1}{\mathbb{1}^T D y}\, Dy$ *für alle* $y \in \Sigma$.

(d) $T_k(\Sigma) = \Sigma$.

(e) $P_k := T_k(P) = T_k(\Sigma \cap \Omega) = \Sigma \cap \Omega_k$, *wobei* $\Omega_k = \{y \in \mathbb{R}^n \mid ADy = 0\}$.

(f) $T_k(x^k) = \frac{1}{\mathbb{1}^T\mathbb{1}}\, D^{-1}x^k = \frac{1}{n}\mathbb{1} \in P_k$. △

Die Transformation T_k ist also eine bijektive Abbildung $P \longrightarrow P_k$ und $\Sigma \longrightarrow \Sigma$, die den Punkt x^k in das Zentrum $\frac{1}{n}\mathbb{1}$ von Σ abbildet. Aus (5.15) folgt

$$\begin{array}{rcccl} \min c^T x & = & \min c^T x & = & \min\ c^T T_k^{-1}(y) \\ Ax = 0 & & x \in P & & y \in P_k = T_k(P) \\ \mathbb{1}^T x = 1 & & & & \\ x \geq 0 & & & & \end{array} \quad = \quad \min\ \frac{1}{\mathbb{1}^T Dy}\, c^T Dy \qquad \begin{array}{r} ADy = 0 \\ \mathbb{1}^T y = 1 \\ y \geq 0 \end{array} \tag{5.16}$$

Das letzte Programm in (5.16) ist das in der geometrisch-anschaulichen Erläuterung weiter oben genannte Programm mit nichtlinearer Zielfunktion $\frac{1}{\mathbb{1}^T Dy}\, Dy$. Wir linearisieren diese durch Weglassen des Nenners wie folgt:

$$\overline{c}^T := c^T D \tag{5.17}$$

und erhalten das lineare (Hilfs)-Programm

$$\begin{array}{r} \min\ \overline{c}^T y \\ ADy = 0 \\ \mathbb{1}^T y = 1 \\ y \geq 0 \end{array} \tag{5.18}$$

Wir vereinfachen (5.18) dadurch, dass wir die Nichtnegativitätsbedingung in (5.18) durch die Bedingung ersetzen, dass y in einer Kugel um $\frac{1}{n}\mathbb{1}$ liegt. Die größte Kugel mit diesem Zentrum, die man dem Simplex Σ einbeschreiben kann, hat den Radius $1/\sqrt{n(n-1)}$. Wie bereits erwähnt, müssen wir aus technischen Gründen eine kleinere Kugel wählen. Es wird sich zeigen, dass man mit der Hälfte dieses optimalen Radius auskommt. Wir betrachten also das Programm

$$\begin{array}{r} \min\ \overline{c}^T y \\ ADy = 0 \\ \mathbb{1}^T y = 1 \\ \left\| y - \frac{1}{n}\mathbb{1} \right\| \leq \frac{1}{2}\, \frac{1}{\sqrt{n(n-1)}} \end{array} \tag{5.19}$$

Das Minimum von (5.19) kann explizit bestimmt werden.

(5.20) Lemma. *Die Optimallösung von* (5.19) *ist der Vektor*

$$y^{k+1} := \frac{1}{n}\,\mathbb{1} - \frac{1}{2}\, \frac{1}{\sqrt{n(n-1)}}\, \frac{(I - DA^T(AD^2A^T)^{-1}AD - \frac{1}{n}\mathbb{1}\mathbb{1}^T)Dc}{\|(I - DA^T(AD^2A^T)^{-1}AD - \frac{1}{n}\mathbb{1}\mathbb{1}^T)Dc\|}. \qquad \triangle$$

Beweis. Zunächst projizieren wir $\bar{c}$ orthogonal auf den linearen Raum $L := \{x \in \mathbb{R}^n \mid ADx = 0, \mathbb{1}^T x = 0\}$. Setzen wir

$$B = \begin{pmatrix} AD \\ \mathbb{1}^T \end{pmatrix},$$

so ist die Projektion von $\bar{c}$ auf L gegeben durch

$$\bar{\bar{c}} = (I - B^T(BB^T)^{-1}B)\,\bar{c},$$

denn offenbar gilt $B\bar{\bar{c}} = 0$, $(\bar{c} - \bar{\bar{c}})^T\bar{\bar{c}} = 0$. Beachten wir, dass $AD\mathbb{1} = Ax^k = 0$ gilt, so folgt durch einfaches Ausrechnen

$$\begin{aligned} BB^T &= \begin{pmatrix} AD^2A^T & AD\mathbb{1} \\ (AD\mathbb{1})^T & \mathbb{1}^T\mathbb{1} \end{pmatrix} = \begin{pmatrix} AD^2A^T & 0 \\ 0 & n \end{pmatrix} \\ (BB^T)^{-1} &= \begin{pmatrix} (AD^2A^T)^{-1} & 0 \\ 0 & \frac{1}{n} \end{pmatrix} \\ B^T(BB^T)^{-1}B &= DA^T(AD^2A^T)^{-1}AD + \frac{1}{n}\mathbb{1}\mathbb{1}^T, \end{aligned}$$

und somit

$$\bar{\bar{c}} = (I - DA^T(AD^2A^T)^{-1}AD - \frac{1}{n}\mathbb{1}\mathbb{1}^T)\,\bar{c}.$$

Für $y \in L$ gilt nach Konstruktion $\bar{c}^T y = \bar{\bar{c}}^T y$ und somit ist das Minimum von (5.19) gleich dem Minimum der Funktion $\bar{\bar{c}}^T y$ unter den Nebenbedingungen von (5.19). Aufgrund unserer Konstruktion ist dieses Minimum gleich dem Minimum von

$$\begin{aligned} &\min\ \bar{\bar{c}}^T y \\ &\left\|y - \tfrac{1}{n}\mathbb{1}\right\| \le \frac{1}{2}\,\frac{1}{\sqrt{n(n-1)}}, \end{aligned}$$

also dem Minimum einer linearen Zielfunktion über einer Kugel. Offenbar wird das Minimum hier durch den Vektor angenommen, den man durch einen Schritt vom Mittelpunkt $\frac{1}{n}\mathbb{1}$ aus in Richtung $-\bar{\bar{c}}$ mit der Länge des Kugelradius erhält, also durch

$$y^{k+1} \;=\; \frac{1}{n}\mathbb{1} - \frac{1}{2}\,\frac{1}{\sqrt{n(n-1)}}\,\frac{1}{\|\bar{\bar{c}}\|}\,\bar{\bar{c}}.$$

Hieraus folgt die Behauptung. □

Damit haben wir ein Minimum von (5.19) analytisch bestimmten können und unser vereinfachtes Hilfsproblem gelöst. Den nächsten Iterationspunkt erhält man durch Anwendung der inversen projektiven Transformation T_k^{-1} auf den in (5.20) bestimmten Vektor y^{k+1}:

$$x^{k+1} \;:=\; T_k^{-1}(y^{k+1}) \;=\; \frac{1}{\mathbb{1}^T D y^{k+1}}\,D y^{k+1}. \tag{5.21}$$

Dem Hilfsprogramm (5.19) kann man übrigens noch eine andere geometrische Interpretation geben. Setzen wir

$$z := Dy, \quad \text{bzw.} \quad y := D^{-1}z,$$

so kann man (5.19) wie folgt schreiben

$$\begin{aligned} \min\ & c^T z \\ & Az = 0 \\ & \mathbb{1}^T D^{-1} z = 1 \\ & z \in E\left(\tfrac{1}{4n(n-1)} D^2, \tfrac{1}{n} x^k\right). \end{aligned} \tag{5.22}$$

Denn wegen (5.17) gilt $\bar{c}^T y = c^T z$ und aus $D^{-1}x^k = \mathbb{1}$ folgt:

$$\begin{aligned} \left\|y - \tfrac{1}{n}\mathbb{1}\right\| \le \frac{1}{2}\frac{1}{\sqrt{n(n-1)}} &\iff \left\|D^{-1}z - \tfrac{1}{n}D^{-1}x^k\right\| \le \frac{1}{2}\frac{1}{\sqrt{n(n-1)}} \\ &\iff \left\|D^{-1}(z - \tfrac{1}{n}x^k)\right\| \le \frac{1}{2}\frac{1}{\sqrt{n(n-1)}} \\ &\iff (z - \tfrac{1}{n}x^k)^T D^{-2}(z - \tfrac{1}{n}x^k) \le \frac{1}{4n(n-1)} \\ &\iff (z - \tfrac{1}{n}x^k)(4n(n-1))D^{-2}(z - \tfrac{1}{n}x^k) \le 1 \\ &\overset{(4.16)}{\iff} z \in E\left(\tfrac{1}{4n(n-1)} D^2, \tfrac{1}{n} x^k\right) \end{aligned}$$

Gehen wir vom Ursprungsproblem (5.11) aus, so bedeutet dies also, dass wir in unserem Hilfsprogramm die Originalzielfunktion c und den linearen Raum Ω unverändert lassen. Wir ersetzen die Hyperebene E durch $E_k = \{z \mid \mathbb{1}^T D^{-1} z = 1\}$ und die Nichtnegativitätsbedingung $x \in \mathbb{R}^n_+$ durch das Ellipsoid $E(\frac{1}{4n(n-1)} D^2, \frac{1}{n} x^k)$ und optimieren hierüber. Aus einer Optimallösung, sagen wir z^{k+1}, von (5.22) erhalten wir eine Optimallösung von (5.19) durch

$$y^{k+1} = D^{-1} z^{k+1}.$$

Wie im Beweis von (5.20) kann man zeigen, dass (5.22) durch eine direkte Formel gelöst werden kann (das folgt natürlich auch aus (5.20) durch die obige Gleichung), und zwar gilt:

$$z^{k+1} \;=\; \frac{1}{n}x^k - \frac{1}{2}\frac{1}{\sqrt{n(n-1)}} \frac{D(I - DA^T(AD^2A^T)^{-1}AD - \frac{1}{n}\mathbb{1}\mathbb{1}^T)Dc}{\|(I - DA^T(AD^2A^T)^{-1}AD - \frac{1}{n}\mathbb{1}\mathbb{1}^T)Dc\|}. \tag{5.23}$$

Hieraus ergibt sich über (5.21) eine neue (einfache) Formel zur Berechnung des nächsten Iterationspunktes.

$$x^{k+1} \;:=\; \frac{1}{\mathbb{1}^T D y^{k+1}} D y^{k+1} \;=\; \frac{1}{\mathbb{1}^T z^{k+1}} z^{k+1}. \tag{5.24}$$

Damit haben wir die wesentlichen Bestandteile eines Iterationsschrittes des Karmarkar-Algorithmus beschrieben und können ihn zusammenfassend darstellen, wobei noch das Abbruchkriterium zu begründen ist.

(5.25) Algorithmus (Karmarkar-Algorithmus).
Eingabe: $A \in \mathbb{Q}^{(m,n)}$ und $c \in \mathbb{Q}^n$. Zusätzlich wird vorausgesetzt, dass $\frac{1}{n}A\mathbb{1} = 0$ und $c^T\mathbb{1} > 0$ gilt.
Ausgabe: Ein Vektor x mit $Ax = 0$, $\mathbb{1}^T x = 1$, $x \geq 0$ und $c^T x \leq 0$ oder die Feststellung, dass kein derartiger Vektor existiert.

1. **Initialisierung.** Setze

$$x^0 := \frac{1}{n}\mathbb{1}$$
$$k := 0$$
$$N := 3n(\langle A\rangle + 2\langle c\rangle - n)$$

2. **Abbruchkriterium.**
   (2.a) Gilt $k = N$, dann hat $Ax = 0$, $\mathbb{1}^T x = 1$, $x \geq 0$, $c^T x \leq 0$ keine Lösung, STOP!
   (2.b) Gilt $c^T x^k \leq 2^{-\langle A\rangle - \langle c\rangle}$, dann ist eine Lösung gefunden. Falls $c^T x^k \leq 0$, dann ist x^k eine Lösung, andernfalls kann wie bei der Ellipsoidmethode (Satz (4.29)) aus x^k ein Vektor $\overline{x}$ konstruiert werden mit $c^T\overline{x} \leq 0$ $A\overline{x} = 0$, $\mathbb{1}^T\overline{x} = 1$, $\overline{x} \geq 0$, STOP!

3. **Update.**
   (3.a) $D := \operatorname{diag}(x^k)$
   (3.b) $\overline{\overline{c}} := (I - DA^T(AD^2A^T)^{-1}AD - \frac{1}{n}\mathbb{1}\mathbb{1}^T)Dc$ (siehe Beweis von (5.20))
   (3.c) $y^{k+1} := \frac{1}{n}\mathbb{1} - \frac{1}{2}\frac{1}{\sqrt{n(n-1)}}\frac{1}{\|\overline{\overline{c}}\|}\overline{\overline{c}}$
   (3.d) $x^{k+1} := \frac{1}{\mathbb{1}^T Dy^{k+1}}Dy^{k+1}$
   (3.e) $k := k + 1$

4. Gehe zu 2. △

Die Geschwindigkeit des Algorithmus hängt natürlich nur von der Implementierung des Schrittes 3 ab. Wir haben hier die aus Lemma (5.20) gewonnene Formel (5.21) für den nächsten Iterationspunkt x^{k+1} gewählt. Man kann x^{k+1} natürlich auch über (5.23), (5.24) bestimmen. Wesentlich ist, dass man eine Update-Formel findet, in der möglichst wenig arithmetische Operationen auftreten.

Es ist offensichtlich, dass die Hauptarbeit von 3 im Schritt (3.b) liegt. Führt man ihn kanonisch aus, so werden $O(n^3)$ Rechenschritte benötigt. Man beachte, dass sich in jedem Schritt nur die Diagonalmatrix D ändert. Diese Tatsache kann man, wie Karmarkar gezeigt hat, ausnutzen, um 3 in $O(n^{2.5})$ Rechenschritten zu erledigen. Die Laufzeit beträgt dann insgesamt $O(n^{2.5}(\langle A\rangle + \langle c\rangle))$.

Wie bei der Ellipsoidmethode können bei der Ausführung des Karmarkar-Algorithmus irrationale Zahlen (durch Wurzelziehen in 3.c) auftreten. Wir sind also gezwungen, alle

Rechnungen approximativ auszuführen. Die dadurch auftretenden Rundungsfehler akkumulieren sich natürlich. Wir müssen also unsere Rundungsvorschriften so einrichten, dass beim Abbruch des Verfahrens eine korrekte Schlussfolgerung gezogen werden kann. Auch das kann man wie bei der Ellipsoidmethode erledigen. Dies sind (auf Abschätzungstechniken der linearen Algebra und Analysis beruhende) Tricks, mit deren Hilfe Verfahren (5.25) in einen polynomialen Algorithmus abgewandelt werden kann. Da insgesamt höchstens N Iterationen durchgeführt werden, folgt

(5.26) Satz. *Die Laufzeit des (geeignet modifizierten) Karmarkar-Algorithmus* (5.25) *zur Lösung von Problemen des Typs* (5.10) *ist* $O(n^{3.5}(\langle A\rangle + \langle c\rangle)^2)$. △

Wir wollen im Weiteren die Rundungsfehler vernachlässigen und annehmen, dass wir in perfekter Arithmetik arbeiten. Es bleibt noch zu zeigen, dass Algorithmus (5.25) mit einem korrekten Ergebnis endet.

Zunächst überlegen wir uns, dass alle Punkte x^k im relativ Inneren von $\Omega \cap E \cap \mathbb{R}^n_+$ enthalten sind.

(5.27) Lemma. *Für alle* $k \in \{0, 1, \ldots, N\}$ *gilt* $Ax^k = 0$, $x^k > 0$, $\mathbb{1}^T x^k = 1$. △

Beweis. Durch Induktion über k! Für $k = 0$ gilt die Behauptung nach Voraussetzung. Für $k+1$ ist $\overline{\overline{c}}$ die Projektion von c auf $\{x \mid ADx = 0, \mathbb{1}^T x = 0\}$, siehe Beweis von (5.20). Daraus folgt $AD\overline{\overline{c}} = 0$, $\mathbb{1}^T\overline{\overline{c}} = 0$. Mithin ergibt sich

$$\begin{aligned}(\mathbb{1}^T Dy^{k+1})Ax^{k+1} = \; ADy^{k+1} \; &= \; AD\left(\frac{1}{n}\mathbb{1} - \frac{1}{2}\,\frac{1}{\sqrt{n(n-1)}}\,\frac{1}{\|\overline{\overline{c}}\|}\,\overline{\overline{c}}\right)\\ &= \; AD\left(\frac{1}{n}\mathbb{1}\right) \; = \; Ax^k = 0.\end{aligned}$$

Daraus folgt $Ax^{k+1} = 0$.

Um $x^k > 0$ zu zeigen, stellen wir zunächst fest, dass

$$K := \left\{y \mid \|y - \frac{1}{n}\mathbb{1}\| \le \frac{1}{2}\,\frac{1}{\sqrt{n(n-1)}}\right\} \subseteq \mathbb{R}^n_+.$$

Denn gibt es ein $\overline{y} \in K$ mit einer nichtpositiven Komponente, sagen wir $\overline{y}_1 \le 0$, so gilt

$$\sum_{i=2}^{n}\left(\overline{y}_i - \frac{1}{n}\right)^2 \le \frac{1}{4n(n-1)} - \left(\overline{y}_1 - \frac{1}{n}\right)^2 \le \frac{1}{4n(n-1)} - \frac{1}{n^2} = \frac{-3n+4}{4n^2(n-1)} < 0,$$

ein Widerspruch. Daraus folgt $y^{k+1} > 0$, und (5.21) ergibt direkt $x^{k+1} > 0$.

Aus (5.21) folgt ebenfalls sofort, dass $\mathbb{1}^T x^{k+1} = 1$ gilt. □

Entscheidend für die Abschätzung der Anzahl der Iterationsschritte ist die folgende Beobachtung.

(5.28) Lemma. *Gibt es ein* $x \ge 0$ *mit* $Ax = 0$, $\mathbb{1}^T x = 1$ *und* $c^T x \le 0$, *dann gilt für alle* k*:*

$$\frac{(c^T x^{k+1})^n}{\prod_{i=1}^n x_i^{k+1}} \; \le \; \frac{2}{e}\,\frac{(c^T x^k)^n}{\prod_{i=1}^n x_i^k}.$$ △

Wir wollen uns nun überlegen, wieviele Iterationen im Verfahren (5.25) höchstens durchgeführt werden müssen. Der erste Iterationspunkt ist der Vektor

$$x^0 = \frac{1}{n}\mathbb{1}.$$

Daraus folgt mit einer groben Abschätzung aus (4.6)(a)

$$c^T x^0 \;\le\; \frac{1}{n}\sum_{i=1}^{n}|c_i| \;\le\; \frac{1}{n}\sum_{i=1}^{n}2^{\langle c_i\rangle-1} \;<\; \frac{1}{n}2^{\langle c\rangle-n}.$$

Aus Lemma (5.28) ergibt sich (unter den dort gemachten Voraussetzungen)

$$\frac{(c^T x^k)^n}{\prod_{i=1}^n x_i^k} \;\le\; \left(\frac{2}{e}\right)^k \frac{(c^T x^0)^n}{\prod_{i=1}^n x_i^0}$$

und somit wegen $\prod_{i=1}^n x_i^k \le 1$ und $\prod_{i=1}^n x_i^0 = (\frac{1}{n})^n$:

$$c^T x^k \;<\; \left(\frac{2}{e}\right)^{\frac{k}{n}} \left(\frac{\prod x_i^k}{\prod x_i^0}\right)^{\frac{1}{n}} \frac{1}{n}\,2^{\langle c\rangle-n} \;\le\; \left(\frac{2}{e}\right)^{\frac{k}{n}} 2^{\langle c\rangle-n}. \tag{5.29}$$

In jedem Schritt schrumpft also der Zielfunktionswert um den nur von der Dimension n abhängigen Faktor $\sqrt[n]{\frac{2}{e}}$. Setzen wir

$$N := 3n(\langle A\rangle + 2n\langle c\rangle - n) \tag{5.30}$$

dann gilt:

(5.31) Satz. *Es gibt ein $x \ge 0$ mit $Ax = 0$, $\mathbb{1}^T x = 1$, $c^T x \le 0$ genau dann, wenn es ein $k \in \{0,\dots,N\}$ gibt mit*

$$c^T x^k < 2^{-\langle A\rangle-\langle c\rangle}.$$

△

Beweis. „$\Longrightarrow$“: Angenommen, es gibt ein $x \in P = \Sigma \cap \Omega$ mit $c^T x \le 0$, dann sind die Voraussetzungen von Lemma (5.28) erfüllt und folglich gilt mit (5.29) und (5.30)

$$c^T x^N \;<\; \left(\frac{2}{e}\right)^{\frac{N}{n}} 2^{\langle c\rangle-n} \;<\; 2^{-\langle A\rangle-\langle c\rangle},$$

wobei wir die Tatsache ausgenutzt haben, dass $3 > \frac{-1}{\log(\frac{2}{e})}$ gilt.

„$\Longleftarrow$“: Angenommen $c^T x^k < 2^{-\langle A\rangle-\langle c\rangle}$ gilt für ein $k \in \{0,\dots,N\}$. Da $P \neq \emptyset$ ein Polytop ist, wird $\min\{c^T x \mid x \in P\}$ in einer Ecke von P angenommen. Nach Satz (4.11) ist der Optimalwert dieses Programms eine rationale Zahl $\frac{p}{q}$ mit

$$1 \;\le\; q \;\le\; 2^{\langle A\rangle+\langle\mathbb{1}\rangle+\langle c\rangle-n^2-n} \;<\; 2^{\langle A\rangle+\langle c\rangle}.$$

Aus $\frac{p}{q} \le c^T x^k < 2^{-\langle A\rangle-\langle c\rangle}$ folgt dann $\frac{p}{q} \le 0$. □

5.2 Primaler Pfadverfolgungsalgorithmus

Wir skizzieren nun ein Beispiel einer Klasse von Innere-Punkte-Methoden, die primale Pfadverfolgungsalgorithmen genannt werden. Wir folgen dabei der Darstellung dieses Themas in (Bertsimas and Tsitsiklis, 1997, Kapitel 9). Weitere Literatur hierzu findet sich in Roos et al. (2006) und Wright (1997).

Die Idee der Algorithmen ist die Folgende. Wir gehen aus von einem LP in Standardform (P) und seinem dualen Programm (D) mit zusätzlich eingeführten Schlupfvariablen s in der folgenden Form:

$$\begin{array}{lll} \min\ c^T x & & \max\ u^T b \\ Ax = b & \text{(P)} \quad \text{und} & u^T A + s^T = c^T \quad \text{(D)} \\ x \geq 0 & & s \geq 0 \end{array}$$

Wir starten mit einem für (P) zulässigen Punkt x^0, dessen Komponenten strikt positiv sind. Dieser Punkt $x^0 \in \mathring{P} := \{x \in \mathbb{R}^n \mid Ax = b, x > 0\}$ ist ein innerer Punkt im Sinne dieser Vorlesung, d. h. ein innerer Punkt in der Relativtopologie des Zulässigkeitsbereichs. Wir versuchen nun eine Folge von Punkten $(x^k)_{k\geq 1}$ zu generieren, die alle in $\mathring{P}$ liegen, die Abstand vom Rand des Polyeders halten und gegen einen Optimalwert konvergieren. Bei unserem speziellen Polyedertyp sind alle Randpunkte in mindestens einer der Mengen $\{x \in \mathbb{R}^n \mid x_i = 0\}$, $i = 1, \ldots, n$ enthalten. Die gewünschte Abstandswahrung vom Rand versuchen wir dadurch zu erreichen, dass eine Annäherung an den Rand in einer Komponente x_i^k bestraft wird. Eine Funktion, die so etwas leistet, ist der Logarithmus.

Für $\mu > 0$ definieren wir die *Barriere-Funktion*

$$B_\mu(x) := \begin{cases} c^T x - \mu \sum_{j=1}^{n} \log x_j & \text{für } x > 0, \\ \infty & \text{falls } x_j \leq 0 \text{ für ein } j = 1, \ldots, n. \end{cases} \tag{5.32}$$

Das *Barriere-Problem* lautet dann:

$$\begin{array}{l} \min\ B_\mu(x) \\ Ax = b \end{array} \tag{5.33}$$

Diese Definition impliziert, dass alle Punkte, die $Ax = b$ erfüllen, aber nicht in $\mathring{P}$ liegen, einen unendlichen Zielfunktionswert haben. Falls es also Optimallösungen von (5.33) mit endlichem Wert gibt, so sind diese zulässig für (P). Durch wiederholte Lösung von (5.33) bei Veränderung des Barriere-Parameters μ wird versucht, eine Annäherung an eine Optimallösung von (P) zu erreichen.

Die Barriere-Funktion (5.32) ist strikt konvex. Dies impliziert, dass eine Optimallösung mit endlichem Wert, falls eine solche existiert, eindeutig ist. Wir treffen die Annahme, dass alle Barriere-Probleme für $\mu > 0$ eine Optimallösung $x(\mu)$ haben, die somit eindeutig ist. Mit „Standardtricks“ kann man erreichen, dass diese Annahme beim Beginn des Verfahrens erfüllt ist. Beim Simplex-Verfahren haben wir derartige Methoden ausführlich diskutiert, hier verzichten wir darauf.

(5.34) Definition. *Für alle $\mu > 0$ seien Barriere-Probleme* (5.33) *gegeben, deren Optimallösungen jeweils mit $x(\mu)$ bezeichnet seien. Der* zentrale Pfad *ist die Menge $\{x(\mu) \mid \mu > 0\}$ und der (wohldefinierte) Punkt $\lim_{\mu\to\infty} x(\mu)$ heißt* analytisches Zentrum. △

Das analytische Zentrum ist ein Punkt, der – im Sinne der logarithmischen Bestrafungsfunktion – so weit wie möglich von allen Rändern des zulässigen Bereichs entfernt ist, siehe Abbildung 5.2. $\lim_{\mu\to 0} x(\mu)$ ist die Optimallösung unseres originalen LPs.

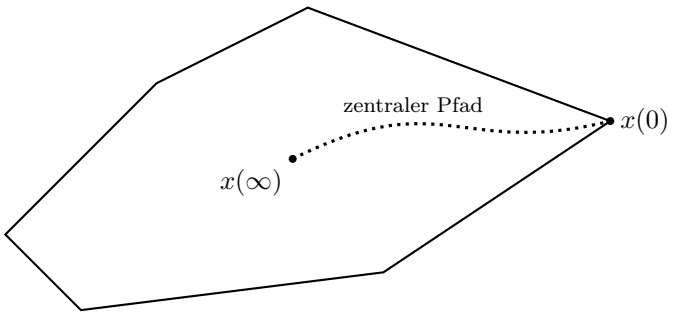


Abbildung 5.2: Zentraler Pfad

(5.35) Beispiel.

(a) Betrachte das Minimierungsproblem

$$\begin{aligned} \min\ & x \\ & x \geq 0 \end{aligned}$$

Die Barriere-Funktion lautet: $B_\mu(x) = x - \mu \log x$, $(x > 0)$. Zum Auffinden des Minimums leiten wir ab:

$$\frac{d}{dx} B_\mu(x) = 1 - \frac{\mu}{x} = 0 \iff x = \mu,$$

Die Optimallösung zum Barriere-Problem lautet also $x(\mu) = \mu$, d.h.

$$\lim_{\mu\to 0} x(\mu) = 0.$$

(b) Betrachte das Minimierungsproblem

$$\begin{aligned} \min\ & x_2 \\ & x_1 + x_2 + x_3 = 1 \\ & x_1, x_2, x_3 \geq 0 \end{aligned}$$

Das Barriere-Problem lautet:

$$\begin{aligned} \min\ & x_2 - \mu \log x_1 - \mu \log x_2 - \mu \log x_3 \\ & x_1 + x_2 + x_3 = 1 \end{aligned}$$

Einsetzen der Gleichheitsbedingung in die Zielfunktion führt zu folgendem zentralem Pfad:

$$x(\mu) \;=\; \frac{1}{2}\begin{pmatrix} \sqrt{9\mu^2+2\mu+1}-3\mu \\ 1+3\mu-\sqrt{9\mu^2+2\mu+1} \\ \sqrt{9\mu^2+2\mu+1}-3\mu \end{pmatrix}$$

mit

$$\lim_{\mu\to 0} x(\mu) = \begin{pmatrix} 1/2 \\ 0 \\ 1/2 \end{pmatrix}, \qquad \lim_{\mu\to\infty} x(\mu) = \begin{pmatrix} 1/3 \\ 1/3 \\ 1/3 \end{pmatrix},$$

siehe Abbildung 5.3. An diesem Beispiel sieht man, dass bei diesem Ansatz eine

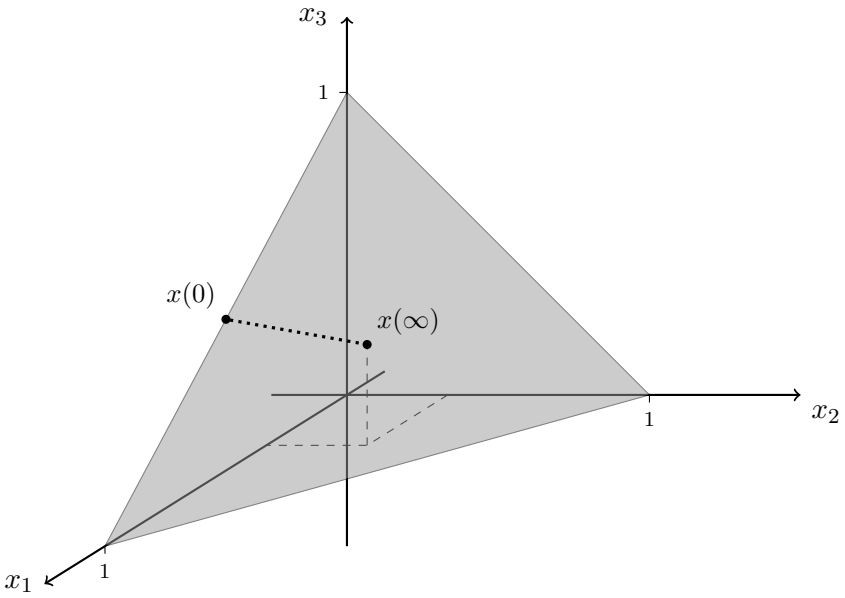


Abbildung 5.3: Beispiel (5.35)(b)

Optimallösung angestrebt wird, die „in der Mitte“ der optimalen Seitenfläche liegt.△

Zum Dualproblem (D) kann man analog ein zugehöriges duales Barriere-Problem definieren:

$$\begin{aligned} \max\; & u^T b + \mu \sum_{j=1}^{n} \log s_j \\ & u^T A + s^T = c^T \end{aligned} \tag{5.36}$$

Wendet man den Kuhn-Tucker-Satz aus der nichtlinearen Optimierung (dieser ist eine Verallgemeinerung des Dualitätssatzes der linearen Optimierung) auf die nichtlinearen Programme (5.33) und (5.36) an, so ergibt sich:

(5.37) Satz (Kuhn-Tucker). *$x^* \in \mathbb{R}^n$, $u^* \in \mathbb{R}^m$ und $s^* \in \mathbb{R}^n$ sind optimal bezüglich* (5.33) *und* (5.36) *genau dann, wenn die folgenden Bedingungen (auch genannt* Karush-Kuhn-Tucker-Bedingungen*) gelten:*

$$\begin{aligned} Ax^* &= b \\ x^* &\geq 0 \\ A^T u^* + s^* &= c \\ s^* &\geq 0 \\ x_j^* s_j^* &= \mu \qquad \forall j = 1, \ldots, n \end{aligned}$$ △

Beweis. Siehe (Bertsimas and Tsitsiklis, 1997, S. 421). □

Wir nehmen nun an, dass $x \in \mathring{P}$ gilt. Die logarithmische, und damit schwer zu behandelnde primale Barriere-Funktion ersetzen wir für $x > 0$ durch die Approximation durch die ersten Glieder ihrer Taylorreihe.

$$\begin{aligned} B_\mu(x+d) &\approx B_\mu(x) + \sum_{i=1}^n \frac{\partial}{\partial x_i} B_\mu(x) d_i + \frac{1}{2} \sum_{i,j=1}^n \frac{\partial^2}{\partial x_i \partial x_j} B_\mu(x) d_i d_j \\ &= B_\mu(x) + \sum_{i=1}^n \Big(c_i - \frac{\mu}{x_i}\Big) d_i + \frac{1}{2} \sum_{i=1}^n \frac{\mu}{x_i^2} d_i^2 \\ &= B_\mu(x) + (c^T - \mu \mathbb{1}^T X^{-1}) \cdot d + \frac{1}{2}\, \mu d^T (X^2)^{-1} d \end{aligned}$$

mit

$$X := \operatorname{diag}(x) = \begin{pmatrix} x_1 & & \\ & \ddots & \\ & & x_n \end{pmatrix}, \qquad d = \begin{pmatrix} d_1 \\ \vdots \\ d_n \end{pmatrix}.$$

Anstatt die Barriere-Funktion direkt zu minimieren, suchen wir eine Richtung d, die diese (abgebrochene) Taylorentwicklung von $B_\mu(x+d)$ minimiert, so dass $x + d$ wieder die Nebenbedingung $A(x+d) = b$, und das heißt $Ad = 0$, erfüllt:

$$\begin{aligned} \min\ & (c^T - \mu \mathbb{1}^T X^{-1}) \cdot d + \frac{1}{2} \mu d^T (X^2)^{-1} d \\ & Ad = 0 \end{aligned} \tag{5.38}$$

Das Problem (5.38) wird nun mit Hilfe von Lagrange-Multiplikatoren gelöst. Wir definieren eine Lagrange-Funktion mit Lagrange-Multiplikatoren $\Lambda^T = (\lambda_1, \ldots, \lambda_m)$ wie folgt:

$$L(d, \Lambda) = (c^T - \mu \mathbb{1}^T X^{-1}) \cdot d + \frac{1}{2} \mu d^T (X^2)^{-1} d - \Lambda^T A d.$$

Eine Lösung finden wir dadurch, dass wir die folgenden Gleichungen lösen:

$$\frac{\partial}{\partial d_i} L(d, \Lambda) = 0 \quad \forall i = 1, \ldots, n, \qquad \frac{\partial}{\partial \lambda_j} L(d, \Lambda) = 0 \quad \forall j = 1, \ldots, m.$$

Es gilt:

$$\begin{pmatrix} \frac{\partial}{\partial d_1} \\ \vdots \\ \frac{\partial}{\partial d_n} \end{pmatrix} L(d,\Lambda) = c - \mu X^{-1}\mathbb{1} + \mu(X^2)^{-1}d - A^T\Lambda, \qquad \begin{pmatrix} \frac{\partial}{\partial \lambda_1} \\ \vdots \\ \frac{\partial}{\partial \lambda_m} \end{pmatrix} L(d,\Lambda) = Ad,$$

was auf das folgende lineare Gleichungssystem führt:

$$\begin{pmatrix} \mu(X^2)^{-1} & -A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} d \\ \Lambda \end{pmatrix} = \begin{pmatrix} \mu X^{-1}\mathbb{1} - c \\ 0 \end{pmatrix}.$$

Die Lösung dieses Gleichungssystems kann man explizit angeben:

$$\begin{aligned} d(\mu) &= \left(I - X^2A^T(AX^2A^T)^{-1}A\right)\left(X\mathbb{1} - \frac{1}{\mu}X^2c\right), \\ \Lambda(\mu) &= \left(AX^2A^T\right)^{-1} A\left(X^2c - \mu X\mathbb{1}\right). \end{aligned}$$

Hat man eine zulässige Lösung zum aktuellen Barriere-Problem (5.33) für ein bestimmtes μ, so bekommt man auf diese Weise eine neue zulässige Lösung $x + d(\mu)$ für (5.33). Der Vektor $d(\mu)$ heißt *Newton-Richtung*, die obige Berechnung ein *Newton-Schritt*, da er als Schritt in der Anwendung des Newton-Verfahrens der nichtlinearen Optimierung aufgefasst werden kann. Iteriert man dieses Vorgehen, dann konvergiert die Folge der so erzeugten Lösungen gegen die (beweisbar eindeutige) Optimallösung $x(\mu)$ des Barriere-Problems (5.33) zum Parameter μ.

Um das Originalproblem zu lösen, bedient man sich dieser Idee, geht aber etwas anders vor: Man verringert μ in jeder Iteration durch Multiplizieren mit einem festen Parameter α, wobei $0 < \alpha < 1$. Ein allgemeiner Schritt im Gesamtverfahren, das mit einem Startpunkt $x^0 \in \mathring{P}$, einem Barriere-Parameter $\mu > 0$ und einem Skalierungsparameter $0 < \alpha < 1$ startet, sieht dann wie folgt aus (hierbei ist $k = 0, 1, 2, \ldots$ der Iterationszähler): Wir haben einen gegenwärtigen Punkt $x^k \in \mathring{P}$ und den aktuellen Barriere-Parameter $\alpha^k\mu$. (Achtung! In α^k ist k als Exponent zu verstehen, wohingegen k an den Variablen x, u und s in diesem Abschnitt als oberer Index steht.) Wenn wir von x^k ausgehend den Punkt $x(\alpha^k\mu)$ (die Lösung des Barriere-Problems (5.33) zum Parameter $\alpha^k\mu$) berechnen wollten, würden wir eine Folge von Newton-Schritten machen, die gegen $x(\alpha^k\mu)$ konvergiert. Wir machen jedoch nur den ersten Schritt dieser Folge und bestimmen von x^k ausgehend die erste Näherungslösung $x^{k+1} := x^k + d(\alpha^k\mu)$ von $x(\alpha^k\mu)$. Statt nun weiter auf $x(\alpha^k\mu)$ zuzusteuern, verändern wir den Barriere-Parameter von $\alpha^k\mu$ auf $\alpha^{k+1}\mu$. Nun bestimmen wir von x^{k+1} ausgehend die erste Näherungslösung $x^{k+2} := x^{k+1} + d(\alpha^{k+1}\mu)$ von dem Punkt $x(\alpha^{k+1}\mu)$ auf dem zentralen Pfad, usw., siehe Abbildung 5.4.

Mit dieser Interpretation ist klar, dass wir ein Verfahren vor uns haben, das versucht dem zentralen Pfad zum Optimalpunkt zu folgen, aber (zur Aufwandsreduzierung) jeweils nur Punkte in der Nähe des zentralen Pfades erzeugt.

In jedem Schritt k erhält man übrigens auch eine passende duale Lösung zu $x^{k+1} = x^k + d(\alpha^k\mu)$ durch

$$\begin{aligned} u^{k+1} &= \Lambda(\alpha^k\mu), \\ s^{k+1} &= c - A^Tu^{k+1}. \end{aligned}$$

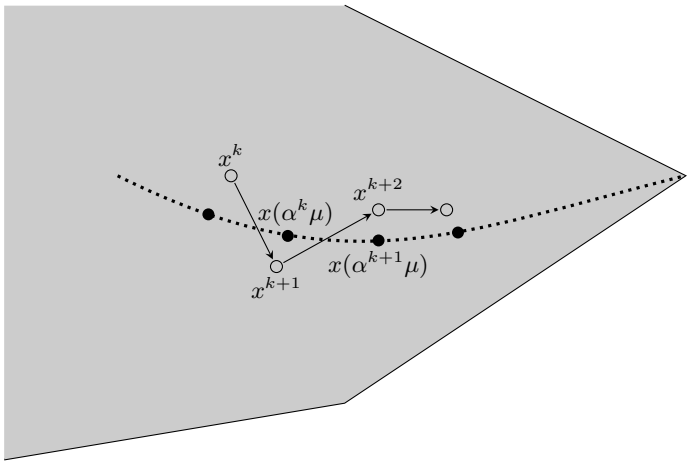


Abbildung 5.4: Newton-Schritte

Wir fassen die Überlegungen zusammen:

(5.39) Algorithmus (Primaler Pfadverfolgungsalgorithmus).
Eingabe: A, b, c, Genauigkeitsparameter $\varepsilon > 0$, $0 < \alpha < 1$, Startparameter $\mu^0 > 0$, zulässige primale und duale Startlösungen $x^0 > 0$, $s^0 > 0$, u^0
Ausgabe: ε-genaue primale und duale Lösungen x, s, u

1. Setze $k := 0$

2. (Optimalitätstest)
   Falls $(s^k)^T x^k < \varepsilon$, setze $x := x^k$, $u := u^k$, $s := s^k$. STOP

3. (Update Barriere-Parameter)
   Setze
   $$X_k := \begin{pmatrix} x_1^k & & \\ & \ddots & \\ & & x_n^k \end{pmatrix}, \qquad \mu^{k+1} := \alpha\mu^k$$

4. (Berechnung Newton-Richtung)
   Löse das lineare Gleichungssystem
   $$\begin{pmatrix} \mu^{k+1}(X_k^2)^{-1} & -A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} d \\ \Lambda \end{pmatrix} = \begin{pmatrix} \mu^{k+1} X_k^{-1}\mathbb{1} - c \\ 0 \end{pmatrix}.$$
   in den Variablen d und Λ.

5. (Update Lösungen)
   Setze

$$\begin{aligned} x^{k+1} &:= x^k + d, \\ u^{k+1} &:= \Lambda, \\ s^{k+1} &:= c - A^T\Lambda. \end{aligned}$$

6. Setze $k := k + 1$ und gehe zu Schritt 2. △

Zur Konvergenz: Man kann zeigen, dass während des Algorithmus gilt

$$\sum_{i=1}^{n}(x_i^k s_i^k - \mu^k)^2 \leq (\mu^k)^2\beta$$

für einen festen Wert $\beta < 1$. Wenn also μ^k nahe genug an 0 ist, muss $x_i^k s_i^k$ nahe an 0 sein für alle $i = 1, \ldots, n$ und die Bedingung für den komplementären Schlupf ist näherungsweise erfüllt. Genauer gilt:

(5.40) Satz. *Für $0 < \beta < 1$ sei*

$$\alpha \;=\; 1 - \frac{\sqrt{\beta} - \beta}{\sqrt{\beta} + \sqrt{n}},$$

$\mu^0 > 0$, $\varepsilon > 0$, und $x^0 > 0$, $s^0 > 0$, u^0 seien zulässige Startlösungen mit

$$\left\| \frac{1}{\mu^0} X_0 S_0 \mathbb{1} - \mathbb{1} \right\| \leq \beta,$$

wobei

$$X^0 \;=\; \begin{pmatrix} x_1^0 & & \\ & \ddots & \\ & & x_n^0 \end{pmatrix}, \qquad S^0 \;=\; \begin{pmatrix} s_1^0 & & \\ & \ddots & \\ & & s_n^0 \end{pmatrix}.$$

Dann findet Algorithmus (5.39) *eine primal-duale Lösung x^K, s^K, u^K mit Genauigkeitslücke $(s^K)^T x^K \leq \varepsilon$ nach*

$$K = \left\lceil \frac{\sqrt{\beta} + \sqrt{n}}{\sqrt{\beta} - \beta} \; \log \frac{(s^0)^T x^0 (1 + \beta)}{\varepsilon(1 - \beta)} \right\rceil$$

Iterationen. △

Beweis. Siehe (Bertsimas and Tsitsiklis, 1997, S. 427–429). □

Um das Verfahren zu initialisieren, braucht man noch Startlösungen. Man bekommt zulässige Lösungen $x^0 > 0$, $s^0 > 0$ mit

$$\left\| \frac{1}{\mu^0} X_0 S_0 \mathbb{1} - \mathbb{1} \right\| \leq \beta = \frac{1}{4}$$

aus einem Hilfs-LP, das sich mit „offensichtlicher“ Startlösung lösen lässt, ähnlich wie in Phase I des Simplexalgorithmus.

Nach Satz (5.40) beläuft sich mit $\beta = \frac{1}{4}$ die Anzahl an notwendigen Iterationen, um die Genauigkeitslücke von $\varepsilon_0 = (s^0)^T x^0$ auf ε zu verkleinern, auf

$$\left\lceil (2 + 4\sqrt{n}) \log \left(\frac{\varepsilon_0}{\varepsilon} \cdot \frac{5}{3} \right) \right\rceil,$$

wobei in jeder Iteration $O(n^3)$ Operationen anfallen (Lösung des Gleichungssystems mit $m \leq n$). Damit ist die Gesamtlaufzeit von Algorithmus (5.39) in

$$O\Big(n^3 \sqrt{n} \log \frac{\varepsilon_0}{\varepsilon} \Big).$$

5.3 Primal-dualer Pfadverfolgungsalgorithmus

Die heute verwendeten Barriere-Methoden bauen fast alle auf dem primal-dualen Pfadverfolgungsalgorithmus auf. Die Grundform dieses Verfahrens stammt u. a. von Kojima et al. (1989). In diesem Abschnitt wird das Grundprinzip eines solchen Algorithmus skizziert. Für mehr Details siehe z. B. Wright (1997).

Die Idee besteht darin, die KKT-Bedingungen in (5.37) mit Hilfe des Newton-Verfahrens direkt zu lösen. Ganz allgemein bestimmt das Newton-Verfahren für eine gegebene Funktion $F : \mathbb{R}^r \to \mathbb{R}^r$ eine Nullstelle z^* von F, d. h. ein $z^* \in \mathbb{R}^r$ mit $F(z^*) = 0$, und funktioniert wie folgt: Angenommen, wir haben bereits eine Näherung z für z^*, dann suchen wir eine Richtung $d \in \mathbb{R}^r$, so dass $z + d$ eine bessere Näherung für z^* ist. Approximation von F durch die Taylorentwicklung ergibt wieder:

$$F(z + d) \approx F(z) + J_F(z) \cdot d$$

mit der Jacobi-Matrix

$$J_F(z) \; = \left(\frac{\partial}{\partial z_j} F_i(z) \right)_{1 \leq i,j \leq r}.$$

Das Newton-Verfahren besteht darin, in jedem Schritt ein d (die *Newton-Richtung*) zu suchen, so dass $F(z) + J_F(z) \cdot d = 0$.

In unserem Fall wird die Funktion F durch die Karush-Kuhn-Tucker-Bedingungen bestimmt: ein Punkt $(x^*, u^*, s^*) \in \mathbb{R}^{2n+m}$ mit $x^* \geq 0$ und $s^* \geq 0$ sowie

$$\begin{aligned} Ax^* &= b \\ A^T u^* + s^* &= c \\ X^* S^* \mathbb{1} &= \mu \mathbb{1} \end{aligned}$$

(vgl. (5.37)) ist eine Nullstelle der Funktion

$$F : \mathbb{R}^{2n+m} \to \mathbb{R}^{2n+m}, \qquad F(x, u, s) \; = \begin{pmatrix} Ax - b \\ A^T u + s - c \\ XS\mathbb{1} - \mu \mathbb{1} \end{pmatrix},$$

wobei wieder $X^* := \mathrm{diag}(x^*)$, $S^* := \mathrm{diag}(s^*)$. Zu lösen ist dann in jedem Newton-Schritt das lineare Gleichungssystem

$$\begin{pmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S & 0 & X \end{pmatrix} \cdot \begin{pmatrix} d_x \\ d_u \\ d_s \end{pmatrix} = -F(x,u,s) = \begin{pmatrix} 0 \\ 0 \\ \mu \mathbb{1} - XS\mathbb{1} \end{pmatrix}, \tag{5.41}$$

falls die aktuelle Lösung (x, u, s) primal und dual zulässig ist.

Das Newton-Verfahren verwendet keine Schrittlängen. Bei dem hier skizzierten Ansatz kann es daher passieren, dass der nächste Iterationspunkt $(x^{k+1}, u^{k+1}, s^{k+1})$ nicht zulässig ist. Deswegen müssen Schrittlängen eingebaut werden. Es hat sich als sinnvoll erwiesen, für primale und duale Variablen unterschiedliche Schrittlängen β_P und β_D zu wählen. Die neue Lösung ergibt sich dann aus der aktuellen Lösung (x^k, u^k, s^k) durch

$$x^{k+1} = x^k + \beta_P d_x\ , \quad u^{k+1} = u^k + \beta_D d_u\ , \quad s^{k+1} = s^k + \beta_D d_s.$$

Die Nichtnegativitätsbedingung $x^k + \beta_P d_x \geq 0$ übersetzt sich in

$$\beta_P \leq -\frac{x_i}{(d_x)_i} \qquad \forall i \text{ mit } (d_x)_i < 0$$

(analog für β_D), so dass man für die Schrittlängen

$$\beta_P = \min\left\{1, \alpha \min_{\{i|(d_x)_i<0\}} \Big(-\frac{x_i}{(d_x)_i}\Big)\right\},$$
$$\beta_D = \min\left\{1, \alpha \min_{\{i|(d_s)_i<0\}} \Big(-\frac{s_i}{(d_s)_i}\Big)\right\},$$

mit $0 < \alpha < 1$ wählen kann. ($\alpha < 1$, um nicht auf den Rand des Zulässigkeitsbereichs zu stoßen und numerische Probleme zu vermeiden.)

Wie im primalen Pfadverfolgungsalgorithmus wird auch der Barriere-Parameter in jedem Schritt angepasst, z. B. durch

$$\mu^k = \rho_k \frac{(x^k)^T s^k}{n} \tag{5.42}$$

mit $0 < \rho_k \leq 1$. Für die Wahl von ρ_k gibt es verschiedene Möglichkeiten; häufig wird z. B. $\rho_k = 1$ gewählt, solange der Algorithmus relativ große Fortschritte macht, und $\rho_k < 1$ sonst.

(5.43) Algorithmus (Primal-dualer Pfadverfolgungsalgorithmus).

Eingabe: A, b, c, Genauigkeitsparameter $\varepsilon > 0$, zulässige primale und duale Startlösungen $x^0 > 0$, $s^0 > 0$, u^0

Ausgabe: ε-genaue primale und duale Lösungen x, s, u

1. Setze $k := 0$
2. (Optimalitätstest)
   Falls $(s^k)^T x^k < \varepsilon$, setze $x := x^k$, $u := u^k$, $s := s^k$. STOP

3. (Berechnung Newton-Richtung)
   Für ein geeignetes $\rho_k \in (0,1]$ setze

$$\mu^k := \rho_k \frac{(x^k)^T s^k}{n}, \qquad X_k := \begin{pmatrix} x_1^k & & \\ & \ddots & \\ & & x_n^k \end{pmatrix}, \qquad S_k = \begin{pmatrix} s_1^0 & & \\ & \ddots & \\ & & s_n^0 \end{pmatrix}$$

   und löse das lineare Gleichungssystem

$$\begin{pmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S_k & 0 & X_k \end{pmatrix} \cdot \begin{pmatrix} d_x^k \\ d_u^k \\ d_s^k \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \mu^k \mathbb{1} - X_k S_k \mathbb{1} \end{pmatrix}.$$

4. (Bestimmung Schrittlängen)
   Setze

$$\beta_P^k = \min\left\{1, \alpha \min_{\{i|(d_x^k)_i<0\}} \Big(- \frac{x_i^k}{(d_x^k)_i} \Big) \right\},$$
$$\beta_D^k = \min\left\{1, \alpha \min_{\{i|(d_s^k)_i<0\}} \Big(- \frac{s_i^k}{(d_s^k)_i} \Big) \right\}.$$

5. (Update Lösungen)
   Setze

$$x^{k+1} := x^k + \beta_P d_x^k,$$
$$u^{k+1} := u^k + \beta_D d_u^k,$$
$$s^{k+1} := s^k + \beta_D d_s^k.$$

6. Setze $k := k+1$ und gehe zu Schritt 2 △

Bezüglich der Komplexität des Algorithmus kann man zeigen, dass die Anzahl Iterationen bei Start-Genauigkeit $\varepsilon_0 = (s^0)^T x^0$ im worst case in $O(\sqrt{n} \log \frac{\varepsilon_0}{\varepsilon})$ ist. In der Praxis wurde sogar eine durchschnittliche Laufzeit von $O(\log n \log \frac{\varepsilon_0}{\varepsilon})$ beobachtet, aber bisher nicht bewiesen.

Mehrotras Prediktor-Korrektor-Methode

Die heute in kommerziellen Codes meist verwendeten Barriere-Verfahren sind üblicherweise Varianten der „Prediktor-Korrektor-Methode“ von Merothra, siehe Mehrotra (1992), die im Prinzip folgendermaßen funktionert. In jeder Iteration wird die nächste Richtung mittels Lösen des Gleichungssystems (5.41) bestimmt, wobei jedoch statt μ ein skalierter Wert $\sigma\mu$ benutzt wird. Der Skalierungsfaktor $0 < \sigma < 1$ wird vorher bestimmt, indem ein Newton-Schritt „simuliert“ wird (ohne die primale und duale Gültigkeit von x, u, s vorauszusetzen) und σ je nach Länge des erlaubten Schrittes gewählt wird.

Geometrische Interpretation: Der „simulierte“ Schritt berechnet eine Richtung, in die man gehen möchte (Prediktor-Schritt). Wenn sich diese Richtung als „gut“ erweist, d. h. man durch (5.42) eine deutliche Verringerung von μ erreichen würde, wird σ nahe an 0 gewählt, ansonsten nahe an 1. Danach wird die tatsächliche Richtung unter Verwendung von σ berechnet (Korrektor-Schritt). In der Regel führt dieses Vorgehen zu einer deutlichen Verringerung der Anzahl an Iterationen. Der zusätzliche Aufwand in jedem Schritt hält sich in Grenzen: es ist zwar ein zusätzliches Gleichungssystem zu lösen, dessen linke Seite stimmt aber mit dem ohnehin zu lösenden überein, so dass man die Faktorisierung der Matrix wiederverwenden kann.

Die Prediktor-Korrektor-Methode liefert keine neuen theoretischen Erkenntnisse, hat sich aber in der Praxis als das bisher effizienteste Verfahren zur Lösung von großen LPs herausgestellt. Natürlich gibt es auch hier wieder viele verschiedene Varianten und es sind eine Menge numerischer Details zu beachten, auf die hier nicht näher eingegangen werden kann; für mehr Informationen siehe Wright (1997).

Aussagen führender LP-Software-Anbieter

Im Frühjahr 2013 habe ich einige der führenden kommerziellen Anbieter von Optimierungssoftware gefragt, welche Version der Innere-Punkte- bzw. Barriere-Methoden in ihrem LP-Paket implementiert ist. Bob Bixby (Gurobi) antwortete:

> „We use a primal-dual log-barrier long-path method with predictor corrector, where two important practical issues are the choice of starting point and the handling of dense columns. And, for these last two issues there really isn't any clear winner to point to. Gurobi does also offer the option to run the homogeneous algorithm.... This version has theoretical advantages when handling infeasibilities, though, as it turns out, the standard algorithm can also be slightly modified to produce infeasibility proofs.“

Christian Bliek (IBM, software package CPLEX) schrieb:

> „For lp/qp our barrier solver uses Mehrotra predictor corrector, and Gondzio multiple corrections. We recently switched to the homogeneous method for our default, as I found it to be numerically more robust. For this I had to work a bit to reduce the extra cost associated with this method. We still work with the normal equations, and the normal matrix is factorized using Cholesky. We have three orderings; approximate minimum degree, approximate minimum fill or nested dissection. Our default automatically selects what it thinks the best ordering algorithm is for the problem at hand. Dense columns and free variables are handled in a special way. The challenge for these is numerical stability. Implementers will typically not share too much information about this, but you can find some approaches described in “Primal-Dual Interior-Point Methods” by Stephen Wright or in papers by Csaba Meszaros or Erling Andersen.“

Wie in der Vorlesung mehrfach erwähnt, die Implementierung von Barriere-Methoden ist nicht-trivial, viele theoretisch interessante Varianten sind verfügbar, welche Kombination von welchen „Tricks“ in der Praxis erfolgreich ist, ist theoretisch nicht entscheidbar. Hier weisen nur Rechenexperimente mit relevanten Praxisdaten den Weg. Testläufe mit zufällig erzeugten Daten sind wenig hilfreich.

Christian Blieck hat noch eine Zusatzbemerkung zum Vergleich zwischen dem CPLEX barrier solver und dem von CPLEX angebotenen Simplexverfahren angefügt:

> „On our lp testset barrier beats simplex when the solve times are above 10 sec. Break even is even a bit less than that. One of the advantages of barrier is that it can take advantage of parallelism, while simplex doesn't.“

Ähnliche Aussagen machen auch alle anderen führenden Anbieter von Optimierungssoftware. Dies gilt für das „reine Lösen“ von LPs. Wenn aber ein LP-Löser als Unterroutine in ein Verfahren zur Lösung von MIPs eingebunden ist, sieht die Sache anders aus. Dieser Aspekt wird in den nächsten beiden Kapiteln behandelt.

Literaturverzeichnis

D. Bertsimas and J. N. Tsitsiklis. *Introduction to Linear Optimization.* Athena Scientific, 1997.

C. Gonzaga. An algorithm for solving linear programming in $O(n^3L)$ operations. In N. Megiddo, editor, *Progress in Mathematical Programming: Interior Points and Related Methods*, pages 1–28. Springer Verlag, New York, 1989.

N. Karmarkar. A new polynomial time algorithm for linear programming. *Combinatorica*, 4(4):373–395, 1984.

M. Kojima, S. Mizuno, and A. Yoshise. A primal-dual interior point method for linear programming. In N. Megiddo, editor, *Progress in Mathematical Programming: Interior Points and Related Methods*, pages 29–47. Springer Verlag, New York, 1989.

S. Mehrotra. On the implementation of a primal-dual interior point method. *SIAM Journal on Optimization*, (2):575–601, 1992.

C. Roos, T. Terlaky, and J.-P. Vial. *Interior point methods for linear optimization.* Springer Verlag, 2 edition, 2006.

D. Shanno. Who invented the interior-point method? In M. Grötschel, editor, *Optimization Stories*, DOCUMENTA MATHEMATICA, Journal der Deutschen Mathematiker-Vereinigung, pages 55–64. Bielefeld, 2012. Online: `www.math.uni-bielefeld.de/documenta/vol-ismp/20_shanno-david.pdf`.

S. J. Wright. *Primal-Dual Interior-Point Methods.* SIAM, 1997.

6 Branch & Bound-Verfahren

Wir wollen uns nun mit ganzzahliger und gemischt-ganzzahliger Optimierung beschäftigen und werden eine Verfahrensklasse angeben, mit der man derartige Probleme exakt lösen kann. Sie basiert auf der Strategie, alle möglichen Lösungen auf geschickte Weise zu enumerieren und durch Benutzung von Primal- und Dualheuristiken den Rechenaufwand zu beschränken.

Die generelle Idee dieser sogenannten Branch & Bound-Verfahren ist ganz einfach. Wir betrachten ein Minimierungsproblem. Man benutze zunächst eine Primal- und eine Dualheuristik (Berechnung von Bounds), sind die Werte der gefundenen Lösung gleich, ist man fertig. Falls nicht, wird die Lösungsmenge partitioniert (Branching), und die Heuristiken werden auf die Teilmengen angewandt. Ist für eine (oder mehrere) dieser Teilmengen die für sie berechnete untere Schranke nicht kleiner als die beste bisher gefundene obere Schranke, braucht man die Lösungen in dieser Teilmenge nicht mehr zu betrachten (man erspart sich also die weitere Enumeration). Ist die untere Schranke kleiner als die beste gegenwärtige obere Schranke, muss man die Teilmenge weiter zerteilen. Man fährt so lange mit der Zerteilung fort, bis für alle Lösungsteilmengen die untere Schranke mindestens so groß ist wie die (global) beste obere Schranke.

Techniken dieser Art erscheinen in der Literatur unter einer Vielzahl von Namen wie z. B.:

- Branch & Bound-Verfahren,
- Verfahren der implizierten Enumeration,
- Divide-and-Conquer-Methoden,
- Backtracking-Methoden,
- Zerlegungs-Verfahren, etc.

Durch geeignete Darstellung kann man zeigen, dass jede dieser Techniken ein Spezialfall der übrigen ist, d. h. im Prinzip tun diese Verfahren alle das gleiche, sie unterscheiden sich nur bezüglich der jeweils angewendeten Strategie bzw. Philosophie. Manche Autoren machen sehr feine Unterschiede zwischen diesen Methoden, wir wollen jedoch alle Verfahren, die Enumerationstechniken benutzen, *Branch & Bound-Algorithmen* nennen.

Das Prinzip der Branch & Bound-Verfahren ist trivial, theoretisch sind sie nicht sonderlich interessant, aber es scheint so, dass man ohne sie einfach nicht auskommt. Bei den meisten in der Praxis erfolgreichen Methoden zur Lösung schwieriger kombinatorischer oder ganzzahliger Optimierungsprobleme wird irgendwann einmal Branch & Bound eingesetzt.

Noch eine Warnung! So simpel die Idee ist, so schwierig ist es i. A. Branch & Bound-Methoden effizient zu implementieren. Eine Hauptschwierigkeit besteht darin, „gute“ Zerlegungstechniken und einfache Datenstrukturen zu erfinden, die die effiziente Abarbeitung und Durchsuchung der einzelnen Teilmengen ermöglichen. Ganz allgemein kann man diesen Algorithmentyp formalisiert wie folgt darstellen.

(6.1) Algorithmus (Allgemeines Branch & Bound-Verfahren).

Eingabe: Eine Menge L von zulässigen Lösungen (implizit gegeben) und eine Zielfunktion $c : L \to \mathbb{K}$, die jedem $S \in L$ einen Wert $c(S)$ zuordnen.

Ausgabe: Lösung von

$$\max_{S \in L} c(S). \tag{P}$$

Annahme: Wir nehmen an, dass es ein „Universum“ $\overline{L}$ gibt mit $L \subseteq \overline{L}$, dass $c(S)$ wohldefiniert ist für alle $S \in \overline{L}$ und dass das relaxierte Problem

$$\max_{S \in \overline{L}} c(S) \tag{RP}$$

„einfach“ lösbar ist. Ferner soll die Lösung von (RP) in „nicht-trivialem“ Zusammenhang mit der Lösung von (P) stehen. Wir nehmen also an, dass wir P relaxieren können und (durch die Lösung von (RP)) eine effiziente Dualheuristik zur Verfügung steht.

1. Berechne eine untere Schranke U für (P). Man wende z. B. eine Primalheuristik zur Auffindung einer zulässigen Lösung an oder setze $U := -\infty$.

2. Setze $\mathcal{K} = \{L\}$ ($\mathcal{K}$ ist die Menge der Kandidatenmengen).

3. Falls $\mathcal{K} = \emptyset \longrightarrow$ STOP (die beste gegenwärtige Lösung ist eine Optimallösung von (P)).
   Andernfalls wähle eine Menge $M \in \mathcal{K}$. (Diesen Schritt nennt man *Branching* oder *Verzweigung*. Man muss sich hier genau überlegen, welche der noch nicht bearbeiteten Kandidatenmengen untersucht werden soll.)

4. **Bounding:** Wähle eine „geeignete“ Menge $\overline{M} \subseteq \overline{L}$ mit $M \subseteq \overline{M}$, so dass das relaxierte Problem

   $$\max_{S \in \overline{M}} c(S) \tag{RPM}$$

   „einfach“ zu lösen ist. (Das auf M eingeschränkte Ursprungsproblem wird also relaxiert. Wegen $M \subseteq \overline{M}$ bildet der Wert $c(S^*)$ der Optimallösung S^* von (RPM) eine obere Schranke (*Bound*) für den Wert der Optimallösung von

   $$\max_{S \in M} c(S) \tag{PM}$$

5. Ist $c(S^*) \leq U$, so hat keine Lösung aus $\overline{M}$ und somit auch keine aus M einen besseren Wert als die beste bereits bekannte Lösung von (P), d. h. die Lösungen aus M brauchen nicht weiter betrachtet zu werden. Entferne also M aus $\mathcal{K}$ und gehe zu 3.

6. Ist $S^* \in L$ und $c(S^*) > U$, so ist eine zulässige Lösung für (P) gefunden, deren Wert besser als U ist. Da die beste Lösung aus M gefunden ist, braucht M nicht mehr weiter untersucht zu werden. Entferne M aus $\mathcal{K}$, setze $U := c(S^*)$ und gehe zu 3.

7. **Separation** (Zerlegung): Es gilt also $c(S^*) > U$ und $S^* \notin L$. In diesem Falle war die Berechnung der unteren Schranke nutzlos, da wir keine verwertbare Aussage machen können. Wir zerlegen daher M in „geeignete“ kleinere Mengen $M_1, M_2, \ldots, M_k$, entfernen M aus $\mathcal{K}$, fügen $M_1, M_2, \ldots, M_k$ zu $\mathcal{K}$ hinzu und gehen zu 3. △

Falls M in den Schritten 5 oder 6 verworfen wird, so sagt man, dass M *ausgelotet* (fathomed) worden ist.

Ist man in Schritt 7 angelangt, so ist es häufig nützlich zu versuchen, aus S^* eine zulässige Lösung S von M zu konstruieren, um damit (durch eine Primalheuristik) die obere Schranke verbessern zu können.

Branch & Bound-Verfahren unterscheiden sich vor allem durch die Wahl verschiedener Relaxierungen $\overline{L}$, durch unterschiedliche Zerlegungsstrategien in 7. und durch die Auswahl der nächsten Kandidatenmenge in 3. Mit jedem Branch & Bound-Verfahren kann man einen sogenannten *Branch & Bound-Baum* assoziieren, der über die Verzweigungen Auskunft gibt. Ein derartiger Verzweigungsbaum ist in Abbildung 6.1 gezeigt.

Bei der Lösung des Teilproblems 1 wird eine obere Schranke mit Wert 20 gefunden und durch eine Heuristik eine zulässige Lösung von (P) mit Wert 7. Links vom jeweiligen Knoten ist die Reihenfolge vermerkt, in welcher die Teilprobleme gelöst wurden, rechts neben den Knoten die jeweils berechnete obere Schranke. Bei der Lösung des 6. Teilproblems wird eine zulässige Lösung mit Wert 10 gefunden. Das 15. Teilproblem liefert die Optimallösung.

Das Hauptproblem bei der Implementierung von Branch & Bound-Verfahren besteht darin, dass man sich alle Kandidatenmengen merken muss. Dies muss auf geschickte Weise geschehen, um nicht zu viel suchen zu müssen und nicht zu viel Speicherplatz zu vergeuden. Außerdem muss man dafür Sorge tragen, dass der Baum nicht zu groß wird und der Speicherplatz ausreicht. Man muss also in vielerlei Hinsicht Kompromisse schließen, um ein praktikables Verfahren zu entwerfen.

Das Branch & Bound-Verfahren ist so organisiert, dass in jedem Schritt folgendes gilt:

> Eine zulässige Lösung S gehört entweder zu einer bereits verworfenen Menge oder sie ist in genau einer der Kandidatenmengen enthalten.

Falls also die Menge der zulässigen Lösungen endlich ist und jedes Subproblem (RPM) in endlicher Zeit gelöst wird, bricht das Verfahren nach endlich vielen Schritten ab, wenn wir die (sinnvolle) Konvention treffen, dass einelementige Mengen M nicht relaxiert werden.

Wir wollen uns nun überlegen, wie wir die Branch & Bound-Technik bei ganzzahligen Programmierungsproblemen einsetzen können.

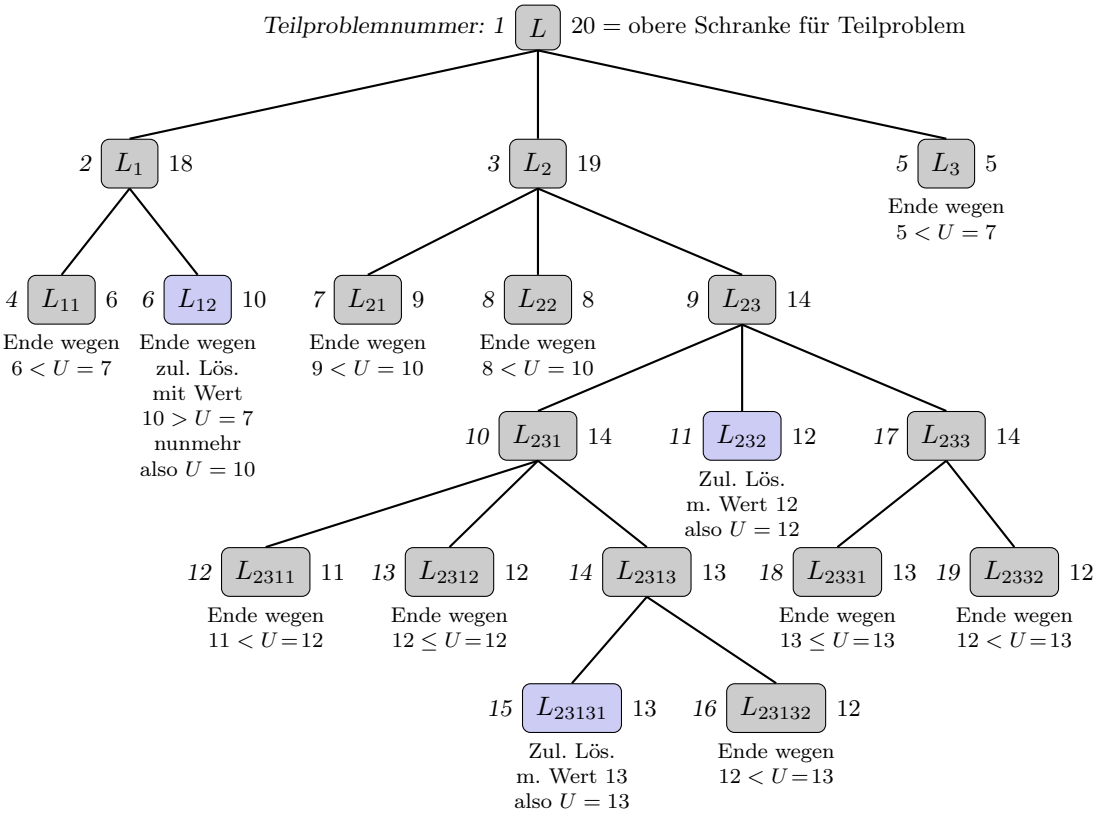


Abbildung 6.1: Ein Branch & Bound-Baum. Der Baum wird in der Reihenfolge der Problemnummern durchlaufen. Zu Beginn hat die untere Schranke U den Wert 7, in den farbigen Knoten wird U erhöht. Die Optimallösung (mit Wert 13) wird im Knoten L_{23131} gefunden.

(6.2) Algorithmus (Branch & Bound-Verfahren von Dakin für gemischt-ganzzahlige Programme).

Eingabe: Gemischt-ganzzahliges Programm mit rationalen Daten

$$\begin{aligned} \max\ & c^T x \\ & Ax = b \\ & x \geq 0 \\ & x_i \text{ ganzzahlig für alle } i \in N_1 \end{aligned} \tag{MIP$^=$}$$

$$P_0 := \{x \mid Ax = b, x \geq 0, x_i \text{ ganzzahlig für alle } i \in N_1\}$$

Ausgabe: Lösung von (MIP$^=$)

1. Setze:

untere Schranke	$U := -\infty$
Kandidatenmengen	$\mathcal{K} := \{P_0\}$
Zählindex	$k := 0$
gegenwärtig beste Lösung	$\overline{x}$ (am Anfang leer)

2. Falls $\mathcal{K} = \emptyset$, dann STOP.

   Ergebnis: Falls $U = -\infty$, so hat (MIP$^=$) keine Lösung.
   Falls $U > -\infty$, so ist U der Optimalwert und $\overline{x}$ eine Optimallösung.

3. **Branching:** Wähle eine Menge $P_j \in \mathcal{K}$.

4. **Bounding:** Löse das durch Weglassen der Ganzzahligkeitsbedingung in P_j entstehende lineare Programm
$$\max\ c^T x,\ x \in LP_j \qquad (\text{LMIP}_j^=)$$

5. Ist $LP_j = \emptyset$ oder (LMIP$_j^=$) unbeschränkt, so sei $c^* = -\infty$, andernfalls sei x^* die Optimallösung von (LMIP$_j^=$) und c^* der Optimalwert.

6. Ist $c^* \leq U$, so entferne P_j aus $\mathcal{K}$.

7. Ist $c^* > U$ und $x_i^* \in \mathbb{Z}\ \forall i \in N_1$, so setze $U := c^*$, $\overline{x} := x^*$ und entferne P_j aus $\mathcal{K}$.

8. **Separation:** Im Falle $c^* > U$ und $x_i^* \notin \mathbb{Z}$ für einige $i \in N_1$ wähle ein $i \in N_1$ mit $x_i \notin \mathbb{Z}$. Entferne P_j aus $\mathcal{K}$ und setze
$$\begin{aligned} P_{k+1} &:= P_j \cap \{x \mid x_i \leq \lfloor x_i^* \rfloor\}, \\ P_{k+2} &:= P_j \cap \{x \mid x_i \geq \lceil x_i^* \rceil\}, \\ k &:= k + 2 \end{aligned}$$
   und gehe zu 2. △

(6.3) Bemerkung. Die Daten des gemischt-ganzzahligen Programms (MIP$^=$) seien – wie in (6.2) verlangt – rational und (LMIP$^=$) sei das durch Weglassen der Ganzzahligkeitsbedingung in (MIP$^=$) entstehende lineare Programm.

(a) Ist (LMIP$^=$) unzulässig oder unbeschränkt, so bricht das Verfahren von Dakin ab und zeigt an, dass (MIP$^=$) keine Lösung oder keine endliche Optimallösung besitzt.

(b) Hat (LMIP$^=$) ein endliches Optimum, und ist $P_0 \neq \emptyset$, so findet das Verfahren von Dakin nach endlich vielen Schritten eine Optimallösung von (MIP$^=$), da alle ganzzahligen Werte der x_i, $i \in N_1$, durchsucht werden.

(c) Hat (LMIP$^=$) ein endliches Optimum, und ist $P_0 = \emptyset$, so kann (theoretisch) durch Einführung einer unteren Schranke für den Zielfunktionswert ein endlicher Abbruch erzwungen werden.

Folglich ist das Dakin-Verfahren ein endliches Verfahren zur Lösung gemischt-ganzzahliger Programme. △

(6.4) Bemerkung. Das ursprüngliche lineare Programm (LP-Relaxierung von (MIP$^=$)) wird bei den verschiedenen Verzweigungsschritten jeweils um sehr einfache Ungleichungen, nämlich um obere bzw. untere Schranken der Variablen erweitert. Für diesen Spezialfall kennen wir bereits eine Variante des Simplexalgorithmus (dualer Simplexalgorithmus mit oberen Schranken, (upper-bounding-technique), siehe Kapitel 3), die es erlaubt, die oberen Schranken durch Transformationen im Tableau implizit zu berücksichtigen, ohne die Ungleichungen explizit hinzuzufügen. △

(6.5) Beispiel. Wir betrachten das folgende rein-ganzzahlige Problem

$$\begin{aligned} \max\ & -7x_1 - 3x_2 - 4x_3 \\ & x_1 + 2x_2 + 3x_3 - x_4 = 8 \\ & 3x_1 + x_2 + x_3 - x_5 = 5 \\ & x_i \geq 0,\ i = 1, \ldots, 5 \\ & x_i \text{ ganzzahlig},\ i = 1, \ldots, 5 \end{aligned}$$

P_0 sei die Lösungsmenge dieses Programms. Eine optimale Lösung des zugehörigen linearen Programms ist gegeben durch $x_3 = x_4 = x_5 = 0$ und

$$x_1 = \frac{2}{5}, \qquad x_2 = \frac{19}{5}, \qquad c^* = -\frac{71}{5}.$$

Wir merken uns bei jedem Programm den Wert der Optimallösung und den gegenwärtigen Wert der unteren Schranke U. Da das Problem rein ganzzahlig ist, muss der Zielfunktionswert rein ganzzahlig sein, wir können also den Wert des LP-Optimums abrunden und erhalten damit eine untere Schranke für alle ganzzahligen Lösungen des gegenwärtig gelösten LP. Für P_0 erhalten wir

$$c^* = -15, \qquad U = -\infty$$

Zur Verzweigung wählen wir die Variable x_2 und fügen einerseits $x_2 \leq 3$ und andererseits $x_2 \geq 4$ zu den Nebenbedingungen von P_0 hinzu. Wir erhalten auf diese Weise neue Kandidatenmengen P_1, P_2. P_0 wird aus $\mathcal{K}$ entfernt. Wir merken uns, was wir getan haben bildlich in dem in Abbildung 6.2 gezeigten Branch & Bound-Baum.

Unsere Kandidatenmenge $\mathcal{K}$ enthält nunmehr die Mengen P_1, P_2. Wir entscheiden uns zur Lösung von $\max c^T x$, $x \in P_1$. Die Optimallösung der LP-Relaxierung dieses Problems ist die folgende: $x_4 = x_5 = 0$ und

$$x_1 = \frac{1}{2}, \qquad x_2 = 2, \qquad x_3 = \frac{1}{2}, \qquad c^* = -\frac{29}{2} \ \text{(Abrunden ergibt } c^* = -15).$$

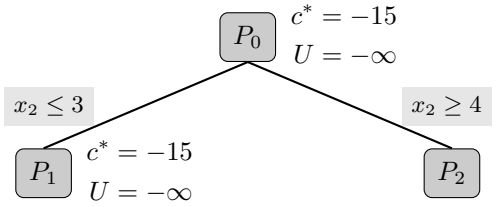


Abbildung 6.2: Erster Separationsschritt im Beispiel (6.5)

Wir tragen dieses Resultat in den Branch & Bound-Baum 6.2 ein. Da die obige Lösung nicht ganzzahlig ist, müssen wir P_1 weiter zerlegen. Wir wählen dazu die Variable x_1, wobei wir zu den Restriktionen von P_1 einerseits $x_1 \geq 1$, andererseits $x_1 \leq 0$ hinzufügen können. Wir entfernen P_1 aus $\mathcal{K}$ und fügen die neuen Kandidatenmengen P_3 und P_4 zu $\mathcal{K}$ hinzu, siehe Abildung 6.3.

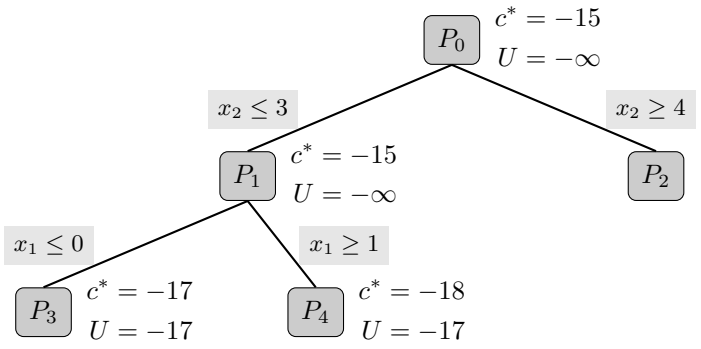


Abbildung 6.3: Zweiter Separationsschritt im Beispiel (6.5)

Die Kandidatenmenge enthält nun P_2, P_3 und P_4. Wir lösen P_3 und erhalten $x_1 = x_5 = 0$ und

$$x_2 = 3, \qquad x_3 = 2, \qquad x_4 = 4, \qquad c^* = -17.$$

Die optimale Lösung der LP-Relaxierung von P_3 ist also ganzzahlig. Wir haben die untere Schranke verbessert und setzen $U := -17$, außerdem ist die Menge P_3 vollständig erledigt, wir können sie aus $\mathcal{K}$ eliminieren.

Es verbleiben P_2 und P_4 in $\mathcal{K}$. Wir entscheiden uns zur Bearbeitung von P_4. Das Optimum der LP-Relaxierung von $\max c^T x$, $x \in P_4$ ist: $x_4 = 0$ und

$$x_1 = 1, \quad x_2 = 3, \quad x_3 = \frac{1}{3}, \quad x_5 = \frac{4}{3}, \quad c^* = -\frac{52}{3} \text{ (Abrunden ergibt } c^* = -18).$$

Also ist der Wert des LP-Optimums kleiner als der Wert der gegenwärtig besten ganzzahligen Lösung. P_4 ist damit auch vollständig erledigt. Die oben erzielten Resultate sind auch in Abbildung 6.3 eingetragen.

In der Kandidatenmenge ist nur noch P_2. Die Optimallösung der LP-Relaxierung von $\max c^T x$, $x \in P_2$ ist $x_3 = x_5 = 0$ und

$$x_1 = \frac{1}{3}, \qquad x_2 = 4, \qquad x_4 = \frac{1}{3}, \qquad c^* = -\frac{43}{3} \text{ (Abrunden ergibt } c^* = -15).$$

Die Lösung ist weder ganzzahlig noch unterschreitet der Optimalwert die gegenwärtig beste Schranke. Wir verzweigen bezüglich x_1 und fügen einerseits $x_1 \leq 0$ und andererseits $x_1 \geq 1$ zu P_2 hinzu. Wir erhalten neue Mengen P_5, P_6, die wir zu $\mathcal{K}$ addieren. P_2 wird aus $\mathcal{K}$ entfernt. Der gegenwärtige Branch & Bound-Baum ist in Abbildung 6.4 gezeigt.

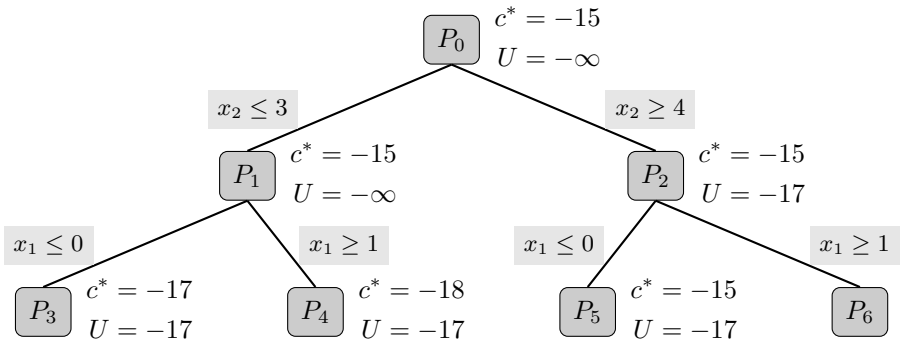


Abbildung 6.4: Dritter Separationsschritt im Beispiel (6.5)

Wir lösen die LP-Relaxierung von $\max c^T x$, $x \in P_5$ und erhalten

$$x_2 = 5, \qquad x_4 = 2, \qquad x_i = 0 \text{ sonst}, \qquad c^* = -15.$$

Diese Lösung ist ganzzahlig, und wir sind mit unserem Algorithmus fertig, da P_6 keine bessere ganzzahlige Lösung enthalten kann, denn eine ganzzahlige Lösung in P_2 hat höchstens den Wert -15. Damit ist eine optimale Lösung des Ausgangsproblems gefunden. △

Ich glaube, dass damit das Prinzip der Branch & Bound-Verfahren klar ist. Wichtig ist, eine Relaxierung zu wählen, die schnell lösbar ist und gute Schranken liefert, also eine gute duale Heuristik auszuwählen. Im Dakin-Verfahren haben wir einfach die kanonische LP-Relaxierung gewählt.

Ein Branch & Bound-Verfahren für das Travelling-Salesman-Problem

Der folgende Teil, in dem wir ein einfaches Beispiel eines Branch & Bound-Verfahrens für das asymmetrische TSP angeben, wird in ADM II nicht mehr behandelt, dafür aber im nächsten Semester in ADM III.

Der Großvater aller Branch & Bound-Algorithmen ist das Verfahren von Little, Murty, Sweeny & Karel zur Lösung asymmetrischer Travelling-Salesman-Probleme, das 1963 veröffentlicht wurde und in dem der Name Branch & Bound-Algorithmus geprägt wurde.

Weil dieses Verfahren so einfach ist, wollen wir es hier kurz besprechen, jedoch darauf hinweisen, dass es zur Lösung großer Probleme nicht besonders geeignet ist.

(6.6) Algorithmus (Verfahren von Little et al. für das asymmetrische Travelling-Salesman-Problem).

Eingabe: Asymmetrisches TSP auf n Städten gegeben durch Distanzmatrix $c = (c_{ij})$ mit $c_{ii} = \infty$, $i = 1, \ldots, n$ und $c_{ij} \geq 0$.

Ausgabe: Optimallösung des gegebenen TSPs.

Verfahrensweise: Zeilen- und Spaltenreduktion der Matrix C und sukzessive Erzeugung von Teilproblemen. Ein Teilproblem ist definiert durch

$$P(\textsc{Eins}, \textsc{Null}, I, J, C, L, U),$$

wobei

$$\begin{aligned} \textsc{Eins} &= \text{die Menge der auf 1 fixierten Bögen,} \\ \textsc{Null} &= \text{die Menge der auf 0 fixierten Bögen,} \\ C &= \text{die Distanzmatrix des Teilproblems,} \\ I &= \text{die Zeilenindizes von } C, \\ J &= \text{die Spaltenindizes von } C, \\ L &= \text{die untere Schranke für das Teilproblem,} \\ U &= \text{die obere Schranke für das Globalproblem.} \end{aligned}$$

1. Definiere das Anfangsproblem

   $$P(\emptyset, \emptyset, \{1, \ldots, n\}, \{1, \ldots, n\}, C, 0, +\infty),$$

   wobei C die Ausgangsmatrix ist, und setze dieses Problem auf die Problemliste.

2. Sind alle Teilprobleme gelöst $\longrightarrow$ STOP. Andernfalls wähle ein ungelöstes Teilproblem $P(\textsc{Eins}, \textsc{Null}, I, J, C, L, U)$ aus der Problemliste.
   Regel: Wähle das Problem mit kleinster unterer Schranke oder das Problem, bei dem Eins am meisten Elemente enthält. Bei ersterer Wahl erhält man hoffentlich eine optimale Tour, bei zweiterer hat man das Problem schnell abgearbeitet.

3. **Bounding:**

   a) **Zeilenreduktion:**
   FOR all $i \in I$ DO

   Berechne das Minimum der i-ten Zeile von C

   $$c_{ij_0} = \min_{j \in J} c_{ij}$$

   Setze $c_{ij} := c_{ij} - c_{ij_0}$ $\forall j \in J$ und $L := L + c_{ij_0}$.

   END

   b) **Spaltenreduktion:**
   FOR all $j \in J$ DO

Berechne das Minimum der j-ten Spalte von C

$$c_{i_0 j} = \min_{i \in I} c_{ij}$$

Setze $c_{ij} := c_{ij} - c_{i_0 j}\ \forall i \in I$ und $L := L + c_{i_0 j}$.

END

c) Ist $L \geq U$, so entferne das gegenwärtige Teilproblem aus der Problemliste und gehe zu 2.

d) Definiere den Nulldigraphen $G_0 = (V, A)$ mit

$$A := \textsc{Eins} \cup \{(i,j) \in I \times J \mid c_{ij} = 0\}.$$

e) Enthält G_0 keine Tour so gehe zu 4.

f) Enthält G_0 eine Tour, so hat die Tour die Länge L.

f_1) Entferne alle Teilprobleme aus der Problemliste, deren untere Schranke größer gleich L ist.

f_2) Setze in allen noch nicht gelösten Teilproblemen $U := L$.

f_3) Gehe zu 2.

(I. A. wird der Nulldigraph G_0 nicht berechnet, sondern so lange enumeriert bis nur noch $(2,2)$-Probleme übriggeblieben sind.)

4. **Branching:**

a) Wähle nach einem Plausibilitätskriterium einen Bogen (i,j), der 0 oder 1 gesetzt wird.
Z. B.: Definiere $u(i,j) := \min\{c_{ik} \mid k \in J \setminus \{j\}\} + \min\{c_{kj} \mid k \in I \setminus \{i\}\} - c_{ij}$, d. h. $u(i,j)$ sind die Zusatzkosten (Umwegkosten), die entstehen, wenn wir von i nach j gehen, ohne (i,j) zu benutzen.
Branching-Regel: Wähle $(i,j) \in I \times J$, so dass

$$\begin{aligned} c_{ij} &= 0 \qquad \text{und} \\ u(i,j) &= \max\{u(p,q) \mid c_{pq} = 0.\} \end{aligned}$$

(Wähle also einen Bogen, der zu möglichst hohen Umwegkosten führt.)

b) Definiere die neuen Teilprobleme

b_1) $P(\textsc{Eins} \cup \{(i,j)\}, \textsc{Null}, I \setminus \{i\}, J \setminus \{j\}, C', L, U\}$,
wobei C' aus C durch Streichen der Zeile i und der Spalte j entsteht und $c'_{ji} := \infty$ (zur Vermeidung des Kurzzyklus $\langle i,j \rangle$).

b_2) $P(\textsc{Eins}, \textsc{Null} \cup \{(i,j)\}, I, J, C'', L, U)$,
wobei C'' aus C entsteht durch $c_{ij} := \infty$.

Füge diese Teilprobleme zur Problemliste hinzu und gehe zu 2. △

Das Hauptproblem bei (6.6) ist – wie immer – die Buchhaltung der Teilprobleme. Die Anzahl der Touren in 4.b$_1$) ist kleiner als die in 4.b$_2$), die Wahl von (i, j) wird so getroffen, dass „hoffentlich“ in 4.b$_1$) eine optimale Tour ist.

(6.7) Beispiel. Wir beginnen mit $P = (\emptyset, \emptyset, \{1, \ldots, 6\}, \{1, \ldots, 6\}, C, 0, +\infty\}$

Anfangsmatrix						Zeilenmin							
						↓							
∞	5	1	2	1	6	1		∞	4	0	2	0	6
6	∞	6	3	7	2	2		4	∞	4	1	5	0
1	4	∞	1	2	5	1	$\longrightarrow$	0	3	∞	0	1	4
4	3	3	∞	5	4	3		1	0	0	∞	2	1
1	5	1	2	∞	5	1		0	4	0	1	∞	4
6	2	6	4	5	∞	2		4	0	4	2	3	∞
						10	Spaltenmin	0	0	0	0	0	0

Die Zeilenreduktion (Schritt 3.a) ergibt insgesamt einen Wert von 10, die Spaltenreduktion erbringt nichts, da nach der Zeilenreduktion bereits jede Spalte eine Null enthält. Die untere Schranke ist somit $L = 10$. Wir können nun den Nulldigraphen G_0 definieren. Er ist in Abbildung 6.5 dargestellt (alle Bögen mit dem Wert 0 sind eingezeichnet.). G_0

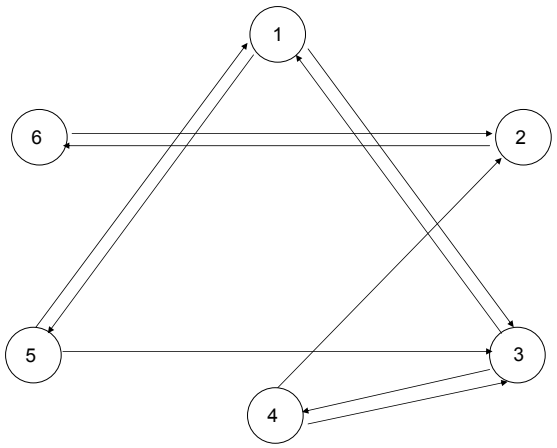


Abbildung 6.5: Nulldigraph G_0

enthält keine Tour.

Wir verwenden die in Schritt 4.a angegebene Branchingregel und wählen den Bogen $(i, j) = (2, 6)$ mit $u(2, 6) = 2$. Das Ausgangsproblem ist nun erledigt. Wir definieren zwei neue Teilprobleme.

Erstes neues Problem: $P = (\{(2,6)\}, \emptyset, \{1,3,4,5,6\}, \{1,2,3,4,5\}, C', 10, \infty)$

		1	*2*	*3*	*4*	*5*	Zeilenmin		*1*	*2*	*3*	*4*	*5*
	1	∞	4	0	2	0	0		∞	4	0	2	0
	3	0	3	∞	0	1	0		0	3	∞	0	1
C' :	*4*	1	0	0	∞	2	0	$\longrightarrow$	1	0	0	∞	2
	5	0	4	0	1	∞	0		0	4	0	1	∞
	6	4	∞	4	2	3	2		2	∞	2	0	1

Die untere Schranke (nach Zeilen- und Spaltenreduktion) für dieses Teilproblem ist $L = 12$.

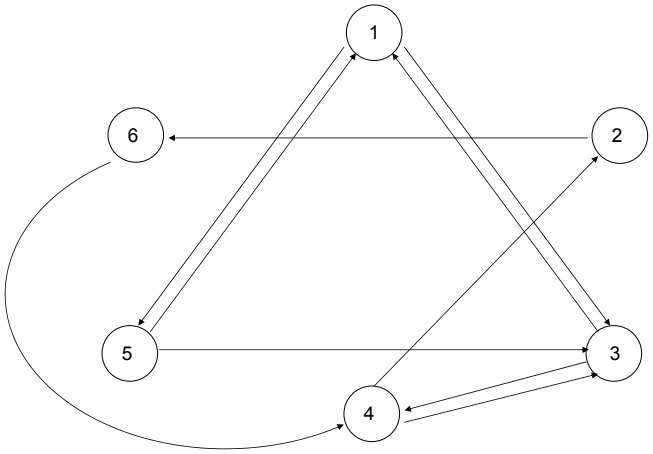


Abbildung 6.6: Nulldigraph ?

Zweites neues Problem: $P = (\emptyset, \{(2,6)\}, \{1,2,3,4,5,6\}, \{1,2,3,4,5,6\}, C'', 10, \infty)$

	∞	4	0	1	0	5	0		∞	4	0	1	0	4
	4	∞	4	1	5	∞	1		3	∞	3	0	4	∞
	0	3	∞	0	1	4	0		0	3	∞	0	1	3
C'' :	1	0	0	∞	2	1	0	$\longrightarrow$	1	0	0	∞	2	0
	0	4	0	1	∞	4	0		0	4	0	1	∞	3
	4	0	4	2	3	∞	0		4	0	4	2	3	∞
	0	0	0	0	0	1	$u = 12$							

△

Die untere Schranke für das zweite Problem ist ebenfalls 12. Das Verfahren wird auf diese Weise fortgeführt. Insgesamt erhält man den in Abbildung 6.7 dargestellten Branch & Bound-Baum.

Das Verfahren (6.6) ist sehr simpel und (daher) in der Praxis nicht sonderlich effizient. Wesentlich bessere Ergebnisse erzielt man bei Benutzung der Zuordnungsrelaxierung, die wir bisher noch nicht besprochen haben.

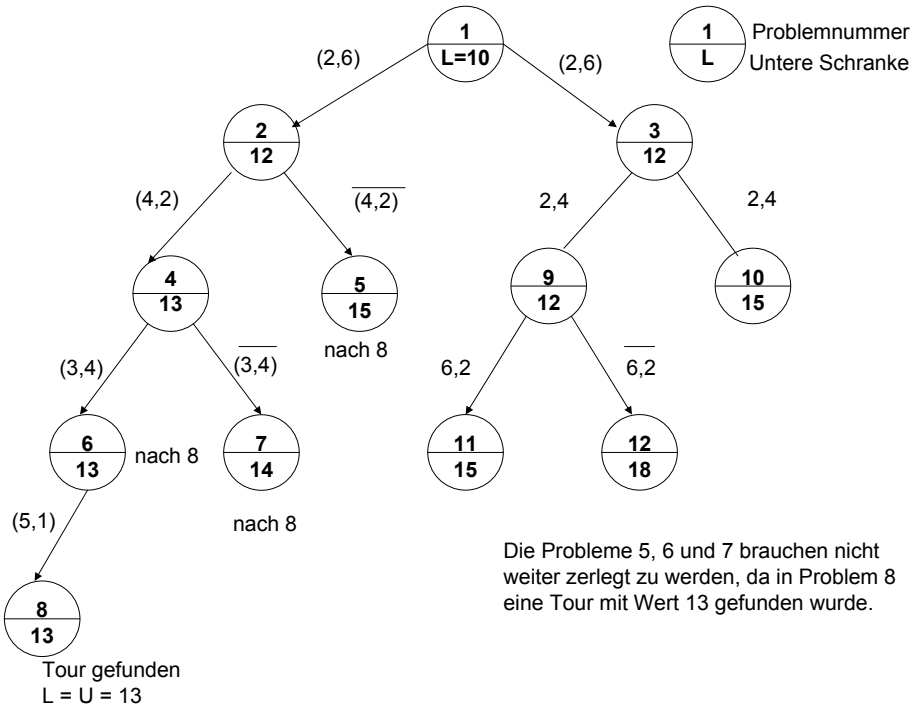


Abbildung 6.7: Branch & Bound-Baum zu Beispiel (6.7)