Einführung in die Mathematische Optimierung

Manuskript zur Vorlesung

SS 1996

Uwe Zimmermann
Abteilung Mathematische Optimierung
Technische Universität Braunschweig

Braunschweig, den 5. Oktober 1998

Inhaltsverzeichnis

Kapitel I

Lineare Optimierung

In der Linearen Optimierung sind die zulässigen Alternativen als Lösungen eines linearen Ungleichungssystems $Ax \leq b$ gegeben. Die Lösungsmenge ist daher ein Polyeder. Untersuchungen von Polyedern haben eine lange Geschichte in der Mathematik. Fragen der elementaren Polyedertheorie haben schon ägyptische und griechische Mathematiker beschäftigt. Ein typisches Resultat ist der im 18. Jahrhundert von Euler formulierte Zusammenhang der Anzahl der Ecken (v), Kanten (e) und Flächen (f) eines Polyeders, $v + f = e + 2$.

Heutzutage liegt die zentrale Aufgabe der Angewandten Mathematik in der

Entwicklung und Analyse konstruktiver Verfahren zur konkreten Lösung mathematisch modellierter Probleme.

In der Mathematischen Optimierung handelt es sich dabei um Optimierungsmodelle, mit deren Hilfe komplexe Probleme, z. B. aus den Naturwissenschaften, den Ingenieurwissenschaften oder den Wirtschaftswissenschaften und deren Anwendungsgebieten in mathematischer Sprache beschrieben werden. Ein solches Modell beschreibt eine Menge von zulässigen Lösungen mit unterschiedlichem Wert. Die Bestimmung der gesuchten optimalen Lösung erfordert wegen der Komplexität der Modelle den Einsatz von Computern. Daher müssen die konstruktiven Verfahren letztlich in einer dem Computer verständlichen Sprache formuliert werden.

Bei der Analyse der entwickelten Algorithmen können verschiedene Schwerpunkte gesetzt werden. Da reelle Zahlen auf Computern nur näherungsweise darstellbar sind, untersucht man in der Numerik sehr genau die auftretenden Rundungsfehler. Dabei unterscheidet man die Fehler, die allein aufgrund der Formulierung des Problemes unvermeidbar in Lösungen auftreten, sorgsam von den Fehlern, die bei Ablauf des jeweiligen Algorithmus entstehen. Die letzteren kann man ja durch Wahl und Formulierung des Algorithmus beeinflussen.

In der Optimierung steht häufig die Komplexität der behandelten Probleme im Vordergrund. Das liegt teilweise daran, daß in vielen Optimierungsproblemen keine Rundungsfehler auftreten, z.B. bei diskreten Problemen, teilweise daran, daß die Probleme äußerst schwierig sind, so daß man zunächst einmal daran interessiert ist, überhaupt eine Lösung zu finden. Natürlich muß letzlich auch der Einfluß von Rundungsfehlern untersucht werden, wenn man über die Qualität der Lösungen ein Urteil fällen will. Das gilt insbesondere bei den im dritten Kapitel behandelten nichtlinearen Optimierungsproblemen, deren Untersuchung sich daher auch in vielen Büchern über Numerik findet. Auch bei der Diskussion der Komplexität muß deutlich zwischen der Komplexität des Problems und der Komplexität des untersuchten Algorithmus zur Lösung des Problems unterschieden werden.

Im Gegensatz zur Nichtlinearen Optimierung lassen sich optimale Lösungen durch notwendige und hinreichende Bedingungen charakterisieren. Solche Bedingungen wurden bereits durch Cournot (1827) für die Gleichgewichtslagen von Massepunkten beschrieben (cf. Abbildung I.1).

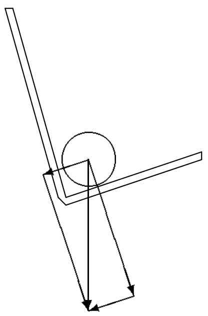


Abbildung I.1: Kugel im Gleichgewicht: die Schwerkraft ist positive Summe der Gradienten aktiver Restriktionen

Die strenge mathematische Formulierung dieses Sachverhalts ist das Lemma von Farkas (1894):

$$(Ay \leq 0 \Rightarrow b^T y \leq 0 \quad \forall\, y) \Leftrightarrow (x^T A = b^T \quad \exists\, x \geq 0).$$

Bereits 1939 beschrieb Kantorovich Produktionsprozesse mit Hilfe linearer Optimierungsaufgaben. Das Simplexverfahren zur Lösung solcher Probleme fand Dantzig 1947. Obwohl dieses Verfahren in der Praxis sehr erfolgreich angewendet wird, konnten Klee und Minty (1971) Beispiele finden, für die das Verfahren exponentiell viele Operationen erfordert. Die sich daran anschließende Frage, ob es überhaupt ein im theoretisch strengen Sinn effizientes Verfahren zur Lösung linearer Optimierungsprobleme gibt, konnte erst 1979 in einer Arbeit von Khachian positiv beantwortet werden. Allerdings gelang es bis heute nicht, aus der dort angegebenen Ellipsoidmethode ein praktikables Verfahren zu entwickeln. 1984 beschrieb Karmarkar ein weiteres theoretisch effizientes Verfahren, eine innere Punkte Methode, deren Weiterentwicklung zu Verfahren geführt hat, die das Simplexverfahren bei der Lösung speziell strukturierter Aufgaben übertreffen. Im hier gesetzten Rahmen werden wir uns nur mit dem Simplexverfahren beschäftigen.

1 Lineare Modelle

Mit Hilfe linearer Restriktionen und Zielfunktionen lassen sich viele Probleme aus den Anwendungen in den Wirtschaftswissenschaften, den Ingenieurwissenschaften und der Informatik modellieren. Wir betrachten zunächst einige wenige Beispiele.

Beispiele

Beim Mischungsproblem stehen Legierungen $L_1, L_2, \ldots, L_n$ mit unterschiedlichem Bleigehalt $a_1, a_2, \ldots, a_n$ (in Prozent) und unterschiedlichen Preisen $c_1, c_2, \ldots, c_n$ zur Verfügung. Daraus soll eine Legierung mit b% Bleigehalt und minimalen Kosten gemischt werden. Sei x_j der unbekannte Anteil von L_j an L. Dann kann man die gestellte Aufgabe in folgender Form linear modellieren:

$$\min\{\sum c_j x_j \mid a^T x = b, 1^T x = 1, x \geq 0\}.$$

Stiegler modellierte 1945 ausgewogene Ernährungspläne. Ein optimaler Ernährungsplan sei aus den unbekannten Mengen $x_1, x_2, \ldots, x_n \geq 0$ der verfügbaren Nahrungsmittel $1, 2, \ldots, n$ zusammengesetzt. Die entstehenden zu minimierenden Kosten sind dann

$$c_1 x_1 + \ldots + c_n x_n.$$

In einem ausgewogenen Ernährungsplan sollen gewisse Anteile an Grundbestandteilen enthalten sein. Im Nahrungsmittel j sind a_{ij} Einheiten (z.B. Gramm) des gewünschten i-ten Bestandteils (Vitamine, Kohlehydrate) vorhanden. Der Bedarf an Grundbestandteilen kann dann linear durch

$$\sum a_{ij} x_j \geq b_i \quad \forall\, i, \qquad x \geq 0$$

modelliert werden.

Das Modell hat also in Matrix- und Vektorschreibweise die folgende Form:

$$\begin{array}{lrcl} \min & c_x^T & & \\ \text{unter} & Ax & \geq & b \\ & x & \geq & 0 \end{array}$$

Kantorovich diskutiert 1939 Produktionsmodelle. $0 \leq x_j$ bezeichne die Menge des j-ten Produkts, $j = 1, \ldots, n$, die in einem Betrieb erzeugt wird. Rohstoffbedarf, Zeitbedarf, etc. der einzelnen Produkte sei bekannt, etwa a_{ij} pro erzeugte Einheit. Bezeichnet b_i die vorhandene Menge der i-ten Resource, so muß die Restriktion

$$\sum a_{ij} x_j \leq b_i$$

berücksichtigt werden. Der bei Verkauf von $x \geq 0$ zu erzielende Gewinn soll durch die Produktionsplanung maximiert werden, d.h.

$$\max \sum c_j x_j.$$

Das resultierende Modell läßt sich in kompakter Form angeben:

Die Kanonische Form des LP

Maximiere $c^T x$ unter $Ax \leq b, x \geq 0$.

Umformungen auf die kanonische Form

Andere mögliche Formen von linearen Restriktionen oder Zielfunkionen lassen sich auf die kanonische Form transformieren.

1. Minimierung von $c^T x$ ist äquivalent zur Maximierung von $(-c)^T x$.
2. Eine Ungleichung der Form $a^T x \geq b$ kann durch $(-a)^T x \leq (-b)$ ersetzt werden.
3. Eine Gleichung $a^T x = b$ kann durch zwei Ungleichungen $a^T x \leq b, a^T x \geq b$ und daher durch Ungleichungen des kanonischen Typs ersetzt werden.
4. Nichtvorzeichenbeschränkte Variable x_j lassen sich durch Paare von beschränkten Variablen modellieren: $x_j = x_j' - x_j''$ mit $x_j', x_j'' \geq 0$.

Daher kann jedes LP in kanonische Form $\max\{c^T x \mid Ax \leq b, x \geq 0\}$ gebracht werden, wobei $c \in \boldsymbol{R}^n, b \in \boldsymbol{R}^m$ und A eine $m \times n$-Matrix ist.

Separable, stückweise lineare konvexe Funktionen

Separable Funktionen sind von der Form $F(x) = \sum_{j=1}^n f_j(x_j)$. Die Komponentenfunktionen f_j seien konvex und stückweise linear, d.h. es gibt eine endliche Intervalleinteilung $0 \leq \alpha_1 \leq \alpha_2 \leq \ldots$ des Definitionsbereichs von f_j und auf jedem Intervall ist die Komponentenfunktion linear (cf. Abbildung I.2). Die Konvexität impliziert, daß die Koeffizienten der linearen Funktionen von Intervall zu Intervall streng anwachsen.

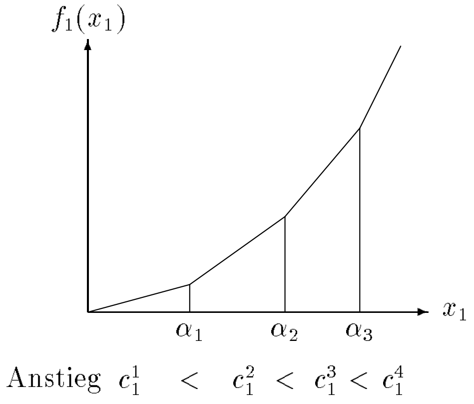


Abbildung I.2: Separable, stückweise lineare, konvexe Funktionen

Wir stellen die jeweilige Variable als Summe von Variablen auf den zugehörigen Intervallen dar. In Abbildung I.2 benutzen wir also vier Variable y_1, y_2, y_3, y_4 und setzen $y_1 + y_2 + y_3 + y_4 = x_1$, wobei $0 \leq y_1 \leq \alpha_1, 0 \leq y_2 \leq \alpha_2 - \alpha_1, 0 \leq y_3 \leq \alpha_3 - \alpha_2, 0 \leq y_4$. Falls die Bedingung

$$(**) \quad y_k > 0 \Rightarrow y_\mu \text{ maximal} \quad \forall\, \mu < k$$

gilt, so sind die zu einem Wert von x_1 gehörenden y-Werte eindeutig festgelegt. In einer optimalen Lösung der mit y-Variablen reformulierten Aufgabe $\min\{f(x) \mid x \in P\}$ ist die Bedingung (**) aufgrund der Konvexität stets erfüllt.

Nichtnegativ gewichtete Summen absoluter Beträge

Auch zunächst nichtlinear erscheinende Aufgaben lassen sich in besonderen Fällen auf lineare Optimierungsaufgaben zurückführen, z.B.

$$\min\{\sum c_j|x_j| \mid Ax = b\}$$

für $c \geq 0$. Wir stellen jede Variable als Differenz zweier beschränkter Variablen dar. Sei $x_j = x_j^+ - x_j^-$, $x_j^+, x_j^- \geq 0$. Zusätzlich soll die nichtlineare Komplementaritätsbedingung $x_j^+ \cdot x_j^- = 0$ erfüllt werden. Dann gilt

$$x_j^+ = \begin{cases} 0 & x_j \leq 0 \\ x_j & x_j > 0 \end{cases}$$

$$x_j^- = \begin{cases} 0 & x_j \geq 0 \\ -x_j & x_j < 0 \end{cases}$$

Wegen $|x_j| = x_j^+ + x_j^-$ lautet die äquivalente neue Aufgabe

$$\min\{\sum c_j(x_j^+ + x_j^-) \mid Ax^+ - Ax^- = b, \quad x^+, x^- \geq 0, \quad x^{+T}x^- = 0\}.$$

Minimierung ohne Beachtung der Komplementaritätsbedingung führt auf eine Lösung $\hat{x}^\pm$. Dann ist wegen $c \geq 0$ die durch

$$\tilde{x}_j^\pm := \hat{x}_j^\pm - \min(\hat{x}_j^+, \hat{x}_j^-)$$

definierte Lösung ebenfalls zulässig, optimal und außerdem komplementär. Falls $c > 0$, ist bereits $\hat{x}^\pm$ komplementär.

Geometrische Darstellung in zwei Variablen

In zwei Variablen lautet die Aufgabe:

$$\begin{array}{lll} \max & c_1x_1 + c_2x_2 & \\ \text{unter} & a_{i1}x_1 + a_{i2}x_2 & \leq \; b_i, \quad \forall\, i \in \{1, 2, \ldots, m\}, \\ & x_1 \geq 0, x_2 \geq 0. & \end{array}$$

Die Menge der Punkte, die die Ungleichungen erfüllen, wird mit P bezeichnet.

Sei $(\alpha_1, \alpha_2) \neq 0$. Dann heißt $H_\leq := \{x \mid \alpha_1x_1 + \alpha_2x_2 \leq \beta\}$ *abgeschlossene Halbebene.* Analog ist $H_\geq$ definiert. Der Schnitt der beiden möglichen Halbräume ist die Gerade $H_=$. Halbebenen und ihre Schnitte sind konvex.

Schnitt von Halbebenen, Polyeder

Bereits beim Schnitt zweier Halbebenen treten verschiedene Fälle auf. Restriktionen können widersprüchlich oder überflüssig sein. Anderenfalls entsteht ein Kegel.

Beim Schnitt endlich vieler Halbräume entstehen Polyeder, die man im beschränkten Fall auch als Polytope bezeichnet.

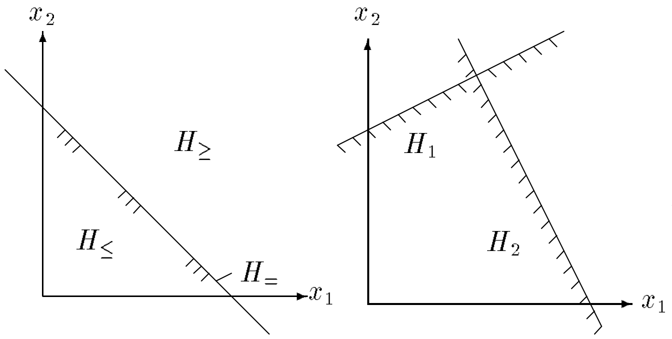


Abbildung I.3: Halbebene, Kegel, beschränktes Polyeder

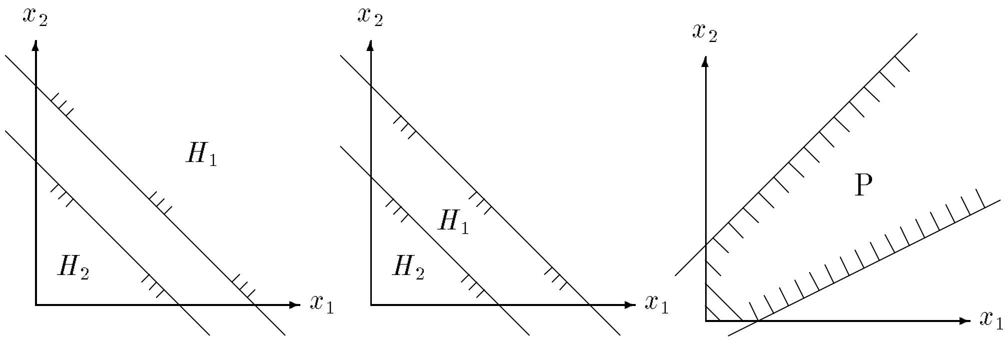


Abbildung I.4: Inkonsistente, redundante Restriktionen und unbeschränktes Polyeder

Zielfunktion

Die Zielfunktion kann man sich als Schar von Geraden

$$\{x \mid c_1x_1 + c_2x_2 = z\}$$

für $z \in \boldsymbol{R}$ veranschaulichen.

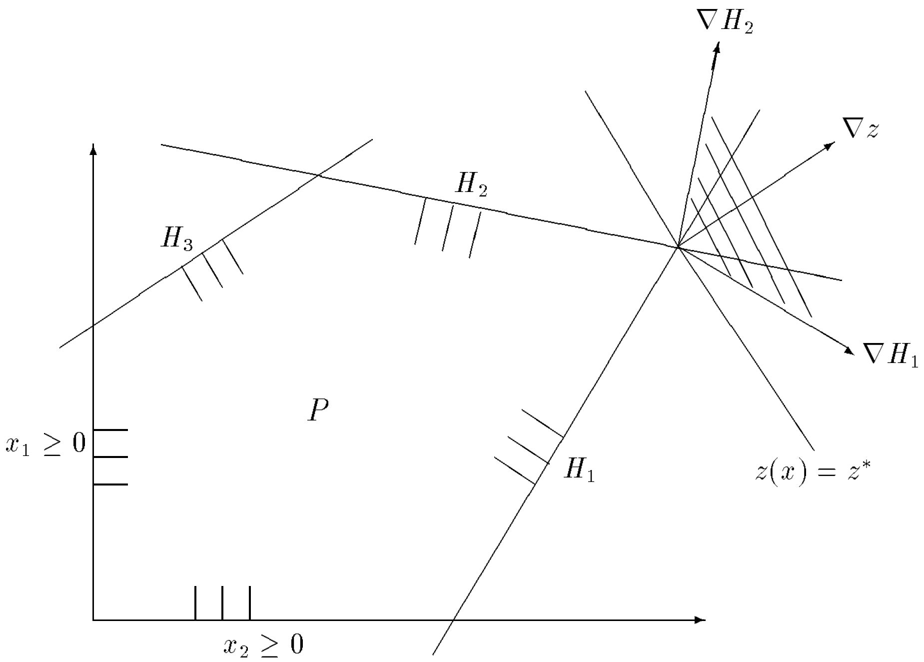


Abbildung I.5: Graphische Lösung in zwei Variablen

Es gilt, die Gerade mit maximalem z zu finden, die mit P nichtleeren Durchschnitt hat. Ist die Optimallösung eindeutig, so berührt diese Gerade das Polyeder in genau einem Punkt, einer Ecke des Polyeders. Dieser Punkt ist als Schnittpunkt der aktiven Restriktionen festgelegt. Der Gradient der Zielfunktion läßt sich als nichtnegative d.h. $\nabla z = \lambda_1 \nabla H_1 + \lambda_2 \nabla H_2$ mit $\lambda_1, \lambda_2 \geq 0$, oder in ausführlicher Form:

$$c_1 = \lambda_1 a_{11} + \lambda_2 a_{21},$$

$$c_2 = \lambda_1 a_{12} + \lambda_2 a_{22}.$$

Die Optimallösung kann man mit Hilfe der aktiven Restriktionen H_1, H_2 berechnen:

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} x_* = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

2 Polyeder und Lineare Programme

Polyeder und lineare Programme lassen sich sowohl über den reellen als auch den rationalen Zahlen weitgehend in gleicher Form untersuchen. Dementsprechend werden wir den reellen und den rationalen Fall gleichzeitig behandeln. Dazu sei $K \in \{\boldsymbol{R}, \boldsymbol{Q}\}$.

Polyeder

Sei A eine $m \times n$-Matrix und b ein m-Vektor über K. Dann heißt

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

rationales bzw. reelles *Polyeder*.

Polyeder sind konvex, denn aus $x, y \in P(A,b)$ folgt $[x,y] \subseteq P(A,b)$, wobei wir definieren: $[x,y] := \{\lambda x + (1-\lambda)y \mid \lambda \in [0,1], \lambda \in K\}$.

Seiten von Polyedern

Zu $I \subseteq \{1,\ldots,m\}$ sei $H_I := \{x \mid a_i^T x = b_i \quad \forall\, i \in I\}$. $H_I = K^n$, falls $I = \emptyset$. Falls $\emptyset \neq S := H_I \cap P$, so heißt S *Seite* von P.

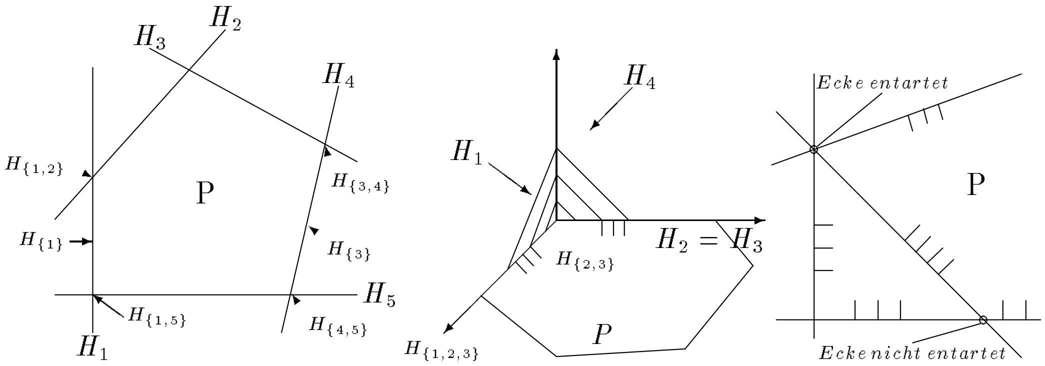


Abbildung I.6: Seiten von Polyedern, insbesondere Ecken

Zu $S \neq \emptyset$ sei $I(S) := \{i \mid S \subseteq H_i\}$. Die *Dimension* einer Seite S ist gegeben durch $\dim S := \dim H_{I(S)}$.

Enthält eine Seite keine andere Seite echt, so nennt man sie *minimale* Seite. Es gilt:

1. S Seite $\Leftrightarrow S = H_{I(S)} \cap P$,

2. S minimale Seite $\Leftrightarrow S = H_{I(S)}$.

Ecken

Ecken sind Seiten der Dimension 0. Facetten sind Seiten der Dimension $\dim P - 1$. Ecken S heißen *entartet*, falls $|I(S)| > n$. Polyeder mit Ecken heißen *spitz*. Ecken sind als minimale Seiten allein durch den Schnitt der sie tragenden Hyperebenen festgelegt.

(Halb-) Geraden

Geraden in Polyedern und Ecken von Polyedern sind eng miteinander verknüpft. Analog zu den Intervallen bezeichnen wir für Vektoren $u \neq v$ Geraden und Halbgeraden mit

$$\begin{aligned} < u,v > &:= \{u + \lambda(v-u) \mid \lambda \in K\}, \\ [u,v > &:= \{u + \lambda(v-u) \mid \lambda \geq 0, \lambda \in K\} \end{aligned}$$

Lemma 2.1 *Sei $x \in P(A,b)$, G Gerade mit $x \in G \subseteq H_{I(x)}$. $S := H_{I(x)} \cap P$. Dann gibt es $c, d \in S \cap G$ mit $x \in (c,d)$.*

Beweis.

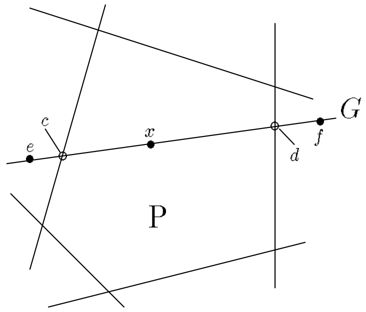


Abbildung I.7: Zu Lemma 2.1

OBdA $G = < e, f >$ mit $x \in (e,f)$. OBdA $f \notin P$, also $a_i^T f > b_i$ und $a_i^T x < b_i$ für $i \notin I(x)$. Daher gibt es einen Schnittpunkt $d_i = G \cap H_i$. Wähle i, so daß (x, d_i) keinen dieser Schnittpunkte enthält. Dann gilt $d_i \in P$, so daß wir $d = d_i$ wählen können. Analog kann man c finden. □

Satz 2.2 (Spitze Polyeder)

1. *Ein nichtleeres Polyeder ist spitz genau dann, wenn es keine Gerade enthält.*
2. *Jede Seite eines spitzen Polyeders enthält eine Ecke.*
3. *Ein Polyeder hat höchstens endlich viele Ecken.*

Beweis.

1. P enthalte keine Gerade. Wähle $x_0 \in P$. Falls x_0 Ecke, ist nichts weiter zu zeigen. Anderenfalls gilt $\dim S \geq 1$ für $S := H_{I(x_0)} \cap P$. Wähle eine Gerade $G = < c, d > \subseteq H_{I(x_0)}$ mit $c, d \in S$. Da S keine Gerade enthält, sei etwa $[x_0, d > \not\subseteq S$. Wir können d wie im Lemma 2.1 als Schnittpunkt von G mit einer der Randhyperebenen H_i, $i \notin I(x_0)$, wählen. Insbesondere gilt dann $x_0 \notin H_i$. Es folgt:

$$|I(d)| > |I(x_0)|, \quad \dim H_{I(d)} < \dim H_{I(x_0)}.$$

   Ist d wiederum keine Ecke, so ersetzen wir x_0 durch d und wiederholen die Konstruktion. Da die Dimension von $H_{I(x_0)}$ in jeder Iteration echt fällt, konstruieren wir auf diese Weise in höchstens n Schritten eine Ecke von P.

   Sei P spitz, $\bar{x}$ Ecke. Wähle $I \subseteq I(\bar{x})$ mit $|I| = n$ und $\mathrm{rg} A_I = n$. Aus der Annahme $< u, v > \subseteq P$, $u \neq v$ folgt $A_I u + \lambda(A_I v - A_I u) \leq b_I$ für alle $\lambda \in K$. Da λ nicht beschränkt ist, muß dann $A_I u = A_I v$ und damit $u = v$ gelten. Widerspruch!

2. Eine Seite S eines spitzen Polyeders P kann wegen $S \subseteq P$ nach 1. keine Gerade enthalten. Da auch S ein Polyeder ist, enthält es nach 1. eine Ecke.

3. Da $\{1, 2, \ldots, m\}$ nur endlich viele Teilmengen enthält, gibt es nur endlich viele Seiten, insbesondere höchstens $\binom{m}{n}$ Ecken. □

Insbesondere Polyeder der Form $P = \{x \mid Ax \leq b, x \geq 0\}$ sind spitz. Eine Funktion $f\colon K^n \to K$ heißt *(affin) linear*, wenn $f(x) = \alpha + c^T x$ mit $\alpha \in K, c \in K^n$. Wir werden im weiteren das Prefix „ affin“ weglassen, auch wenn $\alpha \neq 0$.

Lemma 2.3 *Seien f linear, $a, b \in K^n, a \neq b$. Dann gilt:*

1. *f nimmt auf $[a, b]$ Maximum und Minimum in $\{a, b\}$ an.*

2. *f nimmt auf $[a, b >$ Maximum oder Minimum in a an.*

Beweis. Mit $x = a + \lambda(b - a)$, $0 \leq \lambda \leq 1$ bzw. $0 \leq \lambda$, gilt $f(x) = \alpha + c^T a + \lambda c^T(b - a) =: \beta + \lambda\gamma$. Die Fallunterscheidung: $\gamma = 0$, $\gamma < 0$ oder $\gamma > 0$ liefert die Behauptung. □

Satz 2.4 *Nimmt eine affin lineare Funktion f auf einem Polyeder P ihr Maximum (Minimum) in $\bar{x} \in P$ an, so in allen Punkten der Seite $S = H_{I(\bar{x})} \cap P$. Also auch in einer Ecke, falls P spitz.*

Beweis. OBdA sei $\bar{x}$ Maximum, $\bar{x} \neq a \in S$. Dann ist $G =< a, \bar{x} >$ eine Gerade in $H_{I(\bar{x})}$. Nach Lemma 2.1 wähle $c, d \in S \cap G$ mit $\bar{x} \in (c, d), [c, d] \subseteq S$. OBdA liege in c das Maximum von f auf $[c, d]$. Da $\bar{x} \in G$, gilt $f(c) \geq f(\bar{x})$. Da $c \in P$, gilt $f(c) \leq f(\bar{x})$. Also, $f(c) = f(\bar{x})$, d.h. f ist konstant auf $< c, d >$. Insbesondere $f(a) = f(\bar{x})$. Da $a \in S$ beliebig gewählt war, ist f konstant auf S. □

Bemerkung (Spitze Polyeder): Für Polyeder mit Ecken kann man daher die lineare Optimierungsaufgabe auf eine Optimierungsaufgabe mit nur endlich vielen zulässigen Punkten zurückführen:

1. Besitzt $\max\{c^T x \mid x \in P\}$ eine endliche Optimallösung $\bar{x}$, so gilt $c^T\bar{x} = \max\{c^T x \mid x \in V\}$, wobei V die Menge der Ecken von P bezeichnet.

2. Wir werden später nachweisen, daß anderenfalls entweder $P = \emptyset$ oder $c^T x$ auf P unbeschränkt nach oben sein muß.

3 Simplexverfahren und Revidiertes Simplexverfahren

Für lineare Optimierungsprobleme in kanonischer Form $\max\{c^T x \mid x \in P\}$ liegt die Menge der zulässigen Punkte $P := \{x \mid Ax \leq b, x \geq 0\}$ im K^n. Durch Einführung von *Schlupfvariablen* $y := b - Ax \in K^m$, die also die Lücke in den Ungleichungen füllen, betten wir den K^n affin-linear als $\{\left[\begin{smallmatrix} x \\ y \end{smallmatrix}\right] \mid Ax + y = b\}$ in den K^{n+m} ein. Das bijektive Bild von P ist $\hat{P} := \{\left[\begin{smallmatrix} x \\ y \end{smallmatrix}\right] \mid Ax + y = b, y \geq 0, x \geq 0\}$. Die zugehörigen Abbildungen sind

$$p^{-1}: K^n \to K^{n+m}, x \mapsto \begin{bmatrix} x \\ b - Ax \end{bmatrix}, \quad p: K^{n+m} \to K^n, \begin{bmatrix} x \\ y \end{bmatrix} \mapsto x.$$

Die Einbettung erhält die Dimension der Seiten des Polyeders, insbesondere gehen Ecken in Ecken über.

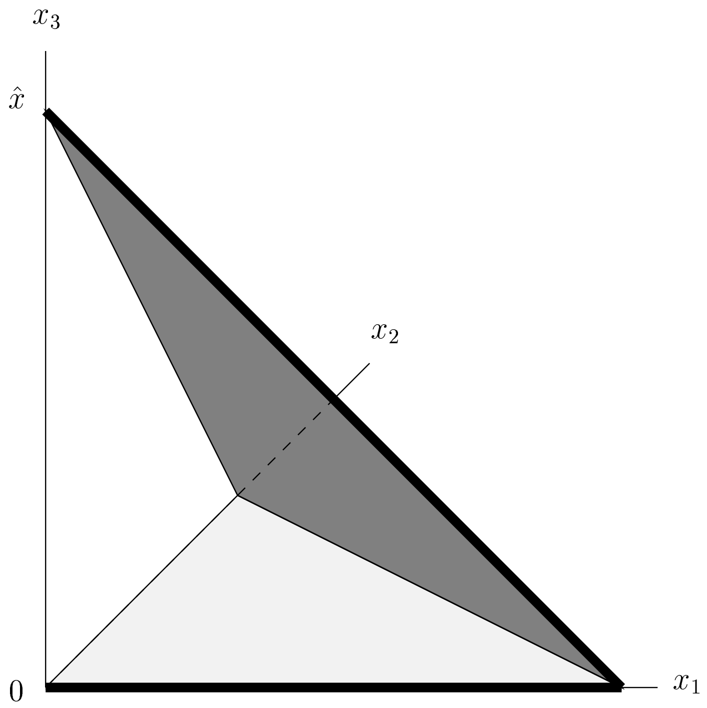


Abbildung I.8: Polyeder und Schlupfvariablen

Beispiel: Durch die Ungleichungen

$$x_1 + x_2 \leq 1, \quad x_1, x_2 \geq 0$$

ist ein Polyeder im K^2 in kanonischer Form beschrieben. Die resultierende Beschreibung ist

$$x_1 + x_2 + x_3 = 1, \quad x \geq 0.$$

Die Ecke $0 \in K^2$ wird auf die Ecke $\left[\begin{smallmatrix} 0 \\ 1 \end{smallmatrix}\right] \in K^3$ abgebildet, die Seite $\{x \in P \mid x_2 = 0\}$ in die Seite $\{\left[\begin{smallmatrix} x \\ y \end{smallmatrix}\right] \in \hat{P} \mid x_2 = 0\}$.

Der Teil der erweiterten Koeffizientenmatrix, der zu den Schlupfvariablen gehört, ist eine $m \times m$ Einheitsmatrix E. Die Spalten von E sind gerade die Einheitsvektoren E_j

des K^m, d.h. $E := [E_1|\cdots|E_m]$. Analoges gilt für die Zeilen e_i, $i = 1, \ldots, m$. Wir werden diese Bezeichnungsweise für Einheitsmatrizen und Einheitsvektoren jeder Dimension verwenden, wobei sich die Dimension jeweils aus dem Zusammenhang ergibt. Durch $y^T =: (x_{n+1}, \ldots, x_{n+m})$ erhalten wir eine einheitliche Bezeichnung der Variablen.

Die resultierende *Standardform* des linearen Optimierungsproblems lautet

$$\max\{c^T x \mid Ax = b, x \geq 0\}.$$

wobei $c_j := 0$ für $j > n$, $A := [A_{alt} \mid E]$. Es gilt für die erweiterte Koeffizientenmatrix: $\text{Rang}(A) = m < n + m$ (= Anzahl der Spalten). Im folgenden betrachten wir lineare Optimierungsprobleme, die in Standardform gegeben sind, und setzen voraus, daß die $m \times (n+m)$ Koeffizientenmatrix maximalen Rang, also m, besitzt. P bezeichnet im folgenden entsprechend das Polyeder des Standardproblems im K^{n+m}.

Basen

Eine *Basis* B ist ein m-Vektor von Spaltenindizes zu A mit linear unabhängiger Spaltenmenge $\{A_{B(i)} \mid i \in \{1, 2, \ldots, m\}\}$. Gibt es ein i mit $j = B(i)$ schreiben wir kurz $j \in B$. Eine Variable x_j mit $j \in B$ heißt *Basisvariable*, die übrigen Variablen heißen *Nichtbasisvariablen*. Die Indizes der Nichtbasisvariablen werden in einem n-Vektor N, der *Nichtbasis*, zusammengefaßt. Variablen, Koeffizienten und Matrizen können wir nun in Blockschreibweise notieren:

$$x_B, x_N, \quad c_B, c_N, \quad A_B, A_N$$

Ein Vektor $x \in K^{n+m}$ heißt *Basislösung* des LP, wenn es eine Basis B mit

$$A_B x_B = b, \quad x_N = 0$$

gibt. Eine Basislösung heißt *zulässig*, falls $x_B \geq 0$. Offenbar gehört eine Basislösung genau dann zu P, wenn sie zulässig ist. Die Existenz von Basislösungen folgt aus der Rangmaximalität der Koeffizientenmatrix.

Beispiel: $\hat{x} := \begin{bmatrix} 0 \\ 1 \end{bmatrix} \in K^3$ ist zulässige Basislösung zu $B = (3), N = (12)$. $\hat{x}$ ist eine Ecke, denn $\{\hat{x}\} = \{x \mid x_1 = 0\} \cap \{x \mid x_2 = 0\} \cap \{x \mid x_1 + x_2 + x_3 = 1\}$.

Satz 3.1 *Jeder zulässigen Basislösung entspricht eine Ecke. Ist die Ecke nicht entartet, so ist die zugehörige Basis eindeutig bestimmt.*

Beweis. Die Bedingung $\hat{x}_N = 0$ bestimmt ein eindeutig festgelegtes $\hat{x}$, da A_B regulär, und wegen $\hat{x}_B \geq 0$ liegt $\hat{x}$ in P. Also gilt $\{\hat{x}\} = \bigcap_{i \in N}\{x \mid x_i = 0\} \cap \{x \mid Ax = b\} \cap \{x \mid x \geq 0\}$. Also ist $\hat{x}$ eine Ecke von P. Umgekehrt erfüllt eine Ecke $\hat{x}$ die Vorzeichenbedingungen und liegt sowohl auf der affinen Ebene $\{x \mid Ax = b\}$ und als auch auf (mindestens) n Vorzeichenhyperebenen $\hat{x}_i = 0, i \in N$, wobei $\hat{x}$ gerade der Schnitt aller Ebenen ist. Dann muß A_B regulär sein, d.h. die Ecke ist eine Basislösung. Ist die Ecke nicht entartet, so liegt sie auf genau n Vorzeichenebenen, d.h. N und damit B ist eindeutig. □

In Abbildung (I.9) finden sich Beispiele für Ecken und die zugehörigen Hyperebenen der Nichtbasen N in kanonischer Darstellung. Die übliche Startbasis entspricht dem Ursprung, d.h. $N = \{1, 2, \ldots, n\}, B = \{n+1, \ldots, n+m\}$. In Standardform gilt dann $x_N = 0, x_B = b$. Diese Startbasis ist zulässig, falls $b \geq 0$.

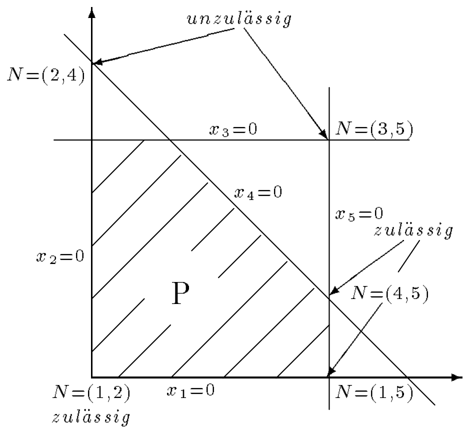


Abbildung I.9:

Darstellung von P in einer Ecke

B sei eine zugehörige Basis. Bei Entartung ist B nicht eindeutig festgelegt. Multiplikation der Gleichung

$$A_B x_B + A_N x_N = b$$

mit A_B^{-1} führt auf die äquivalente Gleichung

$$x_B = A_B^{-1} b - A_B^{-1} A_N x_N.$$

Wir definieren $\tilde{A}_N := A_B^{-1} A_N$ und $\tilde{b} := A_B^{-1} b$. Dann läßt sich P in der Ecke durch

$$x_B = \tilde{b} - \tilde{A}_N x_N, \quad x_B \geq 0, x_N \geq 0 \tag{3.1}$$

beschreiben. Die *Darstellungskoeffizienten* $t_{ij}, i = 1, \ldots, m, j = 1, \ldots, n$ der rechten Seite legen wir durch

$$x_{B(i)} =: t_{i0} + \sum_{j=1}^{n} t_{ij} x_{N(j)} \quad i = 1, \ldots, m. \tag{3.2}$$

fest. Jede mögliche Lösung kann durch Angabe von x_N eindeutig angegeben werden. Entsprechend nennt man die Basisvariablen auch *abhängige* und die Nichtbasisvariablen *unabhängige* Variablen. Wir können das Polyeder bijektiv in den Raum der unabhängigen Variablen x_N projizieren. Die Darstellung in diesem Raum ist gegeben durch die zugehörige kanonische Form

$$\tilde{A}_N x_N \leq \tilde{b}, \quad x_N \geq 0.$$

Beispiel (Basis $B = (2, 4)$): Multiplikation mit der Basisinversen entspricht der Auflösung nach den Basisvariablen. Die Auflösung von

$$\begin{aligned} x_1 + x_2 + x_3 &= 1 \\ -x_1 + x_2 + x_4 &= 1 \end{aligned}$$

zur Basis $B = (34)$ ist besonders einfach:

$$x_3 = 1 - x_1 - x_2$$
$$x_4 = 1 + x_1 - x_2$$

Die Auflösung zur Basis $B = (24)$ beginnt in der Zeile der die Basis verlassenden Variable, wodurch wir eine Darstellung der neuen Basisvariablen erhalten. Diese wird dann zur Elimination in die restlichen Zeilen eingesetzt.

$$x_2 = 1 - x_1 - x_3$$
$$x_4 = 1 + x_1 - (1 - x_1 - x_3) = 0 + 2x_1 + x_3$$

Die Projektion in den Raum der unabhängigen Variablen liefert das Polyeder

$$\begin{array}{r} x_1 + x_3 \leq 1 \\ -2x_1 - x_3 \leq 0 \end{array}, \quad x_1 \geq 0, x_3 \geq 0.$$

Darstellung der Zielfunktion $z(x)$ in einer Ecke

B sei eine zugehörige Basis. Der Zielfunktionswert sei gegeben durch $z = c_0 + c^T x$, wobei c_0 eine von den unabhängigen Variablen nicht berührte Konstante erfassen soll. In der ersten Zeile der folgenden Gleichungen wird x_B aus Gleichung (3.1) eingesetzt.

$$\begin{aligned} z &= c_0 + c_B^T x_B + c_N^T x_N \\ &= (c_0 + c_B^T A_B^{-1} b) - c_B^T A_B^{-1} A_N x_N + c_N^T x_N \\ &= \tilde{c}_0 + (c_N^T - c_B^T A_B^{-1} A_N) x_N \\ &= \tilde{c}_0 + \tilde{c}_N^T x_N \end{aligned}$$

Wir definieren durch $\tilde{c}_0 := c_0 + c_B^T A_B^{-1} b$ und $\tilde{c}_N := c_N^T - c_B^T A_B^{-1} A_N$ entsprechende *reduzierte Kostenkoeffizienten* $\tilde{c}_0, \tilde{c}_N$. Die Darstellung

$$z = \tilde{c}_0 + \tilde{c}_N^T x_N \tag{3.3}$$

der Zielfunktion zeigt, wie der Wert jeder zulässigen Lösung allein durch x_N festgelegt ist. Die Darstellungskoeffizienten der rechten Seite werden analog zu 3.2 festgelegt:

$$z =: t_{00} + \sum_{j=1}^{n} t_{0j} x_{N(j)}. \tag{3.4}$$

Beispiel ($B = (2,4), N = (1,3)$): Erweitern wir unser bisheriges Beispiel zu der linearen Optimierungsaufgabe $\max\{x_2 \mid x \in P\}$, so ergibt sich in der Ecke zu $N = (1,3)$ die Darstellung $z = 0 + x_2 = 1 - x_1 - x_3$ der Zielfunktion.

Hinreichendes Optimalitätskriterium in einer Ecke $\bar{x}$

In der Ecke $\bar{x}$ zu B gilt $\bar{x}_B = \tilde{b}, \bar{x}_N = 0$. Aus der Darstellung

$$\max\{\tilde{c}_N^T x_N \mid \tilde{A}_N x_N \leq \tilde{b}, x_N \geq 0\}.$$

der Aufgabe im Raum der unabhängigen Variablen folgt das hinreichende Optimalitätskriterium

(3.5) $$\tilde{c}_N \leq 0 \rightsquigarrow \bar{x} \text{ ist optimal.}$$

In unserem Beispiel ist daher die Lösung $x^T = (0100)$ zur Basis $B = (2,4)$ optimal. Die optimale Lösung $x_1 = 0, x_2 = 1$ des ursprünglichen Problems in kanonischer Form ergibt sich durch Projektion. Den Wert $z = 1$ der optimalen Lösung liefert die Gleichung (3.3).

Transformation aller Daten

Analog zu den bisherigen Definitionen können wir $\tilde{A}_B := A_B^{-1} A_B = E$ und $\tilde{c}_B := c_B^T - c_B^T A_B^{-1} A_B = 0$ festlegen. Wir fassen jetzt alle bisherigen Ergebnisse über die Darstellung des Problems in Abhängigkeit von einer beliebigen Basis B zusammen. Ausgehend von einer Darstellung des LP in Standardform

$$\max\{c_0 + c^T x \mid Ax = b, x \geq 0\}$$

ergibt sich die nach den Basisvariablen in der Basis B aufgelöste Darstellung

$$\max\{\tilde{c}_0 + \tilde{c}^T x \mid \tilde{A}x = \tilde{b}, x \geq 0\}$$

aus der Gleichung

(3.6) $$\begin{pmatrix} 1 & c_B^T A_B^{-1} \\ 0 & A_B^{-1} \end{pmatrix} \cdot \begin{pmatrix} c_0 & c^T \\ b & -A \end{pmatrix} = \begin{pmatrix} \tilde{c}_0 & \tilde{c}^T \\ \tilde{b} & -\tilde{A} \end{pmatrix}$$

Die Transformationsmatrix wird mit $F(B)$, die Matrix der Ausgangsdarstellung mit T und die zu B mit $\tilde{T}$ bezeichnet. Die zur Basis gehörenden Spalten von $\tilde{T}$ müssen nicht berechnet werden, da nach Definition stets gilt

$$\begin{bmatrix} \tilde{c}_B^T \\ \tilde{A}_B \end{bmatrix} \equiv \begin{bmatrix} 0 \\ E \end{bmatrix}.$$

Bei der Transformation ist es gleichgültig, ob T die ursprünglichen Problemdaten oder die Darstellung zu einer anderen Basis enthält. Wichtig ist dabei nur, daß in $F(B)$ dieselben Daten zugrundegelegt werden.

Verbesserung der Zielfunktion, falls $t_{0s} > 0$

Die grundlegende Idee besteht darin, sukzessive einfache Basiswechsel durchzuführen, wobei die Zielfunktion nicht fallen darf. Ist das hinreichende Optimalitätskriterium wegen $t_{0s} > 0$ nicht erfüllt, so wird versucht, die Variable $x_{N(s)}$ möglichst groß zu machen, wobei $x_{N(k)} = 0$ für alle $k \neq s$. In der vorliegenden Darstellung

(3.7) $$z = t_{00} + \sum_{j=1}^{n} t_{0j} x_{N(j)}$$

(3.8) $$x_{B(i)} = t_{i0} + \sum_{j=1}^{n} t_{ij} x_{N(j)}, \quad i = 1, \ldots, m$$

erkennen wir, daß dabei die Zielfunktionswerte wachsen. Falls $t_{is} \geq 0$ für alle $i = 1, \ldots, m$, so wachsen die Basisvariablen mit, d.h. alle Vorzeichenbeschränkungen bleiben erfüllt. In diesem Fall ist die Zielfunktion nicht beschränkt. Anderenfalls fallen die Werte der Basisvariablen $x_{B(i)}$ mit $t_{is} < 0$ und $x_{N(s)}$ kann nur soweit erhöht werden, wie es die Vorzeichenbeschränkungen dieser Basisvariablen erlauben. Aus den entsprechenden Bedingungen $0 \leq x_{B(i)} = t_{i0} + t_{is}x_{N(s)}$ folgern wir, daß

$$x_{N(s)} := \frac{t_{r0}}{-t_{rs}} := \min\{\frac{t_{i0}}{-t_{is}} \mid t_{is} < 0\}$$

der maximale erlaubte Wert von $x_{N(s)}$ ist. Für diesen Wert ist $x_{B(r)} = 0$ und wir erhalten eine passende neue Basis, indem wir die Basisvariable $x_{B(r)}$ gegen die Nichtbasisvariable $x_{N(s)}$ austauschen:

$$\bar{B}(i) = \begin{cases} B(i) & i \neq r \\ N(s) & i = r \end{cases}, \quad \bar{N}(j) = \begin{cases} N(j) & j \neq s \\ B(r) & j = s \end{cases}.$$

In Abbildung (I.10) sind die auftretenden beschränkten und unbeschränkten Fälle geometrisch verdeutlicht. Ausgehend von der Ursprungsecke im Raum der Nichtbasisvariablen bewegen wir uns mit Erhöhung von $x_{N(s)}$ längs einer Kante des Polyeders, bis wir auf eine der Restriktionen stoßen.

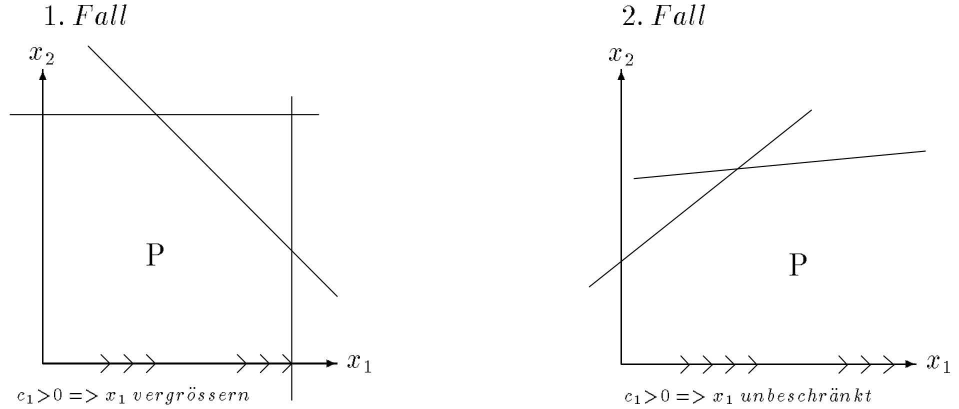


Abbildung I.10: Pivotschritt

Darstellungswechsel bei $(B(r) \leftrightarrow N(s))$

Die neuen Darstellungskoeffizienten zur Basis $\bar{B}$ seien mit $\bar{t}_{ij}$ bezeichnet. Zunächst wird Zeile r aus Darstellung (3.7) nach $x_{N(s)}$ aufgelöst. Wir erhalten

$$\begin{aligned} -t_{rs}x_{N(s)} &= t_{r0} - x_{B(r)} + \sum_{j \neq s} t_{rj}x_{N(j)}, \\ x_{\bar{B}(r)} = x_{N(s)} &= \frac{t_{r0}}{-t_{rs}} + \frac{1}{t_{rs}}x_{B(r)} + \sum_{j \neq s} \frac{t_{rj}}{-t_{rs}}x_{N(j)}, \\ \bar{x}_{\bar{B}(r)} &= \bar{t}_{r0} + \bar{t}_{rs}x_{\bar{N}(s)} + \sum_{j \neq s} \bar{t}_{rj}x_{\bar{N}(j)}. \end{aligned}$$

Die Werte der neuen Darstellungskoeffizienten in der letzten Zeile ergeben sich durch Koeffizientenvergleich mit der vorletzten Zeile. Anschließend ersetzen wir in allen anderen Zeilen der Darstellung (3.7) die Variable $x_{N(s)}$ und bestimmen so die Koeffizienten der neuen Darstellung von $x_{\bar{B}(i)} = x_{B(i)}$:

$$\begin{aligned}
x_{\bar{B}(i)} = x_{B(i)} &= t_{i0} + t_{is}\Big(\frac{t_{r0}}{-t_{rs}} + \frac{1}{t_{rs}} x_{B(r)} + \sum_{j\neq s} \frac{t_{rj}}{-t_{rs}} x_{N(j)}\Big) + \sum_{j\neq s} t_{ij} x_{N(j)}, \\
x_{\bar{B}(i)} &= \Big(t_{i0} - \frac{t_{is}}{t_{rs}} t_{r0}\Big) + \frac{t_{is}}{t_{rs}} x_{\bar{N}(s)} + \sum_{j\neq s}\Big(t_{ij} - \frac{t_{is}}{t_{rs}} t_{rj}\Big) x_{\bar{N}(j)}, \\
x_{\bar{B}(i)} &= \bar{t}_{i0} + \bar{t}_{is} x_{\bar{N}(s)} + \sum_{j\neq s} \bar{t}_{ij} x_{\bar{N}(j)}.
\end{aligned}$$

Die sukzessive neuberechneten Daten können wir in einem Tableau zusammenfassen, dessen Zeilen und Spalten mit den Basis- bzw. Nichtbasisindizes numeriert werden:

$$\begin{array}{|c|c|ccc|}
\hline
 & & N(1) & \cdots & N(n) \\
\hline
 & c_0 & \cdots & c_N^T & \cdots \\
\hline
B(1) & & & & \\
\vdots & b & & -A_N & \\
B(m) & & & & \\
\hline
\end{array}
\equiv
\begin{array}{|c|c|ccc|}
\hline
 & & & N & \\
\hline
 & t_{00} & t_{01} & \cdots & t_{0n} \\
\hline
 & & t_{11} & & \\
B & & & \ddots & \\
 & t_{m0} & & & t_{mn} \\
\hline
\end{array}$$

Der Algorithmus läßt sich jetzt folgendermaßen beschreiben:

Algorithmus 3.1 (Das Simplexverfahren mit Tableauberechnung)

Unter der Voraussetzung $b \geq 0$ soll eine optimale Lösung des linearen Programms in kanonischer Form

$$\max\left\{c^T x \;\middle|\; Ax \leq b, \quad x \geq 0\right\},$$

bestimmt werden. Dazu werden zunächst die Ausgangsdaten in geeigneter Weise zusammengefaßt:

$$(t_{ij}) := \begin{pmatrix} 0 & c^T \\ b & -A \end{pmatrix}, \qquad i = 0, \ldots, m, \quad j = 0, \ldots, n,$$

$$B := (n+1, \ldots, n+m), \qquad N := (1, \ldots, n).$$

Die Schritte einer Iteration des Verfahrens lauten:

1. **Optimalität:**
   Ist ein $t_{0j} > 0$, so gehe nach 2.
   Anderenfalls ist eine Optimallösung erreicht. Setze

   $$x_{B(i)} := t_{i0}, \quad i = 1, \ldots, m; \qquad x_{N(j)} := 0, \quad j = 1, \ldots, n; \qquad z := t_{00}$$

   Stop.

2. **Bestimmung der Austauschspalte $N(j)$:**
   Man wähle den Index s so, daß
   $$t_{0s} = \max_{j=1,\ldots,n} t_{0j}.$$

3. **Endlichkeitskriterium:**
   Sind alle $t_{is} \geq 0$, $i = 1, \ldots, m$, so existiert keine endliche Lösung. Stop.

4. **Bestimmung der Austauschzeile $B(r)$:**
   Man wähle den Index r so, daß
   $$\frac{t_{r0}}{-t_{rs}} = \min\left\{\frac{t_{i0}}{-t_{is}} \mid t_{is} < 0,\, i = 1, \ldots, m\right\}.$$
   Gehe nach 5.

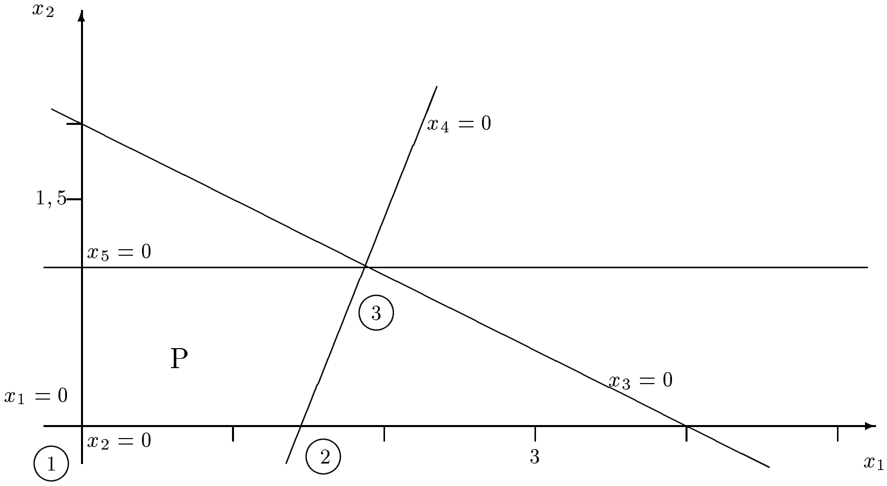
5. **Pivotoperation:**
   Vertausche die r-te Komponente von B mit der s-ten Komponente von N und setze
   $$\begin{array}{lcll} \bar{t}_{rs} & := & \dfrac{1}{t_{rs}}; & \\ \bar{t}_{rj} & := & -\dfrac{t_{rj}}{t_{rs}}, & \text{für } j = 0, \ldots, n,\ j \neq s; \\ \bar{t}_{is} & := & \dfrac{t_{is}}{t_{rs}}, & \text{für } i = 0, \ldots, m,\ i \neq r; \\ \bar{t}_{ij} & := & t_{ij} - \dfrac{t_{is}}{t_{rs}} t_{rj}, & \text{für } i = 0, \ldots, m,\ i \neq r \text{ und } j = 0, \ldots, n,\ j \neq s. \end{array}$$
   Nun ersetze $t_{ij} := \bar{t}_{ij}$, $i = 0, \ldots, m$, $j = 0, \ldots, n$.
   Gehe nach 1.

Beispiel (Tableaumethode):

Bestimme eine optimale Lösung des linearen Programms

$$\begin{array}{lrll} \max & x_1 + x_2 & & \text{Schlupfvariable} \\ \text{unter} & x_1 + 2x_2 & \leq 4 & \quad x_3 \\ & 2x_1 - x_2 & \leq 3 & \quad x_4 \\ & x_2 & \leq 1 & \quad x_5 \\ & x_1,\, x_2 & \geq 0 & \end{array}$$

Das Polyeder ist von folgender Gestalt:



1. Simplextableau:

		x_1	x_2
z	0	1	1
x_3	4	-1	-2
x_4	3	$\boxed{-2}$	1
x_5	1	0	-1

– zulässig, nicht optimal
– Pivot: $s = 1$, $r = 2$

2. Simplextableau:

		x_4	x_2
z	$\frac{3}{2}$	$-\frac{1}{2}$	$\frac{3}{2}$
x_3	$\frac{5}{2}$	$\frac{1}{2}$	$-\frac{5}{2}$
x_1	$\frac{3}{2}$	$-\frac{1}{2}$	$\frac{1}{2}$
x_5	1	0	$\boxed{-1}$

– zulässig, nicht optimal
– Pivot: $s = 2$, $r = 3$

3. Simplextableau:

		x_4	x_5
z	3	$-\frac{1}{2}$	$-\frac{3}{2}$
x_3	0	$\frac{1}{2}$	$\frac{5}{2}$
x_1	2	$-\frac{1}{2}$	$-\frac{1}{2}$
x_2	1	0	-1

– zulässig, optimal
– entartete Lösung, da $x_3 = 0$:
$x_1 = 2,\ x_2 = 1,\ z = 3.$

Matrixform des Darstellungswechsels

Wir setzen jetzt voraus, daß die Ausgangsdarstellung T in der Transformationsgleichung

$$F(B) \cdot T = \bar{T},$$

entsprechend (3.6), die Koeffizienten zur Basis B enthält, d.h. $A_B = E, c_B^T = 0$. Die zur neuen Basis $\bar{B}$ gehörende Basismatrix ergibt sich durch Austausch der r-ten Spalte $A_{B(r)}$

der alten Basismatrix gegen $A_{N(s)}$, die s-te Spalte der Nichtbasismatrix:

$$A_{\bar{B}} = \left[\begin{array}{ccc|c|ccc} 1 & & & a_{1N(s)} & & & \\ & \ddots & & \vdots & & & \\ & & 1 & & & & \\ & & & & & & \\ & & & & 1 & & \\ & & & \vdots & & \ddots & \\ & & & a_{mN(s)} & & & 1 \end{array}\right] \quad \text{(nur } A_{B(r)} \text{ gegen } A_{N(s)} \text{ getauscht).}$$

Die Inverse der neuen Basismatrix besitzt die gleiche einfache Form:

$$A_{\bar{B}}^{-1} = \left[\begin{array}{ccc|c|ccc} 1 & & & & & & \\ & \ddots & & \vdots & & & \\ & & 1 & & & & \\ & & & d & & & \\ & & & & 1 & & \\ & & & \vdots & & \ddots & \\ & & & & & & 1 \end{array}\right] \quad \text{mit } d_i = \begin{cases} -\frac{a_{iN(s)}}{a_{rN(s)}} & i \neq r \\ \frac{1}{a_{rN(s)}} & i = r \end{cases} .$$

Daher gilt

$$c_{\bar{B}}^T A_{\bar{B}}^{-1} = (0, \ldots, 0, c_{N(s)}, 0, \ldots, 0) A_{\bar{B}}^{-1} = (0, \ldots, 0, \frac{c_{N(s)}}{a_{rN(s)}}, 0, \ldots, 0)$$

und die aus Gleichung (3.6) abzulesende Transformationsmatrix $F(\bar{B})$ ist

$$F_{rs} := F(\bar{B}) \left[\begin{array}{c|ccc|c|ccc} 1 & & & & \frac{c_{N(s)}}{a_{rN(s)}} & & & \\ & 1 & & & -\frac{a_{1N(s)}}{a_{rN(s)}} & & & \\ & & \ddots & & \vdots & & & \\ & & & 1 & & & & \\ & & & & \frac{1}{a_{rN(s)}} & & & \\ & & & & & 1 & & \\ & & & & \vdots & & \ddots & \\ & & & & -\frac{a_{mN(s)}}{a_{rN(s)}} & & & 1 \end{array}\right]. \tag{3.9}$$

Die Pivottransformation mit dem sogenannten Pivotelement $a_{rN(s)}$ lautet dann

$$F_{rs} T = \bar{T}. \tag{3.10}$$

Die wesentliche r-te Spalte f von F_{rs} kann leicht mit den Darstellungskoeffizienten t_{is} beschrieben werden:

$$f^T = \frac{1}{-t_{rs}} (t_{0s}, t_{1s}, \ldots, 1, \ldots, t_{ms}).$$

Bei großen Linearen Programmen ist die Anzahl der Pivotsschritte proportional zur Anzahl m der Restriktionen. In der Tableaumethode werden daher, falls $m \ll n$, viele Spalten transformiert, deren Kenntnis im Verlauf des Verfahrens unnötig ist. Wie man an der Gestalt (siehe 3.6) der Transformationsmatrix $F(B)$ leicht abliest, ist im wesentlichen die Kenntnis der Basisinversen A_B^{-1} erforderlich, um Daten des transformierten Tableaus zu berechnen. Bei Unkenntnis der Basisinversen oder einer geeigneten Zerlegung der Basismatrix würde sich der Aufwand zur Lösung der auftretenden Gleichungssysteme um den Faktor m erhöhen. Welche Daten müssen wir nun zur Durchführung eines Iterationsschrittes berechnen? Zur Beantwortung dieser Frage betrachten wir die einzelnen Schritte nochmals.

- **(Optimalitätstest)** Hier müssen die reduzierten Kostenkoeffizienten $\tilde{c}_N^T = c_N^T - c_B^T A_B^{-1} A_N$ berechnet werden. Man berechnet dazu zunächst $\pi^T := c_B^T A_B^{-1}$ und anschließend $\tilde{c}_N^T := c_N^T - \pi^T A_N$. Greift der Optimalitätstest, so erhalten wir die Lösung aus $x_B := A_B^{-1} b, x_N = 0$.
- **(Bestimmung der Pivotspalte s)** Der Index s wird beim Optimalitätstest ermittelt.
- **(Endlichkeitskriterium)** Hier müssen wir die Pivotspalte $\tilde{A}_{N(s)} := A_B^{-1} A_{N(s)}$ berechnen.
- **(Bestimmung der Austauschzeile r)** Um den Index r ermitteln zu können, müssen wir nun noch die Basislösung $\tilde{b} := A_B^{-1} b$ berechnen.
- **(Pivotoperation)** Hier gilt es, die Basisinverse A_B^{-1} oder die verwendete Zerlegung der Basismatrix zu aktualisieren.

Die auftretenden drei Gleichungssysteme sind also:

$$c_B^T =: \pi^T A_B, \quad A_B x_B := b, \quad A_B \tilde{A}_{N(s)} := A_{N(s)}.$$

Ein einfaches, aber numerisch ungünstiges Verfahren beruht auf der expliziten Bereitstellung der Basisinversen. Im folgenden diskutieren wir hierfür den noch fehlenden Aktualisierungschritt.

Aktualisierung der Basisinversen

In revidierten Verfahren kennen wir nur die Daten des Starttableaus T, während die Tableaus $\tilde{T}$ zur aktuellen Basis B sowie $\bar{T}$ zur neuen Basis $\bar{B}$ weitgehend unbekannt sind. Es gilt

$$F(B)T = \tilde{T}, \quad F(\bar{B})T = \bar{T}, \quad \tilde{F}_{rs}\tilde{T} = \bar{T}.$$

Da das Starttableau T maximalen Rang besitzt, folgt hieraus

$$\tilde{F}_{rs} F(B) = F(\bar{B}),$$

d.h. die Transformationsmatrix $F(B)$ transformiert sich genauso wie das Tableau. Fassen wir die nichttrivialen Spalten von $F(B)$ mit der ersten Tableauspalte zusammen, d.h.

$$D(B) := \left[\begin{array}{c|c} c_B^T A_B^{-1} & c_B^T A_B^{-1} b \\ \hline A_B^{-1} & A_B^{-1} b \end{array}\right] \equiv (d_{ij})_{\substack{i=0,\dots,m \\ j=1,\dots,m+1}}$$

so transformieren sich alle d_{ij} auf diese Weise. Enthält das Starttableau T die Darstellungskoeffizienten zu einer zulässigen Basis B so gilt

$$D(B) = \left[\begin{array}{c|c} 0 & 0 \\ \hline E & b \end{array}\right]$$

Dies gilt insbesondere, wenn beim Start ein aus der kanonischen Form abgeleitetes Problem in Standardform mit der Basis $B = (n+1, \dots, n+m)$ vorliegt.

Das resultierende Verfahren läßt sich analog zum Simplexverfahren mit Tableauberechnung strukturieren.

Algorithmus 3.2 (Ein Revidiertes Simplexverfahren)

Unter der Voraussetzung $b \geq 0$ soll eine optimale Lösung des linearen Programms in kanonischer Form

$$\max \left\{ c^T x \mid Ax \leq b, \quad x \geq 0 \right\},$$

bestimmt werden. Die weiteren Startdaten sind

$$\begin{aligned} B &:= (n+1, \dots, n+m), \qquad N := (1, \dots, n), \\ d_{ij} &:= \begin{cases} 1 & \text{falls } i = j,\ i = 1, \dots, m,\ j = 1, \dots, m \\ 0 & \text{falls } i \neq j,\ i = 0, \dots, m,\ j = 1, \dots, m \end{cases} \\ d_{i,m+1} &:= \begin{cases} 0 & \text{für } i = 0 \\ b_i & \text{für } i = 1, \dots, m \end{cases} \end{aligned}$$

Die Schritte einer Iteration des Verfahrens lauten:

1. Berechne für alle $k \in \boldsymbol{N}$: $\tilde{c}_k := c_k - \sum_{j=1}^m d_{0j} a_{jk}$.
2. **Optimalität:**
   Falls für alle $k \in \boldsymbol{N}$ gilt: $\tilde{c}_k \leq 0$, so ist $x_k := 0$ für alle $k \in \boldsymbol{N}$, $x_{B(i)} := d_{i,m+1}$, $i = 1, \dots, m$, eine optimale Basislösung mit Zielfunktionswert $z := d_{0,m+1}$. Stop.
3. **Bestimmung der Austauschspalte s:**
   Wähle $N(\hat{\jmath}) = s$ so, daß $\tilde{c}_s = \max_{k \in \boldsymbol{N}} \tilde{c}_k$.
4. Berechne $\tilde{a}_{is} = \sum_{j=1}^m d_{ij} a_{js}$ für alle $i = 1, \dots, m$.
5. **Endlichkeitskriterium:**
   Falls $\tilde{a}_{is} \leq 0$ für alle $i = 1, \dots, m$, so gibt es keine endliche Optimallösung. Stop.

6. **Bestimmung der Austauschzeile** $B(r)$**:**
   Wähle r so, daß

$$\frac{d_{r,m+1}}{\tilde{a}_{rs}} = \min\left\{ \frac{d_{i,m+1}}{\tilde{a}_{is}} \middle| \tilde{a}_{is} > 0; \quad i = 1, \ldots, m \right\}.$$

7. **Pivotoperation** Wir überschreiben die alten Koeffizienten in der folgenden Reihenfolge durch die neuen Koeffizienten:

$$\begin{aligned}
d_{rj} &:= \frac{d_{rj}}{\tilde{a}_{rs}} \quad \text{für } j = 1, \ldots, m+1, \\
d_{0j} &:= d_{0j} + \tilde{c}_s d_{rj} \quad \text{für } j = 1, \ldots, m+1, \\
d_{ij} &:= d_{ij} - \tilde{a}_{is} d_{rj} \quad \text{für } i = 1, \ldots, m, (i \neq r); \text{ und } j = 1, \ldots, m+1, \\
N(\hat{\jmath}) &:= B(r); \\
B(r) &:= s
\end{aligned}$$

   und gehe nach 1.

Wir verwenden diese Methode zur Lösung der Aufgabe

$$\begin{array}{lrcl}
\max & x_1 + 2x_2 + 2x_3 - x_4 + 3x_5 + x_6 - x_7 + x_8 & & \\
\text{unter} & x_1 + x_2 + x_3 - x_4 + x_5 + x_6 + x_7 + x_8 & \leq & 4 \\
& x_2 - x_3 + 3x_4 + x_5 + x_6 + x_7 + x_8 & \leq & 2 \\
& x & \geq & 0
\end{array}$$

1. Iteration Für $N = (1\ 2\ 3\ 4\ 5\ 6\ 7\ 8)$ ist der maximale reduzierte Kostenkoeffizient c_5. Im ersten Pivotschritt erfolgt daher der Austausch $x_{10} \leftrightarrow x_5$

$$\underbrace{\begin{bmatrix} 1 & 0 & 3 \\ 0 & 1 & -1 \\ 0 & 0 & 1 \end{bmatrix}}_{\tilde{F}_{rs}} \cdot \underbrace{\begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 4 \\ 0 & 1 & 2 \end{bmatrix}}_{D} = \underbrace{\begin{bmatrix} 0 & 3 & 6 \\ 1 & -1 & 2 \\ 0 & 1 & 2 \end{bmatrix}}_{\tilde{D}}$$

2. Iteration Für $N = (1\ 2\ 3\ 4\ 10\ 6\ 7\ 8)$ ergeben sich aus $\tilde{c}_N^T = c_N^T - d_{0,1\ldots m}^T A_N$ die reduzierten Kosten:

$$\begin{aligned}
\tilde{c}_N^T &= (1, 2, 2, -1, \overbrace{0}^{\text{zu } x_{10}}, 1, -1, 1) - (0, 3) \cdot \begin{bmatrix} 1 & 1 & 1 & -1 & 0 & 1 & 1 & 1 \\ 0 & 1 & -1 & 3 & 1 & 1 & 1 & 1 \end{bmatrix} \\
&= (1, -1, \underbrace{5}_{\text{maximal}}, -10, -3, -2, -4, -2)
\end{aligned}$$

Die Pivotspalte $A_B^{-1} A_3$ zu x_3 berechnen wir aus:

$$\begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ -1 \end{bmatrix} := \begin{bmatrix} 2 \\ -1 \end{bmatrix}$$

Mit Hilfe des in $\tilde{D}$ enthaltenen x_B finden wir den Pivotschritt $x_9 \leftrightarrow x_3$

$$\begin{bmatrix} 1 & \frac{5}{2} & 0 \\ 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 & 3 & 6 \\ 1 & -1 & 2 \\ 0 & 1 & 2 \end{bmatrix} = \begin{bmatrix} \frac{5}{2} & \frac{1}{2} & 11 \\ \frac{1}{2} & -\frac{1}{2} & 1 \\ \frac{1}{2} & \frac{1}{2} & 3 \end{bmatrix}$$

3. Iteration Für $N = (1\ 2\ 9\ 4\ 10\ 6\ 7\ 8)$ erfüllen die reduzierten Kostenkoeffizienten

$$\tilde{c}_N^T = (-\frac{3}{2}, -1, -\frac{5}{2}, 0, -\frac{1}{2}, -2, -4, -2)$$

das hinreichende Optimalitätskriterium.

Bemerkung:

1. Alle Operationen können innerhalb von K durchgeführt werden. Dies ist nicht bei allen Methoden zur Lösung linearer Optimierungsprobleme der Fall, z.B. bei der Ellipsoid-Methode. Bei Pivotelementen aus $\{-1, 1\}$ erfordert der Algorithmus nur Additionen und Subtraktionen.

2. Offene Probleme:
   – Falls in der zu Beginn gegebenen Darstellung zu einer Basis $b \ngeq 0$, muß eine geeignete Startlösung bestimmt werden.
   – Falls bei Entartung Zyklen von Basen auftreten, ist die Endlichkeit des Verfahrens in Frage gestellt.

4 Anfangslösung und Endlichkeit

Auffinden einer zulässigen Startbasis

Zum Finden einer zulässigen Basis stellen werden hier drei verschiedene Methoden vorgestellt: die Zweiphasen-, die Mehrphasen- und die M-Methode.

Die Zweiphasenmethode

Beim Simplexverfahren haben wir bislang vorausgesetzt, daß eine zulässige Startbasis bekannt ist. Für das Polyeder der in kanonischer Form gegebenen Aufgabe

$$\begin{array}{lrcr} \text{max} & x_1 + x_2 & & \\ \text{unter} & -\frac{1}{2}x_1 + x_2 & \leq & 2 \\ & -x_1 - x_2 & \leq & -1 \\ & 2x_1 - x_2 & \leq & 1 \\ & x_1,\, x_2 & \geq & 0 \end{array}$$

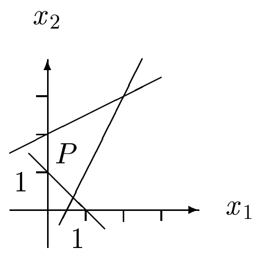


Abbildung I.11: Unzulässiger Ursprung

ist diese Annahme verletzt.

Ziel: Finden einer zulässigen Basislösung.

Einfügen von zusätzlichen Hilfsvariablen u, die die Schlupfvariablen zu den Ungleichungen mit negativen rechten Seiten zu einer Basis ergänzen.

Hierzu: Partitionierung der Restriktionen (aufgelöst nach Basisvariablen):

Sei allgemein $b = \begin{bmatrix} b^1 \\ b^2 \end{bmatrix}$ mit $b^1 < 0$, $b^2 \geq 0$.

$$\begin{aligned} u &= -b^1 + A^1 x + y^1 \\ y^2 &= +b^2 - A^2 x \end{aligned}$$

für Variable $x, y^1, y^2, u \geq 0$.

Gelingt es, die Hilfsvariablen auf $u = 0$ zu verringern, so liegt eine zulässige Basislösung des ursprünglichen Polyeders vor.

Maximiere aus diesem Grund die Hilfszielfunktion

$$z_H = -1^T u = 1^T b^1 - 1^T A^1 x - 1^T y^1.$$

Die ursprüngliche Zielfunktion

$$z = c^T x$$

muß dabei mittransformiert werden, damit bei Erreichen einer zulässigen Basislösung derzeitige Darstellung der ursprünglichen Zielfunktion bekannt ist.

In Tableauform liegen dann die Darstellungskoeffizienten

		x	y^1
z_H	$1^T b^1$	$-1^T A^1$	-1^T
z	0	c^T	0
u	$-b^1$	A^1	E
y^2	b^2	$-A^2$	0

vor. Für das obige Beispiel ergibt sich

		x_1	x_2	y_1^1
z_H	-1	1	1	-1
z	0	1	1	0
u_1	1	-1	-1	1
y_1^2	2	$\frac{1}{2}$	-1	0
y_2^2	1	-2	1	0

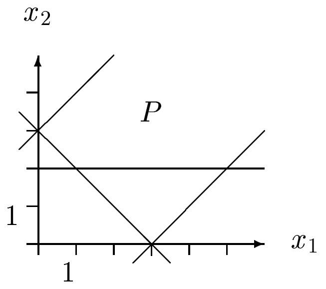
Die Hilfszielfunktion ist nach oben durch 0 beschränkt. Wird die Schranke für eine optimale Basislösung angenommen, so ist $u = 0$ und (x, y) eine zulässige Lösung für das Ausgangsproblem; anderenfalls gilt $P = \emptyset$.

Beispiel (Zur Zweiphasenmethode):

Wir bestimmen eine optimale Lösung des linearen Programms

$$\begin{array}{lrl} \max & -x_1 - 2x_2 & \\ \text{unter} & x_1 + x_2 & \geq 3 \\ & x_2 & \geq 2 \\ & -x_1 + x_2 & \leq 3 \\ & x_1 - x_2 & \leq 3 \\ & x_1, x_2 & \geq 0 \end{array}$$

dessen Polyeder vom Ursprung wegbeschränkt ist.



Als Ausgangstableau finden wir

		x_1	x_2	y_1	y_2
z_H	-5	1	2	-1	-1
z	0	-1	-2	0	0
u_1	3	-1	-1	1	0
u_2	2	0	$-\boxed{1}$	0	1
y_3	3	1	-1	0	0
y_4	3	-1	1	0	0

wobei z_H die Hilfszielfunktion bezeichnet. Die ursprünglichen Zielfunktion z wird mittransformiert. Im ersten Pivotschritt mit $r = s = 2$ verläßt die Hilfsvariable u_2 die Basis. Wenn wir die jeweils aus der Basis austretenden Hilfsvariablen sofort eliminieren, ergeben sich in der Hilfsphase die folgenden Tableaus der Darstellungskoeffizienten:

Tableau 1:

		x_1	y_1	y_2
z_H	-1	1	-1	1
z	-4	-1	0	-2
u_1	1	$-\boxed{1}$	1	-1
x_2	2	0	0	1
y_3	1	1	0	-1
y_4	5	-1	0	1

Tableau 2: $r = s = 1$

		y_1	y_2
z_H	0	0	0
z	-5	-1	-1
x_1	1	1	-1
x_2	2	0	1
y_3	2	1	-2
y_4	4	-1	2

Tableau 2 ist optimal für die Hilfszielfunktion und die gefundene zulässige Basis enthält keine Hilfsvariablen.

Dieses Tableau ist in diesem Beispiel auch optimal für das Ausgangsproblem, denn die reduzierten Kostenkoeffizienten der ursprünglichen Zielfunktion sind nichtpositiv. Die optimale Lösung ist $x_1 = 1, x_2 = 2$ mit Zielfunktionswert $z = -5$.

Die Mehrphasenmethode

Die Mehrphasenmethode benötigt für jede negative Komponente von b eine eigene Phase, jedoch keine Hilfsvariablen. Jede Phase dient dazu, eine Schlupfvariable (die wegen $y^1 = b^1 - A^1 x$ zunächst negativ ist) auf einen positiven Wert zu erhöhen.

Wir betrachten nun die Phase μ (in der die μ-te Komponente von y^1 bearbeitet wird): Wir lösen hier das Problem

$$c_\mu := \max\{y^1_\mu \mid A^2 x + y^2 = b^2, x \geq 0, y_2 \geq 0\}$$

(mit $y^1_\mu = b - a^1_\mu x$, wobei a^1_μ die μ-te Zeile von A_1 bezeichnet) und berechnen die Transformationen für die ursprüngliche Zielfunktion sowie für die nicht berücksichtigten Restriktionen mit.

Während der Berechnung von c_μ führen folgende Fälle zum Ende der Phase:

1. $y^1_\mu \geq 0$

   Hier haben wir eine für die bisherigen Restriktionen zulässige Basislösung gefunden. Wir fügen nun die Zeile $a^1_\mu x + y^1_\mu = b^1_\mu$ den bisherigen Restriktionen hinzu und starten die nächste Phase mit der erweiterten Basis.

2. $y^1_\mu = c_\mu < 0$

Es gibt keine Lösung für das Problem, obwohl noch gar nicht alle Restriktionen berücksichtigt worden sind. Dann kann es aber auch keine Lösung geben, die alle Restriktionen erfüllt. Das Verfahren wird mit dem Resultat $P = \emptyset$ abgebrochen.

3. y^1_μ ist nach oben unbeschränkt

   Es treten in diesem Fall im wesentlichen folgende Tableaudaten auf:

$$\begin{array}{rcl} y^1_\mu &=& \underbrace{t_{\mu 0}}_{<0} + \underbrace{t_{\mu s}}_{>0} \cdot x_{N(s)} \\ x_{B(i)} &=& \underbrace{t_{i0}}_{\geq 0} + \underbrace{t_{is}}_{\geq 0} \cdot x_{N(s)} \end{array}$$

   Dieses läßt einen Pivotschritt mit $B(r) := \mu$ zu, also mit einem *positiven* Pivotelement aus der *Zielfunktionszeile*. Wir erhalten

$$x_{\bar{B}_r} = -\frac{t_{\mu 0}}{t_{\mu s}} > 0.$$

   Auf diese Weise wird die Zeile μ zulässig. Wir nehmen sie zu den bisherigen Restriktionen hinzu und starten analog zu Fall 1 die nächste Phase.

Bei diesem Verfahren erhalten wir also eine monotone Zunahme der Menge der Restriktionen. Wenn alle Phasen zu einem $y^1_\mu \geq 0$ geführt haben, erhalten wir schließlich eine zulässige Basis für das Ausgangsproblem. Wir setzen dann mit dem „normalen“ Simplexverfahren fort.

Beispiel zur Mehrphasenmethode

Wir betrachten noch einmal die Aufgabe

$$\begin{array}{lrrrrrl} \max & - & x_1 & - & 2x_2 & & \\ \text{unter} & & x_1 & + & x_2 & \geq & 3 \\ & & & & x_2 & \geq & 2 \\ & - & x_1 & + & x_2 & \leq & 3 \\ & & x_1 & - & x_2 & \leq & 3 \\ & & x_1, & & x_2 & \geq & 0 \end{array}$$

In den folgenden Tableaus sind die aktuellen Restriktionen mit $*$, die aktuelle Zielfunktion mit $\dagger$ bezeichnet.

$$\begin{array}{|c|c|cc|c} \hline & & x_1 & x_2 & \\ \hline z & 0 & -1 & -2 & \\ \hline y_1 & -3 & 1 & 1 & \dagger \\ y_2 & -2 & 0 & 1 & \\ y_3 & 3 & 1 & -1 & * \\ y_4 & 3 & \boxed{-1} & 1 & * \\ \hline \end{array} \longrightarrow \begin{array}{|c|c|cc|c} \hline & & y_4 & x_2 & \\ \hline z & -3 & 1 & -3 & \\ \hline y_1 & 0 & -1 & 2 & * \\ y_2 & -2 & 0 & \boxed{1} & \dagger \\ y_3 & 6 & -1 & 0 & * \\ x_1 & 3 & -1 & 1 & * \\ \hline \end{array} \longrightarrow \begin{array}{|c|c|cc|c} \hline & & y_4 & y_2 & \\ \hline z & -9 & 1 & -3 & \dagger \\ \hline y_1 & 4 & -1 & 2 & * \\ x_2 & 2 & 0 & 1 & * \\ y_3 & 6 & -1 & 0 & * \\ x_1 & 5 & -1 & 1 & * \\ \hline \end{array}$$

Das letzte Simplextableau ist zulässig; mit den gewohnten Pivotschritten findet man die Optimallösung.

Die M-Methode

Für dieses Verfahren wird genau eine Hilfsvariable benutzt. Wir lösen statt des Problems

$$\max\{c^T x \mid y = b - Ax, x \geq 0, y \geq 0\}$$

das um eine Variable $\tilde{x}$ ergänzte Problem

$$\max\{c^T x - M \cdot \tilde{x} \mid y = b - Ax + (1, \ldots, 1)^T \tilde{x};\ x, y, \tilde{x} \geq 0\}.$$

Intuitiv scheint klar, daß neue Problem für genügend großes M die gleiche Optimallösung besitzt (mit $\tilde{x} = 0$). Tatsächlich läßt sich ein solches M theoretisch finden, jedoch ist es für praktische Probleme normalerweise so groß, daß es sich im Rechner nicht mehr darstellen läßt. Man benutzt dann ein kleineres M. Die Schwierigkeiten, die sich hieraus ergeben, werden später noch erläutert.

Die Startbasis mit den y-Variablen (entsprechend die Nichtbasis mit den x-Variablen und $\tilde{x}$) ist unzulässig. Wir führen einen Pivotschritt $\tilde{x} \leftrightarrow y_r$ mit $b_r := \min\{b_i \mid 1 \leq i \leq m\}$ durch. Somit ergibt sich

$$\begin{aligned} y_i = x_{\bar{B}(i)} &= b_i - b_r \geq 0 \qquad i \neq r \\ \tilde{x} = x_{\bar{B}(r)} &= -b_r \geq 0, \end{aligned}$$

d.h. die Basis $\bar{B}$ ist zulässig. Nun wird das gewohnte Simplexverfahren ausgeführt. Im Falle einer endlichen Optimallösung ergeben sich zwei Fälle:

1. Optimallösung mit $\tilde{x} = 0$

   In diesem Fall ist die Optimallösung auch für das Ausgangsproblem optimal (da wir die Lösungsmenge durch das Einfügen von $\tilde{x}$ nur vergrößert haben).

2. Optimallösung mit $\tilde{x} > 0$

   Aus theoretischer Sicht (wenn wir also das M genügend groß gewählt haben) ist das Ausgangsproblem unlösbar. Da im Computer jedoch (wie wir bereits diskutiert haben) ein kleinerer Wert für M angenommen wird, kann diese Aussage nicht übertragen werden. Man erhält also als Resultat nur „Problem unlösbar oder M zu klein“.

Sobald die Variable $\tilde{x}$ aus der Basis austritt, muß sie nicht mehr berücksichtigt werden.

Beispiel zur M-Methode

Wir betrachten dieselbe Aufgabe wie bei der Mehrphasenmethode. Man kann zeigen daß $M = 3$ ein ausreichend großer Wert ist. Es ergeben sich die folgenden Tableaus:

		x_1	x_2	$\tilde{x}$
z	0	-1	-2	-3
y_1	-3	1	1	$\boxed{1}$
y_2	-2	0	1	1
y_3	3	1	-1	1
y_4	3	-1	1	1

$\longrightarrow$

		x_1	x_2	y_1
z	-9	2	1	-3
$\tilde{x}$	3	-1	-1	1
y_2	1	$\boxed{-1}$	0	1
y_3	6	0	-2	1
y_4	6	-2	0	1

$\longrightarrow$

		y_2	x_2	y_1
z	-7	-2	1	-1
$\tilde{x}$	2	1	$\boxed{-1}$	0
x_1	1	-1	0	1
y_3	6	0	-2	1
y_4	4	2	0	-1

$\longrightarrow$

		y_2	y_1
z	-5	-1	-1
x_2	2	1	0
x_1	1	-1	1
y_3	2	-2	1
y_4	4	2	-1

Das letzte Tableau ist optimal mit $\tilde{x} = 0$ (da $\tilde{x} \in N$). Wir haben also eine optimale Lösung für das Ausgangsproblem gefunden.

Erzwingen der Endlichkeit des Verfahrens

Da die Anzahl der Ecken endlich ist, ist das Simplexverfahren endlich, falls die Zielfunktionswerte in jedem Schritt echt wachsen. Wegen $t_{0s} > 0$, $t_{rs} < 0$ wächst der neue Zielfunktionswert $\bar{t}_{00} = t_{00} + \frac{t_{0s}}{-t_{rs}} t_{r0}$ nur dann nicht, wenn $t_{r0} = x_{B(r)} = 0$, d.h. wenn die Basis B entartet ist. Das folgende Beispiel zeigt, daß das Simplexverfahren in der Tat in einer entarteten Ecke durch die zugehörigen Basislösungen kreisen kann.

Beispiel (Kreisen beim Simplexverfahren, Gass[1964]):

$$
\begin{array}{llcrcrcrcl}
\max & \frac{3}{4}x_1 & - & 150x_2 & + & \frac{1}{50}x_3 & - & 6x_4 & & \\
\text{unter} & \frac{1}{4}x_1 & - & 60x_2 & - & \frac{1}{25}x_3 & + & 9x_4 & \leq & 0 \\
 & \frac{1}{2}x_1 & - & 90x_2 & - & \frac{1}{50}x_3 & + & 3x_4 & \leq & 0 \\
 & & & & & x_3 & & & \leq & 1 \\
 & & & & & \multicolumn{3}{r}{x_1, x_2, x_3, x_4} & \geq & 0
\end{array}
$$

Mit dem Simplexverfahren durchlaufen wir eine zyklische Folge von Basislösungen, deren zugehörige Darstellungen wir in Tableauform angeben:

Anfangstableau:

		x_1	x_2	x_3	x_4
z	0	$\frac{3}{4}$	-150	$\frac{1}{50}$	-6
x_5	0	$-\boxed{\frac{1}{4}}$	60	$\frac{1}{25}$	-9
x_6	0	$-\frac{1}{2}$	90	$\frac{1}{50}$	-3
x_7	1	0	0	-1	0

Tableau 1:

		x_5	x_2	x_3	x_4
z	0	-3	30	$\frac{7}{50}$	-33
x_1	0	-4	240	$\frac{4}{25}$	-36
x_6	0	2	$-\boxed{30}$	$-\frac{3}{50}$	15
x_7	1	0	0	-1	0

Tableau 2:

		x_5	x_6	x_3	x_4
z	0	-1	-1	$\frac{2}{25}$	-18
x_1	0	12	-8	$-\boxed{\frac{8}{25}}$	84
x_2	0	$\frac{1}{15}$	$-\frac{1}{30}$	$-\frac{1}{500}$	$\frac{1}{2}$
x_7	1	0	0	-1	0

Tableau 3:

		x_5	x_6	x_1	x_4
	0	2	-3	$-\frac{1}{4}$	3
x_3	0	$\frac{75}{2}$	-25	$-\frac{25}{8}$	$\frac{525}{2}$
x_2	0	$-\frac{1}{120}$	$\frac{1}{60}$	$\frac{1}{160}$	$-\boxed{\frac{1}{40}}$
x_7	1	$-\frac{75}{2}$	25	$\frac{25}{8}$	$-\frac{525}{2}$

Tableau 4:

		x_5	x_6	x_1	x_2
	0	1	-1	$\frac{1}{2}$	-120
x_3	0	$\boxed{-50}$	150	$\frac{125}{2}$	-10500
x_4	0	$-\frac{1}{3}$	$\frac{2}{3}$	$\frac{1}{4}$	-40
x_7	1	50	-150	$-\frac{125}{2}$	10500

Tableau 5:

		x_3	x_6	x_1	x_2
	0	$-\frac{1}{50}$	2	$\frac{7}{4}$	-330
x_5	0	$-\frac{1}{50}$	3	$\frac{5}{4}$	-210
x_4	0	$\frac{1}{150}$	$\boxed{-\frac{1}{3}}$	$-\frac{1}{6}$	30
x_7	1	-1	0	0	0

Tableau 6:

		x_3	x_4	x_1	x_2
	0	$\frac{1}{50}$	-6	$\frac{3}{4}$	-150
x_5	0	$\frac{1}{25}$	-9	$-\frac{1}{4}$	60
x_6	0	$\frac{1}{50}$	-3	$-\frac{1}{2}$	90
x_7	1	-1	0	0	0

In Tableau 6 erhält man dieselbe Basislösung wie im Ausgangstableau; daher ist das Simplexverfahren für dieses Beispiel nicht endlich und kreist auf einer Ecke!

Um die Endlichkeit des Verfahrens zu erzwingen, müssen wir Zusatzregeln finden, die das Kreisen unmöglich machen. Dazu betrachten wir noch einmal eine Pivottransformation, und zwar für die Zeilen unserer Darstellung zur Basis B, oBdA. $B = (1, \dots, m), N = (m+1, \dots, m+n)$:

$$\begin{aligned} c_0 &= z - c_N^T x_N, \\ b &= x_B + A_N x_N. \end{aligned}$$

Wir fassen deren Koeffizienten zu Zeilenvektoren $h_0, h_1, \dots, h_m \in K^{2+m+n}$ und diese zu einem langen Tableau H zusammen. H enthält im wesentlichen die gleichen Informationen wir die gewohnten Tableaus (teilweise mit umgekehrtem Vorzeichen). Beim Start hat H das folgende Aussehen:

c_0	1	$0\dots0$	c_N^T
b	0	E	A_N

Der Pivotschritt mit dem Element $-t_{r,s-m-2} = h_{r,s} > 0$ wird dann mit den folgenden Zeilenoperationen beschrieben:

$$\bar{h}_r := \frac{1}{h_{rs}} h_r; \quad \bar{h}_i := h_i - h_{is}\bar{h}_r, \qquad \text{für } i = 0, \dots, m, i \neq r. \tag{4.1}$$

Wir ersetzen nun die Monotonie des Zielfunktionswertes (die man offenbar nicht erreichen kann) durch eine noch zu definierende „Monotonie" der Zielfunktionszeile. Dazu definieren wir eine Ordnung von Vektoren und untersuchen, ob sich die Pivotelemente so wählen lassen, daß die Zielfunktionszeile bezüglich dieser Ordnung streng monoton fällt.

Lexikographische Ordnung von Vektoren

1. Ein Vektor v, $0 \neq v \in K^k$, heißt *lexikographisch positiv* ($v \succ 0$), falls seine erste nichtverschwindende Komponente positiv ist, z.B. ist $(0, 0, 1, -5)^T \succ 0$.

2. Wir nennen u *lexikographisch größer* als v ($u \succ v$), falls $u - v \succ 0$. Man kann auf diese Weise alle Vektoren anordnen. Die lexikographische Ordnung $\succeq$ ist durch $u \succeq v :\Leftrightarrow u = v \vee u - v \succ 0$ definiert.

$(K^k, +, \succ)$ ist eine angeordnete abelsche Gruppe, d.h. die Addition ist mit der lexikographischen Ordnung verträglich:

$$v \succeq w \Rightarrow u + v \succeq u + w$$

für alle $u \in K^k$. Darüberhinaus ist K^k sogar ein angeordneter Vektorraum, denn

$$u \succ v \Rightarrow \alpha u \succ \alpha w$$

für alle $\alpha \in K, \alpha > 0$.

Die Zeilenvektoren $h_i, i > 0$ unserer Anfangsdarstellung (4.1) sind, wenn wir wieder $b \geq 0$ voraussetzen, wegen der enthaltenen Einheitsvektoren offenbar lexikographisch positiv und linear unabhängig.

Die lexikographische Zeilenauswahlregel

Die Pivotspalte s wird unverändert bestimmt. Die Wahl der Pivotzeile r erfolgt jedoch durch die neue aufwendigere Regel

$$\frac{1}{h_{rs}} h_r := \operatorname{lexmin}\{\frac{1}{h_{is}} h_i \mid i = 1, \dots, m, h_{is} > 0\} \tag{4.2}$$

Dadurch wird r eindeutig fixiert, denn die lineare Unabhängigkeit der Zeilen verbietet $\frac{1}{h_{is}} h_i = \frac{1}{h_{js}} h_j$ für $i \neq j$. Die Regel verfeinert offenbar die frühere Zeilenauswahlregel.

Lemma 4.1 *Für das Simplexverfahren mit lexikographischer Zeilenauswahlregel (4.2) gilt:*

1. *h_0 nimmt streng lexikographisch ab,*
2. *h_i ist stets für alle $i = 1, \dots, m$ lexikographisch positiv.*

Beweis. Zu Beginn ist die 2. Bedingung erfüllt. Ist dies vor einer Iteration richtig, d.h. $h_i \succ 0$, für alle $i = 1, \dots, m$, so zeigen wir, daß dies auch für die neuen Zeilen $\bar{h}_i, i = 1, \dots, m$, in (4.1) gilt.

Für $i = r$ folgt $\bar{h}_r \succ 0$ aus $h_{rs} > 0$. Für $i \neq r$ mit $h_{is} \leq 0$ gilt $\bar{h}_i = h_i - h_{is}\bar{h}_r \succeq h_i \succ 0$. Für $i \neq r$ mit $h_{is} > 0$ folgt aus der lexikographischen Zeilenauswahlregel $\frac{1}{h_{is}} h_i \succ \frac{1}{h_{rs}} h_r$. Also ist $h_i \succ \frac{h_{is}}{h_{rs}} h_r = h_{is}\bar{h}_r$, woraus die Behauptung folgt.

Aus $\bar{h}_r \succ 0$, $h_{0s} < 0$ folgt $\bar{h}_0 = h_0 - h_{0s}\bar{h}_r \succ h_0$. □

Satz 4.2 *Das Simplexverfahren mit lexikographischer Zeilenauswahlregel (4.2) ist endlich.*

Beweis. Folgt unmittelbar aus Lemma 4.1. □

Beispiel (Lexikographische Zeilenauswahlregel):

Verwendet man beim Beispiel von Gass die lexikographische Zusatzregel, so erhält man nach 5 Iterationen die Optimallösung. Da in diesem Beispiel bereits die Zeilen von A_N lexikographisch positiv sind, können wir die Schlupfvariablen, wie z.B. in Standardform üblich, hinten anfügen. Die Zeilen $h_i, i = 0, \dots, m$ von

$$\begin{aligned} c_0 &= \quad z \quad -c_N^T x_N, \\ b &= \qquad\quad +A_N x_N + x_B \end{aligned}$$

fassen wir zu einem Tableau H zusammen:

Anfangstableau H:

	z	x_1	x_2	x_3	x_4	x_5	x_6	x_7
0	1	$\frac{3}{4}$	-150	$\frac{1}{50}$	-6	0	0	0
0	0	$\boxed{\frac{1}{4}}$	-60	$-\frac{1}{25}$	9	1	0	0
0	0	$\frac{1}{2}$	-90	$-\frac{1}{50}$	3	0	1	0
1	0	0	0	1	0	0	0	1

Tableau 1:

	z	x_1	x_2	x_3	x_4	x_5	x_6	x_7
0	1	0	30	$\frac{7}{50}$	-33	-3	0	0
0	0	1	-240	$-\frac{4}{25}$	36	4	0	0
0	0	0	$\boxed{30}$	$\frac{3}{50}$	-15	-2	1	0
1	0	0	0	1	0	0	0	1

Tableau 2: Pivotwahl durch lexikographische Regel geändert!

	z	x_1	x_2	x_3	x_4	x_5	x_6	x_7
0	1	0	0	$\frac{2}{25}$	-18	-1	-1	0
0	0	1	0	$\frac{8}{25}$	-84	-12	8	0
0	0	0	1	$\boxed{\frac{1}{500}}$	$-\frac{1}{2}$	$-\frac{1}{15}$	$\frac{1}{30}$	0
1	0	0	0	1	0	0	0	1

Tableau 3:

	z	x_1	x_2	x_3	x_4	x_5	x_6	x_7
0	1	0	-40	0	2	$\frac{5}{3}$	$-\frac{7}{3}$	0
0	0	1	-160	0	-4	$-\frac{4}{3}$	$\frac{8}{3}$	0
0	0	0	500	1	-250	$-\frac{100}{3}$	$\frac{50}{3}$	0
1	0	0	-500	0	$\boxed{250}$	$\frac{100}{3}$	$-\frac{50}{3}$	1

Tableau 4: Hier wird die Ecke verlassen und z wächst.

	z	x_1	x_2	x_3	x_4	x_5	x_6	x_7
$-\frac{1}{125}$	1	0	-36	0	0	$\frac{7}{5}$	$-\frac{4}{5}$	$-\frac{1}{125}$
$\frac{2}{125}$	0	1	-168	0	0	$-\frac{4}{5}$	$\frac{12}{5}$	$\frac{2}{125}$
1	0	0	0	1	0	0	0	1
$\frac{1}{250}$	0	0	-2	0	1	$\boxed{\frac{2}{15}}$	$-\frac{1}{15}$	$\frac{1}{250}$

Tableau 5:

	z	x_1	x_2	x_3	x_4	x_5	x_6	x_7
$-\frac{1}{20}$	1	0	-15	0	$-\frac{21}{2}$	0	$-\frac{3}{2}$	$-\frac{1}{20}$
$\frac{1}{25}$	0	1	-180	0	6	0	2	$\frac{1}{25}$
1	0	0	0	1	0	0	0	1
$\frac{3}{100}$	0	0	-15	0	$\frac{15}{2}$	1	$-\frac{1}{2}$	$\frac{3}{100}$

Geometrische Interpretation (ϵ-Störung)

In der Darstellung zur Startbasis stören wir $b_i, i = 1, \ldots, m$ entsprechend der Zeilen h_i des langen Tableaus H:

$$b_i \mapsto b_i(\epsilon) := b_i + \epsilon^{i+1} + a_{i1}\epsilon^{m+2} + \ldots + a_{in}\epsilon^{m+n+1} = \sum_{j=0}^{m+n} h_{ij}\epsilon^j.$$

Für genügend kleines ϵ ist $b_i(\epsilon) > 0$ genau dann, wenn $h_i \succ 0$. Die Beziehung bleibt auch nach einem Pivotschritt gültig, da die neuen Zeilen $\bar{H}$ Linearkombination der Zeilen von H sind, d.h.

$$\bar{b}_i(\epsilon) = \sum_{j=0}^{m+n} \bar{h}_{ij}\epsilon^j.$$

Da bei der lexikographischen Methode nur endlich viele zulässige Basislösungen durchlaufen werden und die Zeilen h_i der zugehörigen langen Tableaus lexikographisch positiv sind, kann man ϵ klein genug wählen, so daß $\bar{b}_i(\epsilon), i = 1, \ldots, m$ für alle zulässigen Basislösungen positiv ist, d.h. das gestörte Problem besitzt keine entartete Ecke. Die lexikographische Zeilenauswahlregel für das ungestörte Problem liefert dann die gleiche Folge von Pivotschritten wie die ursprüngliche Regel für das gestörte Problem.

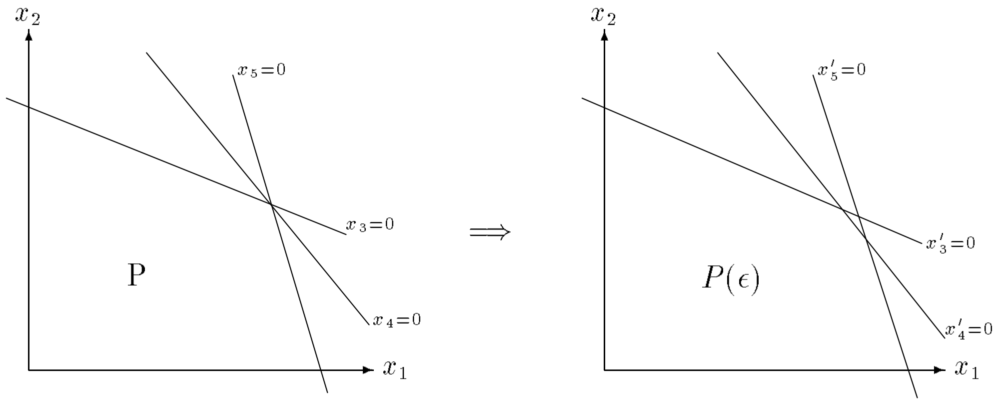


Abbildung I.12: Geometrische Interpretation der lexikographischen Zeilenauswahlregel

Aufgrund der Endlichkeit des Simplexverfahrens mit lexikographischer Zeilenauswahlregel liefern die drei Bedingungen, unter denen das Simplexverfahren abbrechen kann, drei grundlegende Alternativen, von denen für jedes (LP) genau eine erfüllt ist.

Blands Regel

Alternativ kann die Endlichkeit auch durch die folgende Regel von Bland gewährleistet werden:

Wähle die Pivotspalte s durch

$$N(s) := \min\{N(j) \mid t_{0j} > 0\}$$

und dann die Pivotzeile r durch

$$B(r) := \min\{B(i) \mid \tfrac{t_{i0}}{-t_{is}} = \min_{\substack{t_{\mu s}<0 \\ 1\le\mu\le m}} \tfrac{t_{\mu 0}}{-t_{\mu s}}\}.$$

Satz 4.3 *Das Simplexverfahren mit Blands Regel ist endlich.*

Beweis: Annahme: Das SV kreise auf einer Ecke mit den Basen $B_0, B_1, \ldots, B_0$. T sei die Indexmenge der Variablen x_j, die in die Basen des Zyklus eintreten ($j \in T \Rightarrow x_j = 0$ im Zyklus).

$q := \max T$. OBdA verlasse x_q beim Basiswechsel von $B = B_0$ nach B_1 die Basis. $x_{N(s)}$ trete dafür ein (also $B(r) = q \in T$, $N(s) = p \in T$), $p < q$. Beim Wechsel von $\bar{B} = B_\mu$ nach $B_{\mu+1}$ trete x_q wieder in die Basis ein.

Da auf der Ecke der Zielfunktionswert konstant ist, gilt $t_{00} = \bar{t}_{00}$ in:

$$\left.\begin{array}{ll} (1) & z = t_{00} + \sum t_{0j} x_{N(j)} \\ (2) & x_{B(i)} = t_{i0} + \sum t_{ij} x_{N(j)} \end{array}\right\} \quad \text{Basis } B,\ t_{0s} > 0$$

$$\left.\begin{array}{ll} (3) & z = \bar{t}_{00} + \sum \bar{t}_{0j} x_{\bar{N}(j)} \\ (4) & x_{\bar{B}(i)} = \bar{t}_{i0} + \sum \bar{t}_{ij} x_{\bar{N}(j)} \end{array}\right\} \quad \text{Basis } \bar{B},\ \bar{t}_{0\varrho} > 0,\ \bar{N}(\varrho) = q$$

Aus (1) und (3) folgt $t_{00} + c^T x = t_{00} + \bar{c}^T x$, wobei $c_B \equiv 0$, $\bar{c}_{\bar{B}} \equiv 0$.
Für $x_{N\setminus p} \equiv 0$ ergibt sich daraus und aus (2)

$$c_p x_p = \bar{c}_p x_p + \sum_{i=1}^m \bar{c}_{B(i)} (t_{i0} + t_{is} x_p) \text{ für alle } x_p \in \boldsymbol{R}.$$

Somit folgt

$$\left(c_p - \bar{c}_p - \sum_{i=1}^m \bar{c}_{B(i)} t_{is}\right) x_p = \sum_{i=1}^m \bar{c}_{B(i)} t_{i0} \text{ für alle } x_p \in \boldsymbol{R}.$$

Diese Gleichung kann nur dann für alle $x_p \in \boldsymbol{R}$ gültig sein, wenn beide Seiten verschwinden, es gilt daher $c_p - \bar{c}_p - \sum \bar{c}_{B(i)} t_{is} = 0$ und $\sum \bar{c}_{B(i)} t_{i0} = 0$.

Nun ist $c_p = t_{0s} > 0$ sowie $\bar{c}_p \le 0$, da zwar $p < q$, aber x_p nicht in die Basis eintritt. Es gilt daher

$$\sum_{i=1}^m \bar{c}_{B(i)} t_{is} = c_p - \bar{c}_p > 0.$$

Daher ist $\bar{c}_{B(k)} t_{ks} > 0$ für mindestens ein k mit $1 \le k \le m$. Insbesondere ist $\bar{c}_{B(k)} \ne 0$, und damit muß $B(k) \in \bar{N}$ sein. Es folgt $B(k) \in T$, $B(k) \le q$, $x_{B(k)} = 0$.

Andererseits: $\bar{c}_q = \bar{t}_{0\varrho} > 0$ (da x_q in der entsprechenden Iteration in die Basis eintritt) und $t_{rs} < 0$ ($x_{B(r)}$ verläßt die Basis), und somit gilt $\bar{c}_{B(r)} t_{rs} < 0$, woraus $r \ne k$, $B(k) < q$ folgt.

Da $x_{B(k)}$ nicht in die Basis eintritt, gilt $\bar{c}_{B(k)} \le 0$, also $t_{ks} < 0$. Dann ist $x_{B(k)}$ aber Kandidat, um B zu verlassen im Widerspruch zur Wahl von x_q (da $B(k) < q$)! $\square$

Satz 4.4 *Für das LP* $\max\{c^T x \mid Ax \leq b, x \geq 0\}$ *gilt eine der folgenden strengen Alternativen:*

1. $\{x \mid Ax \leq b, x \geq 0\} = \emptyset$,
2. $\{c^T x \mid Ax \leq b, x \geq 0\}$ *ist nach oben unbeschränkt,*
3. *Es gibt eine optimale Basis mit* $\tilde{c}_N \leq 0$.

Bemerkung: In der Praxis wird die Endlichkeit des Verfahrens durch Störungen bei der Wahl der Pivotzeile erzwungen, falls die Iterationen lange Zeit nicht von einer Ecke wegführen.

5 Der Aufwand des Simplexverfahrens

Die Anzahl der Iterationen des Simplexverfahrens ist abhängig von der Wahl der jeweiligen Pivotspalte. Zur Auswahl steht jede Nichtbasisvariable mit positiven reduzierten Kosten.

Steilster Anstieg in x_N

Dantzig hat ursprünglich vorgeschlagen, die Kante zu wählen, längs der die Zielfunktion im Raum der Nichtbasisvariablen am schnellsten wächst:

$$\tilde{c}_s := \max_{j \in N} \tilde{c}_j \tag{5.1}$$

Da die Nichtpositivität als Optimalitätskriterium auf die eine oder andere Weise überprüft werden muß, erfordert diese Regel nur wenig zusätzlichen Aufwand. Für Probleme mit sehr vielen Variablen kann man diesen Aufwand leicht weiter reduzieren, wenn man in jedem Schritt nur gewisse Teilmengen der Nichtbasisvariablen untersucht, die zyklisch alle Nichtbasisvariablen überdecken.

Größter absoluter Zuwachs

Die Wahl der Kante des steilsten Anstiegs führt wegen der unterschiedlichen Längen dieser Kanten nicht notwendigerweise zu der Nachbarecke mit dem größten absoluten Zuwachs in der Zielfunktion. Eine solche Nachbarecke kann man ermitteln, indem man alle möglichen Pivotschritte vergleicht. Sei $\tilde{c}_j > 0$ mit Pivotzeile $r(j)$, Pivotelement $\tilde{a}_{r(j)j}$. Dann wählt man s mit

$$\frac{\tilde{c}_s \tilde{b}_{r(s)}}{\tilde{a}_{r(s)s}} := \max\left\{\frac{\tilde{c}_j \tilde{b}_{r(j)}}{\tilde{a}_{r(j)j}} \mid \tilde{c}_j > 0, j \in N\right\} \tag{5.2}$$

Diese Regel liefert lokal in jedem Schritt die optimale Wahl, ist dafür aber erheblich aufwendiger als die Regel des steilsten Anstiegs.

Steilster Anstieg

Die Regel des steilsten Anstieg im Raum der Nichtbasisvariablen vernachlässigt die Änderungen der Basisvariablen. Der Anstieg längs der Kantenrichtung zu $x_j, j \in N$, im Raum aller Variablen ist gegeben durch die entsprechende Richtungsableitung der linearen Zielfunktion. Wächst x_j um Δx_j, so ändern sich Zielfunktion und Variable um $\Delta z = \tilde{c}_j \Delta x_j$, $\Delta x_{B(i)} = -\tilde{a}_{ij} \Delta x_j$, für alle $i = 1, \ldots, m$, und $\Delta x_\mu = \delta_{\mu j} \Delta x_j$, $\mu \in N$. Die Richtungsableitung zu $x_j, j \in N$ ist daher:

$$\frac{\Delta z}{\| \Delta x \|} = \frac{\tilde{c}_j}{\sqrt{1 + \tilde{A}_j^T \tilde{A}_j}}.$$

Die Wahl von s nach steilstem Anstieg ergibt sich dann aus:

$$\frac{\tilde{c}_s^2}{1 + \tilde{A}_s^T \tilde{A}_s} := \max \left\{ \frac{\tilde{c}_j^2}{1 + \tilde{A}_j^T \tilde{A}_j} \mid \tilde{c}_j > 0, j \in N \right\} \tag{5.3}$$

Der zusätzliche Aufwand bei dieser Regel liegt im wesentlichen bei der Berechnung der Skalarprodukte.

Kuhn und Quandt haben 1963 die mittlere Iterationenanzahl für die unterschiedlichen Regeln numerisch untersucht:

Pivotspaltenauswahlregel	(5.1)	(5.2)	(5.3)
mittlere Iterationenanzahl	34,6	22,5	18,6

Trotzdem ergibt sich für die Regel (5.1) insgesamt der kleinste Rechenaufwand. Damit die kleinere Iterationenanzahl der Regel (5.3) durchschlägt, muß der Aufwand pro Iteration reduziert werden.

Hilfsgrößen für (5.3)

Die Skalierung der einzelnen Richtungen erfolgt mit Faktoren χ_j, die wir sowohl in revidierter Form als auch mit Hilfe der Darstellungskoeffizienten berechnen können:

$$\chi_j = 1 + \| A_B^{-1} A_{N(j)} \|^2 = 1 + \sum_{i=1}^{m} t_{ij}^2, \tag{5.4}$$

für $j = 1, \ldots, n$. Goldfarb und Reid haben dafür 1977 rekursive Formeln entwickelt.

Lemma 5.1 *Sei t_{rs} das Pivotelement und bezeichne $\bar{\chi}_j := 1 + \sum_{i=1}^m \bar{t}_{ij}^2$ die neuen Faktoren, ausgedrückt durch die neuen Darstellungskoeffizienten. Dann gilt*

$$\begin{aligned} \bar{\chi}_s &:= \left(\tfrac{1}{t_{rs}}\right)^2 \chi_s \\ \bar{\chi}_j &:= \left(\tfrac{t_{rj}}{t_{rs}}\right)^2 \chi_s + \chi_j - 2 \left(\tfrac{t_{rj}}{t_{rs}}\right) \textstyle\sum_{i=1}^m t_{ij} t_{is} \text{ für alle } j \neq s \end{aligned} \tag{5.5}$$

Beweis.

$$
\begin{aligned}
\bar{\chi}_s &= 1+\sum \bar{t}_{is}^2 = 1+\sum_{i\neq r}\left(\frac{t_{is}}{t_{rs}}\right)^2+\left(\frac{1}{t_{rs}}\right)^2 \\
&= \left(\frac{1}{t_{rs}}\right)^2\left[t_{rs}^2+\sum_{i\neq r} t_{is}^2+1\right] \\
\bar{\chi}_j &= 1+\sum_{i=1}^{m}\bar{t}_{ij}^2 = 1+\sum_{i\neq r}\left(t_{ij}-\frac{t_{is}}{t_{rs}}t_{rj}\right)^2+\left(\frac{t_{rj}}{-t_{rs}}\right)^2 \\
&= 1+\sum_{i\neq r}\left[t_{ij}^2+\left(\frac{t_{is}}{t_{rs}}t_{rj}\right)^2-2\frac{t_{rj}}{t_{rs}}t_{ij}t_{is}\right]+\left(\frac{t_{rj}}{t_{rs}}\right)^2 \\
&= \chi_j - t_{rj}^2+\left(\frac{t_{rj}}{t_{rs}}\right)^2\chi_s - t_{rj}^2 - 2\frac{t_{rj}}{t_{rs}}\sum t_{ij}t_{is}+2\frac{t_{rj}}{t_{rs}}t_{rj}t_{rs}\square
\end{aligned}
$$

Bemerkung: Wenn die Darstellungskoeffizienten t_{ij} bekannt sind, kann man zwar die Faktoren mittels der Rekursion (5.5) berechnen, gewinnt aber nichts gegenüber der direkten Berechnung (5.4). Im Gegenteil, wegen der Möglichkeit, die Skalarprodukte nur für die $j \in N$ mit $c_j > 0$ auszuwerten, ist die direkte Berechnung vorzuziehen.

In revidierten Verfahren lohnt sich dagegen die Auswertung der Rekursionsformeln. Bei Kenntnis von A_B^{-1} oder einer entsprechenden Zerlegung der Basismatrix kann man $A_B^{-1}A_{N(s)}$ leicht berechnen. Die Summen ergeben sich nun mit Hilfe von

$$
\begin{aligned}
\sum_{i=1}^{m} t_{ij}t_{is} &= (A_B^{-1}A_{N(j)})^T A_B^{-1}A_{N(s)} \\
&= A_{N(j)}^T(A_B^{-T}A_B^{-1}A_{N(s)}) \\
&=: A_{N(j)}^T w.
\end{aligned}
$$

Hier ist nur einmal w zu berechnen, während man bei direkter Berechnung alle Darstellungskoeffizienten in Spalten zu positiven reduzierten Kostenkoeffizienten berechnen muß. Die auftretenden Rundungsfehler beeinflussen zwar die Wahl der Pivotspalte, aber nicht die Basislösung. Trotzdem ist es empfehlenswert, von Zeit zu Zeit eine Neuberechnung der Faktoren aus (5.5) durchzuführen. Für Probleme der Größenordnung (10000 Nichtnullelemente, $m = 800$, $n = 1000$) berichten Goldfarb und Reid über folgende Erfahrungen:

	(5.1)	(5.3)
Iterationenanzahl	3976	1182

Dabei liegt die mittlere Rechenzeit pro Iteration bei (5.3) nur um den Faktor 1.47 über der mittleren Rechenzeit pro Iteration bei (5.1), so daß die Regel des steilsten Anstiegs im Raum aller Variablen vorzuziehen ist. Für weitere Einzelheiten sei auf die Arbeit von Goldfarb und Reid (*Mathematical Programming* 12 (1977) 361ff.) verwiesen.

Aufwand von Algorithmen (Zeitkomplexität)

Der Laufzeit eines Algorithmus auf einem Computer zur Lösung eines mathematisch modellierten Problems hängt in der Regel von der Größe des jeweiligen Problems ab. Als Maß für die Zeitkomplexität wird die Anzahl notwendiger elementarer Operationen in Abhängigkeit der Problemgröße herangezogen. Definition von Problemgröße und elementaren Operationen variieren je nach Maschinenmodell.

Modell	Problemgröße	elementare Operationen
arithmetisch	Anzahl Eingabedaten	$+, -, \cdot, /, \leq$
binär	Länge $< \dots >$ der binären Codierung der Eingabedaten	Bitoperationen

In der theoretischen Behandlung muß die Größe von rationalen Zahlen, Vektoren und Matrizen klar definiert sein. Die Größe $< z >$ eine ganzen Zahl z ist

$$< z > := 1 + \lceil \log_2 |z| + 1 \rceil \, .$$

Die Größe einer rationalen Zahl $r = \frac{p}{q}$ mit teilerfremden, ganzen p, q ist $< r > := < p > + < q >$. Die Größe von Vektoren und Matrizen ist die Summe der Größen der jeweiligen Koeffizienten, d.h.

$$< A > := \sum_{ij} < a_{ij} >$$

für eine Matrix $A \in \boldsymbol{Q}^{nm}$.

Für ein Lineares Programm ergibt sich also die Größe $L =< A > + < b > + < c >$.

Aufwand des Simplexverfahrens

Für Aufwandsbetrachtungen wird in der Regel die O-Notation verwendet. Es bedeutet

$$f(x) = O(g(x)) :\Leftrightarrow \bigvee_{c>0} \bigvee_{n_0 \in \boldsymbol{N}} \bigwedge_{n \geq n_0} |f(x)| \leq c \cdot g(x).$$

Im arithmetischen Modell ist die Größe allein durch die Anzahl der Eingabedaten bestimmt, also bei $(m+1)(n+1)$ Eingabedaten ist die Problemgröße $O(mn)$. Für eine Iteration des Simplexverfahrens sind ebenfalls $O(mn)$ elementare Operationen notwendig.

Im binären Modell ist die Größe L in jeder Iteration gegeben durch

$$L =< \tilde{A} > + < \tilde{b} > + < \tilde{c} > \, .$$

Der Aufwand einer Iteration des Simplexverfahrens ist $O(mnL)$ Bitoperationen. In beiden Modellen ist der Iterationsaufwand polynomial in der Größe der Eingabedaten.

Die Anzahl der Iterationen ist viel schwieriger abzuschätzen. In empirischen Untersuchungen mit $n = O(m)$ wird die Anzahl der Iterationen meist mit $O(m)$ angegeben.

Theoretische Aussagen über die mittlere Anzahl von Iterationen (average case performance) sind schwierig und nur unter einschränkenden unbefriedigenden Annahmen über die Verteilung der Problemdaten abzuleiten. Für spezielle Varianten des Simplexverfahrens hat Borgwardt 1987 die Schranke $O(m^4 n^{\frac{1}{m-1}})$ angeben können. Adler, Meggido und Karp leiten unter anderen Annahmen die Schranke $O(\min\{m^2, n^2\})$ ab.

Schranken für den theoretisch ungünstigsten Fall (worst case performance) sind vielfältig aber mit negativem Ergebnis untersucht worden. Klee und Minty geben 1972 Beispiele mit $2^n - 1$ Iterationen, in Bezug auf die Länge der Eingabedaten also mit exponentiellem Aufwand. Schon für $n = 50$ auf einem Computer, der 100 Iterationen pro Sekunde schafft, mußte man 350000 Jahre warten, bis die optimale Ecke gefunden ist. Bis heute ist keine Variante des Simplexverfahrens mit garantiert polynomialem Aufwand bekannt. Dies steht natürlich in krassem Gegensatz zum praktischen Erfolg der Methode, der sich in den empirischen Untersuchungen bestens widerspiegelt.

Bis 1979 war nicht bekannt, ob es überhaupt einen polynomialen Algorithmus zur Lösung linearer Optimierungsprobleme gibt. In einer Komplexitätsuntersuchung der Ellipsoidmethode wies Khachian 1979 die Schranke $O(n^4 L^2)$ für die Anzahl der Bitoperationen und damit die polynomiale Lösbarkeit des linearen Optimierungsproblems nach. Jahrelang wurde vergeblich versucht, auf dieser Methode ein praktikables numerisches Verfahren zur Lösung linearer Optimierungsprobleme aufzubauen.

1984 beschrieb Karmarkar die projektive Methode, wies die Schranke $O(n^{3.5} L^2)$ nach und behauptete provokativ, diese Methode wäre für praktische Probleme um den Faktor 100 schneller als das Simplexverfahren. Die resultierende kontroverse Diskussion löste eine Flut von Veröffentlichungen über Innere-Punkte Methoden aus, die inzwischen tatsächlich zu praktikablen Alternativen zum Simplexverfahren geführt hat. Auch wenn die Entwicklung hier noch lange nicht abgeschlossen ist, kann man wohl feststellen, daß sehr effektive Varianten beider Methoden zur Lösung linearer Optimierungsaufgaben zur Verfügung stehen. Welches Verfahren bei welcher Struktur der Restriktionen in großen praktischen Anwendungen besonders erfolgreich sein wird, ist erst in Ansätzen bekannt.

6 Alternativ- und Dualitätssätze

In diesem Abschnitt geht es um eine der Theorie linearer Ungleichungen grundliegende Symmetrie. Dieses äußert sich sowohl in den Alternativsätzen über lineare Systeme als auch in den Dualitätssätzen der Linearen Optimierung. Beginnen wir mit einem typischen Alternativsatz, wie er bereits für Gleichungen in der Linearen Algebra diskutiert werden kann.

Satz 6.1 *Es gilt genau eine der folgenden beiden Alternativen:*

1. $\exists x \in K^n : Ax = b,$
2. $\exists y \in K^m : y^T A = 0, y^T b = 1.$

Beweis. Wenn wir annehmen, beide Systeme hätten Lösungen x, y, dann folgt der Widerspruch $0 = y^T Ax = y^T b = 1$.

Wir nehmen nun an, daß das erste System nicht lösbar ist. Dies gilt genau dann, wenn b linear unabhängig von den Spalten von A ist, d.h. wenn $\mathrm{rg}[A \mid b] = \mathrm{rg}A + 1$. Dann gilt auch

$$\mathrm{rg}\begin{bmatrix} A & b \\ 0 & 1 \end{bmatrix} = \mathrm{rg}[A \mid b].$$

Dies ist äquivalent zur linearen Abhängigkeit der letzten Zeile (01) von den Zeilen der Matrix (Ab), d.h. $y^TA = 0, y^Tb = 1$ besitzt eine Lösung. □

Satz 6.2 *Es gilt genau eine der folgenden beiden Alternativen:*

1. $\exists x \in K^n : Ax = b, x \geq 0,$
2. $\exists y \in K^m : y^TA \geq 0, y^Tb < 0.$

Beweis. Wenn wir annehmen, beide Systeme hätten Lösungen x, y, dann folgt der Widerspruch $0 \leq y^TAx = y^Tb < 0$.

Wir nehmen nun an, daß das erste System nicht lösbar ist und betrachten das entsprechende beschränkte, nichtleere Hilfsproblem zur Bestimmung einer solchen Lösung (wobei OBdA $b \geq 0$):

$$\gamma := \max\{\underbrace{-1^Tu}_{=:\, d^Tz} \mid \underbrace{Ax + u}_{=:\, Hz} = b, \underbrace{x \geq 0, u \geq 0}_{z \geq 0}\},$$

Nach Satz 4.4 existiert eine endliche optimale Basislösung B mit $\gamma = d_B^Tz_B < 0$. Sei $y^T := d_B^TH_B^{-1}$. Dann folgt

$$0 \geq d^T - d_B^TH_B^{-1}H = [0^T, -1^T] - y^TH,$$

also $y^TA \geq 0$. Außerdem gilt $0 > \gamma = d_B^TH_B^{-1}b = y^Tb$, d.h. y ist Lösung des zweiten Systems . □

Endlich erzeugte Kegel

Zur geometrischen Interpretation dieses Satzes betrachten wir endlich erzeugte Kegel. Die Vektoren $A_1, A_2, \ldots, A_n \in K^m$ erzeugen den Kegel

$$\mathrm{cone}(A_1, \ldots, A_n) := \{Ax \mid x \geq 0\} =: K(A)$$

wobei $A := [A_1 \mid \ldots \mid A_n]$.

Satz 6.2 zeigt, daß entweder $b \in K(A)$ oder eine Hyperebene $H = \{x \mid y^Tx = 0\}$ trennt b strikt von $K(A)$, d.h. $y^Tb < 0 \leq y^Tz$ für alle $z \in K(A)$, siehe Abbildung I.13. Der Fall $b \in K(A)$ läßt eine weitere Interpretation zu, die auf Farkas zurückgeht und unmittelbar aus Satz 6.2 folgt:

Satz 6.3 (Lemma von Farkas, 1894) *Die folgenden beiden Aussagen sind äquivalent:*

1. $\exists x \in K^n : Ax = b, x \geq 0,$

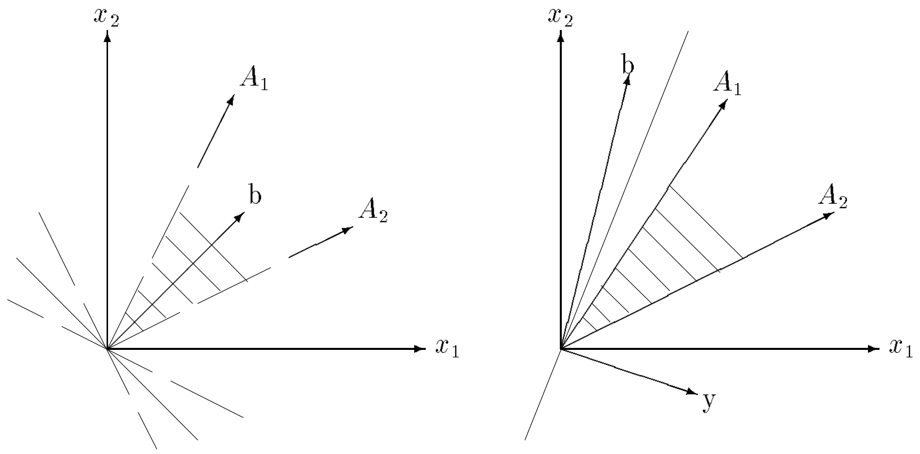


Abbildung I.13: Entweder $b \in K(A)$ oder genau $\exists y : y^T b < 0 < y^T K(A)$

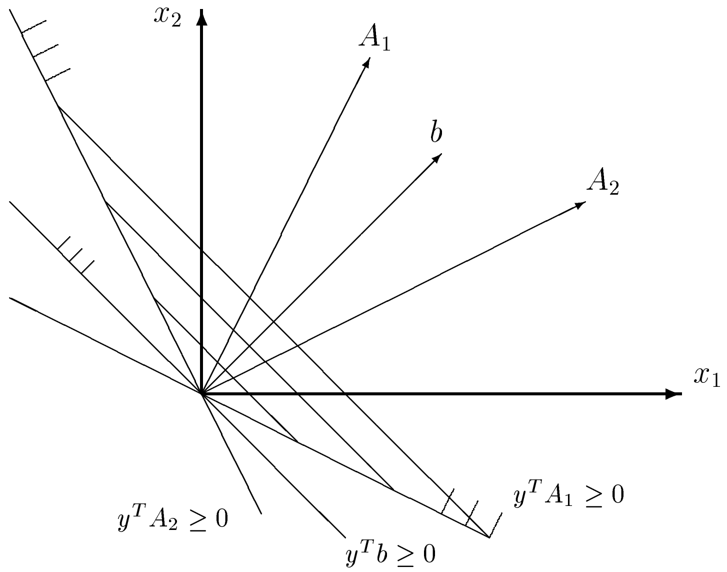


Abbildung I.14: Zum Lemma von Farkas

2. $\forall y \in K^m : (y^T A \geq 0 \Rightarrow y^T b \geq 0)$.

Danach ist das System $Ax = b, x \geq 0$ genau dann lösbar, wenn der Kegel $\{y \mid y^T A \geq 0\}$ vollständig im Halbraum $\{y \mid y^T b \geq 0\}$ enthalten ist (siehe Abbildung I.14).

Alternativsätze liefern Aussagen über die Existenz von Lösungen von Ungleichungssystemen. Eine solche Existenzfrage wollen wir kurz diskutieren.

Wunschtraum eines Politmanagers

In einem Mehrparteiensystem, etwa mit den vier Parteien in $P := \{S, R, G, GR\}$, gibt es in aller Regel von Wahl zu Wahl Wählerbewegungen. Die Wanderungsquoten a_{ij}, für $i, j \in P$

lassen sich tabellarisch darstellen:

nach \ von	S	R	G	GR
S	60	12	5	20
R	10	40	30	20
G	10	15	35	25
GR	20	33	30	35

Aus der Tabelle kann man ablesen, daß z.B. 15% der Wähler der Partei R zur Partei G gewechselt haben. Selbst wenn sich diese Daten über mehrere Wahlen nicht ändern, ändern sich die Mehrheitsverhältnisse in aller Regel von Wahl zu Wahl. Der Wunsch nach stabilen Verhältnissen könnte einen Politmanager fragen lassen, ob es eine Aufteilung der Wähler und damit eine Mandatsverteilung gibt, die für alle Folgewahlen trotz der Wählerbewegungen unverändert bleibt?

Markov-Prozesse

Das zugrundeliegende mathematische Modell ist ein Markov-Prozeß. Für ein System von n möglichen Grundzuständen seien die nichtnegativen Übergangswahrscheinlichkeiten p_{ij} von j nach i bekannt und in einer Übergangsmatrix P zusammengefaßt. Da alle möglichen Grundzustände erfaßt sind, gilt $\sum_i p_{ij} = 1$ für alle Grundzustände j. Zu Beginn des Prozesses liegt mit Wahrscheinlichkeit x_j der Zustand j vor. Diese Wahrscheinlichkeiten beschreiben einen *stochastischen Zustand* x. Hier gilt wiederum $\sum x_j = 1, x_j \geq 0$. Der Prozess besteht aus einer Folge von Übergängen, wobei sich der jeweils neue Zustand y zu dem vorherigen Zustand x aus der Gleichung $y = Px$ ergibt. Offenbar folgt dann $\sum y_i = 1, y \geq 0$.

Ein Zustand x heißt *stabil*, falls $x = Px$. Wir wollen zeigen, daß jede Übergangsmatrix P einen stabilen Zustand besitzt.

Für symmetrische Matrizen P ist der Zustand $x := \frac{1}{n}$ stabil, da dann

$$(Px)_i = \frac{1}{n}\sum_j p_{ij} = \frac{1}{n}\sum_j p_{ji} = \frac{1}{n}.$$

Im allgemeinen Fall ist x ein stabiler Zustand von P genau dann, wenn x eine Lösung von $Ax = b, x \geq 0$ für

$$A = \begin{bmatrix} \\ P - E \\ \\ 1 \dots 1 \end{bmatrix}, \quad b = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}$$

ist. Falls x keinen stabilen Zustand besitzt, muß daher nach Alternativsatz 6.2 gelten

$$\exists y \in K^n : y^T A \geq 0, y^T b < 0.$$

Sei $y = \left[\begin{smallmatrix} z \\ -\lambda \end{smallmatrix}\right]$ entsprechend A partitioniert. Dann gilt $z^T(P-E) \geq \lambda 1^T, \lambda > 0$, d.h.

$$\sum_i z_i p_{ij} - z_j \geq \lambda > 0, \quad (*)$$

für alle $j = 1, \ldots, n$. Für $z_k := \max_i z_i$ gilt andererseits

$$\sum z_i p_{ik} \leq z_k \sum_i p_{ik} = z_k$$

also $\sum z_i p_{ik} - z_k \leq 0$. Dies steht im Widerspruch zu $(*)$ für $j = k$.

Somit folgt: Jede Übergangsmatrix hat einen stabilen Zustand. Also gibt es auch einen stabilen Zustand bei den Wählerbewegungen (konstante Wanderungsquoten vorausgesetzt).

Der Alternativsatz 6.2 läßt sich leicht für Kombinationen von Gleichungen und Ungleichungen verallgemeinern:

Korollar 6.4 *Genau eines der beiden folgenden Systeme ist lösbar:*

$$\text{(1)} \quad \begin{aligned} Ax + By &\leq b \\ Cx + Dy &= d \\ x &\geq 0 \end{aligned} \qquad\qquad \text{(2)} \quad \begin{aligned} u^T b + v^T d &< 0 \\ u^T A + v^T C &\geq 0^T \\ u^T B + v^T D &= 0^T \\ u &\geq 0 \end{aligned}$$

Beweis. Durch Einführung von Schlupfvariablen und durch Aufspaltung der nicht vorzeichenbeschränkten Variablen erkennt man, daß das erste System genau dann lösbar ist, wenn das System

$$\begin{aligned} Ax + By' - By'' + z &= b \\ Cx + Dy' - Dy'' &= d \\ x, y', y'', z &\geq 0 \end{aligned}$$

lösbar ist. Alternativ ist hierzu nach 6.2 die Lösbarkeit von:

$$\begin{aligned} u^T b + v^T d &< 0 \\ u^T A + v^T C &\geq 0^T \\ u^T B + v^T D &\geq 0^T \\ -u^T B - v^T D &\geq 0^T \\ u^T &\geq 0^T \end{aligned}$$

wobei die dritte und vierte Restriktion zu $u^T B + v^T D = 0^T$ äquivalent sind. □

Die transponierte Form dieses Korollars werden wir weiter unten benutzen. Obwohl man sie direkt ablesen kann, soll sie der Übersichtlichkeit halber nochmals separat formuliert werden:

Transponierte Form des Korollars

Genau eines der beiden folgenden Systeme ist lösbar:

$$\begin{array}{rrcrl} (1) & u^T A + v^T C \leq b^T & \qquad & (2) \quad b^T x + d^T y & < 0 \\ & u^T B + v^T D = d^T & & Ax + By & \geq 0 \\ & u \geq 0 & & Cx + Dy & = 0 \\ & & & x & \geq 0 \end{array}$$

Im folgenden Satz erweitern wir die Diskussion auf mehrere strikte Ungleichungen.

Satz 6.5 (Motzkin 1936, Fourier 1826, Kuhn 1956) *Genau eines der beiden folgenden Systeme ist lösbar:*

1. $Ax < b, Cx \leq d$
2. $u^T A + v^T C = 0, u \geq 0, v \geq 0$ *wobei*
   (a) $u^T b + v^T d < 0$ *oder*
   (b) $u^T b + v^T d \leq 0, u \neq 0$

Beweis. Wenn wir annehmen, beide Systeme hätten Lösungen x, u, v, so folgt ein Widerspruch aus

$$0 = (u^T A + v^T C)x = u^T Ax + v^T Cx \leq u^T b + v^T d \leq 0,$$

da wegen (b) bzw. (a) mindestens eine der beiden Ungleichungen strikt ist.

Wir nehmen nun an, daß das zweite System keine Lösung besitzt, und konstruieren eine Lösung für das erste System.

Da es keine Lösung mit (a) gibt, so ist nach Korollar 6.4

$$Ax \leq b, Cx \leq d \quad (*)$$

lösbar. Wir wählen eine Lösung $\bar{x}$. Da es auch keine Lösung mit (b) gibt, ist

$$\begin{array}{ll} u^T A + v^T C & = 0, \quad u \geq 0, v \geq 0 \\ u^T b + v^T d & \leq 0, \quad u_i > 0 \end{array}$$

unlösbar für alle $i = 1, \ldots, m$. Dann sind auch die entsprechenden neuen Systeme

$$\begin{array}{ll} u^T A + v^T C & = -a_i^T, \quad u \geq 0, v \geq 0 \\ u^T b + v^T d & \leq -b_i \end{array}$$

unlösbar für alle $i = 1, \ldots, m$. Anderenfalls liefert eine Lösung u, v des i-ten neuen Systems eine Lösung $u + E_i$, v des i-ten alten Systems. Die Alternative hierzu ergibt sich aus der transponierten Form des Korollars. Da alle Variablen nichtnegativ sind, erhalten wir keinen Gleichungsblock. Entsprechend ist

$$\begin{aligned} Ax + \lambda b &\geq 0 \\ Cx + \lambda d &\geq 0 \\ -a_i^T x - \lambda b_i &< 0 \\ \lambda &\geq 0 \end{aligned}$$

lösbar für alle $i = 1, \ldots, m$. Wir bezeichnen eine Lösung mit x^i, λ_i.

1.Fall: $\lambda_i > 0$. Dann ist $z^i := -\frac{x^i}{\lambda_i}$ eine Lösung des Systems $Az^i \leq b, Cz^i \leq d, a_i^T z^i < b_i$.

2. Fall: $\lambda_i = 0$. Mit einer Lösung $\bar{x}$ zu $(*)$ sei $z^i := \bar{x} - x^i$. Dann gilt $Az^i = A\bar{x} - Ax^i \leq b, Cz^i \leq d, a_i^T z^i < b_i$.

Nun setze $\hat{x} := \frac{1}{m} \sum_{i=1}^m z^i$. Dann folgt $A\hat{x} < b, C\hat{x} \leq d$. □

Der Satz von Motzkin impliziert einige weniger allgemeine Ergebnisse, die hier in einem Korollar zusammengefaßt werden.

Korollar 6.6

1. *(Gordan, 1873)*
   $Ax = 0, x \geq 0, x \neq 0$ ist lösbar genau dann, wenn $y^T A > 0$ unlösbar,
2. *(Stiemke, 1915)*
   $Ax = 0, x > 0$ ist lösbar genau dann, wenn $y^T A \geq 0, y^T A \neq 0$ unlösbar,
3. *(Carver, 1921/22)*
   $Ax < b$ ist lösbar genau dann, wenn $y^T A = 0, y^T b \leq 0, y \geq 0, y \neq 0$ unlösbar.

Duale lineare Programme

Zu jedem linearen Programm (LP) kann man ein zugehöriges lineares Programm (DP) definieren, das man als *duales* Programm bezeichnet. Das Ausgangsproblem (LP) heißt dann auch *primales* Problem. Da man jedes (LP) in kanonische Form bringen kann, genügt es, die Dualisierung eines (LP) in kanonischer Form zu beschreiben. Zu

$$(LP) \qquad z_P := \max\{c^T x \mid \underbrace{Ax \leq b, x \geq 0}_{=: P}\}$$

lautet das duale Problem

$$(DP) \qquad z_D := \min\{y^T b \mid \underbrace{y^T A \geq c^T, y \geq 0}_{=: D}\}.$$

Bemerkung: Das duale Problem zu (DP) ist wiederum (LP): Zunächst formulieren wir ein zu (DP) äquivalentes Problem in Standardform:

$$(LP') \qquad -z_D = \max\{(-b^T)y \mid (-A^T)y \leq -c, y \geq 0\}.$$

Dualisieren wir (LP'), so erhalten wir

$$(DP') \qquad -z_P = \min\{x^T(-c) \mid x^T(-A^T) \geq (-b)^T, x \geq 0\}.$$

(DP') ist offenbar äquivalent zu (LP). Aufgrund dieser Symmetrie folgt aus jeder Aussage über ein (LP) die analoge Aussage für (DP). Wir werden solche Aussagen als dual zueinander bezeichnen.

Satz 6.7 (Schwacher Dualitätssatz)

1. *$x \in P, y \in D \Rightarrow c^T x \leq y^T b$*
2. *$P \neq \emptyset$. Dann gilt: $c^T x$ unbeschränkt nach oben $\Leftrightarrow D = \emptyset$*
3. *$D \neq \emptyset$. Dann gilt: $y^T b$ unbeschränkt nach unten $\Leftrightarrow P = \emptyset$*

Beweis. Für $x \in P, y \in D$ gilt

$$c^T x \leq y^T A x \leq y^T b,$$

woraus 1. folgt. Da 2. und 3. duale Aussagen sind, genügt es, 3. zu zeigen.

Ist $P \neq \emptyset$, so liefert (nach 1.) jedes $x \in P$ eine untere Schranke. Anderenfalls ist $Ax \leq b, x \geq 0$ unlösbar. Nach Korollar zu 6.2 ist dann $y^T A \geq 0, y^T b < 0, y \geq 0$ lösbar, etwa durch $\hat{y}$. Sei $\bar{y} \in D$. Dann ist $\bar{y} + \lambda\hat{y} \in D$ für alle $\lambda \geq 0$. Die Werte der Zielfunktion auf diesem Halbstrahl, gegeben durch $(\bar{y} + \lambda\hat{y})^T b = \bar{y}^T b + \lambda(\hat{y}^T b)$, sind nach unten unbeschränkt. □

Bemerkung: Falls $P \neq \emptyset, D \neq \emptyset$, gibt es nach Satz 4.4 eine endliche Optimallösung x_* für (LP) und y_* für (DP). Nach dem schwachen Dualitätssatz gilt $z_P = c^T x_* \leq y_*^T b = z_D$. Fälls es $x \in P, y \in D$ mit $c^T x = y^T b$ gibt, so ist x optimal für (LP), y optimal für (DP) ist. Wir bezeichnen (x, y) dann als *optimales Paar*. Wir wollen nun untersuchen, ob $z_P < z_D$ gelten kann.

Parametrischer Ansatz

Mit Hilfe von

$$P_z := \{x \mid z \leq c^T x, Ax \leq b, x \geq 0\}$$

kann man ein zu (LP) äquivalentes parametrisches Problem formulieren:

$$\max\{z \mid P_z \neq \emptyset\}.$$

Lemma 6.8 *Sei $P \neq \emptyset$. Dann ist $P_z = \emptyset$ genau dann, wenn $y^T b < z, y^T A \geq c^T, y \geq 0$ lösbar.*

Beweis. Nach Korollar zu 6.2 ist $P \neq \emptyset$ genau dann, wenn

$$(*) \qquad y^T b < 0, y^T A \geq 0, y \geq 0$$

unlösbar ist. Analog ist $P_z = \emptyset$ genau dann, wenn

$$[y_0, y^T] \begin{bmatrix} -z \\ b \end{bmatrix} < 0, [y_0, y]^T \begin{bmatrix} -c^T \\ A \end{bmatrix} \geq 0, y_0 \geq 0, y \geq 0$$

lösbar. Eine Lösung mit $y_0 = 0$ kann es wegen $(*)$ nicht geben. Da die Menge der Lösungen einen Kegel bildet, können wir $y_0 \geq 0$ sogar durch $y_0 = 1$ ersetzen, ohne die Lösbarkeit einzuschränken. Daher ist $P_z = \emptyset$ genau dann, wenn

$$y^T b < z, y^T A \geq c^T, y \geq 0$$

lösbar. □

Satz 6.9 (Starker Dualitätssatz) *Besitzt eines der Probleme (LP) oder (DP) eine endliche Optimallösung, so auch das jeweils andere. Dann gilt $z_P = z_D$.*

Beweis. Da die beiden enthaltenen Aussagen dual sind, genügt es, eine zu zeigen. Sei etwa (DP) endlich lösbar, $y_*^T b = z_D$. Da $y_*^T b$ nach unten beschränkt, ist nach Satz 6.7 $P \neq \emptyset$.

Offenbar ist $y^T b < z_D, y^T A \geq c^T, y \geq 0$ unlösbar, also $P_{z_D} \neq \emptyset$, d.h. es existiert x_* mit $z_D \leq c^T x_*, Ax_* \leq b, x_* \geq 0$. Nach schwachem Dualitätssatz muß $y_*^T b = z_D = c^T x_*$ sein, d.h. x_*, y_* bilden ein optimales Paar. □

In der linearen Optimierung gibt es nach dem starken Dualitätssatz keine sogenannte Dualitätslücke zwischen dem Optimalwert des primalen und dem des dualen Problems. Daher kann man sehr leicht überprüfen, ob zwei gegebene Vektoren x, y Optimallösungen sind. Man testet die primale bzw. die duale Zulässigkeit von x und y und prüft, ob die Zielfunktionswerte übereinstimmen. Solche leicht überprüfbaren notwendigen und hinreichenden Optimalitätsbedingungen gibt es im allgemeinen weder in der Nichtlinearen noch in der Diskreten Optimierung. Man kann auch dort duale Aufgaben formulieren, aber es treten dabei echte Dualitätslücken auf.

Eine weitere Charakterisierung optimaler Paare findet sich im folgenden Satz.

Satz 6.10 (Komplementärer Schlupf) *Sei $x \in P, y \in D$. (x, y) ist ein optimales Paar genau dann, wenn*

1. $x_j > 0 \Rightarrow (y^T A)_j = c_j, \quad j = 1, \ldots, n,$
2. $y_i > 0 \Rightarrow (Ax)_i = b_i, \quad i = 1, \ldots, m.$

Beweis. Nach Satz 6.7 gilt $c^T x \leq y^T Ax \leq y^T b$, da $x \in P, y \in D$. Daher ist (x, y) ein optimales Paar genau dann, wenn beide Ungleichungen Gleichungen sind, also wenn $(c^T - y^T A)x = 0, y^T(Ax - b) = 0$. Da alle Werte nichtnegativ sind, muß jeder Summand verschwinden. □

Äquivalente Dualisierungen

Wie schon früher festgestellt, können wir (LP) in beliebiger Form dualisieren. Ein (LP) in Standardform

$$\max\{c^T x \mid Ax = b, x \geq 0\}$$

ist äquivalent zu

$$\max\{c^T x \mid Ax \leq b, -Ax \leq -b, x \geq 0\}.$$

Dieses ist in kanonischer Form und hat das duale Programm

$$\min\{u^T b - v^T b \mid u^T A - v^T A \geq c^T, u \geq 0, v \geq 0\}.$$

Mit $y := u - v$ vereinfacht sich dies zu

$$\min\{y^T b \mid y^T A \geq c\}.$$

Dualvariablen zu Gleichungen sind offenbar nicht vorzeichenbeschränkt. Wir wollen alle möglichen Fälle von Gleichungen, Ungleichungen, vorzeichenbeschränkten und nicht vorzeichenbeschränkten Variablen in einem starken allgemeinen Dualitätssatz zusammenfassen. Der Beweis ist Routine.

Korollar 6.11 (Allgemeiner starker Dualitätssatz) *Es gilt*

$$\begin{array}{llcll}
\max & d^T x + e^T y + f^T z & = & \min & u^T a + v^T b + w^T c \\
\textit{unter} & Ax + By + Cz \leq a & & \textit{unter} & u^T A + v^T D + w^T G \geq d^T \\
& Dx + Ey + Fz = b & & & u^T B + v^T E + w^T H = e^T \\
& Gx + Hy + Kz \geq c & & & u^T C + v^T F + w^T K \leq f^T \\
& x \geq 0, z \leq 0 & & & u \geq 0, w \leq 0,
\end{array}$$

falls eine der Aufgaben eine endliche Optimallösung besitzt.

Bemerkung:

1. Die endliche Variante des Simplexverfahrens haben wir zu Beginn des Abschnitts benutzt, um einen konstruktiven Beweis des Alternativsatzes 6.2 zu geben. Eine geometrische Formulierung dieses Resultats war das Farkas-Lemma.

2. Aus dem Farkas-Lemma folgen der schwache und starke Dualitätssatz 6.7 und 6.9. Umgekehrt kann man aus den beiden Sätzen das Farkas-Lemma folgern, d. h. die Sätze sind äquivalent zum Farkas-Lemma. Um das einzusehen, betrachten wir die primale Aufgabe
$$z_P := \max\{0^T x \mid Ax = b, x \geq 0\}$$
und die dazu gehörende duale Aufgabe
$$z_D := \min\{y^T b \mid y^T A \geq 0\}.$$

Falls $Ax = b, x \geq 0$ lösbar, ist $z_P = 0$ untere Schranke der dualen Zielfunktion. Daher folgt für dual zulässige y, d.h. für alle y mit $y^TA \geq 0$, $y^Tb \geq 0$.

Anderenfalls ist, da 0 dual zulässig, nach schwachem Dualitätssatz y^Tb unbeschränkt nach unten, d.h. es gibt einen dual zulässigen Halbstrahl $\bar{y} + \lambda\hat{y}, \lambda \geq 0$, auf dem y^Tb mit wachsendem λ echt fällt. Dann gilt $\hat{y}^TA \geq 0, \hat{y}^Tb < 0$.

Berechnung der dualen Optimallösung mittels Simplexverfahren

Mit Hilfe des Simplexverfahrens bestimmen wir eine optimale Basislösung x_B, x_N des (LP)

$$\max\{c^Tx \mid Ax = b, x \geq 0\}.$$

Wir zeigen, daß $y^T := c_B^T A_B^{-1}$ eine optimale Lösung des dualen Problems

$$\min\{y^Tb \mid y^TA \geq c^T\}$$

ist. Da B primal optimal, gilt $0 \geq \tilde{c}^T = c^T - y^TA$, d.h. y ist dual zulässig. Weiterhin gilt $y^Tb = c_B^T A_B^{-1} b = c_B^T x_B$, also ist (x, y) ein optimales Paar.

Wenn man alle Abbruchalternativen diskutiert, kann man mit Hilfe des endlichen Simplexverfahren auch einen direkten konstruktiven Beweis der Sätze 6.7 und 6.9 geben.

Zur Ökonomischen Interpretation der Dualvariablen

Beim bereits früher kurz vorgestellten Ernährungsmodell von Stiegler bezeichnen die primalen Variablen x_j die einzukaufende Menge des j-ten Nahrungsmittels. Die Koeffizienten a_{ij} der Ungleichungen beschreiben den Anteil der Einheiten des i-ten Grundbestandteils im j-ten Nahrungsmittel, die Komponenten b_i den zu deckenden Bedarf des i-ten Grundbestandteils. Das (LP) lautet also

$$z_P = \min\{c^Tx \mid Ax \geq b, x \geq 0\}.$$

Alternativ zu den Nahrungsmitteln soll es nun möglich sein, Grundbestandteile in Pillenform zu erwerben. Dabei bezeichnet y_i den Preis des i-ten Bestandteils. Falls $y^TA \leq c^T$ gilt, kann man jedes Nahrungsmittel durch eine Kombination der Pillen ersetzen, ohne daß Mehrkosten entstehen. Anders ausgedrückt, der Hersteller der Pillen ist konkurrenzfähig, wenn $y^TA \leq c^T$. Um bei bekanntem Bedarf die Einnahmen zu maximieren, löst der Hersteller die duale Aufgabe

$$z_D = \max\{y^Tb \mid y^TA \leq c^T, y \geq 0\}.$$

Die minimalen Einkaufskosten des Verbrauchers sind also der maximale Gewinn des Herstellers. Um die optimalen Preise, die Marktpreise, zu beschreiben, bringen wir das primale Problem mit geeigneten Schlupfvariablen in Standardform

$$\min\{\underbrace{c^Tx + 0^Tz}_{d^Tu} \mid \underbrace{Ax - z = b}_{Hu=b}, \underbrace{x \geq 0, z \geq 0}_{u \geq 0}\}.$$

Ist B eine optimale Basis, so ist die Optimallösung des dualen Problems

$$\max\{y^T b \mid y^T H \leq d^T\}$$

bekanntlich $\bar{y}^T := d_B^T H_B^{-1}$. Die duale Zulässigkeit $\bar{y}^T H \leq d^T$ impliziert $\bar{y} \geq 0$, da die Restriktionen eine negative Einheitsmatrix enthalten und die zugehörigen Koeffizienten der rechten Seite verschwinden.

Nun wird der Verbraucher nur einen Teil, etwa Δb, des Bedarfs b durch Pillenkauf ersetzen. Zur Vereinfachung wollen wir annehmen, daß $H_B^{-1}(b - \Delta b) \geq 0$ gilt. Dann bleibt die Basis B optimal. Die Einkaufskosten bei den Nahrungsmitteln sinken um $\Delta z = \sum_i \Delta z_i$. Die relative Reduzierung der Kosten $\frac{\Delta z_i}{\Delta b_i}$ heißt marginaler Preis von b_i. Es gilt

$$\Delta z = z_P - d_B^T H_B^{-1}(b - \Delta b) = d_B^T H_B^{-1} \Delta b = \bar{y}^T \Delta b.$$

Die marginalen Kosten sind also gerade die Marktpreise $\bar{y}_i$. Wenn Verbraucher und Hersteller optimieren, stehen der Kostenersparnis bei den Nahrungsmitteln gleich hohe Erwerbskosten für die Pillen gegenüber.

Falls $(A\bar{x})_i > b_i$ für die optimale primale Lösung $\bar{x}$ gilt, so ist der Bedarf b_i mehr als gedeckt. Nach dem Satz vom komplementären Schlupf ist der Marktpreis $\bar{y}_i = 0$, d.h. der i-te Bestandteil ist am Markt unverkäuflich.

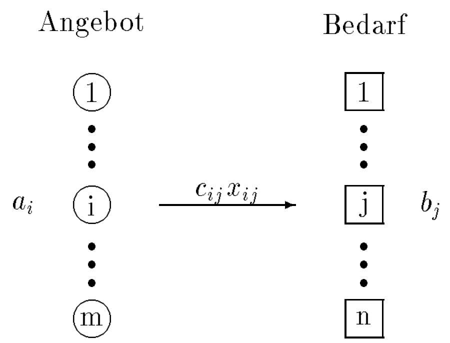


Abbildung I.15: Transportprobleme

Transportprobleme

Ein Handelsunternehmen kauft bei verschiedenen Herstellern $i = 1, \ldots, m$ Warenmengen a_i zum Preis u_i. Die Warenmenge b_i kann bei verschiedenen Kunden $j = 1, \ldots, n$ zum Preis v_j verkauft werden. Beim Transport der Warenmenge x_{ij} vom Hersteller i zum Kunden j entstehen marktübliche externe Transportkosten $c_{ij}x_{ij}$. Damit die Kunden den Einkauf nicht selbst billiger organisieren können, müssen die Preise $v_j \leq u_i + c_{ij}$ erfüllen. Optimale Gewinne erzielt das Handelsunternehmen durch Lösung von

$$\max\{\sum b_j v_j - \sum a_i u_i \mid v_j - u_i \leq c_{ij}\}.$$

Bei Minimierung der entstehenden Transportkosten

$$\min\{\sum c_{ij}x_{ij} \mid \sum_j(-x_{ij}) = -a_i, i = 1, \dots, m, \sum_i x_{ij} = b_j, j = 1, \dots, n, x \geq 0\}$$

wird dem Unternehmen allerdings deutlich, daß ein Gewinn nur dann zu erzielen ist, wenn man den Transport zu Preisen unter Marktniveau, etwa mit einem eigenen Fuhrpark, durchführen kann.

7 Die Simplexinterpretation des Simplexverfahrens

Der Name des Simplexverfahrens geht auf eine unterschiedliche geometrische Interpretation zurück. Wir betrachten dazu (LP) in der Form

$$\min\{c^T x \mid Ax = b, 1^T x = 1, x \geq 0.\} \tag{7.1}$$

Ausgehend von der Standardform kann man jedes (LP) auf diese Form bringen: Für eine genügend große Konstante $M > 0$ lassen sich die Variablen durch $1^T x + \lambda = M$, $\lambda \geq 0$ beschränken. Mit $y_i := x_i/M$, $\bar{\lambda} := \lambda/M$ erhält man ein entsprechendes Problem.

Für nichtleere Polyeder ist dann $\lambda > 0$ in der optimalen Ecke genau dann, wenn das Ausgangsproblem eine endliche Optimallösung besitzt.

Die geometrische Interpretation im K^{m+1} benutzt die Koordinaten $\left[\begin{smallmatrix} z_0 \\ z \end{smallmatrix}\right]$ mit $z_0 \in K, z \in K^m$. Sei $P := \left[\begin{smallmatrix} c^T \\ A \end{smallmatrix}\right]$. Die konvexe Hülle der Spalten von P ist

$$C(P) := \operatorname{conv}\{P_1, \dots, P_n\} = \{Px \mid 1^T x = 1, x \geq 0\}$$

Für eine zulässige Lösung x gilt außerdem $Px = \left[\begin{smallmatrix} z_0 \\ b \end{smallmatrix}\right]$. Das (LP) (7.1) ist daher äquivalent zu

$$\alpha = \min\left\{ z_0 \,\middle\|\, \begin{bmatrix} z_0 \\ b \end{bmatrix} \in C(P) \right\}. \tag{7.2}$$

Der Schnitt der Geraden $G := \left\{ \left[\begin{smallmatrix} z_0 \\ b \end{smallmatrix}\right] \mid z_0 \in K \right\}$ mit $C(P)$ enthält alle möglichen Zielfunktionswerte (siehe Abbildung I.16).

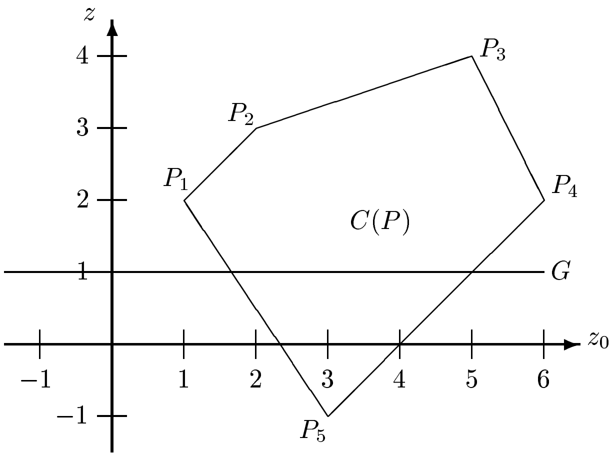
Interpretation des Simplexverfahrens

Auf eine Diskussion von Entartungsfällen wird im folgenden verzichtet.

1. Einer Basis B entsprechen die $m+1$ Punkte P_j, $j \in B$. Die Basis B ist zulässig, falls $C(P_B) \cap G \neq \emptyset$. In Abbildung I.16 entspricht P_3, P_5 einer zulässigen Basis, P_3, P_4 einer unzulässigen Basis.

2. Eine Basis B ist bekanntlich optimal, wenn der durch

$$0 = \tilde{c}_B^T =: c_B^T - y^T \begin{bmatrix} A_B \\ 1_B^T \end{bmatrix}$$



$$\begin{array}{lrcl} \min & x_1 + 2x_2 + 5x_3 + 6x_4 + 3x_5 & & \\ \text{unter} & 2x_1 + 3x_2 + 4x_3 + 2x_4 - x_5 & = & 1 \\ & x_1 + x_2 + x_3 + x_4 + x_5 & = & 1 \\ & x & \geq & 0 \end{array}$$

Abbildung I.16: Geometrische Interpretation

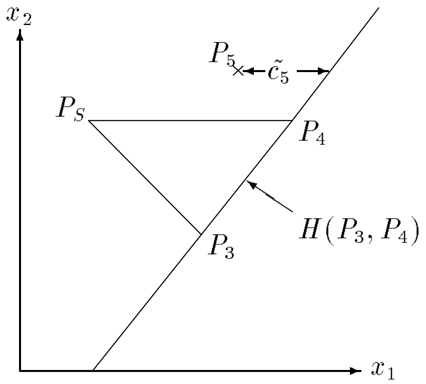


Abbildung I.17: Interpretation der reduzierten Kosten

definierte Vektor $y \in K^{m+1}$ dual zulässig ist, d.h. wenn gilt:

$$0 \leq \tilde{c}_N^T = c_N^T - y^T \begin{bmatrix} A_N \\ 1_N^T \end{bmatrix}.$$

Die zur Basis gehörenden Punkte P_B erzeugen eine affine Hyperebene,

$$H = \left\{ \begin{bmatrix} z_0 \\ z \end{bmatrix} \middle| [1, -y_1, \ldots, -y_m] \begin{bmatrix} z_0 \\ z \end{bmatrix} = y_{m+1} \right\}$$

denn

$$0 = c_B^T - y^T \begin{bmatrix} A_B \\ 1_B^T \end{bmatrix} = [1, -y_1, \ldots, -y_m] P_B - y_{m+1} 1_B^T.$$

Also ist B optimal, wenn alle Punkte P_j, $j = 1, \dots, n$, im Halbraum

$$H_{\geq} = \left\{ \begin{bmatrix} z_0 \\ z \end{bmatrix} \middle| [1, -y_1, \dots, -y_m] \begin{bmatrix} z_0 \\ z \end{bmatrix} \geq y_{m+1} \right\}$$

liegen. Die reduzierten Kosten $\tilde{c}_j = c_k - (y_1, \dots, y_m)A_j - y_{m+1}$ geben dabei die Entfernung des Punkts P_j von H an. Die Entfernung wird parallel zur Geraden G gemessen. In Abbildung I.17 ist dies der horizontale Abstand.

3. Die Pivotspalte entspricht dem Punkt P_s mit kleinstem (negativen) Abstand $\tilde{c}_s$. Sind die $m+2$ Punkte $P_{B\cup s}$ in allgemeiner Lage, so ist $S = C(P_{B\cup s})$ ein Simplex im K^{m+1}.

4. Die Auswahl der Pivotzeile r erfolgt über die Zulässigkeitsbedingung. Die Gerade G schneidet den Rand des Simplex S im allgemeinen in 2 Punkten. Der erste Schnittpunkt liegt auf $C(P_B)$, der zweite Schnittpunkt auf $C(P_{B\cup s\setminus r})$.

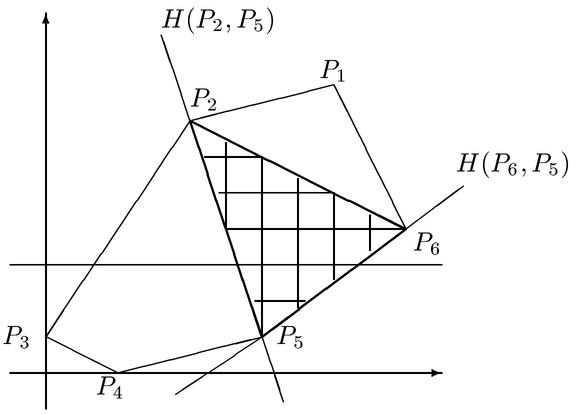


Abbildung I.18: Simplex des Basiswechsels von P_6/P_5 nach P_2/P_5

Kapitel II

Diskrete Optimierung

In der Diskreten Optimierung gilt es, in einer endlichen Menge von Objekten dasjenige zu finden, das bezüglich einer Bewertung optimal ist. Die Schwierigkeit der Problemstellung liegt im wesentlichen in der Größe der in den Anwendungen auftretenden Mengen, die es in der Regel verbietet, alle Elemente zu durchmustern. Wir werden uns hier auf Probleme beschränken, die sich in anschaulicher Weise mit Hilfe von Graphen formulieren lassen. Optimierungsaufgaben auf Graphen gehören zum Kern der Diskreten Optimierung und haben ungezählte Anwendungen. Ihr Schwierigkeitsgrad ist sehr unterschiedlich. Im Rahmen dieser Einführung beschränken wir uns auf relativ leicht lösbare Probleme. Nur im letzten Abschnitt werden wir ein wirklich schweres Problem, das Rundreiseproblem, diskutieren.

1 Beispiele, Graphen

Bei der Verdrahtung von Leiterplatten müssen Lötpunkte miteinander verbunden werden. Die Lötpunkte liegen meist auf einem vorgegebenen Gitter. Aus vielen Gründen (z.B. um Material zu sparen und die Leiterplatten möglichst rasch herzustellen) soll dabei die Länge der Verbindungen so klein wie möglich sein.

5 4

Abbildung II.1: Zwei minimale Verdrahtungen von Leiterplatten

Je nach Art der erlaubten Verbindungen führt diese Forderung auf ein relativ leichtes oder ein schweres Problem. Erlaubt man Verweigungen nur an den Lötstellen, so bilden die kürzesten Verbindungen einen „minimalen Baum“, den man mit einem Algorithmus aus Sektion 2 effizient finden kann. Eine solche minimale Verdrahtung findet sich links in Bild II.1. Erlaubt man Verzweigungen an allen Gitterpunkten auf der Leiterplatte, so bilden die minimalen Verdrahtungen minimale „Steiner-Bäume“, deren Bestimmung viel schwieriger ist. Es gibt bis heute keinen effizienten Algorithmus zu seiner Lösung. Eine entsprechende minimale Verdrahtung findet sich rechts im Bild II.1.

Die benutzten Darstellungen in Bild II.1 sind Beispiele für Graphen.

Graphen

Ein endlicher *Graph* $G = (V, E)$ besteht aus einer endlichen Menge V von Knoten und einer endlichen Menge E von Kanten. Wir bezeichnen die Knoten meist mit $i, j \ldots$ und die Kanten mit $ij, \ldots$, wobei $i, j \in V$. Jede Kante verbindet zwei Knoten des Graphen, die Kante ij die Knoten i, j. Diese Bezeichnungsweise erlaubt nicht, daß zwei Knoten durch mehrere (parallele) Kanten verbunden werden. Eine Kante ii, die den Knoten i mit sich selbst verbindet, heißt *Schlinge*. Endliche Graphen ohne Schlingen und parallele Kanten heißen *einfach*.

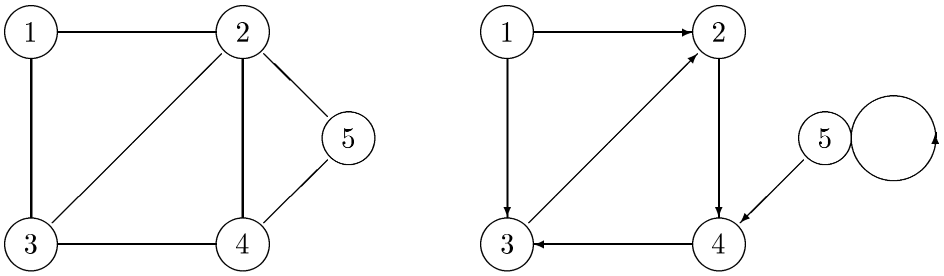


Abbildung II.2: Links ein ungerichteter einfacher Graph, rechts ein gerichteter Graph mit Schlinge

Man unterscheidet gerichtete und ungerichtete Graphen (siehe Abbildung II.2). In einem *gerichteten* Graphen sind alle Kanten gerichtet und die Kante ij führt vom Knoten i zum Knoten j. Dementsprechend bezeichnet man i als Anfangs- und j als Endknoten der Kante ij. Enthält ein gerichteter Graph kein Paar der wohlunterschiedenen Kanten ij und ji, so heißt er *antisymmetrisch*. In *ungerichteten* Graphen sind alle Kanten ungerichtet. Eine ungerichtete Kante ij verbindet ihre Endknoten i, j und unterscheidet sich nicht von der Kante ji. Knoten, die durch eine Kante verbunden sind, heißen *adjazent*. Formal kann man gerichtete Kanten als geordnetes Paar (i, j) und ungerichtete Kanten als Menge $\{i, j\}$ definieren. Da im Kontext immer feststehen wird, ob wir gerade einen ungerichteten oder einen gerichteten Graphen betrachten, benutzen wir durchgehend die kurze Bezeichnung ij.

Zu einem antisymmetrischen gerichteten Graphen erhält man den *zugrundeliegenden* ungerichteten Graphen mit gleichen Knoten und Kanten, indem man die Richtungen der Kanten nicht beachtet. Die Antisymmetrie ist hier vorausgesetzt, damit im ungerichteten Graphen keine parallelen Kanten entstehen.

Die anschauliche Darstellung von gerichteten und ungerichteten Graphen ist außerordentlich hilfreich. Dies gilt nicht nur bei der mathematischen Modellierung von Anwendungsproblemen sondern auch bei der Erläuterung und Verifizierung von Algorithmen zur Lösung von Optimierungsproblemen auf Graphen. Wegen der Größe der Anwendungsprobleme lassen sich allerdings auch die als relativ leicht apostrophierten Probleme nicht ohne Hilfe leistungsfähiger Computer lösen. Dazu müssen Graphen effizient im Computer gespeichert werden.

Darstellung von Graphen auf Computern

Sei $G = (V, E)$ ein Graph mit n Knoten und m Kanten. Wir wollen drei wichtige Darstellungen diskutieren. Als Beispiele dienen jeweils die Graphen aus Abbildung II.2.

1. **Adjazenzliste**: Kompakte, sparsame Speicherung mittels $O(n + m)$ Einträgen in zwei Listen $A[n]$ und $N[m]$ (für gerichtete Graphen) bzw. $N[2m]$ (für ungerichtete Graphen). In der Liste N stehen beginnend an der Position $A[i]$ sukzessive die Knotennummern der zum Knoten i adjazenten Knoten. Für den ungerichteten Graphen ergeben sich die folgenden Listen A und N:

1	3	7	10	13

2	3	5	4	3	1	1	2	4	3	2	5	4	2

   Der gerichtete Graph führt auf die Listen A und N:

1	3	4	5	6

2	3	4	2	3	5	4

2. **Verkettete Adjazenzlisten**: Hier werden die Knoten zu i nicht sukzessive, sondern in Reihenfolge $A[i], \mathrm{NEXT}[A[i]], \mathrm{NEXT}[\mathrm{NEXT}[A[i]]], \ldots$ abgespeichert. Ist der letzte dieser Knoten $N[k]$, so ist $\mathrm{NEXT}[k] = 0$. Die zusätzliche Liste $\mathrm{NEXT}[m]$ bzw. $\mathrm{NEXT}[2m]$ erlaubt Einfügen und Streichen von Kanten oder Knoten in $O(1)$. Der ungerichtete Graph wird durch A, N, NEXT mit

1	2	7	14	13

2	5	4	3	3	1	1	5	2	2	4	2	4	3
4	3	5	0	6	0	9	0	11	0	0	8	10	12

   dargestellt. Für den gerichteten Graphen finden wir A, N, NEXT mit

7	6	1	4	3

2	4	5	3	3	4	2
0	0	2	0	0	0	5

3. **Adjazenzmatrix**: $A[i, j] := 1$ für $i, j \in [1, n]$ genau dann, wenn $ij \in E$. A erlaubt es, die Präsenz einer Kante in $O(1)$ zu überprüfen. Allerdings sind n^2 Einträge notwendig (ein Bit pro Eintrag). Hier ergeben sich die folgenden Adjazenzmatrizen für den ungerichteten bzw.den gerichteten Graphen:

ungerichtet

0	1	1	0	0
1	0	1	1	1
1	1	0	1	0
0	1	1	0	1
0	1	0	1	0

gerichtet

0	1	1	0	0
0	0	0	1	0
0	1	0	0	0
0	0	1	0	0
0	0	0	1	1

Wege

Betrachtet man in einem Stadtplan die Kreuzungen als Knoten und die Straßen als Kanten, so erhält man einen Graphen. In analoger Weise modellieren Graphen das Eisenbahnnetz in Europa oder die Flugverbindungen der Welt. Die Suche einer Verbindung mit kürzester Reisedauer oder kürzester Entfernung in solchen Graphen ist ein alltägliches Problem.

In der Graphentheorie werden solche Verbindungen als Wege bezeichnet. Ein *Weg* ist eine Kantenfolge $e_1, \ldots, e_n$ mit der Eigenschaft, daß jede Kante e_i einen Endknoten mit e_{i-1} und den anderen mit e_{i+1} teilt. Ein Weg heißt *einfach*, falls jeder Knoten des Weges

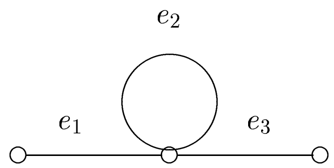
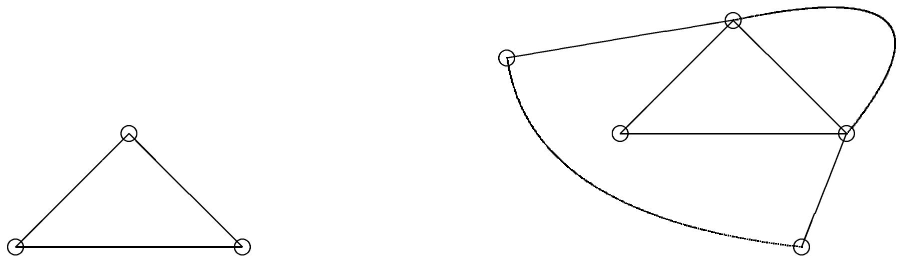


Abbildung II.3: Weg

mit höchstens zwei Kanten des Weges inzidiert. Ein *Kreis* ist ein Weg mit identischem



einfach nicht einfach

Abbildung II.4: Kreise

Anfangs- und Endknoten. Eine Schlinge ist ein Kreis mit nur einer Kante.

Die Anzahl der Kanten eines Weges wird als *Länge* des Weges bezeichnet. Gerichtete Wege werden analog definiert.

Zusammenhang von Graphen

Vor der Suche nach günstigen Verbindungen muß zunächst einmal die Existenz von Verbindungen gesichert sein. In der Graphentheorie wird diese Eigenschaft als Zusammenhang bezeichnet.

Ein ungerichteter Graph $G = (V, E)$ heißt *zusammenhängend*, falls für alle $i, j \in V, i \neq j$ ein Weg zwischen i und j existiert.

Ein gerichteter Graph $G = (V, E)$ heißt *stark zusammenhängend*, falls für alle $i, j \in V, i \neq j$ ein gerichteter Weg von i nach j existiert.

Nennt man zwei Knoten i, j äquivalent, wenn Wege von i nach j und von j nach i existieren, so erhält man eine Äquivalenzrelation auf den Knoten des (gerichteten) Graphen. Ein (gerichteter) Graph ist offenbar genau dann (stark) zusammenhängend, wenn V die einzige Äquivalenzklasse ist.

Für eine Knotenmenge $V' \subseteq V$, ist ein *Schnitt* $\delta(V') := \{ij \mid i \in V', j \in \overline{V'}\}$ die Menge der Kanten, die in V' beginnen und im Komplement $\overline{V'}$ der Menge V' enden. Der Schnitt heißt *nichttrivial*, wenn $\emptyset \subset V' \subset V$. Man kann sich überlegen, daß zu jeder der obigen Äquivalenzklassen ein leerer Schnitt gehört. Also gilt:

Lemma 1.1 *Ein (gerichteter) Graph ist (stark) zusammenhängend genau dann, wenn kein nichttrivialer leerer Schnitt existiert.*

Für einen nicht (stark) zusammenhängenden Graphen $G = (V, E)$ ist es sinnvoll zunächst einmal die obigen Äquivalenzklassen, d.h. maximale Teilmengen $V' \subseteq V$ zu finden, so daß die *induzierten Untergraphen* $G' = (V', E(V'))$ mit $E(V') := \{ij \in E \mid i, j \in V'\}$ (stark) zusammenhängend sind. Diese heißen (starke) *Zusammenhangskomponenten* des Graphen.

Hier sei angemerkt, daß jeder Graph $G' = (V', E')$ mit $V' \subseteq V, E' \subseteq E(V')$ als *Untergraph* von G bezeichnet wird. Ein Untergraph mit $V' = V$ heißt *Teilgraph*.

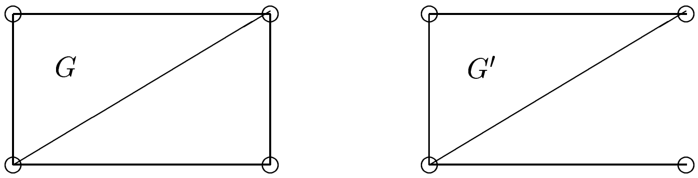


Abbildung II.5: Teilgraph

Versucht man, den Zusammenhang eines Graphen mit möglichst wenigen Kanten sicherzustellen, so ergeben sich spezielle Graphen. Ein zusammenhängender Graph ohne Kreise heißt *Baum*. Ist ein Baum G' ein Teilgraph von G, so heißt er *Gerüst* von G. Bäume lassen sich auf verschiedene Art und Weise charakterisieren:

Satz 1.2 ((Charakterisierung von Bäumen)) *Sei $G = (V, E)$ ein einfacher ungerichteter Graph mit n Knoten und m Kanten. Dann sind äquivalent:*

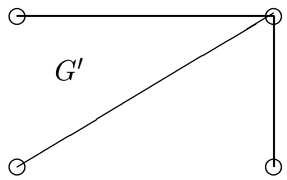


Abbildung II.6: Gerüst

1. *G ist ein Baum.*
2. *G ist kreisfrei, aber fügt man eine Kante hinzu, so ergibt sich ein Kreis.*
3. *Zwischen je zwei verschiedenen Knoten gibt es genau einen einfachen Weg.*
4. *G ist zusammenhängend, aber jeder echte Teilgraph von G ist nicht zusammenhängend.*
5. *G ist kreisfrei und $m = n - 1$.*
6. *G ist zusammenhängend und $m = n - 1$.*

Zur Übung überlege man sich die Implikationen $1 \to 2 \to 3 \to 4 \to 1$ und $1 \to 5 \to 6 \to 1$.

Baukosten eines Netzes

Modelliert man die möglichen Verbindungen eines Telefon- oder Verkehrsnetzes in einem Graphen G und bezeichnet die Baukosten einer Direktverbindung der Knoten i, j mit c_{ij}, so erhält man ein Netz mit minimalen Baukosten, in dem jeder Knoten von jedem anderen Knoten aus erreichbar ist, wenn man ein Gerüst B_* von G mit $\sum_{ij \in B_*} c_{ij} \leq \sum_{ij \in B} c_{ij}$ für alle Gerüste B von G konstruiert. Sind die Baukosten proportional zur Euklidischen

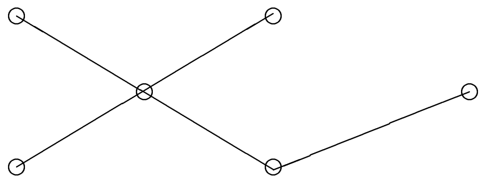


Abbildung II.7: Ein minimales Gerüst

Entfernung der Knoten in der Ebene, so zeigt Abbildung II.7 ein minimales Gerüst.

Ist die Richtung einer Verbindung wesentlich, so kann man analoge Modelle mit gerichteten Graphen entwickeln.

Sei $G = (V, E)$ sei gerichteter Graph. Ein Knoten $w \in V$ heißt *Wurzel* von G, falls für alle $w \neq v$ ein gerichteter Weg von w nach v existiert. Ein antisymmetrischer, gerichter Graph G mit Wurzel heißt *gerichteter Baum*, falls der zugrundeliegende ungerichtete

Graph ein Baum ist. Ist ein gerichteter Baum G' Teilgraph von G, so heißt er *gerichtetes Gerüst* von G.

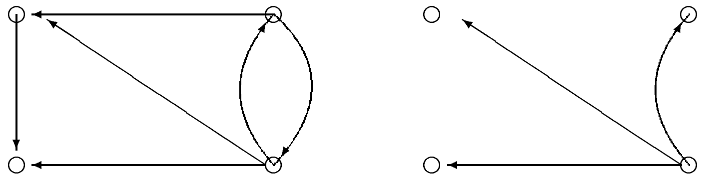


Abbildung II.8: Links ein gerichteter Graph, rechts eines seiner Gerüste.

Satz 1.3 (Charakterisierung gerichteter Bäume) *Es sei $G = (V, E)$ ein gerichteter Graph, dann sind äquivalent:*

1. *G ist ein gerichteter Baum.*
2. *G besitzt eine Wurzel, von der aus genau ein gerichteter Weg zu jedem anderen Knoten führt.*
3. *G besitzt eine Wurzel w mit $|\delta(\overline{w})| = 0$, aber für alle anderen Knoten v gilt $|\delta(\overline{v})| = 1$.*
4. *G besitzt eine Wurzel, aber kein echter Teilgraph besitzt eine Wurzel.*
5. *Der zugrundeliegende ungerichtete Teilgraph von G ist zusammenhängend, G besitzt genau einen Knoten w mit $|\delta(\overline{w})| = 0$, aber für alle übrigen Knoten v gilt $|\delta(\overline{v})| = 1$.*
6. *Der zugrundeliegende ungerichtete Teilgraph von G ist kreisfrei, G besitzt genau einen Knoten w mit $|\delta(\overline{w})| = 0$, aber für alle übrigen Knoten v gilt $|\delta(\overline{v})| = 1$.*

Auch der Beweis dieses Satzes wird zur Übung empfohlen.

Baukosten einer Rohrpost

Modelliert man die möglichen Verbindungen eines Rohrpostsystems in einem gerichteten Graphen G und bezeichnet die Baukosten einer Direktverbindung der Knoten i, j mit c_{ij}, so erhält man ein System mit minimalen Baukosten, in dem jede Nebenstelle von einer Hauptstelle aus erreichbar ist, wenn man ein gerichtetes Gerüst B_* von G mit $\sum_{ij \in B_*} c_{ij} \leq \sum_{ij \in B} c_{ij}$ für alle Gerüste B von G konstruiert.

Das Königsburger Brückenproblem

Euler hat eines der ältesten Wegeprobleme formuliert und gelöst: Kann man über alle sieben Brücken Königsbergs (s. Abbildung II.10, links) gehen, ohne eine mehrfach zu benutzen? Im entsprechenden ungerichteten Graphen (rechts in Abbildung II.10) ist also ein Weg gesucht, der alle Kanten genau einmal durchläuft. Ein solcher Weg heißt *Euler'sch*. Die Antwort auf die Frage hängt allein von der Anzahl der Kanten in den Knoten ab.

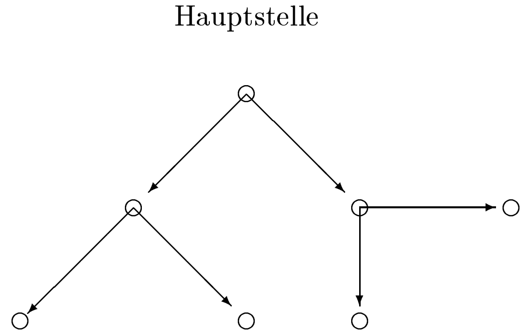


Abbildung II.9: Zum Rohrpostproblem.

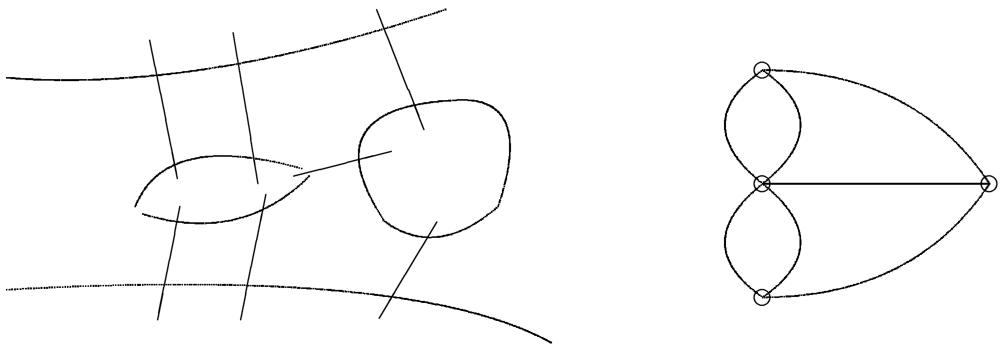


Abbildung II.10: Das Königsberger Brückenproblem.

In gerichteten Graphen ist der *Außengrad* eines Knoten v, $|\delta(v)|$, die Anzahl der Kanten, die in v beginnen. Analog bezeichnet der *Innengrad*, die Anzahl der Kanten, die in v enden, d.h. in $\bar{v}$ beginnen. Analog definiert man den *Grad* von Knoten in ungerichteten Graphen.

Die Existenz Euler'scher Wege bzw. analog definierter gerichteter Euler'scher Wege charakterisiert der folgende Satz (Beweis als Übung):

Satz 1.4 (Existenz Eulerscher Wege)

1. Ein zusammenhängender ungerichteter Graph besitzt einen Euler'schen Weg genau dann, wenn keine oder zwei Knoten ungeraden Grad haben. Im ersten Fall ist jeder Eu-

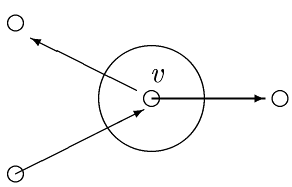


Abbildung II.11: Knotengrade: $|\delta(v)| = 2, |\delta(\bar{v})| = 1$

ler'sche Weg ein Kreis.

2. Ein stark zusammenhängender gerichteter Graph besitzt genau dann einen gerichteten Euler'schen Weg, wenn alle Knoten v oder alle mit Ausnahme zweier Knoten r, w identischen Innen- und Außengrad, d.h.

$$|\delta(\overline{v})| = |\delta(v)|$$

besitzen, wobei im letzteren Fall zusätzlich gilt

$$|\delta(\overline{r})| = |\delta(r)| + 1, |\delta(w)| = |\delta(\overline{w})| + 1.$$

Im ersten Falle ist jeder gerichtete Euler'sche Weg ein gerichteter Kreis.

Eine verwandte Aufgabe stellt sich einem Briefträger, der zur Ablieferung der Post alle Straßen seines Bezirks mindestens einmal durchlaufen und dann zum Ausgangspunkt zurückkehren muß:

Bestimme den kürzesten Kreis, der alle Kanten mindestens einmal durchläuft.

Gibt es keinen Euler'schen Kreis, so müssen Kanten mehrfach so durchlaufen werden, daß die Gesamtlänge minimiert wird. Den Längen können dabei wieder unterschiedliche Zielsetzungen entsprechen, z.B. Länge, Dauer oder Begehbarkeit der Straßen. Probleme dieser Art sind relativ leicht lösbar.

Rundreiseproblem

Dantzig betrachtete 1954 das zunächst verwandt erscheinende Rundreiseproblem. Ein Handelsreisender, der auf einer Rundreise nacheinander eine Reihe von Städten genau einmal aufsuchen muß, versucht etwa seine Reisedauer zu minimieren. Modelliert man das Verkehrsnetz durch einen Graphen, so entsprechen Rundreisen den Kreisen, die alle Knoten genau einmal durchlaufen. Weniger offensichtlich ist die Formulierung als $0-1$-Problem.

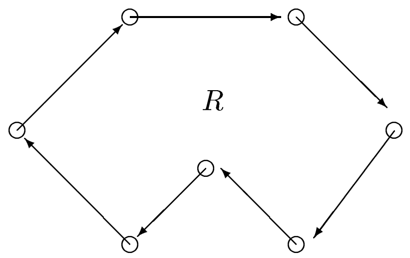


Abbildung II.12: Eine Rundreise R durch 7 Städte

Einer Rundreise R entspricht in eineindeutiger Weise ein Rundreisevektor x:

$$x_{ij} = \begin{cases} 0 & : \quad ij \notin R \\ 1 & : \quad ij \in R \end{cases} .$$

Offenbar kann man mit solchen *Inzidenzvektoren* jede Kantenmenge R eines Graphen als $0-1$-Vektor beschreiben. Zur Abkürzung sei $x(E') := \sum_{e\in E'} x(e)$, $E' \subseteq E$. Dann erfüllt jeder Rundreisevektor die Gleichungen

$$x(\delta(v)) = x(\delta(\bar{v})) = 1$$

für alle $v \in V$. Man kann zeigen, daß die Lösungen dieser Gleichungen die affine Hülle aller Rundreisevektoren bilden. Die affine Hülle enthält aber $0-1$-Vektoren (s. Abbildung II.13), die keine Rundreisevektoren sind.

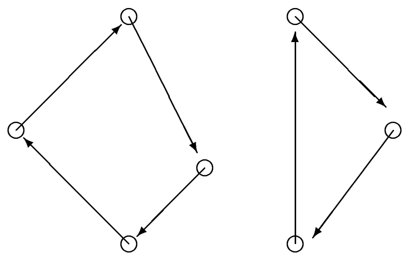


Abbildung II.13: Keine Rundreise

Schränkt man die affine Hülle jedoch weiter durch die Ungleichungen

$$x(\delta(V')) \geq 1,$$

für alle $V' \subset V, V' \neq \emptyset$ ein, so ist jeder verbleibende $0-1$-Lösungsvektor ein Rundreisevektor. Man kann auf diese Weise Rundreisen als $0-1$-Vektoren charakterisieren, die gewisse lineare Gleichungen und Ungleichungen erfüllen.

2 Minimale Bäume

In diesem Abschnitt werden Verfahren zur Konstruktion von minimalen Bäumen in ungerichteten und gerichteten Graphen entwickelt. Zunächst sei $G = (V, E)$ ein ungerichteter zusammenhängender Graph mit Kantengewichten $c : E \to \boldsymbol{R}$. Das Gewicht einer Kantenmenge $E' \subseteq E$ ist $c(E') := \sum_{ij\in E'} c(ij)$. Ein Gerüst (V, E') mit minimalem Gewicht heißt *minimales Gerüst*.

Aufgabe 2.1
Bestimme ein minimales Gerüst (V, E').

In den folgenden Verfahren wird ein minimales Gerüst iterativ durch Hinzunahme von Kanten aufgebaut. Eine Vereinigung knotendisjunkter Bäume $(V_i, B_i), 1 \leq i \leq r$, heißt *Wald*. Ein Wald mit $V = \bigcup_i V_i$ heißt *aufspannend*. In jeder Iteration des Verfahrens ist also ein aufspannender Wald bekannt, der Teilgraph eines minimalen Gerüstes ist. In einer Iteration wird entweder der Wald vergrößert (*Blaue Regel*) oder das minimale Gerüst enger eingegrenzt (*Rote Regel*):

Blaue Regel: In einem Schnitt ohne blaue Kanten färbe eine Kante mit minimalem Gewicht blau.

Rote Regel: In einem Kreis ohne rote Kanten färbe eine Kante mit maximalem Gewicht rot.

Lemma 2.1 (Färbungslemma) *Wendet man sukzessive die rote oder blaue Regel auf einen anfangs ungefärbten zusammenhängenden Graphen an, so gibt es stets ein minimales Gerüst B, das alle blauen, aber keine rote Kante enthält.*

Beweis. Bei ungefärbtem Graphen gilt die Behauptung für jedes minimale Gerüst B. Annahme: Vor Anwendung der Regel im aktuellen Iterationsschritt gilt die Behauptung für ein Gerüst B.

Anwendung der blauen Regel: Eine Kante e eines Schnittes $\delta(S)$ wird blau gefärbt. Falls $e \in B$, gilt die Behauptung weiterhin für B. Anderenfalls enthält $B \cup e$ genau einen Kreis C, der mit dem Schnitt mindestens eine weitere Kante e' gemeinsam hat. Diese ist nicht rot, da im Gerüst, und auch nicht blau, da im Schnitt, weswegen auch $c(e') \geq c(e)$ gilt. Folglich ist $(B \setminus e') \cup e$ ein minimales Gerüst, für das die Behauptung nach Anwendung der Regel gilt.

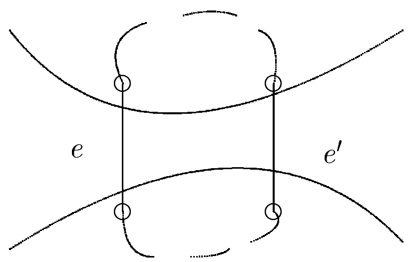


Abbildung II.14: Färbungslemma: Blaue Regel

Anwendung der roten Regel: Eine Kante e eines Kreises C wird rot gefärbt. Falls $e \notin B$, gilt die Behauptung weiterhin mit B. Anderenfalls ist $B \setminus e$ ein aufspannender Wald aus zwei Bäumen B_1, B_2 und der Kreis hat mindestens eine weitere Kante e' mit dem Schnitt $\delta(B_1)$ gemeinsam. Diese ist nicht blau, da nicht im Gerüst, und auch nicht rot, da im Kreis, weswegen auch $c(e') \leq c(e)$ gilt. Folglich ist $(B \setminus e) \cup e'$ ein minimales Gerüst, für das die Behauptung nach Anwendung der Regel gilt. □

Durch sukzessive Anwendung der beiden Regeln kann man alle Kanten des Graphen färben. Nach dem Färbungslemma bilden die blauen Kanten als Teilmenge der Kanten von B stets einen Wald. Ungefärbte Kanten innerhalb eines Baumes kann man dann rot färben, ohne daß B geändert werden muß. Eine der ungefärbten Kanten zwischen verschiedenen Bäumen kann man dann blau färben, wodurch zwei Bäume zusammenwachsen, etc. Beim folgenden Verfahren beginnt man beim Knoten 1 und wählt sukzessive die kleinste Kante, die den derzeitigen Baum mit einem weiteren Knoten verbindet.

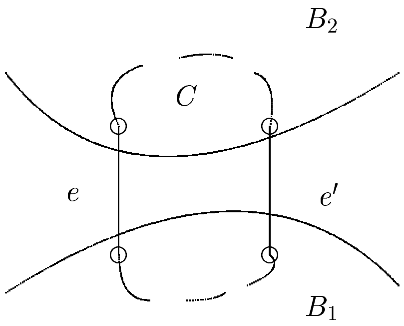


Abbildung II.15: Färbungslemma: Rote Regel

Algorithmus 2.1 (Das Verfahren nach Prim)

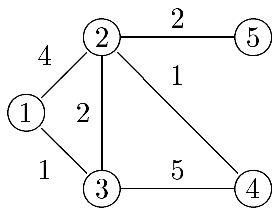
Sei $G = (V, E)$ ein ungerichteter Graph. $N[k] := \{j \mid ij \in \delta(i)\}$ ist die Liste der zu $k \in V$ adjazenten Knoten. Mit T wird die Menge der noch nicht in den Baum aufgenommenen Knoten bezeichnet.

1. $T := V \setminus \{1\}$;
   $u(1) := 0$; $u(j) := c(1j)$, $2 \leq j \leq n$;
   $p(1) := 0$; $p(j) := 1$, $2 \leq j \leq n$;

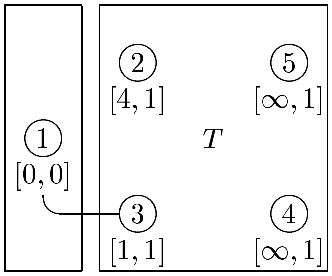
2. Bestimme $k \in T$ mit $u(k) := \min\{u(j) \mid j \in T\}$;
   $T := T \setminus \{k\}$;

3. Falls $T = \emptyset$ stop.
   Für alle $j \in N[k]$: falls $j \in T$ und $u(j) > c(kj)$ setze $u(j) := c(kj)$; $p(j) := k$;
   gehe nach 2.

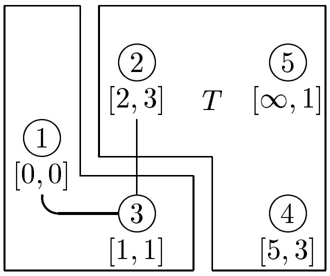
Für den gerichteten Graphen



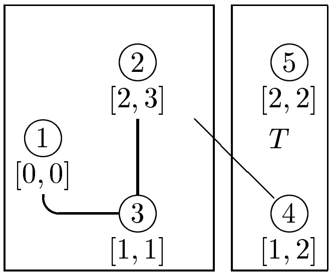
ergeben sich sukzessive die folgenden Markierungen $[u, p]$:



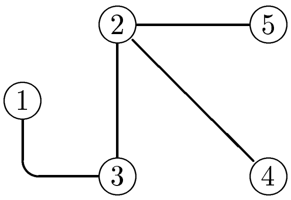
$k = 3$



$k = 2$



$k = 4$ und dann
$k = 5$ mit unveränderten Marken



Lösung

Bemerkung:

1. p liefert Baum $B := \{p(j)j \mid j \neq 1\}$ mit Wurzel 1.

2. Die Komplexität einer trivialen Implementation ist $O(n^2)$.

3. Mit Hilfe von Priority Queues kann man die auftretenden Operationen INSERT, MINIMUM, EXTRACT MINIMUM in $O(\log n)$ durchführen. Wird T als Priority Queue implementiert, so kann man daher in jeder Iteration Schritt 2 in $O(\log n)$, also insgesamt in $O(n \log n)$ durchführen. Schritt 3 mit den Operationen INSERT bzw. DECREASE KEY erfordert ebenfalls den Aufwand $O(\log n)$, also insgesamt $O(m \log n)$.

   Wird T über Fibonacci-Heaps implementiert, so erfordert Schritt 3 nur konstanten Aufwand $O(1)$, so daß sich ein Verfahren der Komplexität $O(m + n \log n)$ ergibt. Abschätzungen dieser Art werden in [16] detailliert abgeleitet.

Beim Verfahren von Kruskal beginnt man mit einem Wald aus Einzelknoten. Dann wählt man sukzessive die kleinste Kante, die keinen Kreis erzeugt. Im Unterschied zum Verfahren von Prim wächst hier nicht ein einziger Baum, sondern in jedem Schritt werden zwei Bäume des Waldes zu einem größeren Baum zusammengefügt.

Algorithmus 2.2 (Das Verfahren von Kruskal)

Sei $G = (V, E)$ ein ungerichteter Graph mit nach Gewicht sortierten Kanten $e_k := i_k j_k$, für $k := \{1, 2, \ldots, m\}$, d.h. es gilt $c(e_k) \leq c(e_{k+1})$ für alle $k := \{1, 2, \ldots, m-1\}$.

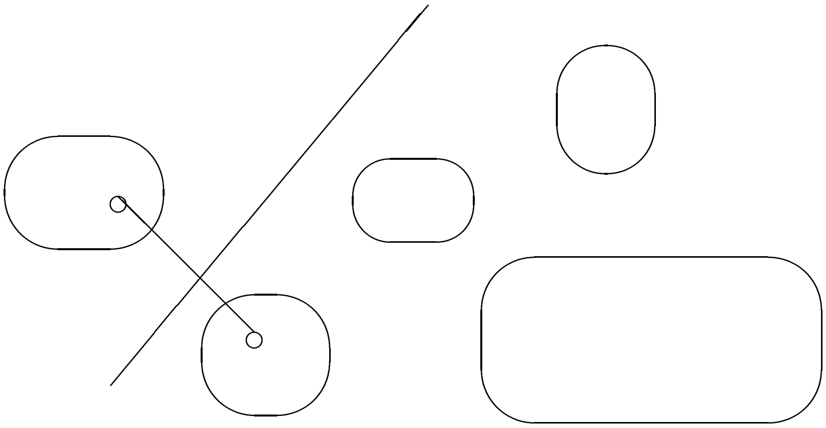
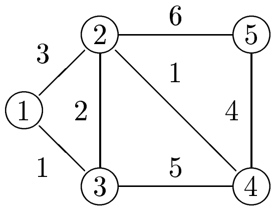


Abbildung II.16: Kruskal

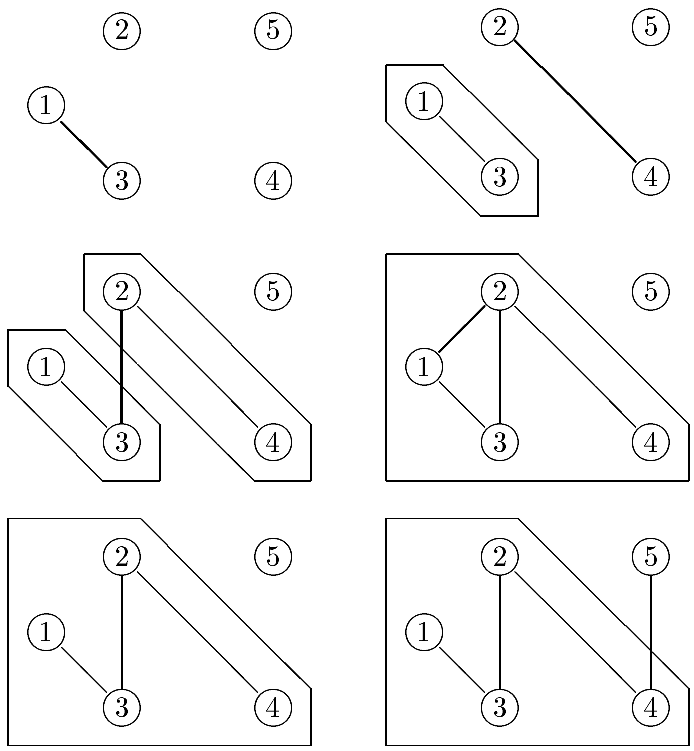
1. Initialisiere Mengen $\mu_j := \{j\}$, $1 \leq j \leq n$; $p(e_k) := 0$, $1 \leq k \leq m$;

2. Für $k := \{1, 2, \ldots, m\}$
   $\mu :=$ Menge(i_k); $\nu :=$ Menge(j_k);
   falls $\mu \neq \nu$ dann bilde Vereinigung(μ, ν); $p(e_k) := 1$;

Für den ungerichteten Graphen



mit der aufsteigend sortierten Kantenfolge $13, 24, 23, 12, 45, 34, 25$ ergeben sich die folgenden Wälder:



Danach ist V der einzige Baum des Waldes und die Untersuchung weiterer Kanten führt zu keiner weiteren Änderung.

Bemerkung (Zur Komplexität des Verfahrens von Kruskal):

Die Kanten kann man in $O(m \log m)$ sortieren. Die jeweils minimale Kante erzeugt einen Kreis genau dann, wenn ihre Endknoten im gleichen Baum liegen. Speichert man daher die Knoten der Bäume als Mengen, so sind m Mengenoperationen erforderlich, die in $O(m \cdot \alpha(m,n)) \leq O(m \log m)$ durchgeführt werden können, wobei α die Inverse der Ackermann-Funktion ist, in der Praxis gilt $\alpha < 4$. Für die Herleitung der Komplexitätsschranke sei wieder auf [16] verwiesen.

Speicherung von Mengen

Disjunkte Teilmengen einer Menge mit n Elementen lassen sich wie gerichtete Bäume abspeichern, wobei jede Teilmenge mit ihrer Wurzel bezeichnet wird. Zwei Teilmengen werden vereinigt, indem man die Wurzel der einen an die der anderen anhängt. Die Höhe der Bäume wird in einem Hilfsfeld $HEIGHT[n]$ näherungsweise mitgerechnet. Um die Höhe klein zu halten, wird bei Vereinigungen der Baum mit kleinerem $HEIGHT$ angehängt. Die Struktur des gerichteten Waldes wird auf einem Feld $A[n]$ gespeichert.

Als Mengenoperationen werden die Initialisierung einelementiger Teilmengen $\{u\}$, die Vereinigung zweier Teilmengen $UNION(\mu,\nu) := \mu \cup \nu$ und die Bestimmung der Teilmenge $SET(u)$, die das Element u enthält, implementiert:

1. **Initialisierung der Menge** $\{u\}$
   $A[u] := u$; $HEIGHT[u] := 0$; *(u ist Wurzel der Menge $\{u\}$)*

2. **Bestimmung von** $\mu = SET(u)$

   $w := u$; solange $w \neq A[w]$ setze $w := A[w]$; *(Wurzelbestimmung)*
   $\mu := w$;
   $w := u$; solange $w \neq A[w]$ setze $v := w; w := A[w]; A[v] := \mu$; *(Pfadkompression)*

3. **Bildung von** $UNION(\mu, \nu) := \mu \cup \nu$

   Falls $\mu \neq \nu$
   falls $HEIGHT[\mu] > HEIGHT[\nu]$ setze $A[\nu] := A[\mu]$;
   sonst setze $A[\mu] := A[\nu]$;
   falls $HEIGHT[\mu] = HEIGHT[\nu]$
   setze $HEIGHT[\nu] := HEIGHT[\nu] + 1$;

Die Pfadkompression bei jeder Bestimmung einer Wurzel trägt wesentlich zur Verringerung der Höhe der Bäume bei. Insgesamt kann man zeigen, daß dann m Mengenoperationen die Komplexität $O(m \cdot \log m)$ besitzen (siehe etwa in [16]).

Die bereits oben zur Illustration des Verfahrens von Kruskal skizzierten Wälder werden nunmehr wie folgt dargestellt:

Start

(1), (2), (3), (4), (5)

	1	2	3	4	5
A	1	2	3	4	5
$HEIGHT$	0	0	0	0	0

(1, 3), (2), (4), (5)

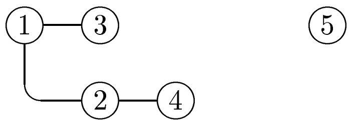
1	2	1	4	5
1	0	0	0	0

(1, 3), (2, 4), (5)

1	2	1	2	5
1	1	0	0	0

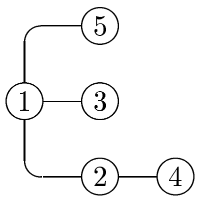
(1, 2, 3, 4), (5)

1	1	1	2	5
2	1	0	0	0



(1, 2, 3, 4, 5)

1	1	1	2	1
2	1	0	0	0



Bei einer anschließenden Bestimmung von $SET(4)$ würde durch die Pfadkompression auch der Knoten 4 unmittelbar an die Wurzel 1 angehängt.

Hierarchische Informationssysteme

Auch hierarchische Informationssysteme lassen sich mit Baumstrukturen modellieren. Allerdings kommt es hier auf die Richtung an. Bei fest vorgegebener Zentrale eignen sich gerichtete Bäume mit fester Wurzel:

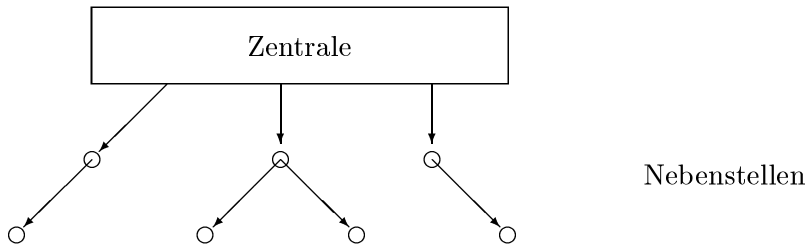


Abbildung II.17: feste Wurzel

Kann die Zentrale frei geplant werden, sucht man einen gerichteten Baum mit freier Wurzel:

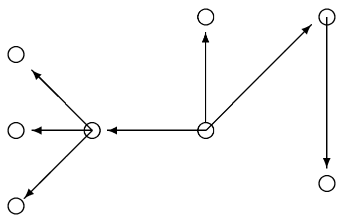


Abbildung II.18: freie Wurzel

Sind mehrere lokale Zentren erlaubt, muß man einen gerichteten Wald konstruieren:

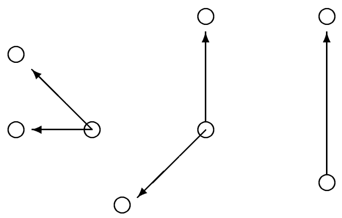


Abbildung II.19: mehrere Wurzeln

Branchings

In einem gerichteten Graphen (V, E) sei (V, B) ein Teilgraph ohne ungerichtete Kreise. Dann heißt (V, B) *Branching*, falls für alle $i \in V$ die "*Innengrad-Bedingung*" gilt:

$$|\delta(\bar{i}) \cap B| \leq 1.$$

Die Suche nach optimalen hierarchischen Strukturen führt auf

Aufgabe 2.2
Bestimme ein Branching mit maximalem Gewicht zu $c\colon E \to \boldsymbol{R}$.

Ein Branching mit maximalem Gewicht wird kurz als maximales Branching bezeichnet. Streicht man in einem Branching nichtpositive Kanten, so erhält man ein Branching mit gleichem oder größerem Gewicht. Daher darf man OBdA $c > 0$ voraussetzen.

Lemma 2.2 *Erfüllt ein gerichteter Graph* $G = (V, H)$ *die Innengradbedingung, so sind alle Kreise in* G *einfach, gerichtet und knotendisjunkt.*

Beweis. Sei C ein Kreis mit k Kanten. Wegen der Innengradbedingung ist jeder der maximal k Kreisknoten Endknoten höchstens einer der k Kanten. Daher besitzt C genau k Knoten und ist einfach. Jeder Kreisknoten ist Endknoten genau einer Kreiskante und Anfangsknoten genau einer anderen Kreiskante. Also ist der Kreis gerichtet. Sind C_1, C_2 zwei verschiedene Kreise, so sind beide insbesondere einfach und gerichtet. Ist $ij \in C_1 \setminus C_2$, so kann wegen der Innengradbedingung j nicht zu C_2 gehören. Durchläuft man C_1 ausgehend von j, so folgt dies ebenso für jeden weiteren Knoten von C_1, d.h. die Kreise sind knotendisjunkt. □

Folgerung: Sei $G = (V, E)$ ein gerichteter Graph ohne Schlingen, der die Innengradbedingung erfüllt. Dann hat G höchstens $\lfloor \frac{n}{2} \rfloor$ Kreise.

Kritische Teilgraphen

Ein Teilgraph (V, H) von G heißt *kritisch*, falls für alle Knoten j mit $\delta(\bar{j}) \neq \emptyset$ gilt:

1. $|\delta(\bar{j}) \cap H| = 1$,
2. $c(ij) \geq c(e) \quad \forall\, ij \in H, e \in \delta(\bar{j})$.

Kritische Teilgraphen erfüllen die Innengradbedingung. Jeder kreisfreie, kritische Teilgraph ist ein maximales Branching.

Folgerung: Sei (V, H) kritisch und sei (V, B_*) ein maximales Branching, das unter allen maximalen Branchings die meisten Elemente von H enthält, d.h. $|H \cap B_*|$ ist maximal. Entweder ist dann $H = B_*$, oder der kritische Graph ist nicht kreisfrei. Da das Branching kreisfrei ist, enthält jeder Kreis C des kritischen Graphen mindestens eine Kante, die nicht zum Branching gehört, d.h.

$$|C \setminus B_*| \geq 1.$$

Jede Kante ij des kritischen Graphen, die nicht zu B_* gehört, kann man zu B_* hinzufügen. Entfernt man gleichzeitig eine möglicherweise in B_* vorhandene Kante kj, so erfüllt $B := (B_* \setminus kj) \cup ij$ die Innengradbedingung und es gilt:

$$c(B) \geq c(B_*) \quad \text{und} \quad |H \cap B| > |H \cap B_*|.$$

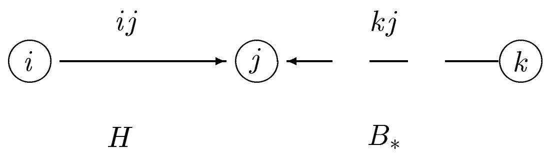


Abbildung II.20: Zur Folgerung

Nach Wahl von B_* muß B einen Kreis enthalten, der wegen der Innengradbedingung einfach und gerichtet ist. Also gibt es für jede Kante $ij \in H \backslash B_*$ einen einfachen, gerichteten Weg W von j nach i mit

$$W \subset B_*.$$

Lemma 2.3 *Sei (V, H) kritisch und sei (V, B_*) ein maximales Branching mit maximalem $|H \cap B_*|$. Dann gilt für alle Kreise C des kritischen Teilgraphen $|C \setminus B_*| = 1$.*

Beweis. Sei C ein gerichteter, einfacher Kreis des kritischen Graphen mit (in Abbildung II.21 fettgedruckten) Kanten $C \setminus B_* = \{u_1 v_1, \ldots, u_k v_k\}$, die entsprechend der Reihenfolge in C numeriert sind (Außenkreis im Bild II.21). Nach obiger Folgerung gilt $k \geq 1$ und B_* enthält für jede fette Kante einen einfachen, gerichteten (normal gezeichneten) Weg P_j von v_j nach u_j. Annahme: $k > 1$.

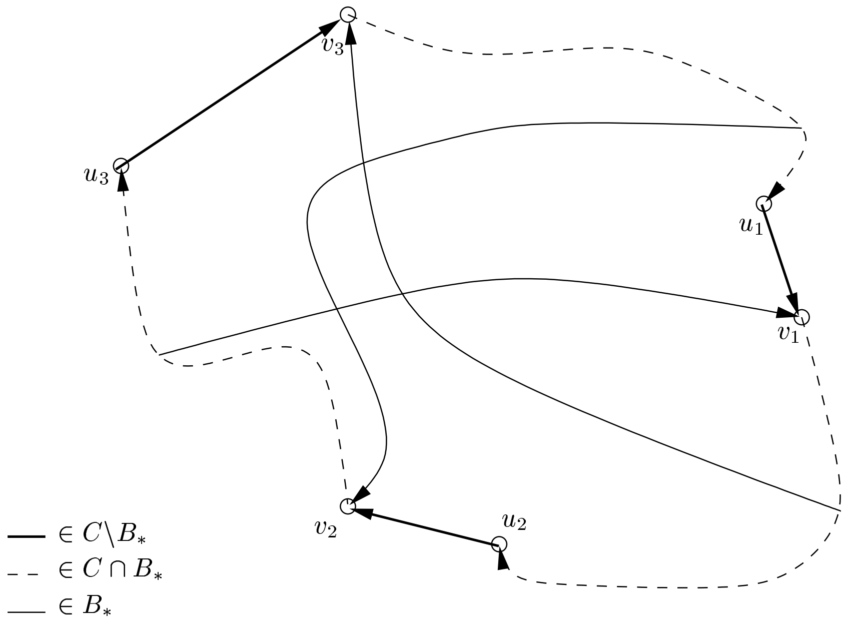


Abbildung II.21: Außen: Kreis C, Innen: Kreis in B_* liefert Widerspruch

Die gestrichelten Kreisstücke gehören zu B_*. Spätestens beim Anfangsknoten u_2 der nächsten fetten Kante verläßt der Weg P_1 den Kreis, beim Endknoten v_k der vorherigen fetten Kante mündet er letztlich wieder in den Kreis ein und läuft weiter bis zum Knoten u_1. Ein späteres Einmünden ist ausgeschlossen, da das Kreisstück zu B_* gehört. Also enthält B_* einen gerichteten, einfachen Weg von v_1 nach v_k. Analog findet man gerichtete

Wege in B_* von v_{j+1} nach v_j für alle $j \in \{1, 2, \ldots, k-1\}$ (fett gezeichnete Wege in Bild II.21). Daraus folgt, daß das Branching B_* einen Kreis enthält, also ein Widerspruch. $\square$

Folgerung: Sei (V, H) kritischer, nicht kreisfreier Teilgraph und sei (V, B_*) ein maximales Branching mit maximalem $|H \cap B_*|$. Ist C ein Kreis in (V, H), so gibt es nach obigem Lemma zwei Fälle:

1. Führt eine Kante $e \in B_*$ in den Kreis, so gilt $C \setminus B_* = \{\tilde{e}\}$,

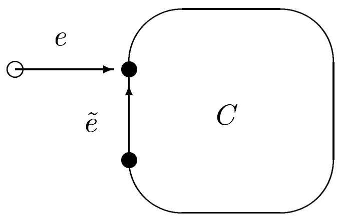


Abbildung II.22: Zur Folgerung

2. führt keine Kante aus B_* in den Kreis, so gilt $C \setminus B_* = \{e_C\}$ für eine Kante e_C mit minimalem Gewicht im Kreis.

Idee eines Verfahrens

Außerhalb von C kann B_* berechnet werden, wenn für alle Kanten $e \in \delta(\bar{C})$ die Gewichte geschickt verändert werden:

$$c_C(e) := c(e) - c(\tilde{e}) + c(e_C).$$

Dann gilt in beiden möglichen in der Folgerung diskutierten Fällen:

$$c_C(B_* \setminus C) = c(B_*) - c(C) + c(e_C).$$

Hat man außerhalb die Optimallösung bestimmt, erhält man aus der Fallunterscheidung die vollständige Lösung. Mit dieser Beobachtung läßt sich das Problem sukzessive auf kleinere Graphen reduzieren.

Schrumpfen von Graphen

Sei $W \subset V$. Der *geschrumpfte Graph* $G_W = (V_W, E_W)$ wird wie folgt definiert:

$$V_W := (V \setminus W) \cup \{v_W\}$$

Der neue Knoten v_W wird als ein "Pseudoknoten" bezeichnet und repräsentiert in dem geschrumpften Graphen alle Knoten, die in der Teilmenge W vorhanden waren. Entsprechend ergibt sich die neue Kantenmenge zu

$$E_W := \{ij \in E \mid i \notin W \vee j \notin W\}$$

Hierbei ist zu beachten, daß für Knoten $j \in W$ die Knotenbezeichnung v_W substituiert wird. Bei Implementation tritt diese Ersetzung nicht explizit auf, da Knoten und Pseudoknoten als Mengen gespeichert werden. Der Pseudoknoten ist also $SET(j)$, wobei $SET(j)$ im Originalgraphen mit j initialisiert wird.

Das Gegenteil von Schrumpfen heißt *Expandieren*.

Bemerkung: Beim Schrumpfen und Expandieren eines Kreises verschieben sich die Gewichte aller Branchings, die mit dem Kreis bis auf eine Kante übereinstimmen, nur um eine Konstante:

Sei (V, H) kritischer Teilgraph mit Kreis C. Wird der Kreis C in G geschrumpft und ist B ein Branching in G, dann wird B zu B_C geschrumpft. Falls $|C \setminus B| = 1$ gilt, ist B_C wieder ein Branching in G_C, denn für jede Kante $e \in B$, die in den Kreis C führt, gibt es eine Kante $\tilde{e} \in C$, die nicht zu B gehört. Also kann höchstens eine Kante von B in den Kreis C führen.

Ist umgekehrt B_C Branching in G_C, so kann B_C leicht zu einem Branching auf G erweitert werden. Sei e_C eine Kante mit minimalem Gewicht im Kreis C. Führt keine Kante aus B_C in den Kreis C, setze $B := B_C \cup (C \setminus e_C)$, sonst setze $B := B_C \cup (C \setminus \tilde{e})$, wobei $\tilde{e}$ die Kante des Kreises C ist, die den gleichen Endknoten wie die in den Kreis C führende Kante von B_C hat. In beiden Fällen gilt wieder

$$c(B) - (c(C) - c(e_C)) = c_C(B_C)$$

wobei c_C die Gewichtsfunktion auf dem geschrumpften Graphen ist.

Knotendisjunkte Kreise können gleichzeitig geschrumpft werden. Sukzessive wird so das Problem auf einen immer kleineren Graphen reduziert, bis die Lösung trivial ist. Dann wird diese Lösung wieder sukzessive expandiert, bis eine Lösung im Originalgraphen konstruiert ist.

Algorithmus 2.3 (Bestimmung maximaler Branchings)

Sei $G = (V, E)$ ein gerichteter Graph.

1. Bestimme einen kritischen Teilgraphen (V, H) in G;
2. Enthält (V, H) Kreise, schrumpfe alle diese Kreise;
   der geschrumpfte Graph sei wieder mit G bezeichnet;
   gehe nach 1.
3. Enthält (V, H) Pseudoknoten, expandiere alle Pseudoknoten;
4. das erweiterte Branching sei wieder mit (V, H) bezeichnet; gehe nach 3.

Ein kleines Beispiel zeigt, wie die gerichteten Kreise der kritischen Teilgraphen schrumpfen, bis ein kritischer Teilgraph ohne Kreise, d.h. ein optimales Branching, gefunden wird. Die kritischen Teilgraphen bestehen aus den dicken Pfeilen, die Pseudoknoten sind quadratisch gezeichnet.

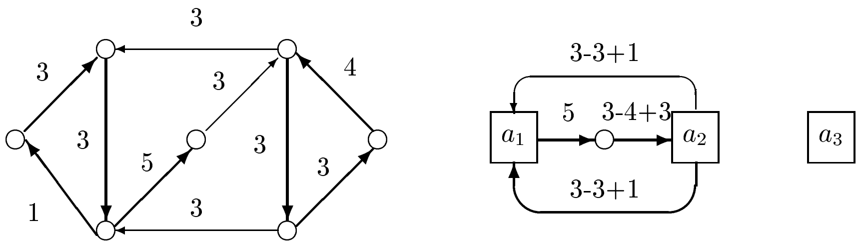


Abbildung II.23: Beispiel: Sukzessives Schrumpfen von Kreisen

Der erste kritische Teilgraph enthält zwei gerichtete Kreise, die simultan zu den Pseudoknoten a_1 und a_2 schrumpfen. Auf Kanten, die in die Pseudoknoten führen, ergeben sich reduzierte Gewichte durch Addition des Kreisgewichts und Subtraktion des Gewichts der in den gleichen Endknoten führenden Kreiskante. Der zweite kritische Teilgraph ist ein gerichteter Kreis, der zu dem Pseudoknoten a_3 schrumpft. Ein Graph mit einem Knoten ohne Kanten ist ein (triviales) optimales Branching. Anschließend müssen die Pseudoknoten wieder expandiert werden, um aus dem Branching im mehrfach geschrumpften Graphen ein optimales Branching im Originalgraphen zu gewinnen:

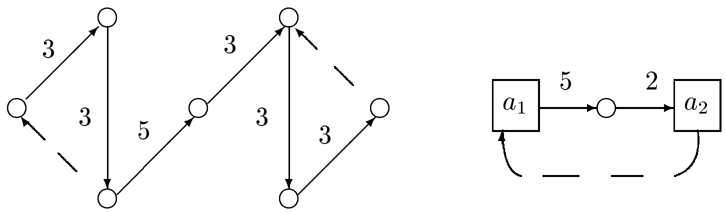


Abbildung II.24: Beispiel: Sukzessive Expansion von Pseudoknoten

Bei Expansion von a_3 wird die minimalgewichtete Kante a_2a_1 des zugehörigen geschrumpften Kreises gestrichen, da keine Kante des aktuellen Branchings im expandierten Knoten endet. Analog verfährt man bei der nachfolgenden Expansion von a_1, während bei Expansion von a_2 die Kante des zugehörigen Kreises gestrichen werden muß, in deren Endknoten bereits eine Kante des aktuellen Branchings endet. Da keine weiteren Pseudoknoten vorhanden sind, ist ein optimales Branching des Originalgraphen gefunden.

Alle Schritte sind in $O(m)$ durchführbar. Da höchstens n Schrumpfungen und danach höchstens n Expansionen möglich sind, ist die Komplexität des Verfahrens $O(nm)$. Mit geeigneten Datenstrukturen konnte Tarjan(siehe [30]) den Aufwand auf $O(m \log n)$ reduzieren.

3 Wege mit festem Anfangsknoten

Die Suche nach Wegen in gerichteten und ungerichteten Graphen gehört zu den grundlegenden kombinatorischen Problemen. In diesem Abschnitt sollen Wege ausgehend von einem fest gewählten Knoten konstruiert werden. Da die Numerierung der Knoten unerheblich ist, wählen wir etwa den Knoten $s := 1$ des Graphen $G = (V, E)$ als Startknoten.

Das einfachste Wegeproblem ist das Entscheidungsproblem, ob es einen Weg von s zu einem anderen Knoten t gibt. Gibt es einen Weg, so sucht man einen Weg kürzester Länge, d. h. einen Weg mit möglichst wenigen Kanten. Ausgehend vom Startknoten kann man den Graphen auf unterschiedliche Weise systematisch durchsuchen. Wir wollen dabei simultan Wege kürzester Länge zu allen erreichbaren Knoten konstruieren.

Beim ersten Verfahren werden sukzessive die Knoten mit Entfernung $0, 1, \cdots$ erreicht. Dazu wird in jeder Iteration ein erreichter Knoten der derzeitigen noch nicht abgearbeiten Entfernungstufe gewählt und alle seine Nachbarn, die noch nicht früher erreicht wurden, als Knoten der nächsten Entfernungsstufe markiert (BFS, “Breitensuche”).

Algorithmus 3.1 (Breadth-First-Search (BFS))

Sei $G = (V, E)$ ein ungerichteter (oder gerichteter) Graph. $N[k] := \{j \mid kj \in \delta(k)\}$ ist die Liste der zu $k \in V$ adjazenten Knoten. S ist die Menge der Knoten der derzeitigen Entfernungsstufe, T die der nächsten Entfernungsstufe. Die Markierungen $p(j)$ und $u(j)$ bezeichnen den Vorgänger des Knoten j auf dem derzeitigen Weg von 1 nach j und die Länge dieses Weges.

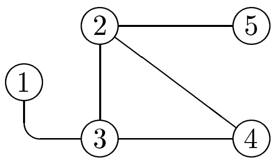
1. $i := 1$; $S := \{1\}$; $T := \emptyset$;
   $u(1) := 0$; $p(1) := 0$; $p(j) := -1$, für alle $2 \le j \le n$;

2. Falls $S = \emptyset$ stop;
   Für alle $k \in S$
       für alle $j \in N[k]$
           falls $p(j) = -1$ setze $u(j) := i$; $p(j) := k$; $T := T \cup \{j\}$;

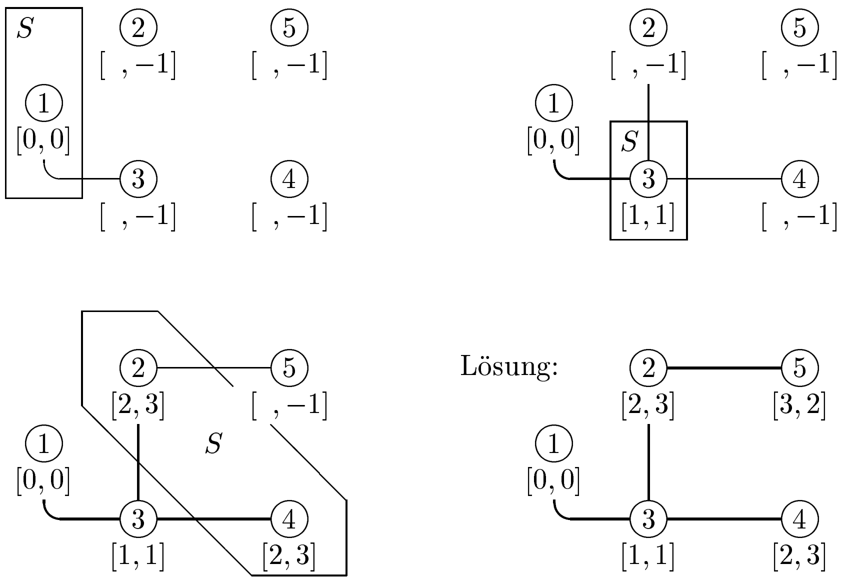
3. $i := i + 1$; $S := T$; $T := \emptyset$;
   gehe nach 2;

Beispiel

Für den ungerichteten Graphen



ergeben sich die folgenden Markierungen $[u, p]$:



Die letzte Iteration mit $S = \{5\}$ ändert nichts mehr. Anschließend bricht das Verfahren wegen $S = \emptyset$ ab.

Die Vorgängermarkierung p definiert einen Baum mit Wurzel 1. Die zu jedem Knoten j in diesem Baum enthaltenen Wege zum Startknoten sind Wege kürzester Länge. Man spricht daher auch von einem „Kürzesten Wegebaum“.

Im Schritt 2 des Verfahrens wird jede Kante höchstens zweimal betrachtet, nämlich von jedem Endknoten aus höchstens einmal. Die übrigen Operationen werden von dem hier anfallenden Aufwand majorisiert. Daher ist die Komplexität des Verfahrens $O(m)$.

Beim zweiten Verfahren wird ausgehend von einem bereits erreichten Knoten einer seiner Nachbarn, der noch nicht früher erreicht wurde, als erreicht markiert. Dann wiederholt man diesen Schritt für diesen Nachbarn. Besitzt der betrachtete Knoten keine bislang unerreichten Nachbarn, betrachtet man als nächsten seinen Vorgänger (DFS, “Tiefensuche”).

Algorithmus 3.2 (Depth-First-Search (DFS))

Sei $G = (V, E)$ ein ungerichteter Graph. $N[k]$ bezeichnet die Liste der zu $k \in V$ adjazenten Knoten, deren Markierung von k aus noch nicht versucht wurde. Nach einem Markierungsversuch, ob erfolgreich oder nicht, wird der Nachbarknoten aus $N[k]$ gestrichen. $num(j)$ gibt an, der wievielte erreichte Knoten der Knoten j ist. $p(j)$ bezeichnet den Vorgänger von j im Wegebaum.

1. $i := 1$; $num(1) = 1$; $k = 1$;
   $p(1) := 0$; $p(j) := -1$, für alle $2 \leq j \leq n$;

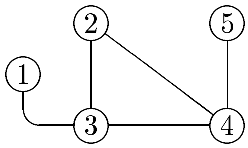
2. Falls $N[k] = \emptyset$ gehe nach 4; (alle Kanten von k benutzt)

3. Wähle $j \in N[k]$; $N[k] := N[k] \setminus \{j\}$;
   falls $p(j) = -1$ setze $p(j) := k$; $k := j$; $i := i + 1$; $num(k) := i$; (Vorwärtsschritt)
   gehe nach 2;

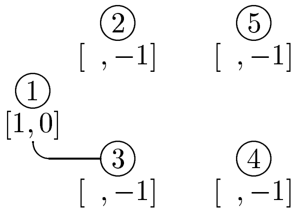
4. Falls $num(k) = 1$ stop; (alle Kanten der Wurzel benutzt)
   $k := p(k)$; gehe nach 2; (Rückschritt zu altem Knoten)

Beispiel

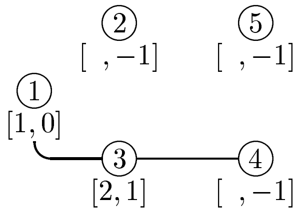
Für den ungerichteten Graphen



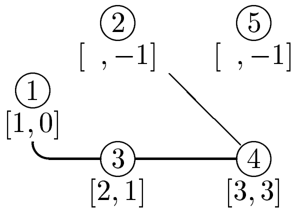
ergeben sich die folgenen Markierungen $[num, p]$:



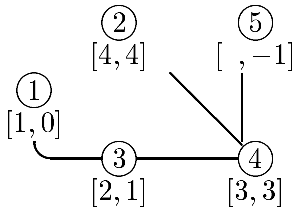
$k = 1$; $3 \in N[1]$



$k = 3$; $4 \in N[3]$



$k = 4$; $2 \in N[4]$



$k = 2$; $3 \in N[2]$ aber $p(3) \neq -1$

Auf den nächsten wegen $4 \in N[2]$ scheiternden Versuch folgt ein Rückschritt zum Knoten 4, da jetzt $N[2] = \emptyset$. Über die nächste Kante wird vom Knoten 4 aus der Knoten 5 erreicht und damit der DFS-Baum vervollständigt.

Sukzessive werden die Listen $N[4]$ und $N[3]$ geleert und zur Wurzel zurückgeschritten. Dann bricht das Verfahren in Schritt 4 ab.

Analog dem BFS-Verfahren besitzt das DFS-Verfahren die Komplexität $O(m)$. Die spezielle Struktur des DFS-Baumes ist bei Lösung verschiedener Probleme in Graphen hilfreich, z. B. bei der Bestimmung der topologische Sortierung und der (starken) Zusammenhangskomponenten (siehe Mehlhorn [16]).

Beide Verfahren können sowohl in ungerichteten als auch in gerichteten Graphen angewandt werden. Der einzige Unterschied liegt in der Liste der abgespeicherten Nachbarn. Im gerichteten Fall wird eine Kante höchstens einmal betrachtet, nämlich ausgehend von ihrem Anfangsknoten. Die konstruierten Wegebäume sind dann gerichtete Bäume.

Optimierungsproblem

Ist auf einem Graphen $G = (V, E)$ Graph zusätzlich eine Gewichtsfunktion $c: E \to \boldsymbol{R}$ auf den Kanten definiert, so kann man neben der Länge eines Weges das Gewicht $c(W)$ zur Bewertung eines Weges W heranziehen:

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

Für jeden Knoten j soll dann ein Weg von 1 nach j mit minimalem Gewicht gefunden werden. Solche Wege werden wieder kürzeste Wege genannt und im folgenden mit $KW_{1\to j}$ bezeichnet.

In den Anwendungen werden sehr unterschiedliche Bewertungen herangezogen. Um alle gleichzeitig behandeln zu können, läßt man Gewichte in angeordneten Monoiden zu. Eine solche angeordnete, algebraische Struktur kann man in kompakter Weise axiomatisch einführen.

Angeordnete Monoide $(H, \otimes, \leq)$

Die Menge H heißt angeordnetes Monoid bezüglich der Operation $\otimes$ und der Ordnungsrelation $\leq$, falls gilt:

1. $(H, \leq)$ ist vollständig angeordnet,
2. $(H, \otimes)$ ist ein *Monoid*, d.h. eine Halbgruppe mit einem neutralem Element,
3. Für alle $a, b, c \in H$ gilt die Verträglichkeitsregel
   $a \leq b \Rightarrow (a \otimes c \leq b \otimes c,$ und $c \otimes a \leq c \otimes b)$.

Das neutrale Element wird mit ϵ bezeichnet, $c(\emptyset) := \epsilon$.

Diese Axiome spiegeln Voraussetzungen über Gewichte wider, die für die sinnvolle Formulierung von Optimierungsaufgaben notwendig sind. Das Gewicht eines Weges $W = (e_1, e_2, \cdots, e_k)$ ist

$$c(W) := c(e_1) \otimes c(e_2) \otimes \cdots \otimes c(e_k).$$

Wegen der in Halbgruppen geltenden Assoziativitätsregel ist dieser Ausdruck auch ohne Angabe von Klammern wohldefiniert. Die Vollständigkeit der Ordnungsrelation garantiert, daß man die Gewichte aller Wege vergleichen kann. Die Verträglichkeitsregel schließlich erlaubt es, aus dem Vergleich von Teilwegen Schlüsse zu ziehen. Diese Bedingung legt außerdem die grundlegende Kopplung zwischen der Operation und der Ordnungsrelation fest und wird in dieser Form in jeder angeordneten algebraischen Struktur vorausgesetzt (siehe Zimmermann [31]).

Das neutrale Elemente ϵ erfüllt $a \otimes \epsilon = \epsilon \otimes a = a$ für alle $a \in H$ und wird als Gewicht leerer Wege, also für Wege der Länge 0, benutzt. Die nichtnegativen Gewichte werden in der Menge H_+ zusammengefaßt, d.h. $H_+ := \{a \mid a \geq \epsilon\}$.

Beispiele

Einige Monoide und zugehörige Anwendungen sind in der folgenden Tabelle zusammengestellt:

Monoid $(H, \otimes, \leq)$	neutrales Element ϵ	$c(ij)$, $ij \notin E$	Gewicht $c(W)$	Anwendung: minimal oder maximal
$(\boldsymbol{R}_+ \cup \{\infty\}, +, \leq)$	0	∞	$\sum_{e \in W} c(e)$	minimale Reisedauer
$([0,1], \cdot, \geq)$	1	0	$\prod_{e \in W} c(e)$	maximale Wahrscheinlichkeit freier Fahrt
$(\{0,1\}, \cdot, \geq)$	1	0	$\prod_{e \in W} c(e)$	Wegexistenz, max
$(\boldsymbol{R}_+ \cup \{\infty\}, \min, \geq)$	∞	0	$\min_{e \in W} c(e)$	maximale Kapazität von Telefonverbindungen

Ein maximales Element ∞ (maximal bezüglich der Anordnung des Monoids) kann man immer adjungieren, falls es im ursprünglichen Monoid fehlt. In der dritten Spalte der Tabelle ist es für die Anwendungen angegeben und wird als Gewichtung nichtvorhandener Kanten benutzt. Das maximale Element “absorbiert” alle anderen Elemente, d.h. es gilt $\infty \otimes a = a \otimes \infty = \infty$ für alle $a \in H$.

Aufgabe 3.1
Sei $G = (V, E)$ ein gerichteter Graph mit nichtnegativen Gewichten, kurz $c \geq \epsilon$. Bestimme kürzeste Wege $KW_{1 \to j}$ für alle $j \neq 1$.

Algorithmus 3.3 (Dijkstra-Verfahren)

Sei $G = (V, E)$ ein gerichteter Graph mit nichtnegativen Gewichten $c \geq \epsilon$. $N[k]$ bezeichnet die Liste der zu $k \in V$ adjazenten Knoten. Die Markierungen $p(j)$ und $u(j)$ bezeichnen den Vorgänger des Knoten j auf dem derzeitigen Weg von 1 nach j und das Gewicht dieses Weges. Die Knotenmenge T enthält alle Knoten, die derzeitig noch nicht in den kürzesten Wegebaum aufgenommen werden konnten.

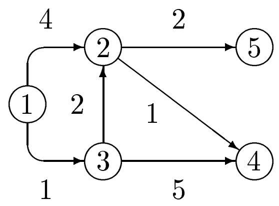
1. $T := V \setminus \{1\}$; $u(1) := \epsilon$; $p(1) := 0$;
   $u(j) := \infty$, $p(j) := 1$, für alle $2 \leq j \leq n$;
   $u(j) := c(1j)$, für alle $j \in N[1]$;

2. Bestimme $k \in T$ mit $u(k) := \min\{u(j) \mid j \in T\}$;
   $T := T \setminus \{k\}$;

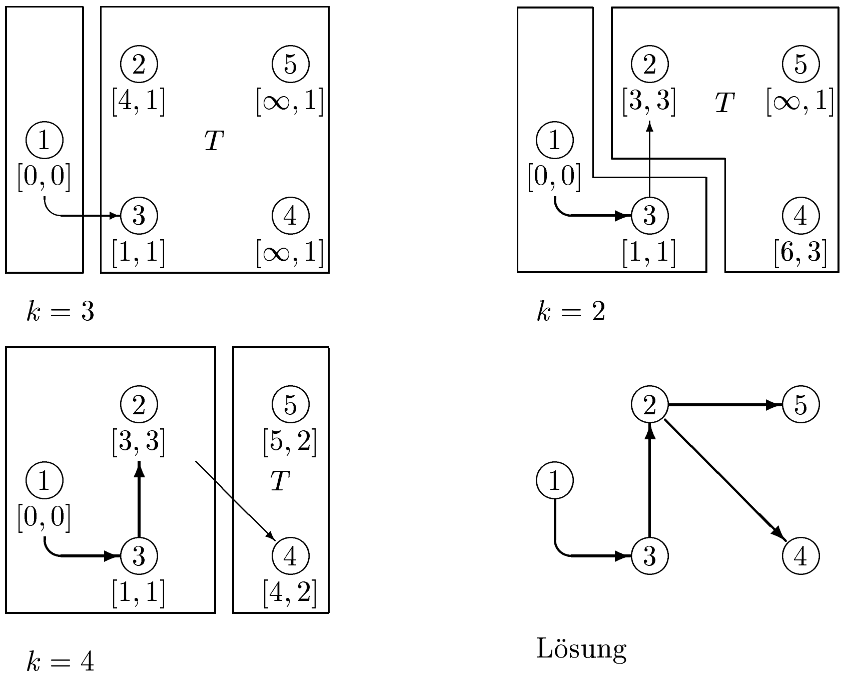
3. Falls $T = \emptyset$ stop;
   Für alle $j \in N[k]$
       falls $j \in T$ und $u(j) > u(k) \otimes c(kj)$
           setze $u(j) := u(k) \otimes c(kj)$; $p(j) := k$;
   gehe nach 2;

Beispiel: **($\otimes := +$)**

Für den gerichteten Graphen



ergeben sich die folgenden Markierungen $[u, p]$:



Die letzte Iteration mit $k = 5$ erfolgt bei unveränderten Marken und liefert die angegebene Lösung.

Lemma 3.1 *Vor Schritt* 2 *des Dijkstra-Verfahrens gilt stets:*

(1) für alle $j \in \bar{T}$: $u(j)$ ist Gewicht eines $KW_{1\to j}$,

(2) für alle $j \in T$: $u(j)$ ist Gewicht eines $KW_{1\to j}$ innerhalb der Knotenmenge $\bar{T} \cup j$,

(3) für alle $j \in V$: $p: V \to \bar{T} \cup 0$ definiert Wege mit Gewicht $u(j)$.

Beweis. Der Beweis erfolgt induktiv über die Iterationen. Nach Schritt 1. vor der ersten Durchführung einer Iteration sind alle Aussagen (1) – (3) richtig. Nach Induktionsannahme seien sie auch vor Beginn einer Iteration, mit den Bezeichnungen T, u, p, vor Schritt 2. gültig. Die in der Iteration geänderten Größen werden mit T', u', p' bezeichnet. Bei Durchführung von Schritt 2. wird nun $k \in T$ bestimmt.

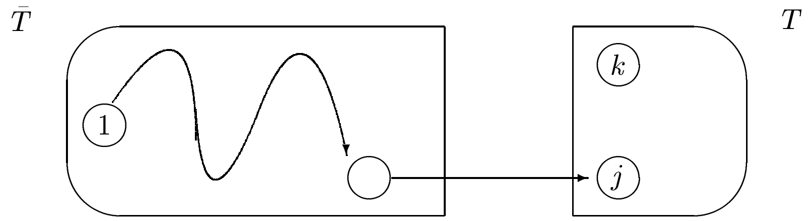


Abbildung II.25: zum Lemma

1. (1) gilt nach Schritt 2. auch für k:
   Sei W Weg $1 \to k$. Falls $W \subseteq \bar{T} \cup k$, so gilt wegen (2) $c(W) \geq u(k)$. Anderenfalls sei j der erste Knoten von W mit $j \notin \bar{T} \cup k$. Dann gilt für den Teilweg $W' \subseteq W$ von 1 bis j wegen Nichtnegativität der Gewichte, wegen (2) und nach Wahl von k die Ungleichungskette $c(W) \geq c(W') \geq u(j) \geq u(k)$.

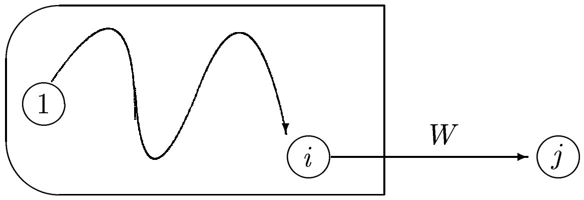


Abbildung II.26: zum Lemma

2. (2) gilt nach Schritt 3. für $T' := T \setminus k$.
   Sei $j \in T \setminus k$ und sei W ein Weg $1 \to j$ innerhalb der Knotenmenge $\bar{T} \cup \{k, j\}$. Sei i der unmittelbare Vorgänger von j auf dem Weg W. Da nach (1) $u(i)$ das minimale Gewicht eines Weges nach i ist, gilt $c(W) \geq u(i) \otimes c(ij)$.

   Es bleibt zu zeigen, daß gilt: $u(i) \otimes c(ij) \geq u'(j)$. Falls $i \neq k$, gilt wegen (2) vor der Iteration bereits $u(i) \otimes c(ij) \geq u(j) \geq u'(j)$. Mit $u \geq u'$, folgt die Behauptung. Anderenfalls ist $i = k$. In Schritt 3. wird $u(j)$, falls $u(k) \otimes c(kj) < u(j)$ auf $u'(j) := u(k) \otimes c(kj)$ abgesenkt, so daß wiederum die Behauptung gilt.

3. p erfüllt (3), weil stets eine Änderung von u eine passende Änderung von p bewirkt.

□

Satz 3.2 *Bei Abbruch des Dijkstra-Verfahrens gilt:*

1. *für alle $j \in V$ ist $u(j)$ das Gewicht eines $KW_{1 \to j}$,*

2. *p definiert einen Kürzesten Wegebaum,*

3. die Folge der aus T eliminierten Knoten $1{=}k_1, k_2, \ldots k_n$ erfüllt $u(k_1) \leq \ldots \leq u(k_n)$.

Beweis. 1. und 2. folgen unmittelbar aus Lemma 3.1. 3. wird zur Übung empfohlen.

Zeitkomplexität

Der Startschritt 1. erfordert $O(n)$ Zuweisungen. Schritt 2. besteht aus einer fallenden Anzahl $n-2+n-3+\ldots = (n-1)\frac{n-2}{2}$ Vergleichen. Schritt 3. enthält $n-2+n-3+\ldots = (n-1)\frac{n-2}{2}$ Vergleiche und $n-2+n-3+\ldots = (n-1)\frac{n-2}{2}$ $\otimes$-Operationen. Die Zeitkomplexität des Dijkstra-Verfahrens ist daher $O(n^2)$. Bessere Abschätzungen ergeben sich, wenn man geeignete Datenstrukturen, wie Heaps und Priority Queues, einsetzt und analysiert (siehe Mehlhorn [16]).

Bemerkung: Das Verfahren ist ohne Schwierigkeiten auf ungerichtete Graphen anwendbar, wenn man die entsprechende Definition der Adjazenzlisten $N[k]$ benutzt. Dagegen ist die Nichtnegativität der Gewichte notwendig. Gibt es negative Gewichte, so versagt das Verfahren. Das Optimierungsproblem bleibt allerdings sinnvoll, solange der Graph keine Kreise mit negativem Gewicht enthält. Anderenfalls kann man solche negativen Kreise beliebig oft durchlaufen, um beliebig kurze Wege zu erhalten.

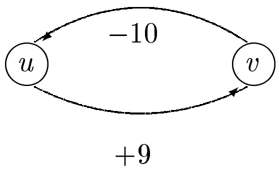


Abbildung II.27: Negativer Kreis mit zwei Kanten

Aufgabe 3.2
Sei $G=(V,E)$ ein gerichteter Graph mit Gewichten $c\colon E \to H$, der keine negativen Kreise enthält, d.h. $c(C) \geq \epsilon$ für alle Kreise in G. Bestimme kürzeste Wege $KW_{1\to j}$ für alle $j \neq 1$.

Algorithmus 3.4 (Bellman-Ford-Verfahren)
Sei $G=(V,E)$ ein gerichteter Graph ohne negative Kreise. Die Markierungen $u^k(j)$ bezeichnen das Gewicht des in der k-ten Iteration bestimmten Weges von 1 nach j.

1. $u^1(1) := \epsilon,\ u^1(j) := \begin{cases} c(1j) & 1j \in E \\ \infty & \textit{anderenfalls} \end{cases}$

2. Für $k = 1, 2, \ldots, n-2$
für alle $j \in V$

$$(*) \quad u^{k+1}(j) := \min\{u^k(j), \min_{ij \in \delta(\bar{j})} u^k(i) \otimes c(ij)\}$$

Lemma 3.3 *$u^k(j)$ ist minimales Gewicht eines Weges $1 \to j$ mit maximal k Kanten.*

Beweis. Der Beweis erfolgt wieder über die Iterationen. Im Startschritt wird offenbar das minimale Gewicht für Wege mit einer Kante zugewiesen. Für den Induktionsschritt sei W ein Weg $1 \to j$ mit $|W| \leq k+1$. Falls $|W| \leq k$, so ist nach Induktionsvoraussetzung $c(W) \geq u^k(j)$. Anderenfalls gilt $|W| = k+1$. Sei i der unmittelbare Vorgänger von j auf W und sei der Teilweg $1 \to i$ mit W' bezeichnet.

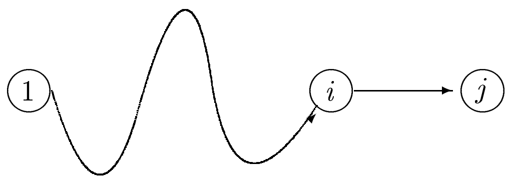


Abbildung II.28: Zum Beweis des Bellman-Ford-Verfahrens

Dann ist die Induktionsvoraussetzung auf W' anwendbar und man erhält

$$c(W) = c(W') \otimes c(ij) \geq u^k(i) \otimes c(ij)$$

Also gilt nach Durchführung der Iteration $c(W) \geq u^{k+1}(j)$.

Wege mit Gewicht $u^{k+1}(j)$ ergeben sich rekursiv, wenn man die p-Markierung entsprechend mitrechnet. Also gilt „=“. □

Dieses Lemma gilt sinngemäß, wenn weitere Iterationen durchgeführt werden.

Satz 3.4 *$u^{n-1}(j)$ ist Gewicht eines $KW_{1\to j}$, falls G keine negativen Kreise enthält.*

Beweis. Die Annahme sichert die Existenz Kürzester Wege mit höchstens $n-1$ Kanten. □

Bemerkung:

1. Da Schritt 2 in $O(m+n)$ ausgeführt werden kann, ist die Zeitkomplexität des Verfahrens $O(nm)$.

2. Da die Iteration nur u^k und u^{k+1} benötigt, genügen zwei Felder als Speicherplatz. Sobald eine Iteration keine Änderung mehr liefert, d.h. $u^k = u^{k+1}$ gilt, sind die Gewichte der kürzesten Wege bestimmt.

3. Analog zum Dijkstra-Verfahren kann man eine Vorgängermarkierung p mitrechnen, die den Kürzesten Wegebaum liefert. Dazu wird im Startschritt $p(1) := 0$ und $p(j) := +1$, für alle $1j \in E$ gesetzt. Die übrigen Knoten werden mit $p(j) := -1$ als noch nicht erreichbar gekennzeichnet. Bei Änderung der Gewichte durch die Formel $(*)$ im Iterationsschritt wird $p(j) := i$ entsprechend geändert.

4. Das Verfahren ist nicht auf ungerichtete Graphen übertragbar, da eine einzelne negative Kante bereits zu einem negativen Kreis im zugehörigen gerichteten Graphen führt.

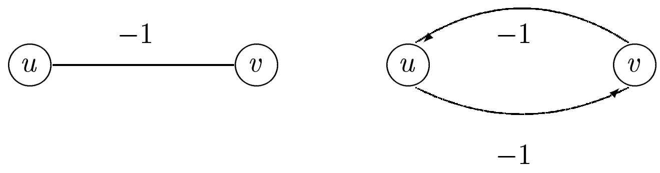


Abbildung II.29: negative ungerichtete Kanten führen auf negative Kreise

Satz 3.5 (Negative Kreise) *H sei eine angeordnete kommutative Gruppe. G enthalte gerichtete Wege von Knoten 1 zu allen anderen Knoten. Dann gibt es einen negativen Kreis genau dann, wenn mindestens ein Knoten j^* mit $u^n(j^*) < u^{n-1}(j^*)$ existiert.*

Beweis. Gibt es keine negativen Kreise, so gilt nach Satz 3.4 $u^{n-1} = u^n$. Da nach Voraussetzung jeder Knoten erreichbar ist, ist u^{n-1} endlich. Falls $u := u^{n-1} = u^n$, gilt $u(j) \leq u(i) \otimes c(ij)$ für alle $i, j \in V$. Längs eines Kreises $C = i_1, \ldots, i_\mu$ ergibt sich daraus

$$a := u(i_1) \otimes \ldots \otimes u(i_\mu) \leq (u(i_\mu) \otimes c(i_\mu i_1)) \otimes \ldots \otimes (u(i_{\mu-1}) \otimes c(i_{\mu-1} i_\mu))) = a \otimes c(C),$$

weil in kommutativen Gruppen Faktoren umgeordnet werden können. Da man in einer Gruppe kürzen kann, folgt $\epsilon \leq c(C)$. Also ist jeder Kreis nichtnegativ. □

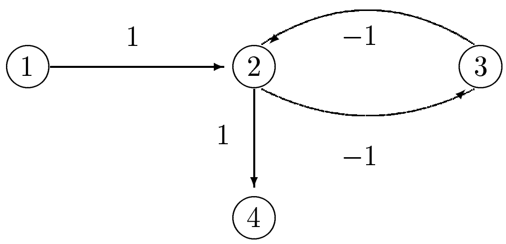


Abbildung II.30: Graph mit negativem Kreis

Unter den Voraussetzungen des Satzes 3.5 findet man einen negativen Kreis mit Hilfe der Markierung p. Für den abgebildeten Graphen mit einem negativen Kreis ergeben sich die in der Tabelle zusammengestellten Markierungen u, p:

Knoten	1	2	3	4
u^3	0	-1	0	2
u^4	0	-1	-2	0
p	0	3	2	2

Man erkennt, daß die Bedingung von Satz 3.5 für die Knoten $3, 4$ erfüllt ist. Ausgehend von einem dieser Knoten, etwa von 4, findet man über p zurück nach $2, 3, 2, \ldots$ und hat einen negativen Kreis gefunden. Dieser negative Kreis hat tatsächlich die Absenkung von u in der letzten Iteration verursacht.

4 Wege zwischen allen Knoten

Die Suche nach Wegen zwischen allen Knoten eines Graphen kann man auf die Suche nach Wegen mit festem Anfangsknoten zurückführen. Das Bellman-Ford-Verfahren etwa liefert dann n verschiedene Wegebäume, einen für jeden Knoten des Graphen, in $O(mn^2)$. Wir werden sehen, wie man alle gesuchten Wege mit weniger Speicherplatz und in kürzerer Zeit mit einer direkten Methode konstruieren kann.

Sei $G = (V, E)$ ein gerichteter Graph mit Knoten ($V = \{1, 2, \ldots, n\}$) und bezeichne P_{ij} die Menge aller Wege von i nach j für alle Knoten $i, j \in V$. Als *transitive Hülle* von $G = (V, E)$ bezeichnet man den durch $\bar{E} := \{ij \mid P_{ij} \neq \emptyset, i, j \in V\}$ gegebenen Graphen $T(G) = (V, \bar{E})$. Um die transitive Hülle zu bestimmen, müssen wir also herausfinden, zwischen welchen Knoten gerichtete Wege existieren.

Ist auf den Kanten des Graphen zusätzlich eine Bewertung $c : E \to R$ gegeben, so sollen jeweils kürzeste Wege bestimmt werden. Wie im letzten Abschnitt finden sich in den Anwendungen sehr unterschiedliche Bewertungen, die man durch Betrachtung von Gewichten in einer hinreichend allgemeinen algebraischen Struktur einheitlich behandeln kann.

Semiringe $(R, \oplus, \otimes)$

Die Menge R heißt *Semiring* bezüglich der additiven bzw. multiplikativen Operationen $\oplus, \otimes$, falls gilt:

1. $(R, \oplus)$ ist eine kommutatives Monoid,
2. $(R, \otimes)$ ist ein Monoid,
3. für alle $a, b, c \in R$ gelten die Distributivgesetze
   $(a \oplus b) \otimes c = (a \otimes c) \oplus (b \otimes c)$ und $c \otimes (a \oplus b) = (c \otimes a) \oplus (c \otimes b)$,
4. Das neutrale Element der additiven Operation ist ein Nullelement.

Das neutrale Element der Addition, mit 0 bezeichnet, ist ein *Nullelement*, d. h. für alle $a \in R$ gilt $0 \otimes a = a \otimes 0 = 0$. Das neutrale Element der Multiplikation wird zur Unterscheidung *Einselement* genannt und mit 1 bezeichnet.

Wie im letzten Abschnitt spiegeln die Axiome die sinnvollen Voraussetzungen entsprechender Optimierungsaufgaben wider und das Gewicht eines Weges $W = (e_1, e_2, \cdots, e_k)$ im Graphen ist

$$c(W) := c(e_1) \otimes c(e_2) \otimes \cdots \otimes c(e_k).$$

Statt der Anordnung $\leq$ von R wird hier nur eine additive Verknüpfung vorausgesetzt. Angeordnete Monoide liefern spezielle Beispiele. Ist $(R, \otimes, \leq)$ ein angeordnetes Monoid mit maximalem Element ∞, so ist $(R, \oplus, \otimes)$ mit

$$a \oplus b := \min(a, b),$$

für alle $a, b \in R$, ein Semiring mit Nullelement ∞.

Um den Wert einer Menge von Wegen, wie z. B. der Menge P_{ij}, festzulegen, muß die additive Verknüpfung für unendlich viele Summanden erkärt sein. Da jedes P_{ij} abzählbar ist, genügen Summen mit abzählbar unendlich vielen Summanden. Ist I eine abzählbare Indexmenge und $(a_i, i \in I)$ ein Tupel mit $a_i \in R$ für $i \in I$, so wird diese Summe mit $\sum_{i \in I} a_i$ bezeichnet.

Abgeschlossene Semiringe

Ein Semiring heißt *abgeschlossen*, wenn Summen $\sum_{i \in I} a_i \in R$ für abzählbare Indexmengen I und für $a_i \in R$ mit $i \in I$ wohldefiniert sind und die folgenden Eigenschaften haben:

1. für endliche Indexmengen $I = \{i_1, \ldots i_k\}$ gilt:

$$\sum_{i \in I} a_i = a_{i_1} \oplus \ldots \oplus a_{i_k},$$

2. für alle Partitionen $(I_j, j \in J)$ von I, d.h. für $I = \dot{\bigcup}_{j \in J} I_j$, gilt:

$$\sum_{i \in I} a_i = \sum_{j \in J} (\sum_{i \in I_j} a_i),$$

3. das Distributivgesetz gilt für alle Summen $\sum_{i \in I} a_i$, $\sum_{j \in J} b_j$:

$$(\sum_{i \in I} a_i) \otimes (\sum_{j \in J} b_j) = \sum_{i \in I} (\sum_{j \in J} a_i \otimes b_j).$$

Beispiele

1. In $B = \{0, 1\}$ ist $0 < 1$. Die entsprechende additive Operation auf B ist max, d.h. $a \oplus b := \max(a, b)$ mit Nullelement 0. Analog sind abzählbar unendliche Summen definiert. Zusammen mit der multiplikativen Operation $a \otimes b := \min(a, b)$ und dem Einselement 1 wird der resultierende abgeschlossene Semiring B als *Bool'scher Semiring* bezeichnet. Zur Bestimmung der transitiven Hülle bewertet man die Kanten des Graphen mit Elementen des Bool'schen Semirings.

2. Die erweiterten reellen Zahlen $\boldsymbol{R} \cup \{\infty, -\infty\}$ bilden mit der additiven Operation „min“, mit Nullelement ∞, den Summen $\sum_{i \in I} a_i := \inf\{a_i \mid i \in I\}$, der multiplikativen Operation „+“ und dem Einselement 0 einen abgeschlossener Semiring. Man beachte, daß aus den Axiomen die Regel $\infty + (-\infty) = \infty$ folgt. Zur Bestimmung der üblichen kürzesten Wege bewertet man die Kanten des Graphen mit Elementen der erweiterten reellen Zahlen.

In abgeschlossenen Semiringen spielen abzählbar unendliche Summen der Form $1 \oplus a \oplus a^2 \oplus a^3 \ldots$ eine wichtige Rolle. Wir definieren daher für alle $a \in R$ den Abschlußoperator „*“ durch

$$a^* := \sum_{i \in \boldsymbol{Z}_+} a^i.$$

Offenbar gilt dann für alle $a \in R$ die Regel $1 \oplus (a^* \otimes a) = 1 \oplus (a \otimes a^*) = a^*$.

Beispiele

1. Der Abschlußoperator des Bool'schen Semirings $B = \{0, 1\}$ ist

$$a^* = \max\{1, a, \min(a, a), \ldots\} = 1.$$

2. In den erweiterten reellen Zahlen $\boldsymbol{R} \cup \{\infty, -\infty\}$ ergibt sich:

$$a^* = \inf\{0, a, a + a, \ldots\} = \begin{cases} 0 & a \geq 0 \\ -\infty & a < 0 \end{cases}$$

Gewicht einer Menge von Wegen

Sei $(R, \oplus, \otimes)$ ein abgeschlossener Semiring, $G = (V, E)$ ein gerichteter Graph mit Knoten $V = \{1, \ldots, n\}$ und Bewertung $c: E \to R$. Bezeichnet $c(p) := c(e_1) \otimes \ldots \otimes c(e_k)$ das Gewicht eines Weges $p = (e_1, \ldots, e_k)$ in G, so ist das Gewicht einer Menge P von Wegen $c(P) := \sum_{p \in P} c(p)$. Das Gewicht der leeren Menge ist das Nullelement.

Aufgabe 4.1
Bestimme $a_{ij} := c(P_{ij})$ *für alle* $i, j \in V$.

Rekursiver Ansatz

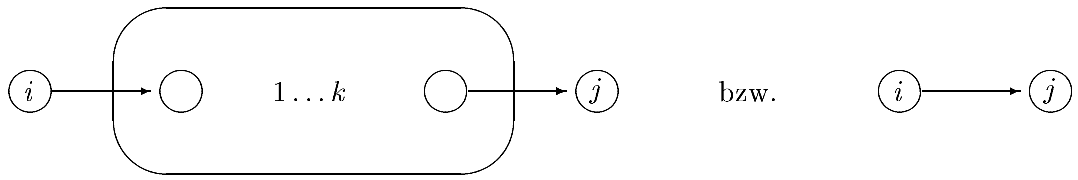


Abbildung II.31: Ansatz zum Algorithmus nach Kleene

Wir betrachten P_{ij}^k, definiert als die Menge aller Wege von i nach j, deren Zwischenknoten nur innerhalb von $\{1, 2, \ldots, k\}$ liegen. Die Mengen der Wege ohne Zwischenknoten, also P_{ij}^0 für $i, j \in V$, sind unmittelbar durch die Kantenmenge E gegeben. Insbesondere gilt

$$c(P_{ij}^0) = \begin{cases} 0 & ij \notin E \\ c(ij) & ij \in E \end{cases} \tag{4.1}$$

Die gesuchten Mengen sind $P_{ij} = P_{ij}^n$, für $i, j \in V$. Für $0 < k$ gilt die triviale Rekursion

$$c(P_{ij}^k) = c(P_{ij}^{k-1}) \oplus c(P_{ij}^k \setminus P_{ij}^{k-1}).$$

Jeder Weg $p \in P_{ij}^k \setminus P_{ij}^{k-1}$ enthält den Zwischenknoten k mindestens einmal. Tritt k genau

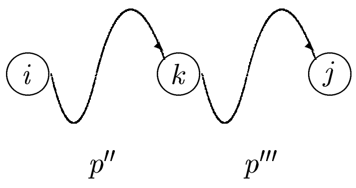


Abbildung II.32: $l = 1$

einmal als Zwischenknoten auf, so gilt (siehe Abbildung II.32)

$$c(p) = c(p'') \otimes 1 \otimes c(p'''),$$

mit $p'' \in P_{ik}^{k-1}$ und $p''' \in P_{kj}^{k-1}$. Enthält p den Zwischenknoten l-fach, $1 < l$, so gilt (siehe

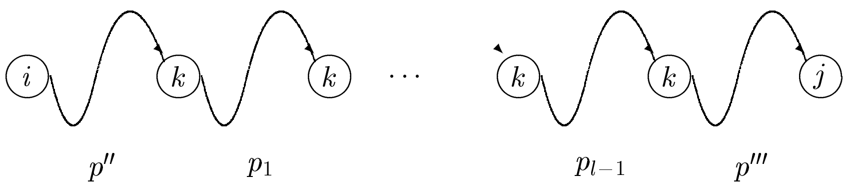


Abbildung II.33: $l > 1$

Abbildung II.33)

$$c(p) = c(p'') \otimes c(p_1) \otimes \ldots \otimes c(p_{l-1}) \otimes c(p'''),$$

mit $p'' \in P_{ik}^{k-1}$, $p''' \in P_{kj}^{k-1}$ und $p_\mu \in P_{kk}^{k-1}$ für alle $\mu = 1, \ldots, l-1$.

Wir bezeichnen das Gewicht aller Wege von k nach k, deren Zwischenknoten nur in $\{1, 2, \ldots, k-1\}$ liegen, mit $a := c(P_{kk}^{k-1})$. Da man Summanden beliebig umsortieren darf, ergibt sich mit Hilfe des Distributivgesetzes

$$c(P_{ij}^k) = c(P_{ij}^{k-1}) \oplus \left[c(P_{ik}^{k-1}) \otimes (1 \oplus a \oplus a^2 \ldots) \otimes c(P_{kj}^{k-1}) \right]$$

woraus unmittelbar die Rekursionsformel

$$c(P_{ij}^k) = c(P_{ij}^{k-1}) \oplus \left[c(P_{ik}^{k-1}) \otimes a^* \otimes c(P_{kj}^{k-1}) \right] \tag{4.2}$$

folgt. Für $k = n$ liefert die Rekursion $c(P_{ij})$, für alle $i, j \in V$.

Algorithmus 4.1 ((Kleene-Verfahren, 1956))

1. *Für alle* $i, j \in V$ *setze* $a_{ij} := \begin{cases} 0 & ij \notin E, \\ c(ij) & ij \in E; \end{cases}$

2. *Für* $k = 1, 2, \ldots, n$ *setze* $\alpha := a_{kk}^*$ *und*

   (a) *für* $i \in V$ *setze* $a_{ik} := a_{ik} \otimes \alpha$;

   (b) *für* $i, j \in V \setminus k$ *setze* $a_{ij} := a_{ij} \oplus (a_{ik} \otimes a_{kj})$;

   (c) *für* $j \in V \setminus k$ *setze* $a_{kj} := \alpha \otimes a_{kj}$;

Für die Gültigkeit des Algorithmus von Kleene ist die Reihenfolge der Zuweisungen in Schritt 2 wesentlich.

Satz 4.1 *Der Algorithmus von Kleene liefert mit* $O(n^3)$ *Semiring-Operationen und* $O(n)$ *Abschlußoperationen die Matrix* A *mit den Elementen* $a_{ij} = c(P_{ij})$ *für alle* $i, j \in V$.

Beweis. Wir zeigen, daß nach Durchführung der k-ten Iteration stets $a_{ij} = c(P_{ij}^k)$ gilt. Für $k = 0$ folgt dies aus Gleichung (4.1) des rekursiven Ansatzes. Der Schluß von $k-1$ auf k erfolgt mit Hilfe der Rekursionsformel (4.2) und der Eigenschaften der Abschlußoperation:

1. ad (a): $a_{ik} = a_{ik} \oplus (a_{ik} \otimes a_{kk}^* \otimes a_{kk}) = a_{ik} \otimes [1 \oplus (a_{kk}^* \otimes a_{kk})] = a_{ik} \otimes a_{kk}^*$.

2. ad (b): Da a_{ik} aus (a) übernommen wird, ist dies genau die Rekursionsformel.

3. ad (c): Analog zu (a).

Schritt 1 besteht aus $O(n^2)$ Zuweisungen. Jede der n Iterationen erfordert $O(n^2)$ Operationen der Form $\otimes, \oplus$, sowie eine Abschlußoperation. □

Abschließend wollen wir das allgemeine Verfahren zur Berechnung der transitiven Hülle und zur Berechnung kürzester Wege spezialisieren.

Konstruktion der Transitiven Hülle, Warshall 1962

Die transitive Hülle $T(G) := (V, \bar{E})$ eines gerichteten Graphen $G = (V, E)$ kann mit Hilfe des Kleene-Algorithmus im Boole'schen Semiring erzeugt werden. Als Kantengewichte auf dem vollständigen gerichteten Graphen $(V, V \times V)$ wählen wir

$$c(ij) := \begin{cases} 1 & ij \in E, \\ 0 & ij \notin E. \end{cases}$$

für alle $i \neq j$. Dann entsprechen die Wege p in G' mit $c(p) = 1$ den Wegen in G. Für eine Menge P von Wegen in G gilt

$$c(P) = \begin{cases} 1 & P \neq \emptyset, \\ 0 & P = \emptyset. \end{cases}$$

Insbesondere können wir wegen $a_{ij} = c(P_{ij})$ die transitive Hülle ablesen:

$$ij \in \bar{E} :\Leftrightarrow a_{ij} = 1$$

Das Verfahren läßt sich wegen $a^* = \max(1, a, \min(a, a), \ldots) = 1$ wesentlich vereinfachen:

1. Für alle $i, j \in V$ setze $a_{ij} := \begin{cases} 1 & ij \in E, \\ 0 & ij \notin E; \end{cases}$

2. Für $k = 1, \ldots, n$, für alle $i, j \in V$ mit $a_{ij} = 0$:
   falls $a_{ik} = a_{kj} = 1$ setze $a_{ij} := 1$;

Der verbleibende Aufwand besteht in $O(n^3)$ Vergleichen und Zuweisungen. Die Komplexität des Verfahrens nach Warshall läßt sich mit Hilfe schneller Matrixmultiplikationen theoretisch verbessern. Damit kann die transitive Hülle in $O(n^{\log 7}(\log n)^2) = O(n^{2.82})$ konstruiert werden (siehe [16]). Der essentiell asymptotische Charakter solcher Schranken hat jedoch zur Folge, daß die resultierenden Verfahren in der Regel wenig praktikabel sind.

Konstruktion kürzester Wege, Floyd 1962

Die kürzesten Wege in einem gerichteten Graphen $G = (V, E)$ mit Bewertung $c : E \to \boldsymbol{R}$ erhält man mit Hilfe des Algorithmus von Kleene in den erweiterten reellen Zahlen, wenn man die Kantengewichte auf den vollständigen Graphen $G' = (V, V \times V)$ überträgt. In G fehlende Kanten werden in G' mit ∞ bewertet. Für eine Menge P von Wegen gilt dann $c(P) = \inf_{p \in P} c(p)$. Insbesondere kann man das Gewicht kürzester Wege aus den Gewichten der Mengen P_{ij} ablesen:

1. falls $c(P_{ij}) \in \boldsymbol{R}$, so ist $c(P_{ij})$ das Gewicht eines kürzesten Weges $i \to j$ in G,

2. falls $c(P_{ij}) = \infty$, so existiert kein Weg $i \to j$ in G,

3. falls $c(P_{ij}) = -\infty$, so liegt ein negativer Kreis auf einem Weg $i \to j$ in G.

Wegen der einfachen Berechnung des Abschlußoperators

$$a^* = \begin{cases} 0 & a \geq 0 \\ -\infty & a < 0 \end{cases}$$

können wir die Schritte (2a)-(2c) des Kleene-Algorithmus effizienter fassen. Falls $\alpha = a^*_{kk} = 0$, so berechnet man die neuen Koeffizienten aus:

$$a_{ij} := \min(a_{ij}, a_{ik} + a_{kj})$$

für alle $i, j \in V \setminus k$. Anderenfalls ist $\alpha = -\infty$. Fast jede Operation mit $-\infty$ hat $-\infty$ als Ergebnis, einzige Ausnahme ist $\infty + (-\infty) = \infty$. Eine Änderung eines von $-\infty$ verschiedenen Koeffizienten, und zwar zu $-\infty$, ergibt sich also genau dann, wenn die anderen beteiligten Koeffizienten endlich sind. In diesem Fall sind also nur Abfragen erforderlich. Die Details finden sich im unten aufgeführten Algorithmus von Floyd.

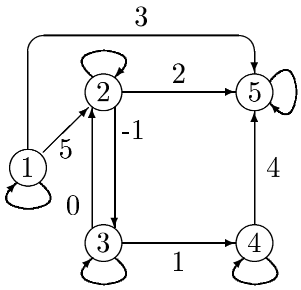
Der verbleibende Aufwand besteht in $O(n^3)$ Vergleichen, Additionen und Zuweisungen.

Algorithmus 4.2 (Das Verfahren von Floyd)

1. Für $i, j \in V$ setze $a_{ij} := \begin{cases} c(ij) & i \neq j,\ ij \in E \\ \infty & i \neq j,\ ij \neq E \\ 0 & i = j \end{cases}$ und $p_{ij} := \begin{cases} j & i = j \vee ij \in E \\ 0 & i \neq j,\ ij \notin E \end{cases}$;

2. Für $k = 1, \ldots, n$
   falls $a_{kk} \geq 0$
     für alle $i, j \in V \setminus \{k\}$
       falls $a_{ij} > a_{ik} + a_{kj}$ setze $a_{ij} := a_{ik} + a_{kj}$, $p_{ij} := p_{ik}$;
   anderenfalls
     für alle $i \in V$
       falls $a_{ik} < \infty$ setze $a_{ik} := -\infty$;
     für alle $i, j \in V \setminus \{k\}$
       falls $a_{ik} < \infty \wedge a_{kj} < \infty$ setze $a_{ij} := -\infty$, $p_{ij} := p_{ik}$;
     für alle $j \in V \setminus \{k\}$
       falls $a_{kj} < \infty$ setze $a_{kj} := -\infty$, $p_{kj} := p_{kk}$;

Beispiel Für den ungerichteten Graphen



ergeben sich nach k Iterationen in Schritt 2 des Verfahrens die folgenden Koeffizienten

$k = 1$:

0	5	∞	∞	3
∞	0	-1	∞	2
∞	0	0	1	∞
∞	∞	∞	0	4
∞	∞	∞	∞	0

(keine Änderungen, da kein Umweg mit endlichem Wert vorhanden)

$k = 2$:

0	5	∞	∞	3
∞	0	-1	∞	2
∞	0	0	1	∞
∞	∞	∞	0	4
∞	∞	∞	∞	0

$k = 3$:

0	5	4	∞	3
∞	0	-1	∞	2
∞	0	-1	1	2
∞	∞	∞	0	4
∞	∞	∞	∞	0

($\alpha = -\infty$, viele Umwege enthalten einen negativen Kreis)

$k = 4$:

0	-∞	-∞	-∞	-∞
∞	-∞	-∞	-∞	-∞
∞	-∞	-∞	-∞	-∞
∞	∞	∞	0	4
∞	∞	∞	∞	0

(keine Änderungen, ebenso für $k = 5$)

Der gefundene negative Kreis mit den Knoten $2, 3$ führt zu beliebig kurzen Wegen vom Knoten 1 zu allen anderen Knoten sowie von den Knoten $2, 3$ zu allen von 1 verschiedenen Knoten:

	1	2	3	4	5
1	0	-∞	-∞	-∞	-∞
2	∞	-∞	-∞	-∞	-∞
3	∞	-∞	-∞	-∞	-∞
4	∞	∞	∞	0	4
5	∞	∞	∞	∞	0

Falls negative Kreise existieren, sollte man die Vorzeichen der Diagonalelemente prüfen. Falls $a_{ii} < 0$ für ein $i \in V$ mit $1 \le k \le n-1$, so liegt i auf einem negativen Kreis. Besitzt G keine negativen Kreise, so modifiziert man die Diagonalelemente in Schritt 1: $a_{ii} := \infty$ für alle $i \in V$. Dann ist bei Abbruch a_{ii} das Gewicht eines kürzesten Kreises, der i enthält. Falls diese Kreise nicht gesucht werden, ist der Start mit $a_{ii} = 0$ vorzuziehen, da im Verfahren weniger Zuweisungen erfolgen.

Berechnung kürzester Wege

Im Verfahren von Floyd werden die kürzesten Wege kompakt durch den Nachfolger p_{ij} von i auf einem jeweils bekannten kürzesten Weg von i nach j abgespeichert:

$$\begin{array}{ccccccccc} i & \longrightarrow & i_1 & \longrightarrow & i_2 & \to \quad \to & i_r & \longrightarrow & j \\ & & = p_{ij} & & = p_{i_1 j} & & = p_{i_{r-1} j} & & = p_{i_r j} \end{array}$$

Abbildung II.34: Kürzeste Wege nach Floyd

5 Lineare Zuordnungsprobleme

Objekte aus verschiedenen Klassen müssen häufig einander zugeordnet werden: im Betrieb Aufgaben und Personal, in der Gesellschaft Frauen und Männer, im Computer Resourcen und Jobs. Solche Aufgaben lassen mit Hilfe eines bipartiten vollständigen Graphen $G = (S\dot{\cup}T, E)$, $E \subseteq S \times T$ modellieren. Die beiden Klassen entsprechen den disjunkten

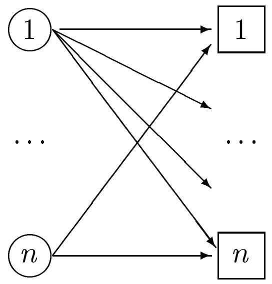


Abbildung II.35: Ein bipartiter Graph

Knotenmengen S, T und mögliche Zuordnungen einzelner Objekte den Kanten. Die zugeordneten Objekte sollen sich nicht überschneiden. Daher nennt man eine Kantenmenge $M \subseteq E$ *Matching (oder Zuordnung)*, falls

$$|M \cap \delta(i))| \leq 1 \quad |M \cap \delta(\bar{j}))| \leq 1$$

für alle $i \in S, j \in T$. Gilt stets „=“, so heißt das Matching *perfekt*. Für die Existenz perfekter Matchings mit n Kanten ist $|S| = |T| = n$ notwendig, was wir stets voraussetzen werden.

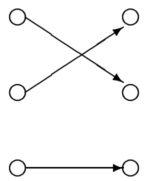


Abbildung II.36: Ein Matching

Bemerkung (Perfekte Matchings und Permutationen): Wir numerieren die Elemente in S, T jeweils von 1 bis n und bezeichnen anschließend Knoten und Kanten entsprechend, z.B sind Knoten $1 \in S$ und Knoten $1 \in T$ durch die Kante 11 verbunden. Dann entspricht dem perfekten Matching $M = \{i_1 j_1, i_2 j_2, \ldots i_n j_n\}$ die Permutation $\phi = \phi(M) : S \to T$, definiert durch $\phi(i_k) = j_k$. ϕ ist also eine Permutation der Menge $\{1, \ldots, n\}$.

Aufgabe 5.1
Für $c: E \to \boldsymbol{R}$ löse die Aufgabe

$$z^* := \min\{c(M) \mid M \text{ perfektes Matching}\}$$

u-v-Reduktion

Wir vereinfachen die Aufgabe durch geschickte Änderungen der Bewertung c. Seien $u_i, v_j \in \boldsymbol{R}$ für alle $i \in S, j \in T$. Dann heißt $\bar{c} = \bar{c}(u,v)$ mit

$$\bar{c}(ij) := c(ij) + u_i - v_j,$$

für alle $ij \in E$, *reduziert* mit Wert $z = z(u,v) := \sum_{j=1}^n v_j - \sum_{i=1}^n u_i$.

Lemma 5.1 *Sei M perfektes Matching und $\bar{c} = \bar{c}(u,v)$ für $u, v \in \boldsymbol{R}^n$. Dann gilt*

1. $c(M) = \bar{c}(M) + z(u,v)$
2. $\bar{c}(M) = 0 \Rightarrow c(M) = z(u,v)$
3. $\bar{c}(M) = 0, \bar{c} \geq 0 \Rightarrow M$ *ist optimal für (5.1).*

Beweis. Sei $\phi := \phi(M)$. Dann folgt (1.) aus

$$\bar{c}(M) = \sum \bar{c}(i\phi(i)) = \sum c(i\phi(i)) + \sum u_i - \sum v_{\phi(i)} = c(M) - z(u,v).$$

(1.) impliziert (2.). Sei nun R ein perfektes Matching. Dann folgt die Optimalität von M aus

$$c(R) = \bar{c}(R) + z(u,v) \geq z(u,v) = c(M).$$

□

Wir wollen sukzessive für $l = 0, 1, \ldots, n$ ein Matching der Form $M = \{1j_1, 2j_2, \ldots, lj_l\}$ und Vektoren $u, v \in \boldsymbol{R}^n$ mit

$$\bar{c}(M) = 0 \qquad \text{und} \qquad \bar{c} \geq 0 \tag{5.1}$$

für $\bar{c} := \bar{c}(u,v)$ bestimmen. Dann ist das letzte Matching optimal und hat den Wert $z(u,v)$.

Zu Beginn sei $M := \emptyset$. Ist $c \geq 0$, wähle $u := 0$, $v := 0$. Anderenfalls wähle $u := 0$ und $v_j := \min_{e \in E} c(e) =: \delta$ für alle $j \in T$. Dann gilt $\bar{c} := \bar{c}(u,v) \geq 0$, denn $\bar{c}(ij) = c(ij) + 0 - \delta \geq 0$ für alle $ij \in E$.

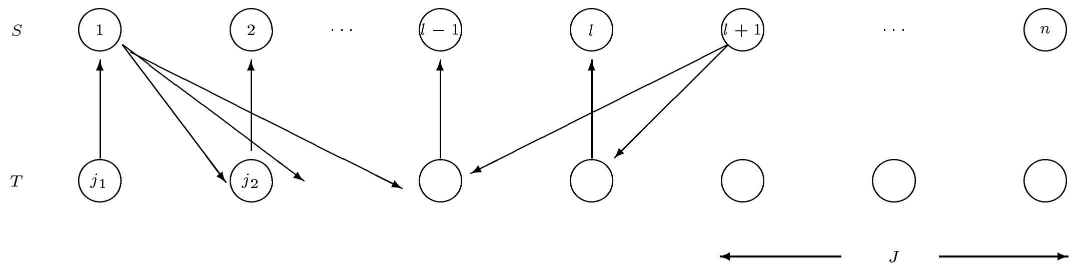


Abbildung II.37: Der Hilfsgraph G_l

Für $l \geq 1$ seien nun M, u, v mit $M = \{1j_1, 2j_2, \ldots, (l-1)j_{l-1}\}$, $\bar{c}(M) = 0$, $\bar{c} \geq 0$ bekannt. Wir betrachten den gerichteten Hilfsgraphen $G_l = (S \cup T, E_B \cup E_R)$ mit Kantenmengen

$$E_B := \{ij \mid ij \in E \setminus M, i \in S, j \in T\} \qquad \text{und} \qquad E_R := \{ji \mid ij \in M, i \in S, j \in T\}$$

Auf dem Hilfsgraphen G_l definieren wir nichtnegative Kantengewichte

$$\tilde{c}(ij) := \begin{cases} \bar{c}(ij) & ij \in E_B \\ 0 & ij \in E_R \end{cases}$$

Offenbar übernehmen wir die Gewichte außerhalb des Matchings und übertragen die Gewichte der Matchingkanten, also 0, auf die entsprechenden neuen Kanten. Anschließend bestimmen wir mit Hilfe des Dijkstra-Verfahrens einen kürzesten Weg von l nach J. Ein solcher Weg ist gefunden, sobald der Index k im Schritt 2 des Dijkstra-Verfahrens aus der Menge J gewählt wird. Damit wird das Dijkstra-Verfahren abgebrochen.

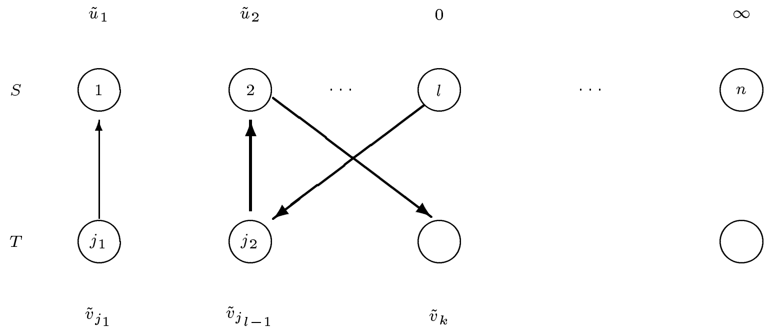


Abbildung II.38: Ein kürzester Weg von l nach $J := T \setminus \{j_1, \ldots, j_{l-1}\}$

Revision von M, u, v

Mit Hilfe des kürzesten Weges W und im Dijkstra-Verfahren bestimmten Markierungen $\tilde{u}, \tilde{v}$ werden M, u und v revidiert:

$$\begin{aligned} M' &:= M \setminus \{ij \mid ji \in W \cap E_R\} \cup \{ij \mid ij \in W \cap E_B\}, \\ u_i' &:= u_i + \min(\tilde{u}_i, \tilde{v}_k), \qquad i \in S, \\ v_j' &:= v_j + \min(\tilde{v}_j, \tilde{v}_k), \qquad j \in T. \end{aligned}$$

Die neue Bewertung wird mit $\bar{c}'$ bezeichnet, d.h. $\bar{c}' = \bar{c}'(u', v')$.

Lemma 5.2 *Gilt (5.1) für $M, \bar{c}, u, v$, so auch für $M', \bar{c}', u', v'$.*

Beweis.

Nach der Revision hat das neue Matching offenbar die gewünschte Form mit $M' = \{1\mu_1, 2\mu_2, \ldots, l\mu_l\}$. Nach (5.1) bleibt zu zeigen:

$$\bar{c}'(M') = 0 \qquad , \qquad \bar{c}' \geq 0.$$

Nach Revision gilt

$$\bar{c}'(ij) = \bar{c}(ij) + \min(\tilde{u}_i, \tilde{v}_k) - \min(\tilde{v}_j, \tilde{v}_k) \quad (+)$$

für alle $ij \in E$. Wir zeigen zunächst $\bar{c}' \geq 0$.

1. Falls $l < i \leq n$ gilt $\tilde{u}_i = \infty$. Die vereinfachte Formel (+) liefert $\bar{c}'(ij) = \bar{c}(ij) + \tilde{v}_k - \min(\tilde{v}_j, \tilde{v}_k) \geq 0$.

2. Anderenfalls ist $1 \leq i \leq l$. Sei $T_D \subseteq V$, die bei Abbruch des Dijkstra-Verfahrens bekannte dort mit T bezeichnete Knotenmenge, d.h. für alle $j \notin T_D$ ist ein kürzester Weg $KW_{l\to j}$ bereits bestimmt. Insbesondere gilt $l \notin T_D$.

   (a) Ist $j \notin T_D$, so ist der Umweg über i nicht kürzer als die Länge $\tilde{v}_j$ des $KW_{l\to j}$. Daher gilt $\bar{c}(ij) + \tilde{u}_i \geq \tilde{v}_j$. Aus Formel (+) folgt wieder $\bar{c}'(ij) \geq 0$.

   (b) Ist $j \in T_D$, so ist nach Wahl von k im Dijkstra-Verfahren $\tilde{v}_j \geq \tilde{v}_k$.

      i. Falls auch $i \in T_D$, so gilt auch $\tilde{v}_j, \tilde{u}_i \geq \tilde{v}_k$. Dann liefert Formel (+) $\bar{c}'(ij) = \bar{c}(ij) \geq 0$.

      ii. Falls jedoch $i \notin T_D$, so ist der Umweg über i nicht kürzer als die Länge $\tilde{v}_j$ eines $KW_{1\to j}$ über Knoten in $\bar{T}_D$, d.h. $\bar{c}(ij) + \tilde{u}_i \geq \tilde{v}_j$. Mit Hilfe der Formel (+) folgt wieder $\bar{c}'(ij) \geq 0$

Nun wollen wir zeigen: $\bar{c}'(M') = 0$. Sei $ij \in M'$. Falls $ij \in M$, so gilt $\bar{c}(ij) = 0$. Der Knoten i ist erreichbar im Hilfsgraphen, allerdings nur über die Kante ji. Daher sind die Längen kürzester Wege nach i und j gleich.

1. Falls $j \notin T_D$, so ist die Länge eines kürzesten Wege nach j bekannt und es gilt $\tilde{u}_i = \tilde{v}_j \leq \tilde{v}_k$ (auch wenn $i \notin T_D$).

2. Anderenfalls ist $j \in T_D$. Dann darf der bisher bekannte kürzeste Weg $KW_{l\to i}$ über Knoten in $\bar{T}_D$ den Knoten j nicht benutzen, d.h. $\tilde{u}_i = \infty \geq \tilde{v}_j \geq \tilde{v}_k$.

In beiden Fällen ist $\bar{c}'(ij) = \bar{c}(ij) = 0$.

Anderenfalls ist $ij \notin M$. Dann ist $ij \in W \cap E_B$. Längs eines kürzesten Weges gilt $\bar{c}(ij) + \tilde{u}_i = \tilde{v}_j$. Die kürzesten Wege nach i und j sind als Teilwege bekannt, so daß weiterhin $i, j \notin T_D, \tilde{u}_i, \tilde{v}_j \leq \tilde{v}_k$ gilt. Mit Hilfe der Formel (+) finden wir $\bar{c}'(ij) = \bar{c}(ij) + \tilde{u}_i - \tilde{v}_j = 0$.
$\square$

Bemerkung: In jeder Iteration des Verfahrens ist der Wert des Matchings $c(M) = z(u, v)$ und M ist optimal unter allen Matchings mit der Eigenschaft $|M \cap \delta(i)| = 1$ für alle $1 \leq i \leq l$. Der Aufwand des Verfahrens ergibt sich aus dem des Dijkstra-Verfahrens und beträgt daher $O(n^3)$.

Beispiel (3×3 Zuordnungsproblem):

Für die Kostenmatrix

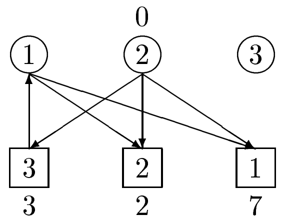
$$C = \begin{pmatrix} 2 & 3 & 1 \\ 7 & 2 & 3 \\ 7 & 1 & 3 \end{pmatrix}$$

liefert die erste Iteration, ausgehend von $M = \emptyset$ und $u = v = 0$, den kürzesten Weg $W = \{13\}$ mit Gewicht 1. Die erste Revision liefert $M = \{13\}$ und $u = (011)$, $v = (111)$.
2. Iteration: Neuberechnung von $\bar{c}$ und Erweiterung des Matchings auf 2 Kanten.

$u \setminus v$	1	1	1
0	2	3	1
1	7	2	3
1	7	1	3

$\Rightarrow \quad \bar{c}_{ij} =$

min	2	2	2
2	1	2	0
0	7	2	3
2	7	1	3

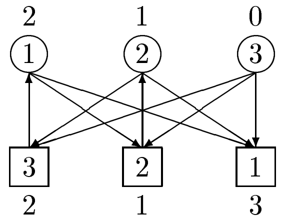


Aus dem kürzesten Weg $W = \{22\}$ ergibt sich das neue Matching $M = \{13, 22\}$.
3. Iteration: Neuberechnung von $\bar{c}$ und Erweiterung des Matchings auf 3 Kanten.

$u \setminus v$	3	3	3
2	2	3	1
1	7	2	3
3	7	1	3

$\Rightarrow \quad \bar{c}_{ij} =$

min	3	1	2
2	1	2	0
1	5	0	1
0	7	1	3



Hier findet man einen kürzesten Weg

$$3 \to \boxed{2} \to 2 \to \boxed{3} \to 1 \to \boxed{1}$$

der Länge 3, der auf die die optimale Zuordnung $M = \{32, 23, 11\}$ mit den reduzierten Kosten

$u \setminus v$	6	4	5
4	2	3	1
2	7	2	3
3	7	1	3

$$\Rightarrow \quad \bar{c}_{ij} = \begin{array}{|ccc|} \hline 0 & 3 & 0 \\ 3 & 0 & 0 \\ 4 & 0 & 1 \\ \hline \end{array}$$

führt. Die Bedingungen $\bar{c} \geq 0$ und $\bar{c}(e) = 0$, für alle $e \in M$, sind offenbar erfüllt. Der Wert des optimalen Matchings ist

$$\sum_{e \in M} c(e) = 6 = 1^T v - 1^T u = z(u, v).$$

Beschreibung als ganzzahliges lineares Optimierungsproblem

Die Aufgabe

$$z_* = \min\{c(M) \mid M \text{perfektes Matching}\}$$

kann man auch als ganzzahlige lineare Optimierungsaufgabe formulieren. Dazu betrachten

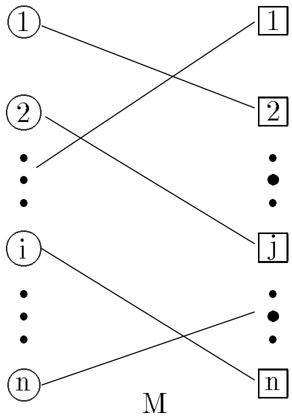


Abbildung II.39: Lösung des Zuordnungsproblems

wir Inzidenzvektoren von Kantenmengen. Insbesondere heißt der durch

$$x_{ij}^M := \begin{cases} 0 & ij \notin M \\ 1 & ij \in M \end{cases}$$

gegebene Vektor x^M *Inzidenzvektor* des perfekten Matchings M. Das Gewicht eines Matchings M ist

$$c(M) = \sum_{ij \in M} c_{ij} = \sum_{ij} c_{ij} x_{ij}^M.$$

Die Bedingung, daß ein perfektes Matching mit jedem Knoten in $S \cup T$ genau eine Kante gemeinsam hat, führt auf die linearen Restriktionen

$$\sum_i x_{i\nu} = 1, \; -\sum_j x_{\mu j} = -1,$$

für alle $\mu, \nu = 1, \ldots, n$. Daher entsprechen perfekte Matchings den ganzzahligen Lösungen des (LP)

$$z_{LP} = \min\{c^T x \mid \sum_i x_{ij} = 1, \; -\sum_j x_{ij} = -1, x \geq 0\}.$$

Das dazu duale Problem (D) ist

$$z_D = \max\{\sum_j v_j - \sum_i u_i \mid v_j - u_i \leq c_{ij}\}.$$

Mit Hilfe des kombinatorischen Verfahrens haben wir neben einem optimalen perfekten Matching M Knotenvariablen $u, v \in \boldsymbol{Z}^n$ mit

$$c(M) = z(u,v) = \sum_j v_j - \sum_i u_i, \quad c_{ij} + u_i - v_j \geq 0,$$

für alle $i, j = 1, \ldots, n$, bestimmt. Offenbar gibt es für diese speziellen linearen Programme stets ein ganzzahliges optimales Paar. Insbesondere gilt hier ein ganzzahliger Dualitätssatz, während im allgemeinen bei ganzzahligen, linearen Programmen Dualitätslücken auftreten.

6 Das Rundreiseproblem

Der Handelsreisende, der nacheinander einige Städte besuchen muß, der Spediteur, der für ein Transportfahrzeug die tägliche Lieferroute plant, oder ein Industrieroboter, der auf einer Bohrplatte eine Anzahl von Löchern zu bohren hat, alle benötigen sie für das zugrundeliegende Problem eine möglichst günstige Reihenfolge, in der die einzelnen Aufgaben abgearbeitet werden. Verbindet man die Städte des Handelsreisenden auf einer Landkarte in der Reihenfolge, in der er gereist und schließlich wieder zurückgekehrt ist, so entsteht eine Rundreise. Gleiches gilt für das Transportfahrzeug und den Bohrer, der am Industrieroboter befestigt ist.

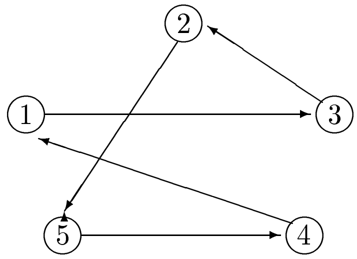


Abbildung II.40: Die Rundreise $\phi = (1\ 3\ 2\ 5\ 4)$

Das mathematische Modell wird mit Hilfe eines gerichteten, vollständigen Graphen $G = (V, E)$ formuliert. Die Bewertung $c(ij)$ der Kante ij entspricht der Entfernung von Stadt i nach Stadt j, wobei $c(ij)$ immer positiv sein soll. Rundreisen entsprechen Kreisen, die jeden Knoten genau einmal durchlaufen. Eine Rundreise ϕ entspricht andererseits einer zyklischen Permutation der Knoten. Permutationen zerfallen allgemein in mehrere Zyklen. Die Zyklenschreibweise der Permutation

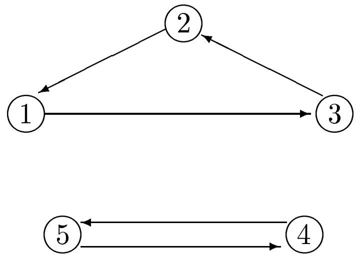


Abbildung II.41: Zyklen

$$\phi = \left| \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \\ \hline 3 & 1 & 2 & 5 & 4 \end{array} \right|$$

ist $\phi = (1\ 3\ 2)\ (5\ 4)$. Das Rundreiseproblem läßt sich daher mit Hilfe von zyklischen Permutationen formulieren:

$$z^* = \min\{\sum_1^n c(i\phi(i)) \mid \phi \text{ zyklische Permutation}\} \tag{6.1}$$

Die Anzahl der Rundreisen entspricht der halben Anzahl der zyklischen Permutationen, die bei n Städten $(n-1)!$ beträgt. Für $n \geq 3$, wächst die Anzahl der Rundreisen also exponentiell:

n	9	20
Anzahl	20160	10^{17}

Im Unterschied zum Zuordnungsproblem ist auch kein effizient auswertbares Optimalitätskriterium bekannt. Kennt man beim Zuordnungsproblem das optimale Matching M, so kann man leicht eine dazu passende $u-v$-Reduktion finden, so daß die Bedingung (5.1) erfüllt ist. Ist die Bedingung für M, u, v erfüllt, so ist M optimal. Für Rundreisen ist es viel schwerer, die Optimalität nachzuweisen. Für eine gegebene Rundreise kennt man keinen prinzipiell besseren Nachweis, als den Vergleich mit allen anderen Rundreisen. Es gibt gute Gründe für die Hypothese, daß es keinen Optimalitätsnachweis gibt, der mit polynomialem Aufwand geführt werden kann. Insbesondere kann man dann natürlich auch keine optimale Rundreise mit polynomialem Aufwand konstruieren. Das Rundreiseproblem gehört damit zu den „harten“ kombinatorischen Optimierungsproblemen.

Da die meisten kombinatorischen Probleme in diesem Sinne als notorisch schwierig gelten, kann man es nicht einfach bei dieser Erkenntnis bewenden lassen. Es kann durchaus so sein, daß für praktisch interessante Fälle eine optimale Rundreise in akzeptabler Zeit konstruiert werden kann, man kann es nur nicht für *alle* möglichen Rundreiseprobleme garantieren. Es gibt zu Rundreiseproblemen sehr viele hilfreiche Untersuchungen, Ideen und Methoden (siehe [12]). Im folgenden wollen wir uns eine naheliegende Idee ansehen, die auf dem Zuordnungsproblem aufbaut.

Untere Schranken: Relaxation

Die zyklischen Permutationen bilden eine Teilmenge aller Permutationen. Daher gilt:

$$\underline{z} := \min\{\sum_1^n c(i\phi(i)) \mid \phi \text{ Zuordnung}\} \leq z^*. \tag{6.2}$$

Da die Permutationen den Zuordnungen entsprechen, kann man eine Lösung dieser Relaxation des Rundreiseproblemes mit Hilfe des Verfahrens für Zuordnungsprobleme in $O(n^3)$ bestimmen.

Obere Schranken: Approximation

Um den Wert einer optimalen Rundreise nach oben zu beschränken, kann man den Wert einer beliebigen Rundreise heranziehen. Bestimmt man eine solche Rundreise mit einem mehr oder weniger ausgeklügelten Verfahren, so bezeichnet man diese als „approximative“ Lösung.

1. Nearest-Neighbour-Approximation
   Die Auswahlregel „Gehe von einer Stadt stets zur nächsten noch nicht besuchten Stadt“ liefert in $O(n)$ die (NN)-Approximation.

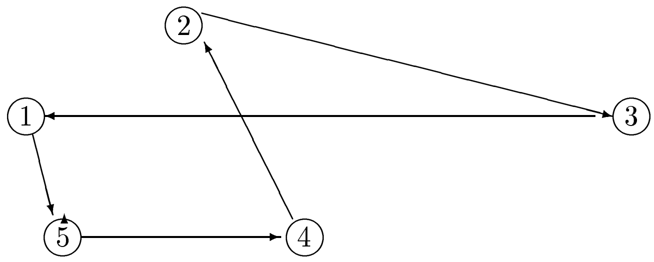


Abbildung II.42: Eine Nearest-Neighbour-Approximation

2. Farthest-Insert-Approximation
   Die Auswahlregel „Nehme in eine Teiltour die entfernteste noch nicht besuchte Stadt auf“ wird im folgenden näher beschrieben. Sei V' Knotenmenge der bereits bestimmten Teiltour. Zu $i \notin V'$ sei

$$c(i, V') := \min\{c(ij) \mid j \in V'\}.$$

Dann wähle $k \notin V'$ mit

$$c(k, V') = \max_{i \notin V'} c(i, V').$$

Man plant also zunächst eine Tour grob vor, um sie anschließend zu verfeinern. In jedem Schritt wird die neue Stadt k jeweils optimal in die Teiltour eingefügt. Nimmt man in der Abbildung II.43 die euklidischen Entfernungen als Bewertung an und beginnt beim Knoten 1, so erhält man zunächst den Knoten $k = 4$ hinzu. Der 2. Schritt liefert zu $V_1 = \{1, 4\}$ den Knoten $k = 2$. Anschließend findet man $k = 5$ und $k = 3$. Mit etwas Nachdenken kann man die (FI)-Approximation in $O(n^2)$

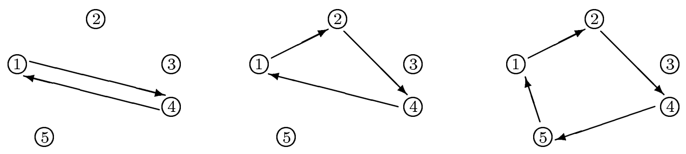


Abbildung II.43: Eine Farthest-Insert-Approximation

konstruieren.

3. Minimum-Tree-Approximation
   Die Auswahlregel „Ein minimales Gerüst doppelt umlaufen und zu einer Rundreise abkürzen“ führt, falls die Dreiecksungleichung erfüllt ist, zu einer Approximation, deren Güte man abschätzen kann.

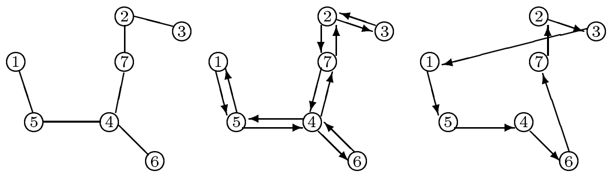


Abbildung II.44: Minimales Gerüst

z_D bezeichne die Länge des doppelten Umlaufs D in der folgenden Reihenfolge: $(1, 5, 4, 6, 4, 7, 2, 3, 2, 7, 4, 5, 1)$. Man kürzt diesen doppelten Umlauf beginnend bei 1 ab, indem man bereits besuchte Städte überspringt. So ergibt sich die Rundreise $R = (1546723)$ mit Länge z_R. Streicht man eine Kante in der optimalen Rundreise, so erhält man ein Gerüst. Daher gilt

$$\frac{1}{2}z_D \leq z^* - \min c(ij) < z^* \leq z_R \leq z_D$$

woraus $z_R - z^* \leq \frac{1}{2}z_D < z^*$ und damit $\frac{z_R - z^*}{z^*} < 1$ folgt. Auch wenn ein relativer Fehler von 100% nicht besonders vielversprechend erscheinen mag, so ist in diesem Falle wegen Kenntnis einer Fehlerabschätzung die Bezeichnung „Approximation“ eher gerechtfertigt als in den beiden anderen Verfahren.

Einschließen

Durch die optimale Zuordnung ϕ_{opt} ergibt sich eine untere Schranke $\underline{z}$, durch die Approximation $\hat{\phi}$ eine obere Schranke $\hat{z}$

$$\underline{z} \leq z^* \leq \hat{z}.$$

Falls die Zuordnung einer Rundreise entspricht, so ist diese optimal. Anderenfalls versucht man das Problem in geschickter Weise aufzuteilen und günstigere Schranken für die Teilprobleme zu finden.

Separation in Teilprobleme

Die Zuordnung zerfalle in $r > 1$ Zyklen, etwa $\phi_{opt} = (i_1, \dots i_{k_1}) \dots (i_{1+k_{r-1}}, \dots, i_{k_r})$, wobei der erste Zyklus C einer von kürzester Länge sei. Die Menge der Rundreisen wird nun durch k Teilmengen überdeckt. Jede Rundreise enthält eine Kante ij mit $i \in C$, $j \notin C$. Ist etwa $i = i_\rho$ für ein $\rho \in \{1, 2, \dots, k\}$, so enthält die Rundreise keine der Kanten $i_\rho i_\mu$, $i_\mu \in C$. Für festes ρ sind diese Rundreisen genau die Rundreisen des Teilgraphen von $G = (V, E)$, der die Kanten $E \setminus \{i_\rho i_\mu \mid i_\mu \in C\}$ enthält.

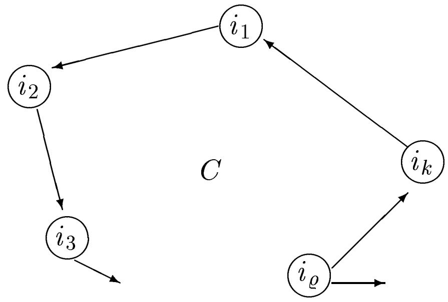


Abbildung II.45: Zyklen

Die resultierenden k Klassen von Rundreisen, für $1 \leq \rho \leq k$, kann man auch als die Rundreisen in G mit endlicher Bewertung bezüglich der neuen Bewertung

$$c^\rho(ij) := \begin{cases} \infty & ij = i_\rho i_\mu, 1 \leq \mu \leq k, \\ c(ij) & \text{anderenfalls.} \end{cases}$$

charakterisieren.

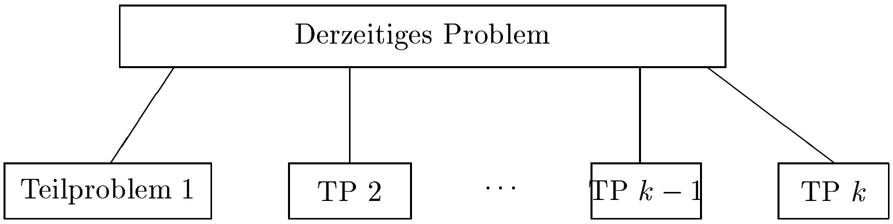


Abbildung II.46: Überdeckung durch Teilprobleme

Behandlung der Teilprobleme

Sei $\hat{z}$ der Wert der bisher besten Approximation $\hat{\phi}$.

1. Exakte Lösung des Teilproblems
   Ist der Wert z der optimalen Rundreise ϕ des Teilproblems bekannt, so erübrigt sich die weitere Separation des Teilproblems. Falls $z < \hat{z}$, so ist gleichzeitig eine bessere Approximation des Problems gefunden.

2. Relaxation des Teilproblems lösen
   Sei $\underline{z}$ der Wert der optimalen Zuordnung ϕ des Teilproblems. Falls $\underline{z} = \infty$, so enthält das Teilproblem keine Rundreisen mit endlichem Wert.

   (a) Falls $\underline{z} \geq \hat{z}$, so ist die bereits bekannte Approximation $\hat{\phi}$ besser als alle Rundreisen des Teilproblems. Daher ist eine weitere Separation nicht sinnvoll.

   (b) Falls $\underline{z} < \hat{z}$, so gewinnt man keine neue Information und muß weiter separieren.

Auswahl des nächsten zu separierenden Teilproblems:

Durch die Separationsschritte entsteht ein Baum, dessen Knoten die Teilprobleme sind. Wir müssen festlegen, welches der zur Separation anstehenden Teilprobleme als nächstes separiert wird. Es gibt dafür unterschiedliche Auswahlregeln.

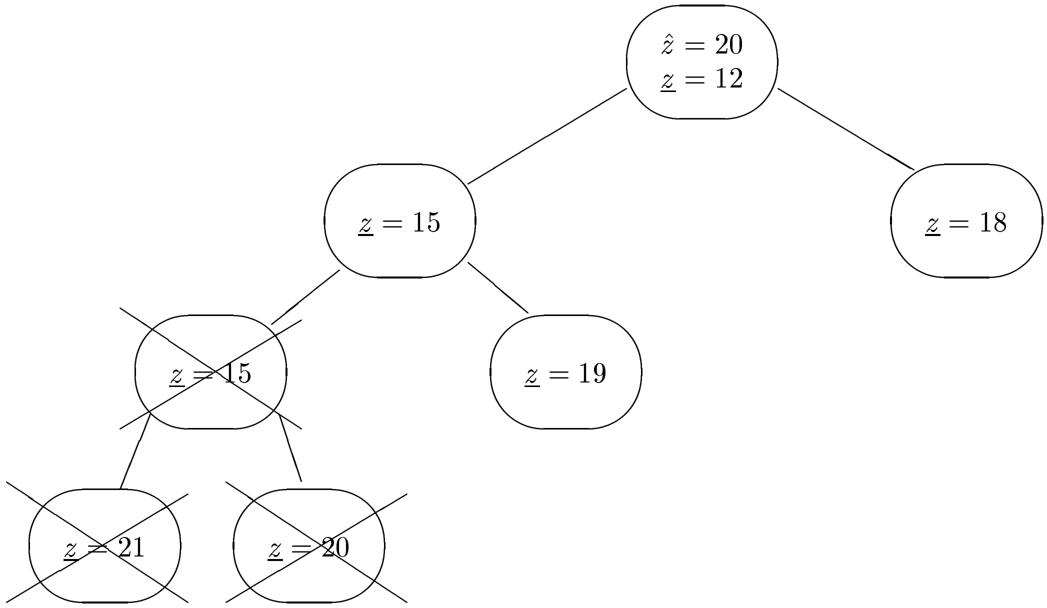


Abbildung II.47: Ein Branch & Bound-Baum

1. Wähle das Teilproblem mit minimaler unterer Schranke zur Separation und berechne anschließend alle fehlenden Schranken. Der Nachteil dieser Regel ist, daß die Menge der zu behandelnden Teilprobleme rasch wächst (Prioritätenregel).

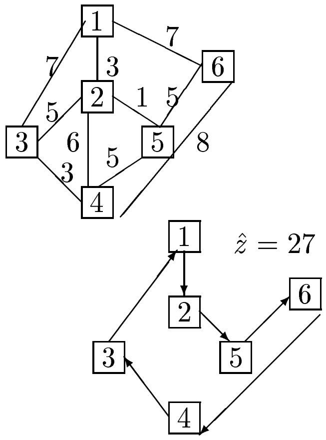
2. FIFO-Regel: Falls Separierung eines Teilproblemes notwendig ist, untersuche *eines* der entstehenden oder bereits entstandenen Nachfolgeprobleme, anderenfalls gehe zum Vorgängerproblem.

Beispiel (Branch & Bound mit Matching-Relaxation):

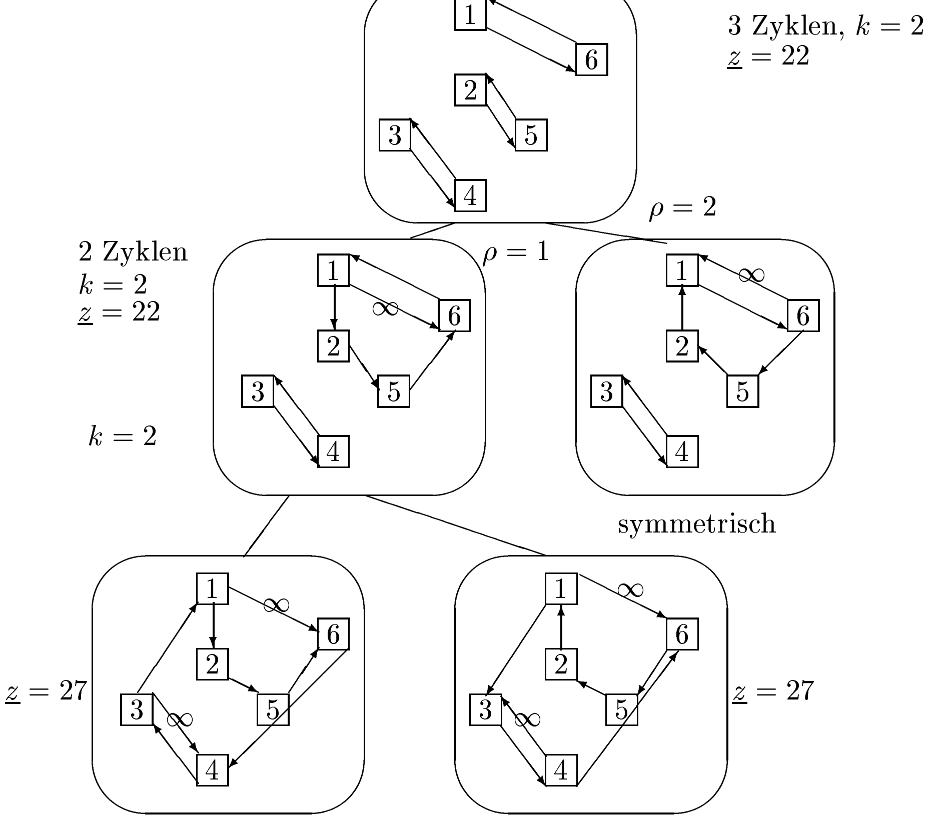
Das Rundreiseproblem mit der Entfernungsmatrix (nicht vorhandene Kanten sind durch $-$ gekennzeichnet)

-	3	7	-	-	7
3	-	5	6	1	-
7	5	-	3	-	-
-	6	3	-	5	8
-	1	-	5	-	5
7	-	-	8	5	-

führt auf den folgenden Graphen mit der NN-Approximation:



Die erste Relaxation liefert eine optimale Zuordnung, die in drei Kreise der Länge 2 zerfällt. Separation für den Kreis $(1, 6)$ führt auf zwei Teilprobleme, in denen die Kanten 16 bzw. 61 verboten sind.



Die optimalen Zuordnungen beider Probleme sind aus Symmetriegründen bis auf die Richtung des Kreises (1652) gleich. Die neue untere Schranke $\underline{z} = 22$ zwingt zu weiterer Separation. Wegen der Symmetrie genügt die Betrachtung einer der möglichen Separationen mit dem Kreis (34). Die resultierenden optimalen Zuordnungen mit Wert $\underline{z} = 27$ entsprechen Rundreisen, weswegen weitere Separation nicht mehr notwendig ist. Untere Schranke und Wert der Approximation sind jeweils gleich. Daher sind diese Rundreisen optimal mit $z^* = 27$. In diesem Beispiel können wir auch ohne Kenntnis der Approximation die Optimalität der bestimmten Rundreisen erkennen, da bis auf symmetrische Rundreisen alle Rundreisen in den beiden letzten Teilproblemen, die wir exakt gelöst haben, enthalten sind.

Kapitel III

Nichtlineare Optimierung ohne Restriktionen

Einleitung

Die im 17. Jahrhundert durch Newton und Leibniz begründete Infinitesimalrechnung liefert das theoretische Fundament der Nichtlinearen Optimierung. Mit entsprechenden analytischen Hilfsmitteln der Analysis lassen sich z.B. Minima von reellen Funktionen $f : \boldsymbol{R}^n \to \boldsymbol{R}$ charakterisieren. Die konkrete Bestimmung der Minima, also die Berechnung von reellen Vektoren, die die charakterisierenden Bedingungen erfüllen, ist in den seltensten Fällen explizit mit theoretischen Mitteln möglich.

In der Einführung zur Mathematischen Optimierung werden nur nichtlineare Optimierungsprobleme ohne explizite Restriktionen behandelt. Bei diesen Aufgaben muß das Minimum einer gegebenen Funktion $f : \boldsymbol{R}^n \to \boldsymbol{R}$ bestimmt werden, d.h. es soll eine Lösung der Aufgabe

$$\min_{x \in \boldsymbol{R}^n} f(x) =: f(\hat{x})$$

gefunden werden. So benutzt man zum Beispiel in der Regressionsrechnung Optimierungsmodelle zur Fixierung unbekannter Parameter.

Regressionsrechnung

Zur mathematischen Formulierung von Gesetzmäßigkeiten, die ein technisches oder physikalisches Phänomen beschreiben, wird häufig eine Hypothese über den möglichen funktionalen Zusammenhang bekannter und beobachtbarer Variablen formuliert, die noch etliche freie Parameter enthält, etwa

$$z = h(x, \lambda).$$

Hierbei ist $x \in \boldsymbol{R}^n$ ein Vektor, der die unabhängigen Variablen eines Experiments enthält, $\lambda \in \boldsymbol{R}^p$ ein Vektor unbekannter Parameter, und z eine Variable, deren Abhängigkeit modelliert werden soll.

Nach Auswertung von m Experimenten kennt man m Paare von Variablenwerten z_i, x_i, die unter Berücksichtigung möglicher Beobachtungsfehler ϵ_i in etwa die funktionale

Abhängigkeit erfüllen sollten, d.h. bei passenden Parametern muß

$$z_i = h(x_i, \lambda) + \epsilon_i$$

für alle $i \in \{1, 2, \ldots, n\}$ gelten. Um möglichst kleine Abweichungen ϵ_i zu erhalten, die man als Beobachtungsfehler deuten kann, versucht man die unbekannten Parameter λ optimal anzupassen, indem man entsprechende Optimierungsaufgaben löst, z.B.

$$\min \left\{ \sum_{i=1}^{m} (z_i - h(x_i, \lambda))^2 \mid \lambda \in \boldsymbol{R}^p \right\}$$

oder

$$\min \left\{ \sum_{i=1}^{m} w_i (z_i - h(x_i, \lambda))^r \mid \lambda \in \boldsymbol{R}^p \right\},$$

wobei $r \geq 1$ gerade, $w > 0$. $w > 0$ ist komponentenweise zu verstehen, wobei 0 den 0-Vektor entsprechender Dimension bezeichnet. Analog benutzen wir diese Konvention auch für andere Vektoren mit identischen Komponenten, insbesondere für den häufig auftretenden 1-Vektor.

1 Minimierung in einer Variablen

Die Lösung nichtlinearer Optimierungsaufgaben im eindimensionalen Fall tritt als Teilproblem in den meisten Verfahren zur Lösung nichtlinearer Optimierungsaufgaben in höheren Dimensionen auf. Zu gegebenem abgeschlossenen Intervall $I \subseteq \boldsymbol{R}$ und gegebener Funktion $f : I \to \boldsymbol{R}$ wird ein $\hat{x} \in I$ mit

(1.1) $$f(\hat{x}) = \min_{x \in I} f(x)$$

gesucht. Um ein solches Minimum mit Hilfe eines Verfahrens effektiv bestimmen zu können, müssen einschränkende Annahmen über die Funktion gemacht werden. Je nach Eigenschaften der Funktion ist dann ein passender Algorithmus zu wählen.

Unimodale Funktionen

Definition (Unimodalität): f heißt *unimodal*, falls für ein $\hat{x} \in I := [a, b]$ gilt:
f streng monoton fallend auf $[a, \hat{x}]$, f streng monoton wachsend auf $[\hat{x}, b]$.

Intervallreduktion

Unimodale Funktionen erlauben nach Auswertung der Funktion an zwei Stellen $x < y$ im Intervall die Reduktion des Intervalls. Unter gewissen Annahmen liefert die rekursive Anwendung dieser Beobachtung einen festgelegten Algorithmus:

1. Im ersten Schritt zwei Auswertungen, in den weiteren Schritten eine weitere Auswertung von f im Restintervall zur Reduktion verwenden,

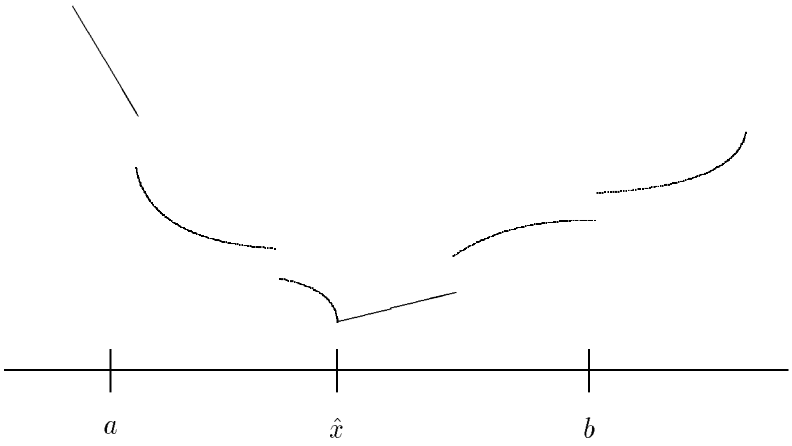


Abbildung III.1: Unimodale Funktionen

2. x und y stets symmetrisch im Restintervall wählen,

3. Reduktionsfaktor σ sei konstant, $\frac{1}{2} < \sigma < 1$.

Durch die symmetrische Wahl von x, y im Restintervall wird der Reduktionsfaktor unabhängig von der Funktion f. Verlangt man einen konstanten Reduktionsfaktor in allen Iterationen, so ist dieser eindeutig bestimmt. Um diesen von f unabhängigen Faktor zu berechnen, genügt es oBdA eine streng monoton wachsende Funktion $f : [0,1] \to \boldsymbol{R}$ zu betrachten. Im ersten Intervall sei oBdA $0 < x < y < 1$, und daher $y = \sigma$, $x = 1 - \sigma$.

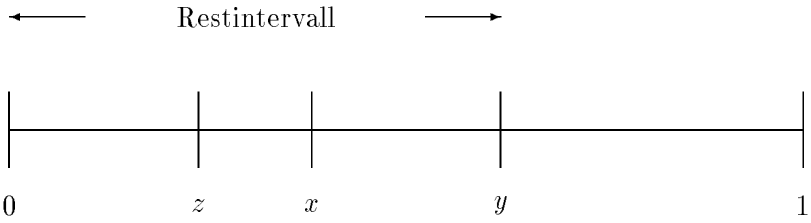


Abbildung III.2: Intervallreduktion bei unimodalen Funktionen

Im ersten Schritt wird das Intervall auf das Restintervall $[0, y]$ reduziert. In diesem Restintervall wird dann z symmetrisch zu x gewählt. Je nach Wahl von y wird $z < x$ oder $x \leq z$.

1. Fall: $\sigma < \frac{2}{3}$. Dann gilt $x = 1 - \sigma > \frac{1}{3}$ und daher $z = y - x = \sigma - (1 - \sigma) < \frac{1}{3} < x$. Konstanz der Reduktion bedeutet nun $\frac{x}{y} = \frac{y}{1}$, d.h. $0 = y^2 - x = \sigma^2 + \sigma - 1$. Die Lösung σ dieser quadratischen Gleichung mit $\frac{1}{2} < \sigma < 1$ ist $\sigma = \frac{1}{2}(\sqrt{5} - 1) = 0{,}618\ldots$.

2. Fall: $\sigma \geq \frac{2}{3}$. Analog zum ersten Fall folgt: $z = \sigma - (1 - \sigma) \geq \frac{1}{3} \geq x$. Also muß zur Konstanz der Reduktion $\frac{z}{y} = \frac{y}{1}$, d.h. $0 = y^2 - z = \sigma^2 - 2\sigma + 1$ gelten. Diese quadratische Gleichung besitzt offenbar keine Lösung σ mit $\frac{1}{2} < \sigma < 1$.

Der im ersten Fall gefundene Reduktionsfaktor $\sigma = 0{,}618\ldots$ ist also unter den getroffenen Annahmen der einzig mögliche und legt den Algorithmus fest.

Algorithmus 1.1 (Goldener Schnitt)

1. **Initialisierung:**
   $i := 0$; $a_0 := a$; $b_0 := b$;
   $x_i := a + (1-\sigma)(b_i - a_i)$; $y_i := a + \sigma(b_i - a_i)$;
   $fx := f(x_i)$; $fy := f(y_i)$.

2. **Iteration:**
   Wenn $fx < fy$, dann

   $a_{i+1} := a_i$; $b_{i+1} := y_i$;
   $x_{i+1} := a_{i+1} + (1-\sigma)(b_{i+1} - a_{i+1})$; $y_{i+1} := x_i$;
   $fy := fx$; $fx := f(x_{i+1})$;

   sonst

   $a_{i+1} := x_i$; $b_{i+1} := b_i$;
   $x_{i+1} := y_i$; $y_{i+1} := b_{i+1} - (1-\sigma)(b_{i+1} - a_{i+1})$;
   $fx := fy$; $fy := f(y_{i+1})$;

   $i := i + 1$;

3. **Abbruch:**
   Wenn $\frac{1}{2}(b_i - a_i) > \epsilon$, dann gehe nach 2.
   $\tilde{x} := \frac{1}{2}(a_i + b_i)$; Stop.

Im obigem Algorithmus wird ein Funktionswert zuviel berechnet; in der Praxis kann man die tatsächliche Berechnung des neuen fx bzw. fy zurückstellen, bis der Wert für den Vergleich benötigt wird.

Mögliche Abbruchkriterien

Wie oben beschrieben bricht der Algorithmus ab, sobald das Minimum bis auf eine vorgegebene Genauigkeit ϵ lokalisiert ist. Da nach $n+1$ Funktionsauswertungen die Länge des Restintervalls auf $\sigma^n(b-a)$ reduziert ist, kann man n in Abhängigkeit von ϵ a priori abschätzen.

Ist f auf dem Restintervall eine quadratische Funktion, so kann man den Wert des Minimums auf dem Restintervall nach unten durch

$$f(\hat{x}) \geq f_{\min} - \sigma(f_{\max} - f_{\min})$$

abschätzen. Liegt die Abweichung innerhalb einer vorgegebenen Genauigkeit, etwa wenn $f_{\max} - f_{\min} \leq \delta$ erfüllt ist, so kann man das Verfahren abbrechen.

Wenn man bereit ist, auf die Konstanz der Reduktion zu verzichten, kann man sich fragen, ob es eine Modifikation des Verfahrens gibt, das bei gleicher Anzahl von Funktionsauswertungen ein kleineres Restintervall liefert. Wir wollen mit L_n die maximale Länge eines mittels n Auswertungen in L_n auf $L_1 = 1$ reduzierbaren Intervalls bezeichnen.

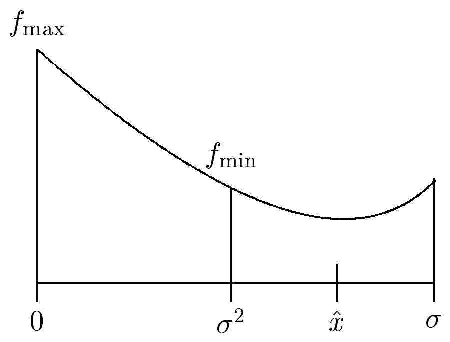


Abbildung III.3: Abbruchkriterien

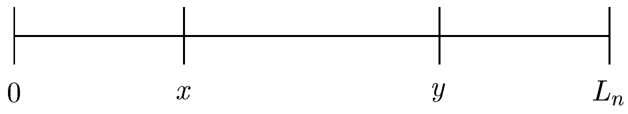


Abbildung III.4: Maximale Intervallreduktion

Durch jede der ersten beiden Stützstellen $x < y$ in L_n, etwa durch x, wird dieses Intervall in zwei Teilintervalle zerlegt. Das linke enthält höchstens $n-2$, das rechte höchstens $n-1$ der n Stützstellen. Entsprechend sind die Längen dieser Teilintervalle höchstens L_{n-2} bzw. L_{n-1}. Also gilt

$$L_n \leq L_{n-1} + L_{n-2}$$

und $L_0 = L_1 = 1$, da mindestens 2 Stützstellen zur Reduktion notwendig sind. Nach dieser Abschätzung erhalten wir die bestmögliche Reduktion, falls eine Lösung die Ungleichungen als Gleichungen erfüllt.

Fibonacci-Zahlen

Die entsprechende Rekursionsformel $F_0 := F_1 := 1, F_n := F_{n-1} + F_{n-2}$ definiert bekanntlich gerade die Fibonacci-Zahlen, deren Eigenschaften man etwa im Buch von Knuth [11] nachlesen kann. Eine Darstellung der Lösung in geschlossener Form ist

$$F_i = \frac{1}{\sqrt{5}} \left(\left(\frac{1+\sqrt{5}}{2} \right)^{i+1} - \left(\frac{1-\sqrt{5}}{2} \right)^{i+1} \right)$$

In Algorithmen erweist sich die Fortsetzung $F_{-2} := 1, F_{-1} := 0$ der Lösung als hilfreich.

Umnormierung auf Intervall $[a, b]$

Die Iterationspunkte bei beliebigem Startintervall $[a, b]$ ergeben sich durch lineare Transformation mit dem Faktor $h := \frac{b-a}{F_n}$. Die Länge des Restintervalls in Iteration $i := 0, 1, \ldots, n-1$ ist $F_{n-i}h$. Im vorletzten Restintervall $[a_{n-2}, b_{n-2}]$ der Länge $2h$ fallen die Stützpunkte zusammen: $x_{n-2} = y_{n-2} = a_{n-2} + h$. Wegen $\hat{x} \in [a_{n-2}, b_{n-2}]$ gilt dann $|\hat{x} - x_{n-2}| \leq h$.

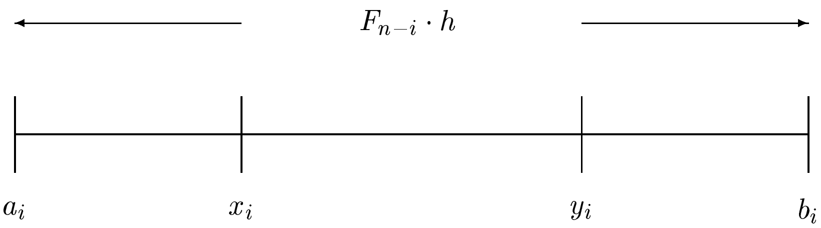


Abbildung III.5: Fibonacci-Suche

Algorithmus 1.2 (Fibonacci-Suche)

1. **Initialisierung:**
   Berechne die Fibonacci-Zahlen F_0 bis F_n.
   $i := 0$; $a_0 := a$; $b_0 := b$; $h := \frac{b-a}{F_n}$;
   $x_0 := a_0 + F_{n-2}h$; $y_0 := a_0 + F_{n-1}h$;
   $fx := f(x_0)$; $fy := f(y_0)$;

2. **Iteration:**
   Wenn $fx < fy$, dann

   $a_{i+1} := a_i$; $b_{i+1} := y_i$;
   $x_{i+1} := a_{i+1} + F_{n-i-3}h$; $y_{i+1} := x_i$;
   $fy := fx$; $fx := f(x_{i+1})$;

   sonst

   $a_{i+1} := x_i$; $b_{i+1} := b_i$;
   $x_{i+1} := y_i$; $y_{i+1} := b_{i+1} - F_{n-i-3}h$;
   $fx := fy$; $fy := f(y_{i+1})$;

   $i := i + 1$.

3. **Abbruch:**
   Wenn $i < n - 2$, dann gehe nach 2.
   $\hat{x} := x_i$; Stop.

Auch im obigen Algorithmus wird ein Funktionswert zuviel berechnet, da der im letzten Iterationschritt berechnete Funktionswert schon bekannt ist.

Ein Beispiel zur Fibonacci-Suche

Die Funktion f, definiert durch $f(x) = \sin(x - 2)$, ist unimodal auf $[a, b] := [0, 2]$. Gesucht wird der Minimalpunkt $\hat{x} = 2 - \frac{\pi}{2} \approx 0,429$. Die Fibonacci-Zahlen F_0 bis F_6 sind: $1, 1, 2, 3, 5, 8, 13$, d.h. bei Fibonacci-Suche mit $n = 6$ findet man ein Restintervall der Länge

$$h := \frac{b-a}{F_n} = \frac{2}{13} \approx 0,154.$$

dessen Genauigkeit hier ausreichen soll.

Die im Verfahren auftretenden Stützstellen (und Intervallgrenzen) sind alle von der Form $a + kh$ mit $k \in \{0, 1, \ldots, F_n\}$. Für eine Stützstelle $t \in [a, b]$ nennt man die entsprechende ganze Zahl $k(t) := \frac{t-a}{h}$ den Fibonacci-Index von t. Beim Start gilt $k(a_0) = 0$, $k(x_0) = F_{n-2}$, $k(y_0) = F_{n-1}$, $k(b_0) = F_n$.

In der Iteration werden die Funktionswerte $f(x_i)$ und $f(y_i)$ verglichen. Die Variable mit kleinerem Funktionswert bleibt Stützstelle, die mit größerem Funktionswert wird neue Intervallgrenze. Der Fibonacci-Index der neuen Stützstelle ergibt sich wegen der symmetrischen Lage der Stützstellen im Restintervall aus $k(x_{i+1}) - k(a_{i+1}) = k(b_{i+1}) - k(y_{i+1})$. Dies erspart bei Handrechnung das Nachschlagen in der Tabelle der Fibonaccizahlen. Ordnet man alle Werte einer Iteration zeilenweise in einer Tabelle an, so verschieben sich diese Werte bei Übergang zur nächsten Iteration nach rechts bzw. nach links ab der Position der neuen Stützstelle:

i	$k(a_i)$	$k(x_i)$	$k(y_i)$	$k(b_i)$	$f(x_i)$	$f(y_i)$
0	0	5	8	13	$-0,943$	$-0,696$
1	0	3	5	8	$-0,999$	$-0,943$
2	0	2	3	5	$-0,993$	$-0,999$
3	2	3	4	5	$-0,999$	$-0,983$
4	2	3	3	4	$-0,999$	$-0,999$

Mit $x_4 = y_4 = 0 + 3h$ bricht das Verfahren ab und es gilt: das Minimum von f ist näherungsweise gegeben durch $\hat{x} = 0,462 \pm 0,154$ ($\hat{x}$ liegt mit Sicherheit im Intervall $[a_4, b_4] = [0 + 2h, 0 + 4h]$).

Vergleich von goldenem Schnitt und Fibonacci-Suche

Aus $1 - \sigma = \sigma^2$ folgt induktiv $\sigma^n = (-1)^n(F_{n-2} - F_{n-1}\sigma)$. Aus $F_0 = 1 = F_1$ und $F_1 = 1 < \sigma^{-1} < 2 = F_2$ erhalten wir $F_2 < \sigma^{-2} \cdot (\sigma^2 + \sigma) < F_3$, also mit Hilfe der ersten Formel $F_2 < \sigma^{-2} < F_3$. Induktiv folgen für $n > 1$ die Abschätzungen:

$$\frac{1}{F_n} > \sigma^n > \frac{1}{F_{n+1}}, \quad \lim_{n\to\infty} \frac{F_{n-1}}{F_n} = \sigma.$$

Demnach ist der Goldene Schnitt bei gleichem Aufwand von vergleichbarer kaum geringerer Genauigkeit. Auf der anderen Seite muß bei Fibonacci-Suche die gewünschte Genauigkeit a priori bekannt sein, um n zu bestimmen.

Beide Verfahren reduzieren die Intervallänge linear, d.h. es gilt $|b_{i+1} - a_{i+1}| \le \lambda |b_i - a_i|$ für ein $\lambda \in (0, 1)$. Dabei ist der Reduktionsfaktor λ durch σ bzw. durch eine Schranke für $\frac{F_{n-1}}{F_n}$ gegeben. Nach

$$n = \left\lceil \frac{1}{\log \lambda} \left(\log(b - a) + \log \frac{1}{\epsilon} \right) \right\rceil$$

Iterationsschritten ist die Länge des Restintervalls auf weniger als ϵ reduziert. Als log wird in solchen Abschätzungen üblicherweise der Logarithmus zur Basis 2 gewählt, da dann die Logarithmen der Eingabedaten als Größe des Problems (Speicherplatz in einem Computer) interpretiert werden können.

Differenzierbare und konvexe Funktionen

Da das Minimum einer auf einem offenen Intervall $I \subseteq \boldsymbol{R}$ stetig differenzierbaren Funktion f notwendigerweise auch eine Nullstelle der Ableitung $g \equiv f'$ ist, versucht man zunächst, ein $\hat{x} \in I$ mit $g(\hat{x}) = 0$ zu finden. Kennt man ein Intervall $[a, b] \subset I$ mit

$$g(a) < 0 < g(b),$$

so folgt aus der Stetigkeit von g die Existenz einer Nullstelle in (a, b). Ist g monoton, so gibt es genau eine Nullstelle und diese ist das globale Minimum von f. Das einfachste Verfahren zur Bestimmung einer Nullstelle in I ist das Bisektionsverfahren.

Algorithmus 1.3 (Das Bisektionsverfahren)

1. **Iteration:**
   $x := \frac{a+b}{2}$; $g_x := g(x)$;
   Wenn $g_x \approx 0$, dann $\hat{x} := x$; Stop.
   Wenn $g_x < 0$, dann $a := x$;
   Wenn $g_x > 0$, dann $b := x$;

2. **Abbruch:**
   Wenn $b - a < \epsilon$, dann $\hat{x} := x$; Stop.
   Gehe nach 1.

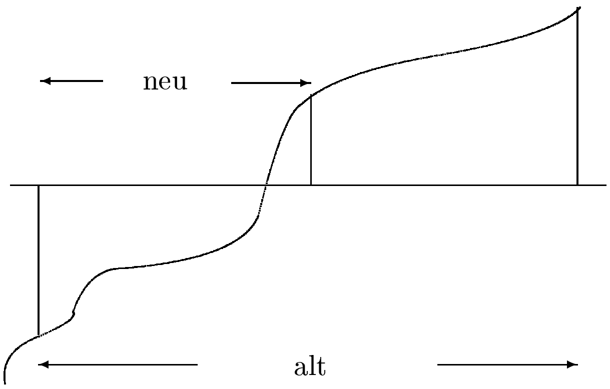


Abbildung III.6: Bisektion

Da das Bisektionsverfahren das Restintervall in jeder Iteration um den Faktor $\lambda = \frac{1}{2}$ reduziert, handelt es sich wieder um ein lineares Verfahren, das unter den angegebenen

Voraussetzungen stets eine Nullstelle von g lokalisiert. Schnellere Verfahren wie das im folgenden beschriebene Newton-Verfahren konvergieren im allgemeinen nur unter strengeren Voraussetzungen.

Sei wieder Intervall $[a,b] \subset I$ mit $g(a) < 0 < g(b)$ bekannt. Außerdem sei nunmehr $f \in C_2(I)$. Das Newton-Verfahren wird mit Hilfe der Iterationsfunktion

$$\Phi(x) := x - \frac{g(x)}{g'(x)}$$

beschrieben.

Algorithmus 1.4 (Newton-Verfahren)

1. $x_0 := b$.

2. Für $k = 0, 1, \ldots$*:*

$$x_{k+1} := \Phi(x_k);$$

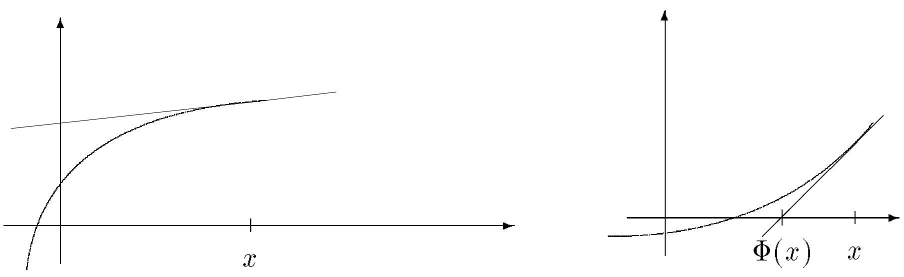


Abbildung III.7: Newton-Verfahren

Selbst wenn g monoton ist und daher eine eindeutige Nullstelle in (a,b) existiert, kann das Newtonverfahren versagen. Im folgenden betrachten wir eine wichtige Klasse von Funktionen, für die das Newton-Verfahren zum Erfolg führt.

Definition (Konvexe Funktionen): $g : I \to \boldsymbol{R}$ heißt *konvex*, falls

$$g(\lambda x + (1-\lambda)y) \leq \lambda g(x) + (1-\lambda)g(y)$$

für alle $\lambda \in (0,1)$, $x, y \in I$.

Im Abschnitt 2 werden konvexe Funktionen im $\boldsymbol{R}^n$ untersucht und auf verschiedene Weise charakterisiert. Für unsere Analyse des Newton-Verfahrens genügt uns die folgende grundlegende Eigenschaft konvexer, differenzierbarer Funktionen:

Lemma 1.1 (Gradientenungleichung) *$g : I \to \boldsymbol{R}$ sei eine konvexe, auf $(a,b) \subseteq I$ stetig differenzierbare Funktion. Dann gilt*

$$g(y) \geq g(x) + g'(x)(y-x)$$

für alle $x, y \in (a,b)$.

Beweis. Die Konvexitätsbedingung $g(\lambda y + (1-\lambda)x) \leq \lambda g(y) + (1-\lambda)g(x)$ impliziert $\frac{g(x+\lambda(y-x))-g(x)}{\lambda(y-x)}(y-x) \leq g(y)-g(x)$. Der Grenzübergang $\lambda \to 0$ führt daher auf $g'(x)(y-x) \leq g(y) - g(x)$. □

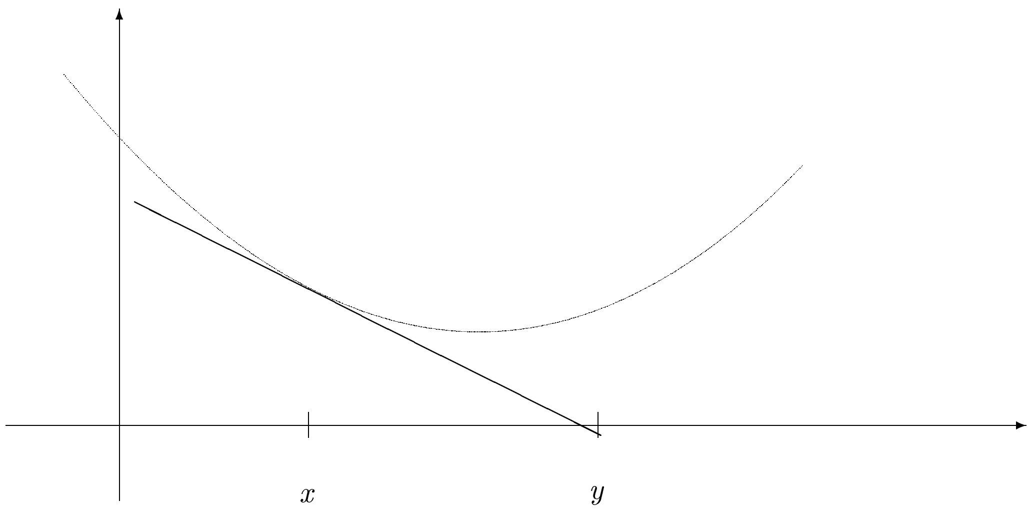


Abbildung III.8: Gradientenungleichung

Folgerung: Differenzierbare konvexe Funktionen besitzen monoton wachsende Ableitungen:

$$x < y \Rightarrow g'(x) \leq g'(y),$$

denn setzt man die Gradientenungleichung mit vertauschten Argumenten, d.h. $g(x) \geq g(y) + g'(y)(x-y)$ in die Gradientenungleichung ein, so folgt $g(y) \geq g(y) + g'(y)(x-y) + g'(x)(y-x)$. Also gilt $g'(y)(y-x) \geq g'(x)(y-x)$, woraus wegen $y - x > 0$ die Behauptung $g'(y) \geq g'(x)$ folgt.

Satz 1.2 (Globale Konvergenz des Newton-Verfahrens) *$g : I \to \boldsymbol{R}$ sei eine konvexe, auf $(a,b) \subseteq I$ stetig differenzierbare Funktion mit $g(a) < 0 < g(b)$. Dann besitzt g eine eindeutige Nullstelle $x_* \in (a,b)$, wobei $g'(x_*) > \frac{-g(a)}{b-a} > 0$, und für die im Newtonverfahren erzeugte Folge aus Algorithmus 1.4 gilt:*

1. *$(x_k) \searrow x_*$ (streng monoton fallend)*

Sei $\epsilon > 0$. Dann gilt zusätzlich:

2. *$g(x_k) \leq \epsilon$ für $2k \geq \left\lceil \log_2 \left(\frac{g'(b)g(b)}{g'(x_*)} \right) + \log_2 \frac{1}{\epsilon} \right\rceil$*

3. *$x_k - x_* \leq \epsilon$ für $2k \geq \left\lceil \log_2 \left(\frac{g'(b)g(b)}{(g'(x_*))^2} \right) + \log_2 \frac{1}{\epsilon} \right\rceil$*

Beweis. Da g stetig, gibt es nach Zwischenwertsatz eine Nullstelle $x_* \in (a,b)$. Angenommen, x'_* ist eine zweite Nullstelle. OBdA $x_* < x'_*$, d.h. $x_* = \lambda a + (1-\lambda)x'_*$ für ein $\lambda \in (0,1)$.

Da g konvex, folgt mit $0 = g(x_*) \leq \lambda g(a) + (1-\lambda)g(x'_*) = \lambda g(a) < 0$ ein Widerspruch. Im Rest des Beweises sei stets $a \leq x < x_* < y \leq b$. Aus der Eindeutigkeit der Nullstelle folgt für solche Argumente:

$$g(x) < 0 < g(y) \tag{1.2}$$

Die Gradientenungleichung liefert $g(a) \geq g(y) + g'(y)(a-y)$. Also folgt

$$g'(y) > \frac{g(y)-g(a)}{y-a} \geq \frac{-g(a)}{b-a} > 0. \tag{1.3}$$

Wegen (1.2) und (1.3) gilt $\Phi(y) < y$. Die Gradientenungleichung $g(y) + g'(y)(x_* - y) \leq g(x_*) = 0$ kann man nun durch $g'(y)$ teilen und erhält $x_* \leq y - \frac{g(y)}{g'(y)} = \Phi(y)$. Also ist die durch Algorithmus (1.4) erzeugte Newtonfolge x_k streng monoton fallend mit $x_k \geq x_*$.

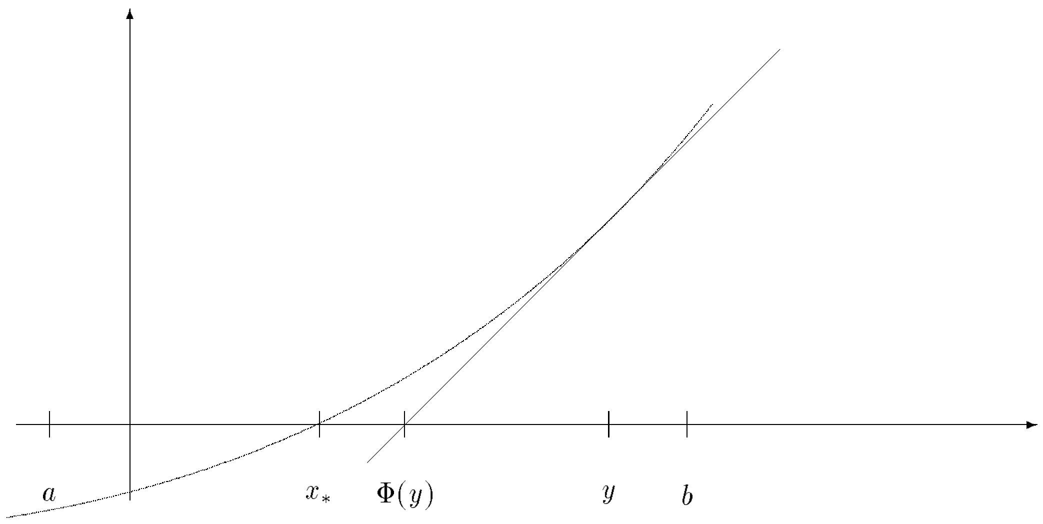


Abbildung III.9: Newton-Verfahren für konvexe Funktionen

Die Gradientenungleichung impliziert weiterhin

$$g(y) \geq g(\Phi(y)) + g'(\Phi(y))(y - \Phi(y)).$$

Mit Division durch $g(y)$ erhält man

$$1 \geq \frac{g(\Phi(y))}{g(y)} + \frac{g'(\Phi(y))}{g'(y)} =: \alpha + \beta.$$

Die Ungleichung zwischen dem geometrischen und dem arithmetischen Mittel ergibt nun $\sqrt{\alpha \cdot \beta} \leq \frac{1}{2}(\alpha+\beta) \leq \frac{1}{2}$, also $\alpha \cdot \beta \leq \frac{1}{4}$. Offensichtlich gilt

$$g\left(\Phi(y)\right) g'\left(\Phi(y)\right) \leq \frac{1}{4} g(y) g'(y),$$

woraus wir nach k Iterationen

$$g(x_k)g'(x_k) \leq (\frac{1}{4})^k g(b)g'(b)$$

erhalten. Zusammen mit $g'(x_k) \geq g'(x_*)$ liefert diese Ungleichung:

$$g(x_k) \leq (\frac{1}{4})^k \frac{g(b)g'(b)}{g'(x_*)}, \tag{1.4}$$

woraus (2.) folgt. Eine weitere Anwendung der Gradientenungleichung zeigt $g(x_k) \geq g(x_*) + g'(x_*)(x_k - x_*)$, also $\frac{g(x_k)}{g'(x_*)} \geq x_k - x_*$. Damit folgt aus (1.4) auch (3.). □

Definition (Konvergenzgeschwindigkeit): Sei (r_n) eine positiveNullfolge, d.h. $r_n \to 0, r_n > 0$. Dann heißt (r_n) *(mindestens) konvergent* von der Ordnung p, falls es ein $b > 0$ und ein $N \in \boldsymbol{N}$ gibt, so daß $r_{n+1} \leq b \cdot r_n^p$ für alle $n > N$.

(r_n) hat die *Konvergenzordnung* $p := \sup\{q > 0 \mid 0 \leq \beta_q = \limsup_{n\to\infty} \frac{r_{n+1}}{r_n^q} < \infty\}$. Dabei heißt β_q *Konvergenzfaktor* zur Ordnung q.

Beispiel:

1. Für $r_{n+1} := \frac{r_n}{2}$ mit $r_0 := 1$ gilt: $p = 1$, $\beta_1 = \frac{1}{2}$.
2. Für $r_{n+1} := r_n^2$ mit $r_0 := \frac{1}{2}$ gilt: $p = 2$, $\beta_2 = 1$.
3. Für $r_n := \frac{1}{n}$ gilt: $\beta_1 = 1$, $p = 1$.
4. Für $r_n := (\frac{1}{2})^{n^2}$ gilt: $\beta_1 = 0$,$p = 1$.

Insbesondere bezeichnet man $p = 1$, $0 < \beta_1 < 1$ als *lineare Konvergenz*, $p \geq 1, \beta_1 = 0$ als *superlineare Konvergenz* und $p = 2$ als *quadratische Konvergenz*. Die bei den bisherigen Verfahren (Goldener Schnitt, Fibonacci-Suche, Bisektionsverfahren) betrachtete Länge des Restintervalls ist jeweils eine linear konvergente Nullfolge, für die das Konvergenzverhalten unmittelbar ab $N = 1$ einsetzt. Im allgemeinen beschreibt die Konvergenzgeschwindigkeit nur das asymptotische Verhalten der Folge. Dies wird auch bei der folgenden Untersuchung des Newton-Verfahrens deutlich, die wir für lipschitzstetig differenzierbare Funktionen durchführen.

Lemma 1.3 (L-Lipschitzstetig differenzierbare Funktionen) *Sei $L > 0$ und g eine auf (a,b) stetig differenzierbare Funktion mit $|g'(y) - g'(x)| \leq L \cdot |y - x|$. Dann gilt:*

$$|g(y) - g(x) - g'(x) \cdot (y - x)| \leq \frac{1}{2}L(y - x)^2$$

für alle $x, y \in (a,b)$.

Beweis.

$$\begin{aligned} |g(y) - g(x) - g'(x)(y - x)| &\leq \int_x^y |g'(z) - g'(x)|dz \\ &\leq L\int_x^y (z - x)dz \\ &= \frac{L}{2}(y - x)^2 \end{aligned}$$

□

Mit Hilfe dieser Abschätzung kann man die Konvergenzgeschwindigkeit des Newtonverfahrens bestimmen.

Satz 1.4 (Lokale Konvergenz des Newton-Verfahrens) *Sei f eine auf (a,b) zweimal stetig differenzierbare Funktion mit L-lipschitzstetig differenzierbarer Ableitung $g \equiv f'$ und mit $|g'| \geq \rho > 0$. Besitzt $g(x) = 0$ eine Lösung $x_* \in (a,b)$, dann existiert ein $\eta > 0$ mit*

1. $|x_0 - x_*| < \eta \quad \Rightarrow \quad (x_k) \to x_*$
2. $|x_{k+1} - x_*| \leq \frac{L}{2\rho}(x_k - x_*)^2$

für die Newtonfolge $x_{k+1} := x_k - \frac{g(x_k)}{g'(x_k)}$.

Beweis. (Übungsaufgabe) □

Der Satz formuliert lokal, hinreichend nahe beim gesuchten Minimum x_*, eine Konvergenzaussage. Dort liefert das Newtonverfahren in jedem Schritt eine Verdopplung der Genauigkeit, d.h. eine typische Entwicklung des Abstandes $|x_{k+1} - x_*|$ ist etwa 10^{-2}, 10^{-4}, $10^{-8}, \ldots$.

Newton-Verfahren mit finiten Differenzen

Approximiert man die zur Durchführung des Newtonverfahrens explizit benötigte Ableitung von f, so erhält man eine Reihe von Varianten. Die geschickte Approximation $g'(x_k) \approx \frac{g(x_k) - g(x_{k-1})}{x_k - x_{k-1}}$ führt auf das Sekantenverfahren:

$$x_{k+1} = x_k - g(x_k)\frac{x_k - x_{k-1}}{g(x_k) - g(x_{k-1})}, \quad k \in \boldsymbol{N}.$$

wobei zur Initialisierung zwei Startpunkte x_0, x_1 gewählt werden müssen. Bei allgemeiner Approximation der Ableitung mit Hilfe des Differenzenquotienten zur Schrittweite h_c spricht man vom Newton-Verfahren mit finiten Differenzen. Der jeweils neue Iterationspunkt x_+ ergibt sich aus dem alten x_c durch

$$a_c := \frac{g(x_c + h_c) - g(x_c)}{h_c}; \quad x_+ := x_c - \frac{g(x_c)}{a_c}$$

Konvergenzabschätzung

Für eine auf (a,b) L-lipschitzstetig differenzierbare Funktion g mit $x_c, x_c + h_c \in (a,b)$ ist die Qualität der Approximation abschätzbar durch:

$$|a_c - g'(x_c)| \leq \frac{L}{2}|h_c|,$$

denn nach Lemma 1.3 gilt $|g(x_c + h_c) - g(x_c) - h_c g'(x_c)| \leq \frac{L}{2}|h_c|^2$. Ist $g(x_*) = 0$, so ergibt sich

$$\begin{aligned} x_+ - x_* &= -\frac{g(x_c)}{a_c} + x_c - x_* \\ &= \frac{1}{a_c}\left[g(x_*) - g(x_c) - a_c(x_* - x_c)\right] \\ &= \frac{1}{a_c}\left[g(x_*) - g(x_c) - g'(x_c)(x_* - x_c) + (g'(x_c) - a_c)(x_* - x_c)\right] \end{aligned}$$

Mit $e_+ := |x_+ - x_*|, e_c := |x_* - x_c|$ folgt $e_+ \leq \frac{1}{a_c}\left[\frac{L}{2}(x_* - x_c)^2 + \frac{L}{2}|h_c|e_c\right]$, und daher

$$e_+ \leq \frac{L}{2|a_c|}(e_c + |h_c|)e_c \tag{1.5}$$

Diese Abschätzung impliziert unter einer Reihe unterschiedlicher Annahmen Konvergenzaussagen:

Satz 1.5 (Lokale Konvergenz mit finiten Differenzen) *Sei f eine auf (a,b) zweimal stetig differenzierbare Funktion mit L-lipschitzstetig differenzierbarer Ableitung $g \equiv f'$ und mit $|g'| \geq \rho > 0$. Besitzt $g(x) = 0$ eine Lösung $x_* \in (a,b)$, dann existieren $\eta, \eta' > 0$, so daß für $0 < |h_k| \leq \eta', |x_0 - x_*| < \eta$ gilt:*

1. *x_k konvergiert mindestens linear gegen x_*,*
2. *falls $h_k \to 0$, so ist die Konvergenz mindestens superlinear,*
3. *falls $|h_k| \leq \alpha|x_k - x_*|$ oder $|h_k| \leq \alpha|g(x_k)|$ für ein $\alpha > 0$, so ist die Konvergenz mindestens quadratisch,*
4. *falls $|h_k| \leq \alpha|x_k - x_{k-1}|$ für ein $\alpha > 0$, so ist die Konvergenz mindestens quadratisch über je zwei Schritte .*

Beweis. (Übungsaufgabe), mittels (1.5) □

Bemerkung (Numerischer Vergleich und numerische Durchführung):

1. Das Newtonverfahren benötigt zwei Funktionsauswertungen (von g und g') pro Iteration und ist mindestens quadratisch konvergent. Unter den gleichen Voraussetzungen benötigt das Sekantenverfahren zwei Auswertungen (von g) in zwei Iterationen und ist wegen (4.) des vorstehenden Satzes, wenn man je zwei Iterationen zusammenfaßt, ebenfalls quadratisch konvergent. Eine genauere Konvergenzanalyse (s. Stoer [29]) zeigt sogar $p = 2.618\ldots = 1 + \frac{\sqrt{5}+1}{2}$ für Doppelschritte beim Sekantenverfahren.
2. Finite Differenzen sind numerisch nicht unproblematisch. Aus $h_k := \alpha|g(x_k)| \to 0$ folgt $x_k + h_k \to x_k$. Eine praktikable Wahl ist etwa $h_k := \sqrt{eps} \cdot \max\{\text{typ } x, |x_k|\}$. Hierbei gibt typ x die zu erwartende Größenordnung von x und eps die Maschinengenauigkeit an.
3. Im Gegensatz zur Nullstellensuche ist bei der Minimierung von Funktionen eine natürliche Qualitätskontrolle zur Hand. Falls $f(x_{k+1}) < f(x_k)$, ist die Iteration sinnvoll. Anderenfalls, falls wenigstens eine Abstiegsrichtung vorliegt, d.h. $f'(x_k)(x_{k+1} - x_k) < 0$ gilt, so stimmt die Richtung des Iterationsschritts und man kann versuchen durch, falls notwendig mehrfache, Halbierung der Schrittweite einen besseren Iterationspunkt zu finden. Ist die Richtung keine Abstiegsrichtung, so kann man in der Gegenrichtung suchen.

4. Als Abbruchkriterien kann man viele verschiedene Bedingungen verwenden, die alle mehr oder minder gut motiviert werden können. Beim Newton-Verfahren benutzt man die auftretenden relativen bzw. absoluten Änderungen:
$$\frac{|x_+ - x_c|}{\max\{\text{typ } x, |x_+|\}} < \tau_1; \quad \left|\frac{f'(x_+)}{f(x_+)}x_+\right| < \tau_2; \quad \left|\frac{f'(x_+)}{f(x_+)}\right| < \tau_3.$$

Interpolation

Sei f zweimal stetig differenzierbar auf dem offenen Intervall I. Gesucht wird wieder $f(x_*) = \min\{f(x) \mid x \in I\}$. Man kann nun f durch ein Interpolationspolynom P zu f approximieren und versuchen, dessen Minimum x_+ mit $P'(x_+) = 0, P''(x_+) > 0$ als Näherung für $\hat{x}$ zu verwenden. Das quadratische Interpolationspolynom P (zu f an der Stelle x_c) und seine Ableitung sind durch

$$P(x) = f(x_c) + f'(x_c)(x - x_c) + \frac{1}{2}f''(x_c)(x - x_c)^2$$
$$P'(x) = f'(x_c) + f''(x_c)(x - x_c)$$

gegeben (Taylorentwicklung). Löst man die zweite Gleichung nach x auf, so erkennt man, daß eine Newton-Iteration eine Nullstelle von P' liefert:

$$x_+ = x_c - \frac{f'(x_c)}{f''(x_c)}.$$

Diese Idee verwendet man in der quadratischen Interpolation iterativ. Allerdings wird statt des Taylorpolynoms das Lagrange'sche Interpolationspolynom zu drei meist äquidistanten Stützstellen benutzt:

$$f(x) \approx \sum_{i=1}^{3} f_i \frac{\prod_{j\neq i}(x - x_j)}{\prod_{j\neq i}(x_i - x_j)} =: P(x)$$

Äquidistante quadratische Interpolation

Zu äquidistanten Stützstellen $x := y - h$, y, $z := y + h$ mit Schrittweite h seien die Stützwerte $f_\chi := f(\chi)$ für $\chi \in \{x, y, z\}$ bekannt. Dann ist die Nullstelle x_+ von P' gegeben durch:

$$x_+ = y + \frac{h}{2}\frac{f_x - f_z}{f_x - 2f_y + f_z}.$$

Der Nenner ist wohldefiniert, wenn die 2. Ableitung nicht verschwindet, denn

$$P''(x_+) = \frac{1}{h^2}(f_x - 2f_y + f_z),$$

woran man auch erkennt, ob ein Minimum oder Maximum bestimmt wurde. Die für ein Minimum hinreichende Bedingung $f_y < \min(f_x, f_z)$ impliziert auch $x_+ \in [x, z]$. Daher wird zunächst eine Näherungslösung y zur Schrittweite h gesucht, für die die Bedingung

$$f_y \leq f_x, f_z$$

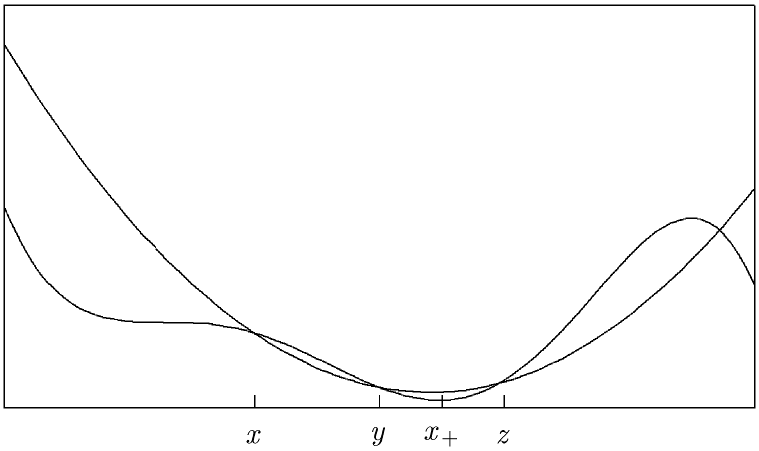


Abbildung III.10: Quadratische Interpolationsfunktionen

erfüllt ist. Dann wird x_+ berechnet und anschließend die Schrittweite h verkleinert. Kennt man näherungsweise den minimalen Funktionswert $f_* \approx f(x_*)$, und die Ableitung $f'(y)$, so kann die Schrittweite mit

$$h := 2\frac{f_* - f_y}{f'(y)} \tag{1.6}$$

automatisch gesteuert werden. Diese Formel erhält man, wenn man die Taylorentwicklung (1. Ordnung) der Ableitung

$$0 = f'(x_*) = f'(y) + f''(y)h$$

mit $(-\frac{h}{2})$ multipliziert, von der Taylorentwicklung (2. Ordnung) der Funktion

$$f_* = f(x_*) = f(y) + f'(y)h + \frac{1}{2}f''(y)h^2$$

abzieht, und die resultierende Gleichung

$$f_* = f(y) + \frac{h}{2}f'(y)$$

nach h auflöst. Im folgenden Verfahren ist die systematische Suche nach einem passenden Näherungswert y mit einer teilweisen Schrittweitensteuerung gekoppelt. Die in Gleichung (1.6) angegebene Schrittweite kann gegebenenfalls in den Schritten (1.) und (6.) verwendet werden.

Algorithmus 1.5 (Das Verfahren nach Davies, Swann und Campey)

1. Wähle Näherung y und Schrittweite h.

2. Falls $f(y) < f(y-h)$ und $f(y) < f(y+h)$, dann gehe nach Schritt 5.

   Anderenfalls wähle das Vorzeichen von h, so daß $f(y) \geq f(y+h)$.

3. $h_0 := h$;

   Solange $f(y+h) \geq f(y+2h)$: $h := 2h$;

   Falls $h = h_0$, dann $y := y + h$ und gehe nach 5.

   $h := \frac{1}{2}h$.

4. Falls $f(y+2h) \leq f(y+3h)$, dann $y := y + 2h$ und gehe nach 5.

   $y := y + 3h$.

5. $x_+ := y + \frac{h}{2} \cdot \frac{f(y-h)-f(y+h)}{f(y-h)-2f(y)+f(y+h)}$

6. Falls $f(y) > f(x_+)$, dann $y := x_+$.

   Falls Abbruchbedingung erfüllt, stop mit Ergebnis y.

   Wähle kürzere Schrittlänge h und gehe nach 2.

2 Lokale und Globale Minima

Stetig differenzierbare Funktionen bezeichnen wir wie üblich kurz als C_1-Funktionen, m-mal stetig differenzierbare als C_m-Funktionen. Ist f eine C_m-Funktion auf $X \subseteq \boldsymbol{R}^n$, so schreiben wir kurz $f \in C_m(X)$. Die Menge der inneren Punkte von X bezeichnen wir mit int$X$. Ist $f \in C_1$, so bezeichnet $\nabla f = (\frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_n})$ den Gradienten, ist $f \in C_2$, so bezeichnet $F = \nabla^2 f = (\frac{\partial^2 f}{\partial x_i \partial x_j})$ die Matrix der zweiten partiellen Ableitungen von f.

In diesem Abschnitt stellen wir Begriffe, Optimalitäts- und Konvergenzbedingungen für Verfahren zulässiger Richtungen zusammen, die als theoretische Grundlage zur Lösung der Aufgabe

$$\min\{f(x) \mid x \in X\}$$

nützlich sind.

Optimalitätsbedingungen

Definition (Minima): $U_\epsilon(x_*) := \{x \mid \| x - x_* \| < \epsilon\}$ heißt *ϵ-Kugel* um x_*.

1. $x_* \in X$ heißt *lokales Minimum* (von f auf X) genau dann, wenn

$$f(x_*) \leq f(x) \quad \exists\, \epsilon > 0 \quad \forall\, x \in X \cap U_\epsilon(x_*), x \neq x_*.$$

2. $x_* \in X$ heißt *globales Minimum* (von f auf X) genau dann, wenn

$$f(x_*) \leq f(x) \quad \forall\, x \in X, x \neq x_*.$$

x_* heißt *strenges lokales bzw. globales Minimum*, wenn entsprechend "$<$" gilt.

In Algorithmen werden häufig Richtungen erzeugt, in denen dann entweder die aktuelle Lösung verbessert oder deren Optimalität in der gewählten Richtung festgestellt werden muß. Natürlich sind nur solche Richtungen wesentlich, in denen zumindest nahe bei der aktuellen Lösung noch zulässige Punkte (in X) existieren.

Definition (Intervalle und zulässige Richtungen):

1. Für $a, b \in \boldsymbol{R}^n$ heißt die Menge $[a, b] := \{\lambda a + (1-\lambda)b \mid 0 \leq \lambda \leq 1\}$ *abgeschlossenes Intervall* von a bis b.

2. Sei $x \in X$. $d \in \boldsymbol{R}^n$ heißt *zulässige Richtung (bei x)* , falls ein positives λ mit $[x, x + \lambda d] \subseteq X$ existiert.

Die Menge aller zulässigen Richtungen (bei x) wird mit $D(x)$ bezeichnet.

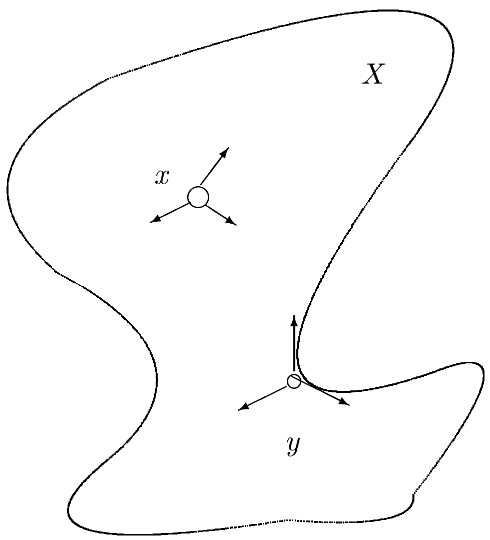


Abbildung III.11: Zulässige Richtungen

Zunächst soll an die aus der Analysis bekannten Optimalitätsbedingungen erinnert werden.

Satz 2.1 (Notwendige Bedingungen) *x_* sei lokales Minimum von $f \in C_1(X)$.*

1. *Da $f \in C_1$, so gilt:* $\quad \nabla f(x_*)d \geq 0 \quad \forall\, d \in D(x_*)$

2. *Ist $f \in C_2$, so gilt außerdem:* $\quad \nabla f(x_*)d = 0 \Rightarrow d^T F(x_*)d \geq 0 \quad \forall\, d \in D(x_*)$

Beweis. zu (1.): Zu $d \in D(x_*)$ mit $\nabla f(x_*)d < 0$ sei $x(\lambda) := x_* + \lambda d \in X$ für $\lambda \in [0, \bar{\lambda}], \bar{\lambda} > 0$. Dann besitzt die Funktion $g(\lambda) := f(x(\lambda))$ auf $[0, \bar{\lambda}]$ ein lokales Minimum in $\lambda = 0$ mit

Ableitung $g'(0) = \nabla f(x_*)d < 0$. Nun ergibt sich aus der Taylorentwicklung (1. Ordnung) der Ableitung für $\lambda > 0$

$$\frac{g(\lambda) - g(0)}{\lambda} = g'(0) + \frac{o(\lambda)}{\lambda},$$

ein Widerspruch, da bekanntlich $\frac{o(\lambda)}{\lambda} \to 0$ für $\lambda \to 0$. Für positive λ ist die linke Seite stets nichtnegativ, die rechte Seite jedoch wird für hinreichend kleine positive λ negativ.

Zu (2.): Hier gilt nun zusätzlich $g'(0) = 0, g''(0) = d^T F(x_*)d < 0$. Daher lautet die Taylorentwicklung (2. Ordnung) nun

$$g(\lambda) - g(0) = \frac{1}{2}g''(0)\lambda^2 + o(\lambda^2)$$

und man erhält einen analogen Widerspruch. □

Eine $n{\cdot}n$-Matrix A heißt bekanntlich *positiv semidefinit*, falls $d^T Ad \geq 0$ für alle $d \in \boldsymbol{R}^n$, und *positiv definit*, falls $d^T Ad > 0$ für alle $d \in \boldsymbol{R}^n \backslash \{0\}$.

Korollar 2.2 *Sei $x_* \in intX$. Ist $f \in C_1$, so gilt $\nabla f(x_*) = 0$. Ist $f \in C_2$, so ist zusätzlich $F(x_*)$ positiv semidefinit.*

Satz 2.3 (Hinreichende Bedingungen) *Sei $x_* \in intX, f \in C_2$. Falls $\nabla f(x_*) = 0$ und $F(x_*)$ positiv definit, so ist x_* strenges lokales Minimum von f auf X.*

Beweis. Da F stetig, ist $F(x)$ positiv definit auf $U_\epsilon(x_*)$ für hinreichend kleine $\epsilon > 0$. Die Taylorentwicklung (2. Ordnung) liefert $f(x_*+d)-f(x_*) = \frac{1}{2}d^T F(y)d$ für ein $y \in [x_*, x_*+d]$. Da nun $F(y)$ positiv definit für alle d mit $\| d \| < \epsilon$, folgt hieraus $f(x_* + d) > f(x_*)$ für alle d mit $\| d \| < \epsilon$. □

Korollar 2.4 *Sei $x_* \in intX, f \in C_2, F(x_*)$ nicht singulär. Dann gilt: x strenges lokales Minimum von f auf X genau dann, wenn $\nabla f(x_*) = 0 und F(x_*)$ positiv semidefinit.*

Beweis. Nach Korollar 2.2 sind die Bedingungen notwendig. Sie sind auch hinreichend, da $F(x_*)$ positiv semidefinit und nicht singulär, also positiv definit ist. □

Konvexe Funktionen

Zunächst wird die Definition der Konvexität auf die hier betrachteten Funktionen ausgedehnt.

Definition (Konvexe Mengen und Funktionen):

1. Sei $X \subseteq \boldsymbol{R}^n$. Dann heißt X konvex, falls $[a, b] \subseteq X$ für alle $a, b \in X$.
2. Sei $X \subseteq \boldsymbol{R}^n$. Dann heißt

   $$convX := \left\{ \sum_{i=1}^{m} \lambda_i x_i | m \in \boldsymbol{N}, x_i \in X, \lambda_i \in \boldsymbol{R}_+, \sum_{i=1}^{m} \lambda_i = 1 \right\}$$

   die *konvexe Hülle* von X.

3. Sei X konvex. Dann heißt $f : X \to \boldsymbol{R}$ konvex (auf X), falls

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y)$$

für alle $\lambda \in [0,1]$, $x, y \in X$.

Bemerkung:

1. $convX$ ist konvex.
2. Sei $X \subseteq Y$,Y konvex. Dann ist $convX \subseteq Y$. D. h. $convX$ ist die kleinste konvexe Obermenge von X.
3. Sei X konvex. Dann ist $convX = X$.

Beweis. (Übungsaufgabe)

Offenbar ist eine Funktion konvex, wenn sie eingeschränkt auf die abgeschlossenen Intervalle ihres konvexen Definitionsbereiches konvex ist.

Lemma 2.5 *Sei f konvex (auf X). Dann sind die Niveaumengen*

$$A_\alpha := \{x \in X \mid f(x) < \alpha\} \quad , \quad B_\alpha := \{x \in X \mid f(x) \leq \alpha\}$$

konvex für alle $\alpha \in \boldsymbol{R}$.

Beweis. Für A_α: Seien $x, y \in A_\alpha, \lambda \in (0,1)$. Dann folgt $f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y) < \alpha$. Für B_α ist der Beweis analog. □

Bemerkung: Sei f konvex (auf X). Es gilt:

$$\forall x \in Y \subset X : f(x) \leq \alpha \Rightarrow \forall x \in convY : f(x) \leq \alpha$$

.

Denn: $Y \subset B_\alpha$, B_α konvex $\Rightarrow convY \subset B_\alpha$.

Eine überraschende Konsequenz der Konvexität zeigt der folgende Satz.

Satz 2.6 (Stetigkeit) *Sei f konvex (auf X). Dann ist f stetig auf $intX$.*

Beweis. $\bar{x} \in \mathrm{int}X$ beliebig. OBdA $\bar{x} = 0$, $f(\bar{x}) = 0$. (Nötigenfalls kann man sonst die Funktion $\tilde{f}(y) := f(y) - f(\bar{x})$ betrachten). Zu zeigen ist daher nur: $\| x \| < \delta(\epsilon) \Rightarrow |f(x)| < \epsilon$.

Sei $W_\rho := \{x \mid \| x \|_\infty \leq \rho\} \subseteq X$ und sei M das Maximum von f auf den endlich vielen Ecken des Würfels W_ρ. Jeder Punkt des Würfels ist Konvexkombination gewisser Ecken V_ρ ($convV_\rho = W_\rho$). Da B_M konvex ist und die Ecken enthält, gilt $W_\rho \subseteq B_M$, d.h. $f|_{W_\rho} \leq M$.

Für x mit $\| x \| = \alpha \leq \rho$ gilt $u = \frac{\rho}{\alpha}x \in W_\rho$. Also ist $f(\frac{\rho}{\alpha}x) \leq M$. Nun liegt x im Interval $[0,u]$. Da f konvex, folgt $f(x) = f\left((1-\frac{\alpha}{\rho})0 + \frac{\alpha}{\rho}u\right) \leq \frac{\alpha}{\rho}M$.

Analog erhält man aus $\| -x \| = \alpha$ die Abschätzung $f(-x) \leq \frac{\alpha}{\rho}M$. Da f konvex und $0 \in [-x, x]$ liegt, folgt $0 = f(0) \leq \frac{1}{2}f(x) + \frac{1}{2}f(-x) \leq \frac{1}{2}(f(x) + \frac{\alpha}{\rho}M)$. Also gilt auch $f(x) \geq -\frac{\alpha}{\rho}M$.

Zusammen ergibt sich $|f(x)| \leq \frac{M}{\rho}\alpha = \frac{M}{\rho} \parallel x \parallel$, woraus man die Stetigkeit im Ursprung abliest. $\square$

Auch Differenzierbarkeitseigenschaften folgen aus der Konvexität, allerdings nur in Richtungen.

Definition (Richtungsableitungen): Sei f eine Funktion auf X, $\bar{x} \in X$, $d \in D(\bar{x})$. Falls der Grenzwert

$$\lim_{\lambda \to 0^+} \frac{f(\bar{x} + \lambda d) - f(\bar{x})}{\lambda}$$

existiert, so heißt er *Richtungsableitung* von f bei $\bar{x}$ in Richtung d und wird mit $f'(\bar{x}; d)$ bezeichnet.

Satz 2.7 (Richtungsdifferenzierbarkeit) *Sei f konvex (auf X), $d \in D(\bar{x})$. Dann existiert $f'(\bar{x}; d)$ und es gilt*

$$f'(\bar{x}; d) = \inf_{\lambda > 0} \frac{f(\bar{x} + \lambda d) - f(\bar{x})}{\lambda}.$$

Beweis. Sei $\lambda > \mu > 0$ hinreichend klein. Dann folgt aus der Konvexität

$$f(\bar{x} + \mu d) = f(\frac{\mu}{\lambda}(\bar{x} + \lambda d) + (1 - \frac{\mu}{\lambda})\bar{x}) \leq \frac{\mu}{\lambda} f(\bar{x} + \lambda d) + (1 - \frac{\mu}{\lambda}) f(\bar{x}),$$

also nach elementarer Umformung

$$\frac{f(\bar{x} + \mu d) - f(\bar{x})}{\mu} \leq \frac{f(\bar{x} + \lambda d) - f(\bar{x})}{\lambda}.$$

Offenbar ist die Funktion $g(\lambda) := \frac{1}{\lambda}(f(\bar{x} + \lambda d) - f(\bar{x}))$ für $\lambda \to 0$ nicht wachsend. Läßt man den uneigentlichen Grenzwert $-\infty$ zu, so existiert $\lim_{\lambda \to 0^+} g(\lambda) = \inf_{\lambda > 0} g(\lambda)$ in jedem Fall. $\square$

Die Konvexitätsungleichung und die Gradientenungleichung schließen die zugrundeliegende konvexe Funktion in jeder zulässigen Richtung in einer Schere ein. Die obere Abschätzung durch die Konvexität wird dabei immer enger, je kürzer die betrachtete Richtung ist (woraus wieder die Stetigkeit folgt).

Korollar 2.8 *Seien $\bar{x}, \bar{x} + d \in X$. Dann gilt für alle $\lambda \in [0, 1]$:*

$$f(\bar{x}) + \lambda f'(\bar{x}; d) \leq f(\bar{x} + \lambda d) \leq f(\bar{x}) + \lambda(f(\bar{x} + d) - f(\bar{x}))$$

Bemerkung: Die Glattheitseigenschaften konvexer Funktionen erlauben es, das eindimensionale Newton-Verfahren auf einem Intervall auch ohne explizite Annahme der stetigen Differenzierbarkeit zu definieren. Seine globale Konvergenz kann ähnlich wie im Beweis von Satz 1.2 nachgewiesen werden.

Das folgende Lemma zeigt, daß jeder Punkt einer konvexen Menge mit inneren Punkten über innere Punkte erreicht werden kann.

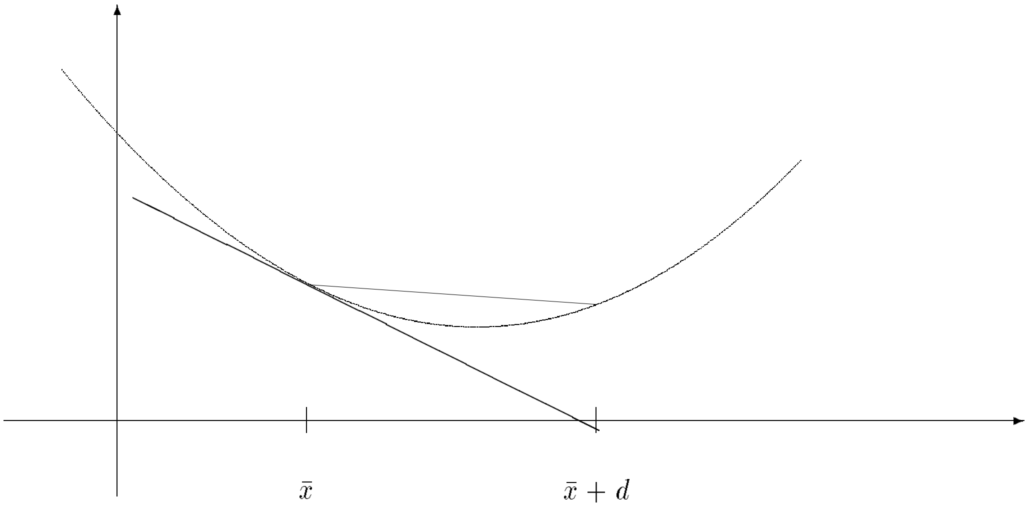


Abbildung III.12: Einschließung konvexer Funktionen

Lemma 2.9 (Erreichbarkeit) *Sei X konvex. Dann gilt $[\tilde{x}, x) \subseteq intX$ für alle $\tilde{x} \in intX$, $x \in X$.*

Beweis. Sei x' ein beliebiger Punkt in $(\tilde{x}, x)$, d.h. $x' = \lambda\tilde{x} + (1-\lambda)x$ mit $\lambda \in (0,1)$. Nach $\tilde{x}$ aufgelöst ergibt sich $\tilde{x} = \frac{1}{\lambda}(x' - (1-\lambda)x)$. Sei $U_\delta(\tilde{x}) \subseteq X$ und $y \in U_{\delta\lambda}(x')$. Sei $z := \frac{1}{\lambda}(y - (1-\lambda)x)$. Dann gilt $\| \tilde{x} - z \| = \frac{1}{\lambda} \| x' - y \|$, also $z \in U_\delta(\tilde{x})$. Da mit $z, x \in X$ auch die Konvexkombination y in X liegt, gilt $U_{\delta\lambda}(x') \subseteq X$. □

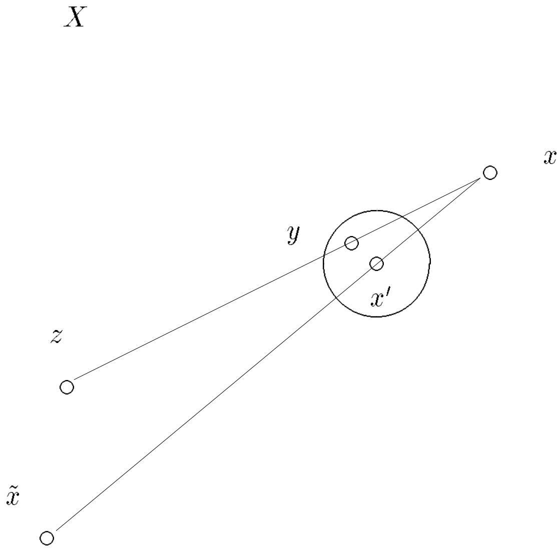


Abbildung III.13: Erreichbarkeitslemma

Das Erreichbarkeitslemma ermöglicht es, die Charakterisierung zweiter Ordung der Konvexität im folgenden Satz unter schwachen Voraussetzungen zu formulieren.

Satz 2.10 (Charakterisierung differenzierbarer konvexer Funktionen)
Sei $X \subseteq \boldsymbol{R}^n$ konvex, $f : \boldsymbol{R}^n \to \boldsymbol{R}$.

1. *(Gradientenungleichung):*
   Sei $f \in C_1$. Dann ist f konvex auf X genau dann, wenn $f(y) \geq f(x) + \nabla f(x)(y-x)$ für alle $x, y \in X$.

2. *Sei $f \in C_2$ und $\mathrm{int}X \neq \emptyset$. Dann ist f konvex auf X genau dann, wenn $F(x)$ positiv semidefinit für alle $x \in X$.*

Beweis. (zu 1.) Sei f konvex, d.h. $f(\alpha y + (1-\alpha)x) \leq \alpha f(y) + (1-\alpha) f(x)$, für $1 > \alpha > 0$. Nach elementaren Umformungen ergibt sich $\frac{1}{\alpha}\left(f(x + \alpha(y-x)) - f(x)\right) \leq f(y) - f(x)$, woraus für $\alpha \to 0_+$ die Gradientenungleichung folgt: $\nabla f(x)(y-x) \leq f(y) - f(x)$. Andererseits gelte die Gradientenungleichung. Für $x, y \in X$, $z := \lambda x + (1-\lambda) y \in [x,y] \subseteq X$ ergeben sich für x, z bzw. y, z:

$$f(x) \geq f(z) + \nabla f(z)(x - z)$$

$$f(y) \geq f(z) + \nabla f(z)(y - z)$$

Multipliziert man die erste Ungleichung mit λ, die zweite mit $(1-\lambda)$ und addiert beide, so ergibt sich

$$\lambda f(x) + (1-\lambda) f(y) \geq f(z) + \nabla f(z)\left[\lambda x + (1-\lambda) y - z\right],$$

wobei die eckige Klammer wegen der Definition von z verschwindet.

(zu 2.) Die Taylorentwicklung (1. Ordnung) liefert ein $\tilde{x} \in [x,y]$ mit

$$f(y) = f(x) + \nabla f(x)(y-x) + \frac{1}{2}(y-x)^T F(\tilde{x})(y-x) \tag{2.1}$$

Da F positiv semidefinit, folgt aus dieser Gleichung die Gradientenungleichung, und aus dieser nach (1.) die Konvexität von f.

Sei andererseits F nicht positiv semidefinit, etwa

$$(y - \bar{x})^T F(\bar{x})(y - \bar{x}) < 0 \tag{2.2}$$

für ein $\bar{x} \in X$ und für ein $y \in \boldsymbol{R}^n$.

OBdA $\bar{x} \in \mathrm{int}X$. Falls $\bar{x} \notin \mathrm{int}X$, so gilt wegen $\hat{x} \in \mathrm{int}X$ nach dem Erreichbarkeitslemma $[\hat{x}, \bar{x}] \subseteq \mathrm{int}X$. Aus Stetigkeitsgründen folgt aus der Ungleichung 2.2

$$(y - x)^T F(x)(y - x) < 0 \quad \forall\, x \in U(\bar{x}) \cap X.$$

Man kann daher in $U(\bar{x}) \cap [\hat{x}, \bar{x}]$ ein passendes neues $\bar{x}$ wählen.

Die Stetigkeit von F und die Ungleichung (2.2) ermöglichen nun die Wahl einer Umgebung $V(\bar{x}) \subseteq X$ mit

$$(y - \bar{x})^T F(\tilde{x})(y - \bar{x}) < 0 \quad \forall\, \tilde{x} \in V(\bar{x}). \tag{2.3}$$

Wir wählen jetzt ein $\bar{y} \in (\bar{x}, y) \cap V(\bar{x})$, d.h. nahe genug bei $\bar{x}$, so daß $[\bar{x}, \bar{y}] \subseteq X$. Ersetzt man dann y in der Ungleichung (2.3) durch $\bar{y}$, bleibt diese gültig. Daher gilt insbesondere

$$(\bar{y} - \bar{x})^T F(\tilde{x})(\bar{y} - \bar{x}) < 0 \quad \forall\, \tilde{x} \in [\bar{x}, \bar{y}].$$

Einsetzen in die Taylorentwicklung (2.1) für $x := \bar{x}, y := \bar{y}$, liefert die Ungleichung $f(y) < f(x) + \nabla f(x)(y - x)$, d.h. die Gradientenungleichung gilt nicht und daher ist nach (1.) f nicht konvex. □

Definition (Pseudokonvexe Funktionen): X konvex, $f \in C_1$. f heißt *pseudokonvex* auf X, falls für alle $x, y \in X$ gilt:

$$\nabla f(x)(y - x) \geq 0 \Rightarrow f(y) \geq f(x)$$

f heißt *pseudokonkav*, wenn $-f$ pseudokonvex ist, d. h. wenn für alle $x, y \in X$ gilt:

$$\nabla f(x)(y - x) \leq 0 \Rightarrow f(y) \leq f(x)$$

Die Gradientenungleichung zeigt, daß konvexe C_1-Funktionen pseudokonvex sind. Die Umkehrung ist im allgemeinen falsch. Wir betrachten als Beispiel die Funktion f mit $f(x) = x + x^3, f'(x) = 1 + 3x^2, f''(x) = 6x$. Da $f'' < 0$ für $x < 0$, ist f nicht konvex. Andererseits ist f pseudokonvex, denn für alle reellen x, y gilt:

$$f'(x)(y - x) \geq 0 \Rightarrow y - x \geq 0 \Rightarrow f(y) \geq f(x).$$

Satz 2.11 *Sei X konvex, f pseudokonvex. Dann ist $x_* \in X$ globales Minimum von f auf X genau dann, wenn $\nabla f(x_*)d \geq 0$ für alle $d \in D(x_*)$.*

Beweis. Nach Satz 2.1 ist die Bedingung notwendig. Andererseits sei $y \in X$ mit $y \neq x_*$. Dann gilt $y - x_* \in D(x_*)$, und daher ist nach Voraussetzung $\nabla f(x_*)(y - x_*) \geq 0$. Da f pseudokonvex, folgt $f(y) \geq f(x_*)$. □

Offenbar ist für pseudokonvexe Funktionen lokale Optimalität identisch mit globaler Optimalität. Abstiegsverfahren, mit deren Hilfe systematisch entsprechende Punkte gesucht werden, die die notwendige Bedingung des Satzes 2.1 erfüllen, lassen sich in folgender Weise schematisch formulieren.

Algorithmus 2.1 (Abstiegsverfahren)

1. ***Initialisierung***
   Finde $x \in X$.

2. ***Wahl der Abstiegsrichtung***
   Falls $\nabla f(x)d \geq 0 \quad \forall\, d \in D(x)$, dann stop.
   Wähle $d \in D(x)$ mit $\nabla f(x)d < 0$.

3. ***Wahl der Schrittweite***
   Wähle $\tilde{x}$ mit $f(\tilde{x}) \approx \min\{f(x + \lambda d) \mid x + \lambda d \in X, \lambda > 0\}$;
   $x := \tilde{x}$ und gehe nach 2.

Konvergenz

Bei Verfahren der Nichtlinearen Optimierung spricht man von Konvergenz, wenn entweder die nach endlich vielen Iterationsschritten bestimmte Lösung x_k oder alle Häufungspunkte der erzeugten Folge das Optimalitätskriterium, z.B. $\nabla f(x)d \geq 0$ für alle $d \in D(x)$, erfüllen. Ist M die Menge der Punkte des $\boldsymbol{R}^n$, die das Optimalitätskriterium erfüllen, so spricht man von *Konvergenz gegen* M. Da der endliche Fall, $x_k \in M$ für ein $k \in \boldsymbol{Z}_+$, trivial ist, wird meist nur untersucht, welche Häufungspunkte auftreten können.

Um Konvergenz gegen kritische Punkte zu erreichen, müssen die Wahl der Abstiegsrichtung und die Wahl der Schrittweite aufeinander abgestimmt werden. Die naheliegende Wahl der optimalen Schrittweite, d.h. die Bestimmung des exakten globalen Optimums $\tilde{x}$ in Schritt 3., ist aus numerischen Gründen meist nicht möglich.

Armijo-Suche (1. Ordnung) Sei $f \in C_1$, $0 < \alpha < 1$. Im Iterationsschritt k sei $\nabla f(x_k)d_k < 0$, $d_k \in D(x_k)$. Dann wähle $\lambda_k := 2^{-i(k)} \geq 0$ mit minimalen $i(k) \geq 0$,d. h. λ_k maximal, so daß

$$f(x_k + \lambda_k d_k) - f(x_k) \leq \alpha \cdot \lambda_k \nabla f(x_k)d_k. \tag{2.4}$$

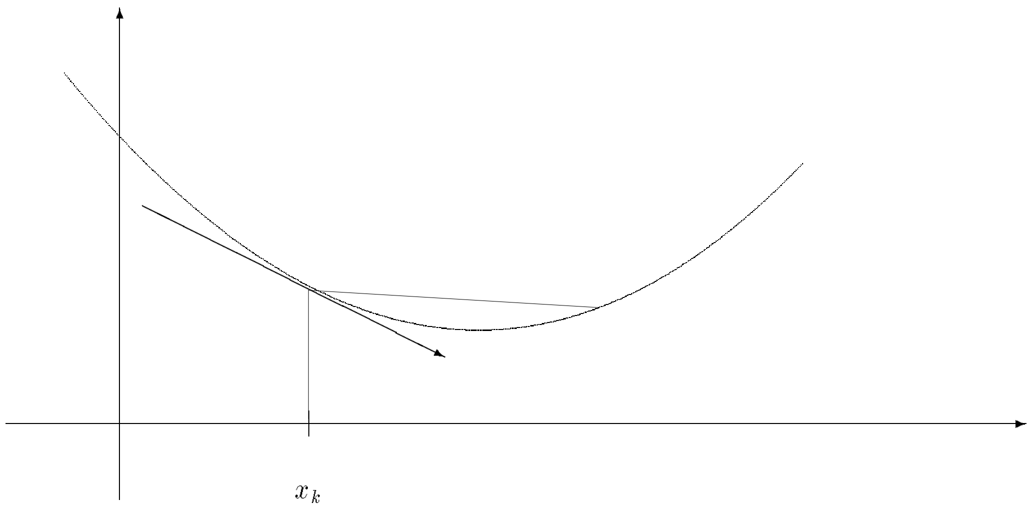


Abbildung III.14: Armijosuche (1. Ordnung). Sei $g(\lambda) := f(x_k+\lambda d_k)$. Hier genügt $i(k) = 2$, denn $g(\frac{1}{4}) \leq g(0) + \alpha\frac{1}{4}g'(0)$.

Unter den Voraussetzungen der Armijosuche ist der Index $i(k)$ wohldefiniert, da $0 < \alpha < 1$. Die Folge der erzeugten Funktionswerte $f(x_k)$ ist wegen (2.4) streng monoton fallend. Konvergiert eine Teilfolge $\{x_k\}_{k \in K}$ gegen $\bar{x}$, so konvergiert die Folge aller Funktionswerte gegen $f(\bar{x})$. Verwendet man die Armijosuche in Abstiegsverfahren, so läßt sich zeigen, daß konvergente Teilfolgen der erzeugten Iterationspunktefolge gegen kritische Punkte konvergieren.

Lemma 2.12 (Armijo-Abstieg 1. Ordnung) *Die Voraussetzungen der Armijo-Suche seien erfüllt und die Teilfolge* $\{x_k\}_{k\in K}$ *konvergiere gegen* $\bar{x}$*. Falls für geeignete Konstanten* $\beta, \delta, \gamma > 0$*:*

1. $\frac{\nabla f(x_k) d_k}{\|\nabla f(x_k)\| \|d_k\|} \leq -\delta \quad \forall\, k \in K$,
2. $\beta \parallel \nabla f(x_k) \parallel \leq \parallel d_k \parallel \leq \gamma \quad \forall\, k \in K$,

so gilt $\nabla f(\bar{x}) = 0$.

Beweis. 1. Fall: Für ein $\bar{\lambda}$ gilt: $\lambda_k \geq \bar{\lambda}$ für alle $k \in K$. Da für alle k die Ungleichung (2.4) und $\nabla f(x_k) d_k < 0$ gilt, folgt die Abschätzung

$$\begin{aligned} f(\bar{x}) - f(x_0) &= \sum_{k=0}^{\infty} (f(x_{k+1}) - f(x_k)) \\ &\leq \alpha \sum_{k=0}^{\infty} \lambda_k \nabla f(x_k) d_k \leq \alpha \bar{\lambda} \sum_{k\in K} \nabla f(x_k) d_k. \end{aligned}$$

Da die Summe beschränkt ist, bilden die Summanden eine Nullfolge. Wegen (1.) und (2.) lassen sich diese weiter abschätzen:

$$\nabla f(x_k) d_k \leq -\delta \parallel \nabla f(x_k) \parallel \parallel d_k \parallel \leq -\delta\beta \parallel \nabla f(x_k) \parallel^2 .$$

Wegen der Stetigkeit von Gradient und Norm folgt aus $\parallel \nabla f(x_k) \parallel \to 0$ unmittelbar $\parallel \nabla f(\bar{x}) \parallel = 0$.

2. Fall: $\{\lambda_k\}_{k\in K'} \to 0$ für eine Teilfolge $K' \subseteq K$. Für $2 \cdot \lambda_k$ ist die Ungleichung 2.4 nicht erfüllt. Mit dem Satz von Taylor (1. Ordnung) findet man:

$$\begin{aligned} \nabla f(x_k) d_k 2\lambda_k + o(\parallel d_k \parallel 2\lambda_k) &= f(x_k + d_k 2\lambda_k) - f(x_k) \\ &> \alpha 2\lambda_k \nabla f(x_k) d_k. \end{aligned}$$

Elementare Umformungen liefern mit Hilfe von (1.):

$$\frac{o(\parallel d_k \parallel 2\lambda_k)}{\parallel d_k \parallel 2\lambda_k} > \frac{(\alpha - 1)\nabla f(x_k) d_k}{\parallel d_k \parallel} \geq (1-\alpha)\delta \parallel \nabla f(x_k) \parallel \geq 0.$$

Für $k \to \infty$ konvergiert der Nenner der linken Seite wegen (2.) gegen 0. Daher konvergiert die linke Seite gegen 0. Also muß $\parallel \nabla f(\bar{x}) \parallel = 0$ gelten. □

3 Gradienten- und Newtonverfahren

In diesem Abschnitt untersuchen wir für einmal oder zweimal stetig differenzierbare Funktionen $f : \boldsymbol{R}^n \to \boldsymbol{R}$ zwei einfache Verfahren zur Lösung der Aufgabe

$$\min\{f(x) \mid x \in \boldsymbol{R}^n\}. \tag{3.1}$$

Gesucht werden dabei statt der globalen Minima Lösungen, die die notwendigen Bedingungen erster oder, bei C_2-Funktionen, zweiter Ordnung für lokale Minima (siehe Satz 2.1) erfüllen. Eines der ältesten Verfahren, das bereits 1847 von Cauchy zur Lösung von Gleichungssystemen formuliert wurde, ist das folgende Abstiegsverfahren.

Algorithmus 3.1 (Methode des steilsten Abstiegs)

1. *Wähle* $x_0 \in \boldsymbol{R}^n$.
2. *Für* $k = 0, 1, \ldots$:

$$d_k := -\nabla f(x_k)^T; \quad x_{k+1} := x_k + \lambda_k d_k.$$

Wegen der Wahl der Abstiegsrichtung nennt man das Verfahren auch Gradientenverfahren.

Wahl der Schrittweite λ_k

1. **(Optimale Schrittweite)**. Man versucht das globale Optimum in der jeweiligen Richtung zu bestimmen:

$$\alpha := f(x_k + \lambda_k d_k) := \min_{\lambda \geq 0} f(x_k + \lambda d_k).$$

   Dann verschwindet bei x_{k+1} die entsprechende Richtungsableitung $\nabla f(x_{k+1})^T d_k$, d.h. sukzessive Suchrichtungen sind orthogonal. Hier kann man Konvergenz gegen kritische Punkte untersuchen.

2. **(ϵ-optimale Schrittweite)**. Man akzeptiert Lösungen, deren Funktionswert nahe bei dem des globalen Optimums liegt:

$$f(x_k + \lambda_k d_k) \leq \alpha + \epsilon.$$

   Hier kann man nur Konvergenz gegen "approximativ kritische" Punkte, d.h. Punkte mit $\| \nabla f(x) \| \leq \epsilon$ erwarten.

3. **(Armijo-Suche 1. Ordnung)**. In den Voraussetzungen der Armijo-Suche (siehe Lemma 2.12) sind mit $\beta = \delta = 1$ zwei der drei Ungleichungen erfüllt, denn

$$\frac{\nabla f(x_k) d_k}{\| \nabla f(x_k) \|_2 \| d_k \|_2} = \frac{- \| d_k \|_2^2}{\| d_k \|_2 \| d_k \|_2} = -1 \quad \text{und} \quad \| \nabla f(x_k) \| = \| d_k \| .$$

Für die praktikable Armijo-Suche wollen wir einen Konvergenzsatz formulieren, dessen Voraussetzungen in typischer Weise die noch fehlende Ungleichung $\| d_k \| \leq \gamma$ im Lemma 2.12 erzwingen.

Satz 3.1 *Bei Schrittweitenwahl nach Armijo ist das Gradientenverfahren konvergent, falls* $N(x_0) := \{x \mid f(x) \leq f(x_0)\}$ *beschränkt ist.*

Beweis. Wegen der Stetigkeit von f ist die Niveaumenge $N(x_0)$ kompakt. Da die Armijosuche fallende Funktionswerte $f(x_k)$ garantiert, gehören alle Iterationspunkte zu $N(x_0)$. Wir betrachten eine konvergente Teilfolge, oBdA. $x_k \to \bar{x}$. Wie bereits oben festgestellt sind zwei der drei Ungleichungen in den Voraussetzungen des Lemmas 2.12 erfüllt für $\delta = \beta = 1$. Außerdem ist ∇f auf dem Kompaktum $N(x_0)$ beschränkt, etwa

$$\| \nabla f(x) \| \leq \gamma \quad \forall\, x \in N(x_0)$$

für ein $\gamma > 0$. Wegen $d_k = -\nabla f(x_k)^T$ ist auch die dritte Ungleichung des Lemmas 2.12 erfüllt. □

Quadratische Funktionen

Sei Q eine reell symmetrische, positiv definite $n \cdot n$-Matrix und sei $b \in \boldsymbol{R}^n$. Dann ist

$$f(x) := \frac{1}{2} x^T Q x - b^T x$$

eine quadratische Funktion.

Mit Hilfe einer zugehörigen quadratischen Funktion können wir eine beliebige Funktion $g \in C_2$ in der Nähe ihres lokalen Minimums $\bar{x}$ modellieren. Bricht man die Taylorentwicklung von g nach dem Term 2. Ordnung ab, so gilt:

$$g(x) \approx g(\bar{x}) + b^T(x - \bar{x}) + \frac{1}{2}(x - \bar{x})^T Q (x - \bar{x}),$$

wobei $b^T := \nabla g(\bar{x}) = 0$ und $Q := G(\bar{x})$ positiv semidefinit.

Eigenschaften quadratischer Funktionen

Eine quadratische Funktionen f läßt sich besonders einfach minimieren.

1. Das Minimum x_* ist gegeben durch die Gleichung $Qx_* = b$, denn $\nabla f(x) = x^T Q - b^T = (Qx - b)^T$.

2. Der minimale Funktionswert ist $f(x_*) = \frac{1}{2}(Q^{-1}b)QQ^{-1}b - b^T Q^{-1} b = -\frac{1}{2} b^T Q^{-1} b$.

3. Das quadratische Modell der Taylorentwicklung (siehe oben) ist exakt, d.h.

$$f(x) = f(x_*) + \frac{1}{2}(x - x_*)^T Q (x - x_*).$$

   Daher kann man zur Vereinfachung die rein quadratische Funktion $\tilde{f}(x) := f(x) - f(x_*)$ untersuchen, die dasselbe Minimum besitzt.

4. Das globale Minimum in einer Richtung $d \neq 0$ kann explizit durch x angegeben werden. Die Funktion $h : \boldsymbol{R} \to \boldsymbol{R}$ mit $h(\lambda) := f(x + \lambda d)$ ist eine konvexe C_1-Funktion. Das Minimum λ_* ist gegeben durch die Bedingung $0 = \frac{dh}{d\lambda}(\lambda_*) = \nabla f(x + \lambda_* d) d = [Q(x + \lambda_* d) - b]^T d$. Auflösung ergibt

$$\lambda_* = -\frac{(Qx - b)^T d}{d^T Q d}$$

   .

Damit läßt sich das Gradientenverfahren für quadratische Funktionen explizit vereinfachen.

Algorithmus 3.2 (Steilster Abstieg für quadratische Funktionen)

1. *Wähle* $x_0 \in \boldsymbol{R}^n$.

2. *Für* $k = 0, 1, \ldots$*:*

$$d_k := -(Qx_k - b); \quad x_{k+1} := x_k + \frac{d_k^T d_k}{d_k^T Q d_k} d_k.$$

Beispiele zum Konvergenzverhalten

Die Konvergenzeigenschaften sind je nach Form der quadratischen Funktion sehr unterschiedlich:

1. Für $f(x) = \frac{1}{2}x^T x$ findet man das Optimum im ersten Iterationsschritt, denn $x_1 = x_0 + \frac{x_0^T x_0}{x_0^T x_0}(-x_0) = 0$.

2. Für $f(x) = x^2 + 100y^2$ konvergiert die Folge der Iterationspunkte linear und es gilt $f(x_{k+1}) \approx 0,63 \cdot f(x_k)$ für alle $k = 0, 1, \ldots$.

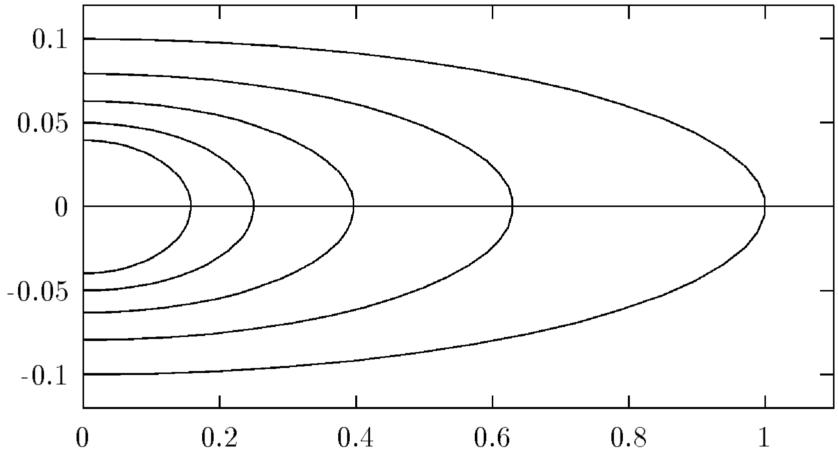


Abbildung III.15: Konvergenz für quadratische Funktionen

Die langsame Konvergenz beim zweiten Beispiel erweist sich als typisch für das Gradientenverfahren. Eine entsprechende Abschätzung liefert der folgende Satz. Hier verwenden wir die bekannte Tatsache, daß eine positiv definite $n \cdot n$-Matrix eine Basis aus paarweise orthonormalen Eigenvektoren zu ihren n positiven reellen Eigenwerten besitzt.

Satz 3.2 *Sei f quadratisch, seien $0 < \mu_1 \leq \ldots \leq \mu_n$ die Eigenwerte von Q, und sei x_* Minimum von f. Dann gilt für die mit Algorithmus 3.2 erzeugte Folge*

$$f(x_{k+1}) - f(x_*) \leq \frac{(\frac{\mu_n}{\mu_1} - 1)^2}{(\frac{\mu_n}{\mu_1} + 1)^2}(f(x_k) - f(x_*)).$$

Beweis. Modulo einer Koordinatenverschiebung können wir oBdA annehmen (siehe Eigenschaften quadratischer Funktionen 3.), daß $b = 0$ und daher $f(x_*) = 0$, $x_* = 0$ und $\nabla f(x) = x^T Q = (Qx)^T$ gelten. Aus der globalen Minimierung in den untersuchten Richtungen folgt:

$$(\dagger) \quad f(x_{k+1}) \leq f(x_k - \lambda Q x_k) \quad \forall\, \lambda \geq 0.$$

Die paarweise orthonormalen Eigenvektoren $a_1, a_2, \ldots, a_n$ zu den Eigenwerten $\mu_1, \mu_2, \ldots, \mu_n$ der reell symmetrischen, positiv definiten Matrix Q bilden eine Orthonormalbasis.

Ist $x_k := \sum_{i=1}^n \alpha_i a_i$ die Koordinatendarstellung von x_k, so ist der Funktionswert gegeben durch

$$(*) \quad f(x_k) = \frac{1}{2} x_k^T Q x_k = \frac{1}{2} \sum_{i=1}^n (\alpha_i)^2 \mu_i.$$

Die Koordinatendarstellung des Argumentes in (†) ist

$$x_k - \lambda Q x_k = \sum_{i=1}^n \alpha_i a_i - \lambda \sum_{i=1}^n \alpha_i Q a_i = \sum_{i=1}^n \alpha_i (1 - \lambda \mu_i) a_i.$$

Daher ergibt sich analog zu $(*)$ für spezielle Wahl der Schrittweite $\bar{\lambda} = \frac{2}{\mu_n + \mu_1}$ der Funktionswert

$$f(x_k - \bar{\lambda} Q x_k) = \frac{1}{2} \sum_{i=1}^n \alpha_i^2 (1 - \bar{\lambda} \mu_i)^2 \mu_i = \frac{1}{2} \sum_{i=1}^n \alpha_i^2 \left[\frac{\mu_n + \mu_1 - 2\mu_i}{\mu_n + \mu_1} \right]^2 \mu_i.$$

Da $\max_i \, (\mu_n + \mu_1 - 2\mu_i)^2 = (\mu_n - \mu_1)^2$, gilt

$$f(x_{k+1}) \leq f(x_k - \bar{\lambda} Q x_k) \leq \left(\frac{\mu_n - \mu_1}{\mu_n + \mu_1} \right)^2 f(x_k).$$

□

Bemerkung (Konvergenzverhalten):

1. Das Verfahren ist bezogen auf die Funktionswerte $r_k := f(x_k) - f(x_*)$ linear konvergent, da $\left(\frac{\mu_n - \mu_1}{\mu_n + \mu_1} \right)^2 < 1$.

2. Es ist praktisch unbrauchbar, falls $\mu_n \gg \mu_1$, da für $r := \frac{\mu_n}{\mu_1} \to \infty$ folgt $(\frac{r-1}{r+1})^2 \to 1$.

3. Die Konvergenzanalyse überträgt sich lokal in der Nähe eines Minimums auf den nichtquadratischen Fall. Mit $x_k \to x_*$ gilt $F(x_k) \to F(x_*) =: Q$, so daß die Eigenwerte von $F(x_*)$ das Konvergenzverhalten nahe x_* bestimmen.

4. Ein weiterer Nachteil des Verfahrens ist seine Skalierungsabhängigkeit. *Skalierung der Koordinaten*, gegeben etwa durch $y_i := \alpha_i x_i$ für $i = 1, \ldots, n$, ist eine spezielle lineare Koordinatentransformation. Nach Koordinatenwechsel $y := H^{-1} x$ mit regulärer Matrix H wird die Funktion g mit $g(y) := f(Hy)$ minimiert. Die zugehörigen Gradienten sind gegeben durch $\nabla f(Hy) H$. In y-Koordinaten ist die neue Suchrichtung $-H^T \nabla f(Hy)^T$. In x-Koordinaten lautet die so gefundene neue Suchrichtung daher $-HH^T \nabla f(x)^T$. Nicht nur die Suchrichtung ist sehr abhängig von den gewählten Koordinaten, auch die Eigenwerte können sich entscheidend ändern, es gilt $G(y) = H^T F(Hy) H$. Durch Skalierung der Koordinaten werden insbesondere die alten Eigenwerte μ_i einer quadratischen Funktion in die Eigenwerte $\frac{\mu_i}{\alpha_i^2}$ transformiert.

Quadratische lokale Approximation

Wir betrachten nun nochmals die abgebrochene Taylorreihe als quadratische Approximation der zu minimierenden Funktion, wobei wir jetzt bei $\bar{x}$, dem momentanen Iterationspunkt, entwickeln:

$$f(x) \approx f(\bar{x}) + \nabla f(\bar{x})(x-\bar{x}) + \frac{1}{2}(x-\bar{x})^T F(\bar{x})(x-\bar{x}) =: Q(x)$$

Falls $F(\bar{x})$ positiv definit ist, so ist durch $x_* - \bar{x} = F(\bar{x})^{-1}[-\nabla f(\bar{x})^T]$ das globale Minimum x_* der quadratischen Funktion $Q(x)$ gegeben. Für nichtlineare C_2-Funktionen kann man x_* als neuen Iterationspunkt betrachten. Das daraus abgeleitete Verfahren ist das

Algorithmus 3.3 (Newton-Verfahren)

1. *Wähle* $x_0 \in \boldsymbol{R}^n$.
2. *Für* $k = 0, 1, \ldots$*:*

$$F(x_k)d_k := -\nabla f(x_k)^T; \quad x_{k+1} := x_k + d_k.$$

Falls $F(x_k)$ positiv definit ist, so liegt wegen

$$\nabla f(x_k)d_k = -\nabla f(x_k)F^{-1}(x_k)\nabla f(x_k)^T$$

eine Abstiegsrichtung vor. Im allgemeinen wird dies aber für nichtlineare Funktionen nicht gelten, so daß globale Konvergenz nur unter zusätzlichen Annahmen zu erzwingen ist. Zur Verbesserung des Verfahrens kann man auch statt der festen Schrittweite $\lambda = 1$ eine Schrittweitensteuerung einbauen. Wir untersuchen zunächst das lokale Konvergenzverhalten. Zur Formulierung eines typischen Konvergenzsatzes benötigen wir Matrixnormen.

Matrixnormen

Im folgenden ist mit $\| \cdot \|$ stets die Euklidische Norm $\| \cdot \|_2$ bezeichnet. Eine aus der euklidischen Norm abgeleitete Matrixnorm ist gegeben durch

$$\| A \| := \max_{\|x\|=1} \| Ax \| .$$

Offenbar gilt $\| A \| = \max_{x \neq 0} \sqrt{\frac{x^T A^T A x}{x^T x}} = \sqrt{\lambda_{\max}(A^T A)}$, wobei $\lambda_{\max}$ der maximale Eigenwert von $A^T A$ ist.

Diese Matrixnorm ist *verträglich* mit der euklidischen Norm, d.h. es gilt

$$\| Ax \| \leq \| A \| \cdot \| x \|,$$

und sie ist *submultiplikativ*, d.h. es gilt

$$\| AB \| \leq \| A \| \cdot \| B \| .$$

Auf diese kanonische Weise kann man zu jeder Vektornorm passende Matrixnormen ableiten.

Satz 3.3 (Lokal quadratische Konvergenz des Newton-Verfahrens) *Sei $X \subseteq \boldsymbol{R}^n$ offen, $\bar{x} \in X$ und sei $f \in C_2(X)$ mit*

1. *$\nabla f(\bar{x}) = 0$, $F(\bar{x})$ regulär,*

2. *es gibt ein $\mu \geq 0$ und eine Umgebung $U(\bar{x}) \subseteq X$, so daß für alle $x \in U(\bar{x})$ gilt*

$$\| F(x) - F(\bar{x}) \| \leq \mu \| x - \bar{x} \| .$$

Dann gibt es ein $\epsilon > 0$, so daß das Newtonverfahren in $U_\epsilon(\bar{x})$ mindestens quadratisch gegen $\bar{x}$ konvergiert.

Beweis. Wähle $U(\bar{x})$ in (2.) klein genug, so daß für alle $x \in U(\bar{x})$ die Norm der Inversen beschränkt, d.h. $\| F^{-1}(x) \| \leq C$ für eine Konstante $C > 0$, und die Iterationsfunktion $g(x) := x - F^{-1}(x)\nabla f(x)^T$ wohldefiniert ist. Um $g(x) - \bar{x}$ abschätzen zu können, untersuchen wir zunächst den Gradienten. Da $\nabla f(\bar{x}) = 0$, gilt

$$-\nabla f(x)^T = \underbrace{-\nabla f(x)^T + F(\bar{x})x}_{=: h(x)} + \underbrace{\nabla f(\bar{x})^T - F(\bar{x})\bar{x}}_{=-h(\bar{x})} - F(\bar{x})(x - \bar{x}).$$

Die Ableitung der hier definierten Funktion h ist $-F(x) + F(\bar{x})$. Aus dem Mittelwertsatz erhalten wir daher die Abschätzung

$$\| h(x) - h(\bar{x}) \| \leq \| x - \bar{x} \| \cdot \sup_{y \in [x, \bar{x}]} \| F(y) - F(\bar{x}) \| \leq \mu \| x - \bar{x} \|^2 .$$

Nun gilt

$$\begin{aligned} F(x)(g(x) - \bar{x}) &= \underbrace{F(x)(g(x) - x)}_{=-\nabla f(x)^T} + F(x)(x - \bar{x}) \\ &= h(x) - h(\bar{x}) + [F(x) - F(\bar{x})](x - \bar{x}). \end{aligned}$$

Multiplikation mit F^{-1}, Anwendung der vorbereiteten Abschätzung und Ausnutzen der Dreiecksungleichung liefern

$$\begin{aligned} \| g(x) - \bar{x} \| &\leq \| F^{-1}(x) \| \cdot [\, \| h(x) - h(\bar{x}) \| + \| F(x) - F(\bar{x}) \| \cdot \| x - \bar{x} \| \,] \\ &\leq C \cdot 2\mu \| x - \bar{x} \|^2 . \end{aligned}$$

Wähle $\epsilon > 0$ hinreichend klein, so daß $0 \leq \epsilon \cdot 2\mu C =: \alpha < 1$ und $U_\epsilon(\bar{x}) \subseteq U(\bar{x})$. Wähle $x_0 \in U_\epsilon(\bar{x})$. Damit erhalten wir

$$\| x_1 - \bar{x} \| = \| g(x_0) - \bar{x} \| \leq 2\mu C \| x_0 - \bar{x} \|^2 < \alpha \| x_0 - \bar{x} \| < \alpha\epsilon,$$

woraus per Induktion ("iterativ") folgt

$$\| x_k - \bar{x} \| < \alpha^k \epsilon \quad \forall\, k \geq 0.$$

Also gilt $x_k \to \bar{x}$. Diese Folge ist mindestens quadratisch konvergent, denn

$$\| x_{k+1} - \bar{x} \| = \| g(x_k) - \bar{x} \| \leq 2\mu C \| x_k - \bar{x} \|^2 .$$

□

Das lokal quadratisch und damit schnell konvergente Newton-Verfahren verlangt einen wesentlich höheren Aufwand als das Gradientenverfahren. Pro Iteration muß neben dem Gradienten die Matrix $F(x)$ berechnet oder approximiert, und das Gleichungssystem $F(x)d := \nabla f(x)^T$ gelöst werden. Darüberhinaus bleibt, wie oben bereits angesprochen, die globale Konvergenz unsicher. Beide Schwierigkeiten versucht man mit geeigneten Modifikationen aus dem Weg zu räumen. Dazu sei G eine stetige Matrixfunktion, die durch $x \mapsto G(x)$ definiert ist, wobei $G(x)$ reell, symmetrisch und positiv definit für alle x.

Algorithmus 3.4 (Modifizierte Newton-Verfahren)

1. *Wähle* $x_0 \in \boldsymbol{R}^n$.
2. *Für* $k = 0, 1, \ldots$*:*

$$G(x_k)d_k := -\nabla f(x_k)^T; \quad x_{k+1} := x_k + \lambda_k d_k.$$

Bemerkung:

1. Die Schrittweite λ_k wird mit Hilfe geeigneter eindimensionaler Minimierungsverfahren gewählt.

2. Die Matrixfunktion wird als Näherung von F gewählt, d.h. $G(x) \approx F(x)$. Für $G \equiv I$ bzw. $G \equiv F$ erhält man formal das Gradienten-, bzw. das Newton-Verfahren, vorausgesetzt F ist positiv definit. Das "vereinfachte" Newton-Verfahren erhält man für $G(x) \equiv F(x_0)$. Praktikabel sind flexible Verfahren, bei denen etwa zunächst Gradientenschritte und dann für einen Zyklus von Schritten vereinfachte Newton-Iterationen durchgeführt werden. Verschlechtert sich das Konvergenzverhalten im Zyklus, wird F erneut approximiert. Wird quadratische Konvergenz erkennbar, wird die Schrittweite auf $\lambda = 1$ fixiert.

3. Ist $f \in C_2(X)$, und $F(x)$ positiv definit auf X, so ist f streng konvex. Dann besitzt f ein eindeutig bestimmtes strenges Minimum, etwa $\bar{x}$. Bei $\bar{x}$ konvergiert das Newton-Verfahren lokal quadratisch.

4. Modifizierte Newton-Verfahren sind stets Abstiegsverfahren, da mit G auch G^{-1} positiv definit sind, woraus $\nabla f(x)d = -\nabla f(x)G(x)^{-1}\nabla f^T(x) < 0$ folgt. Bestimmt man die Schrittweite mit Armijo-Suche und ist die Niveaumenge $N(x_0) := \{x \mid f(x) \leq f(x_0)\}$ beschränkt, so erhält man globale Konvergenz wie in Satz 3.1.

4 Konjugierte Richtungen

Auch in diesem Abschnitt werden wir zunächst wieder quadratische Minimierungsprobleme untersuchen. Sei also Q eine reelle, symmetrische, positiv definite Matrix und sei $f : \boldsymbol{R}^n \to \boldsymbol{R}$ definiert durch $f(x) := \frac{1}{2}x^TQx - b^Tx$ mit Minimum $x_* = Q^{-1}b$ und Ableitung $\nabla f(x)^T = Qx - b =: g(x)$.

Definition (Konjugierte Richtungen): Sei $k \in \boldsymbol{N}$. Vektoren $d_0, \ldots, d_k \in \boldsymbol{R}^n \setminus \{0\}$ heißen *konjugiert*, falls $d_i^TQd_j = 0$ für alle $0 \le i < j \le k$. Zur Abkürzung sei $\rho_i := d_i^TQd_i$ für $0 \le i \le k$.

Konjugierte Vektoren sind immer linear unabhängig, denn es gilt für α mit $\alpha_i \neq 0$: $d_i^TQ\left(\sum \alpha_j d_j\right) = \rho_i\alpha_i \neq 0$.

Lemma 4.1 *Sind $d_0, \ldots, d_{n-1}$ konjugiert, so gilt*

$$Q^{-1} = D := \sum_{k=0}^{n-1} \frac{1}{\rho_k} d_k d_k^T, \quad x_* = \sum_{k=0}^{n-1} \frac{1}{\rho_k} d_k d_k^T b = \sum_{k=0}^{n-1} \frac{d_k^T b}{\rho_k} d_k$$

Beweis. Die Formel für Q^{-1} impliziert die für x_*. Es genügt daher zu zeigen, daß für alle $x \in \boldsymbol{R}^n$ gilt: $DQx = x$. Für $x = \sum_{i=0}^{n-1} \alpha_i d_i$ findet man $d_k^TQx = \sum_{i=0}^{n-1} \alpha_i d_k^TQd_i = \alpha_k\rho_k$. Daher gilt

$$DQx = \sum_{k=0}^{n-1} \frac{1}{\rho_k} d_k d_k^T Qx = \sum_{k=0}^{n-1} \frac{1}{\rho_k} d_k(\alpha_k\rho_k) = x$$

□

Für bekannte konjugierte Richtungen $d_0, \ldots, d_{n-1}$ bietet es sich daher an, sukzessive in diesen Richtungen die Funktion f zu minimieren.

Algorithmus 4.1 (Verfahren konjugierter Richtungen)

1. *Wähle $x_0 \in \boldsymbol{R}^n$.*

2. *Für $k = 0, 1, \ldots, n-1$; $d_0, \ldots, d_{n-1}$ konjugiert:*

$$\lambda_k := -\frac{1}{\rho_k}(Qx_k - b)^T d_k; \quad x_{k+1} := x_k + \lambda_k d_k.$$

Die gewählte Schrittweite λ_k ist optimal. Als Abbruchkriterium kann man das Verschwinden des Gradienten, d.h. $g_k := g(x_k) \approx 0$, testen.

Der von Vektoren $p_0, \ldots, p_k \in \boldsymbol{R}^n$ aufgespannte Unterraum wird in üblicher Weise bezeichnet:

$$< p_0, \ldots, p_k > := \{\sum_{i=0}^{k} \alpha_i p_i \mid \alpha_i \in \boldsymbol{R}; i = 0, \ldots, k\}.$$

Lemma 4.2 *Für $k = 1, \ldots, n$ gilt im Verfahren konjugierter Richtungen*

1. *g_k orthogonal auf $T_k := < d_0, \ldots, d_{k-1} >$,*

2. *x_k ist globales Minimum von f auf $T_k + x_0$,*

3. *$x_* = x_n$.*

Beweis. Aus (1.) folgt (2.), da $g_k^T = \nabla f(x_k)$. Aus (2.) folgt (3.), da $T_n + x_0 = \boldsymbol{R}^n$. Zu zeigen ist also nur noch (1.).

Für $i < k$ gilt $x_k = x_i + \sum_{j=i}^{k-1} \lambda_j d_j$, also auch $Qx_k - b = Qx_i - b + \sum_{j=i}^{k-1} \lambda_j Q d_j$. Multiplikation von links mit d_i^T ergibt

$$d_i^T g_k = d_i^T g_i + \sum_{j=i}^{k-1} \lambda_j d_i^T Q d_j = d_i^T g_i + \lambda_i \rho_i .$$

Wegen $\lambda_i = -\frac{1}{\rho_i} g_i^T d_i$ verschwindet die rechte Seite für alle $i < k$, d.h. $g_k \perp T_k$. □

Lemma 4.3 (Gram-Schmidt-Orthogonalisierung) *Sind für ein r mit $1 \leq r \leq n$ die Vektoren $p_0, \ldots, p_{r-1}$ linear unabhängig, so sind die durch $d_0 := p_0$, und für $k = 0, \ldots, r-2$ durch:*

$$d_{k+1} := p_{k+1} - \sum_{i=0}^{k} \frac{p_{k+1}^T Q d_i}{\rho_i} d_i$$

definierten Vektoren konjugiert und es gilt

$$< p_0, \ldots, p_{r-1} > = < d_0, \ldots, d_{r-1} > .$$

Beweis. Der Beweis wird induktiv über r geführt. Der Induktionsanfang ist trivial wegen $< d_0 > = < p_0 >$. Für den Induktionsschluß $r \to r+1$ sei $j < r$. Dann gilt

$$d_j^T Q d_r = d_j^T Q p_r - \sum_{i=0}^{r-1} \frac{p_r^T Q d_i}{\rho_i} d_j^T Q d_i = 0,$$

d.h. d_r ist konjugiert zu d_j für $j < r$. Nun gilt nach Induktionsannahme $< d_0, \ldots, d_{r-1} > = < p_0, \ldots, p_{r-1} >$, und daher nach Rekursionsformel

$$d_r \in < d_0, \ldots, d_{r-1}, p_r > = < p_0, \ldots, p_r > .$$

Damit folgt $< d_0, \ldots, d_r > \subseteq < p_0, \ldots, p_r >$, wobei "=" gelten muß, da die d_i linear unabhängig sind. □

Mit Hilfe der Gram-Schmidt-Orthogonalisierung kann man also aus jeder Basis eine Basis aus konjugierten Richtungen erzeugen.

Wahl der p_k

Die Freiheit bei der Wahl der Basis nutzt man aus, um die p_k sukzessive so zu bestimmen, daß $g_k^T d_k$ jeweils möglichst klein ist. Zur Normierung dieser Größe sei zunächst $\| p_k \|= 1$ festgelegt. Im ersten Schritt ist der schon aus den Gradientenverfahren bekannte steilste Abstieg minimal, d.h. $d_0 := p_0 := -\frac{g_0}{\|g_0\|}$.

In den Iterationen für $0 \leq k \leq n-2$ ist die Wahl der zu minimierenden Größe eingeschränkt durch die Rekursion. Es gilt

$$g_{k+1}^T d_{k+1} = g_{k+1}^T p_{k+1} - \sum_{i=0}^{k} \frac{p_{k+1}^T Q d_i}{\rho_i} g_{k+1}^T d_i,$$

wobei die letzte Summe wegen $g_{k+1}^T d_i = 0$ verschwindet. Daher wird der Ausdruck minimal für $p_{k+1} := -\frac{g_{k+1}}{\|g_{k+1}\|}$. Da im allgemeinen $p_{k+1} \neq d_{k+1}$, ist im allgemeinen in den Iterationen kein steilster Abstieg mehr möglich. Ohne Berücksichtigung der Normierung kann man die Richtungen also durch

(4.1) $$p_k := -g_k \text{ für } k = 0, \ldots, n-1$$

angeben. Sind diese Richtungen nun auch noch linear unabhängig? Dies kann natürlich nur gelten, wenn keiner der Gradienten verschwindet, da sonst das Verfahren vorzeitig mit der Optimallösung abbricht.

Lemma 4.4 *Verwendet man sukzessive die durch Lemma 4.3 aus den Richtungen in Gleichung (4.1) erzeugten Vektoren im Verfahren 4.1 und verschwindet in keinem Schritt der Gradient, so gilt:*

1. $g_0, \ldots, g_{n-1}$ *paarweise orthogonal,*
2. $< g_0, \ldots, g_k > = < d_0, \ldots, d_k >$ *für* $k = 0, \ldots, n-1$,
3. $d_{k+1} = -g_{k+1} + \frac{1}{\rho_k}(g_{k+1}^T Q d_k) d_k$.

Beweis. (1.) und (2.) lassen sich simultan zeigen: da $g_0 = -d_0$ folgt aus Lemma 4.2 $g_1 \perp g_0$, und daher aus Lemma 4.3 $< g_1, g_0 > = < d_1, d_0 >$. Aus Lemma 4.2 folgt wieder $g_2 \perp < g_1, g_0 >$ usw. Sei $0 \leq j \leq n-2$. Es gilt

(4.2) $$g_{j+1} - g_j = Q(x_{j+1} - x_j) = \lambda_j Q d_j,$$

da nach 2.) $g_{j+1} \neq g_j$ und somit $\lambda_j \neq 0$ gilt. Die Orthogonalität der Gradienten liefert daher für $0 \leq j < k \leq n-2$

$$0 = g_{k+1}^T (g_{j+1} - g_j) = \lambda_j g_{k+1}^T Q d_j.$$

Wegen $\lambda_j \neq 0$ gilt $g_{k+1}^T Q d_j = 0$, woraus folgt

$$d_{k+1} = -g_{k+1} - \sum_{i=0}^{k} \frac{-g_{k+1}^T Q d_i}{\rho_i} d_i = -g_{k+1} + \frac{g_{k+1}^T Q d_k}{\rho_k} d_k.$$

□

Nun können wir für quadratische Funktionen das vollständige Verfahren von Hesteness, Stiefel (1952) beschreiben.

Algorithmus 4.2 (Konjugierte Gradienten bei quadratischen Funktionen)

1. Wähle $x_0 \in \boldsymbol{R}^n$; $\quad g_0 := Qx_0 - b$.
   Falls $g_0 = 0$, dann stop.
   $d_0 := -g_0$; $\quad k := 0$.

2. $\lambda_k := -\frac{g_k^T d_k}{\rho_k}$; $\quad x_{k+1} := x_k + \lambda_k d_k$.

3. $g_{k+1} := g_k + \lambda_k Q d_k$;
   falls $g_{k+1} = 0$, dann stop.

4. $\beta_k := \frac{g_{k+1}^T Q d_k}{\rho_k}$; $\quad d_{k+1} := -g_{k+1} + \beta_k d_k$;
   $k := k+1$; $\quad$ gehe nach 2.

Bemerkung (Zur Konvergenz):

1. Man erzeugt lokal unter allen konjugierten Abstiegsrichtungen die mit dem steilsten Abstieg.

2. Bei etwa doppeltem Aufwand pro Iteration im Vergleich zum Gradientenverfahren erhält man bei quadratischen Funktionen Konvergenz in n Iterationen.

3. Rundungsfehler führen allerdings auf $g_n \neq 0$. Dann kann man für $k \geq n$ weitere Iterationen anschließen und abbrechen, wenn $\| g_k \| \leq \epsilon$.

Im folgenden Lemma werden einige äquivalente Formeln bereitgestellt, die teilweise die notwendigen Berechnungen vereinfachen. Im wesentlichen dienen sie allerdings zur Modifikation des Verfahrens bei nichtquadratischen Funktionen.

Lemma 4.5 (Rechenalternativen) *Für $k = 0, 1, \ldots, n-1$ gilt:*

1. $\lambda_k = \frac{1}{\rho_k} g_k^T g_k$,

2. $\beta_k = \frac{1}{\rho_k} g_{k+1}^T Q d_k = \frac{(g_{k+1} - g_k)^T g_{k+1}}{g_k^T g_k} = \frac{g_{k+1}^T g_{k+1}}{g_k^T g_k}$.

Beweis. zu 1.): Im Verfahren konjugierter Gradienten wurden bereits $d_k = -g_k + \beta_{k-1} d_{k-1}$ und $\lambda_k = -\frac{g_k^T d_k}{\rho_k}$ benutzt. Also gilt

$$-g_k^T d_k = g_k^T g_k - \beta_{k-1} g_k^T d_{k-1} = g_k^T g_k.$$

zu 2.): Unter Verwendung der Gleichung (4.2) finden wir

$$g_{k+1}^T Q d_k = \frac{1}{\lambda_k} g_{k+1}^T (g_{k+1} - g_k) = \frac{\rho_k}{g_k^T g_k} g_{k+1}^T (g_{k+1} - g_k) = \rho_k \frac{g_{k+1}^T g_{k+1}}{g_k^T g_k}.$$

□

Für nichtquadratische Funktionen ersetzt man $g_k \sim \nabla f(x_k)^T$ und $Q \sim F(x_k)$ und berechnet x_{k+1} mit näherungsweise optimaler Schrittweite in Abstiegsrichtung. Die verschiedenen Formeln für β_k liefern dann verschiedene bekannte Verfahren für nichtquadratische Funktionen:

1. **Daniel, 1967:** $\beta_k := \frac{g_{k+1}^T F(x_{k+1}) d_k}{d_k^T F(x_{k+1}) d_k}$,
2. **Polak-Ribière, 1969:** $\beta_k := \frac{(g_{k+1} - g_k)^T g_{k+1}}{g_k^T g_k}$,
3. **Fletcher-Reeves, 1964:** $\beta_k := \frac{g_{k+1}^T g_{k+1}}{g_k^T g_k}$.

Algorithmus 4.3 (Konjugierte Gradienten bei nichtlinearen Funktionen)

1. Wähle $x_0 \in \boldsymbol{R}^n$; $\quad g_0 := \nabla f(x_0)$.
   Falls $g_0 = 0$, dann stop.
   $d_0 := -g_0$; $\quad k := 0$.

2. Bestimme (optimale) Schrittweite λ_k in Richtung d_k;
   $x_{k+1} := x_k + \lambda_k d_k$.

3. $g_{k+1} := \nabla f(x_{k+1})$;
   falls $g_{k+1} = 0$, dann stop;
   falls $k = n - 1$, dann gehe nach 6.

4. Berechne β_k; $\quad d_{k+1} := -g_{k+1} + \beta_k d_k$.

5. Falls $g_{k+1}^T d_{k+1} > 0$, dann gehe nach 6.;
   $k := k + 1$; $\quad$ gehe nach 2.

6. $x_0 := x_{k+1}$; $\quad d_0 := -g_{k+1}$;
   $k := 0$; $\quad$ gehe nach 2.

Konvergenz

1. Globale Konvergenz wird unter den üblichen Voraussetzungen durch den nach n Iterationen eingeschobenen neuen Gradientenschritt erzwungen. Dieser startet einen neuen Zyklus von n Iterationen mit konjugierten Gradienten.

2. Bei geeigneten Voraussetzungen wird quadratische Konvergenz wie beim Newtonverfahren erreicht. Dabei entsprechen n Iterationen einem Newtonschritt.

Index

Literaturverzeichnis

[1] V. Chviátal. *Linear Programming.* W.H.Freeman and Company, New York, 1983.

[2] T. H. Cormen, C.E. Leiserson, and R.L. Rivest. *Introduction to Algorithms.* MIT Press, Cambridge, Ma., 1990.

[3] S. Even. *Graph Algorithms.* Computer Science Press, 1979.

[4] R. Fletcher. *Practical Methods of Optimization.* John Wiley & Sons, Chichester, 1987.

[5] M. Gondran and M. Minoux. *Graphs and Algorithms.* John Wiley & Sons, Chichester/New York, 1984.

[6] M. Groetschel, L. Lovasz, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization.* Springer, Berlin, 1988.

[7] R. Horst. *Nichtlineare Optimierung.* Carl Hanser, Muenchen/Wien, 1979.

[8] R. Horst and H. Tuy. *Global Optimization.* Springer, Berlin, second edition, 1992.

[9] Jr. J. E. Dennis and Robert B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations.* Prentice-Hall Inc., 1983. Prentice-Hall Series in Computational Mathematics.

[10] D. Jungnickel. *Graphen, Netzwerke und Algorithmen.* Bibliographisches Institut, Mannheim, 1987.

[11] D.E. Knuth. *The Art of computer programming, Vol. 1: Fundamental Algorithms.* Series in Computer Science and Information Processing. Addison-Wesley, Reading, 1969.

[12] E.L. Lawler, J.K. Lenstra, A.H.G Rinnooy Kan, and D.B. Shmoys, editors. *The Travelling Salesman Problem.* John Wiley & Sons, Manchester/New York, 1985.

[13] Eugene L. Lawler. *Combinatorial Optimization: Networks and Matroids.* Holt, Rinehart and Winston, New York, 1976.

[14] J. Leeuven, v. *Algorithms and Complexity.* Elsevier, Amsterdam, 1990.

[15] T. Lengauer. *Combinatorial Algorithms for Integrated Circuit Layout.* Teubner, Stuttgart, 1990.

[16] K. Mehlhorn. *Graph Algorithms and NP-Completeness*, volume 2 of *Data Structures and Algorithms.* Springer, Berlin, 1984.

[17] K. Mehlhorn. *Multi-dimensional Searching and Computational Geometry*, volume 3 of *Data Structures and Algorithms.* Springer, Berlin, 1984.

[18] M. Minoux. *Mathematical Programming. Theory and Algorithms.* John Wiley & Sons, Chichester, 1986.

[19] B. M. E. Moret and H. D. Shapiro. *Algorithms from P to NP.* Benjamin Cummings, Redwood City, 1991.

[20] K.G. Murty. *Linear Programming.* John Wiley & Sons, New York/Chichester, 1983.

[21] K.G. Murty. *Linear Complementarity, Linear and Nonlinear Programming.* Heldermann, Berlin, 1988.

[22] George L. Nemhauser and Laurence A. Wolsey. *Integer and Combinatorial Optimization.* John Wiley & Sons, New York, 1988.

[23] G.L. Nemhauser, A.H.G. Rinnooy Kan, and M.J. Todd. *Optimization.* North-Holland, Amsterdam, 1989.

[24] C.H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization (Algorithms and Complexity).* Prentice-Hall, New Jersey, 1982.

[25] R. Gary Parker and Ronald L. Rardin. *Discrete Optimization.* Academic Press, Boston, 1988.

[26] H.M. Salkin and K. Mathur. *Foundations for Integer Programming.* North-Holland, New York, 1989.

[27] A. Schrijver. *Theory of Linear and Integer Programming.* John Wiley & Sons, Chichester/New York, 1986.

[28] Peter Spellucci. *Numerische Verfahren der nichtlinearen Optimierung.* Birkhaeuser, Basel, 1993.

[29] J. Stoer. *Einfuehrung in die Numerische Mathematik I.* Springer, Berlin, 1989.

[30] Robert E. Tarjan. Finding optimum branchings. *Networks*, 7:25–35, 1977.

[31] Uwe Zimmermann. *Linear and Combinatorial Optimization in Ordered Algebraic Structures.* North-Holland, Amsterdam, New-York, Oxford, Tokyo, 1981.